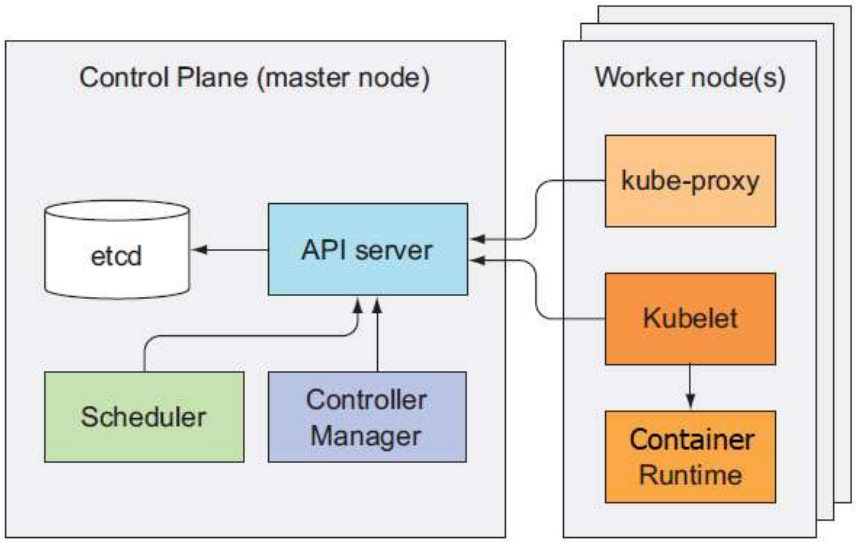
9 week

Authentication and User Management & Authorization & Admission Control

Kubernetes internals – Control Plane

- master node = Control Plane




```
❯ kubectl get componentstatuses
Warning: v1 ComponentStatus is deprecated in v1.19+
NAME                 STATUS      MESSAGE                                                                                       ERROR
controller-manager   Unhealthy   Get "http://127.0.0.1:10252/healthz": dial tcp 127.0.0.1:10252: connect: connection refused
scheduler            Healthy     ok
etcd-0               Healthy     {"health":"true"}
```

※ 참고 : https://livebook.manning.com/book/kubernetes-in-action/chapter-11/34

[별첨] ComponentStatus – 오류 해결

- master node에서 설정 파일 수정 필요

```
❯ sudo nano /etc/kubernetes/manifests/kube-controller-manager.yaml

❯ sudo nano /etc/kubernetes/manifests/kube-scheduler.yaml
```

```
...
 #- --port=0          2개 파일 모두 '- --port=0' 해당 라인 주석 처리
...
```

```
❯ sudo systemctl restart kubelet.service
```

- master node에서 해봐도 좋고, 원격으로 접근해서 해도 좋고 ...

```
❯ kubectl get componentstatuses
Warning: v1 ComponentStatus is deprecated in v1.19+
NAME                 STATUS    MESSAGE             ERROR
controller-manager   Healthy   ok
scheduler            Healthy   ok
etcd-0               Healthy   {"health":"true"}
```

Kubernetes internals – API Server

- **Authentication plugin** : 클라이언트 인증

  . 클라이언트의 사용자 이름, 사용자 ID, 속해 있는 그룹 정보 추출

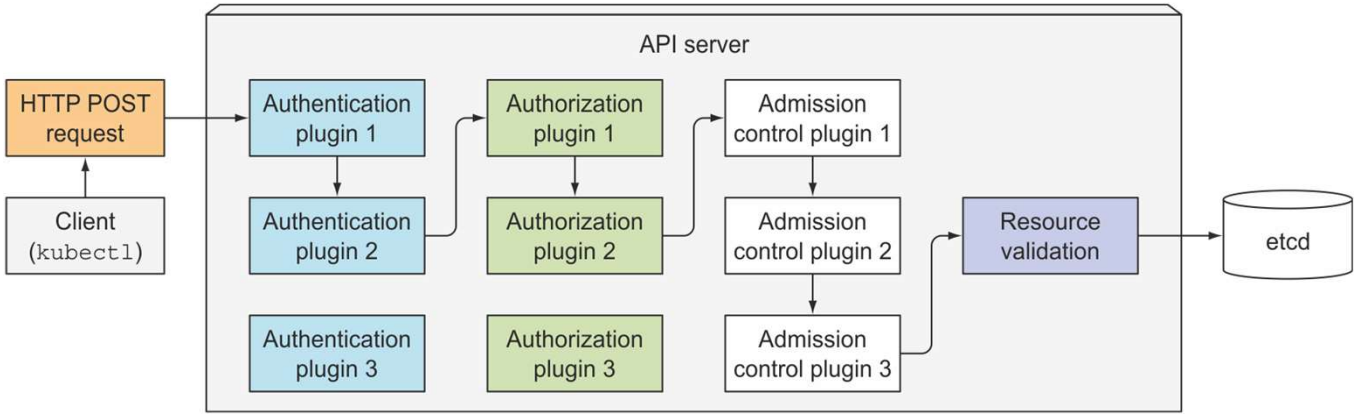
- **Authorization plugin** : 클라이언트 인가

  . 요청한 작업이 요청한 리소스를 대상으로 수행할 수 있는지 판별

- **Admission control plugin** : 요청된 리소스 확인 및 수정

  . 리소스 생성/수정/삭제 요청인 경우에만 수행 (리소스 정의에서 누락된 필드 초기화/재정의 等)

  . ex) AlwaysPullImages, ServiceAccount, NamespaceLifecycle ...

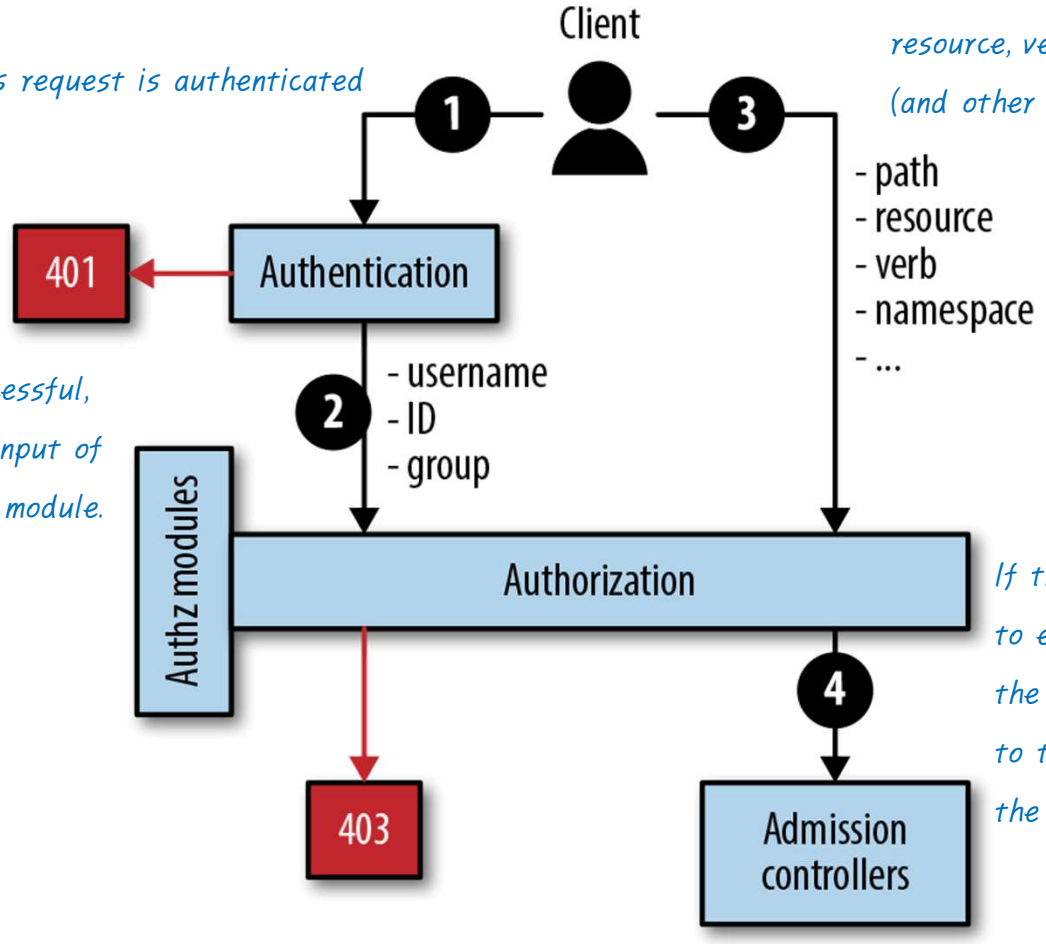


※ 참고 : https://livebook.manning.com/book/kubernetes-in-action/chapter-11/93

Authorization concepts

The client's request is authenticated

The second input to the authorization module is a vector containing the request path, resource, verb, and namespace (and other secondary attributes)



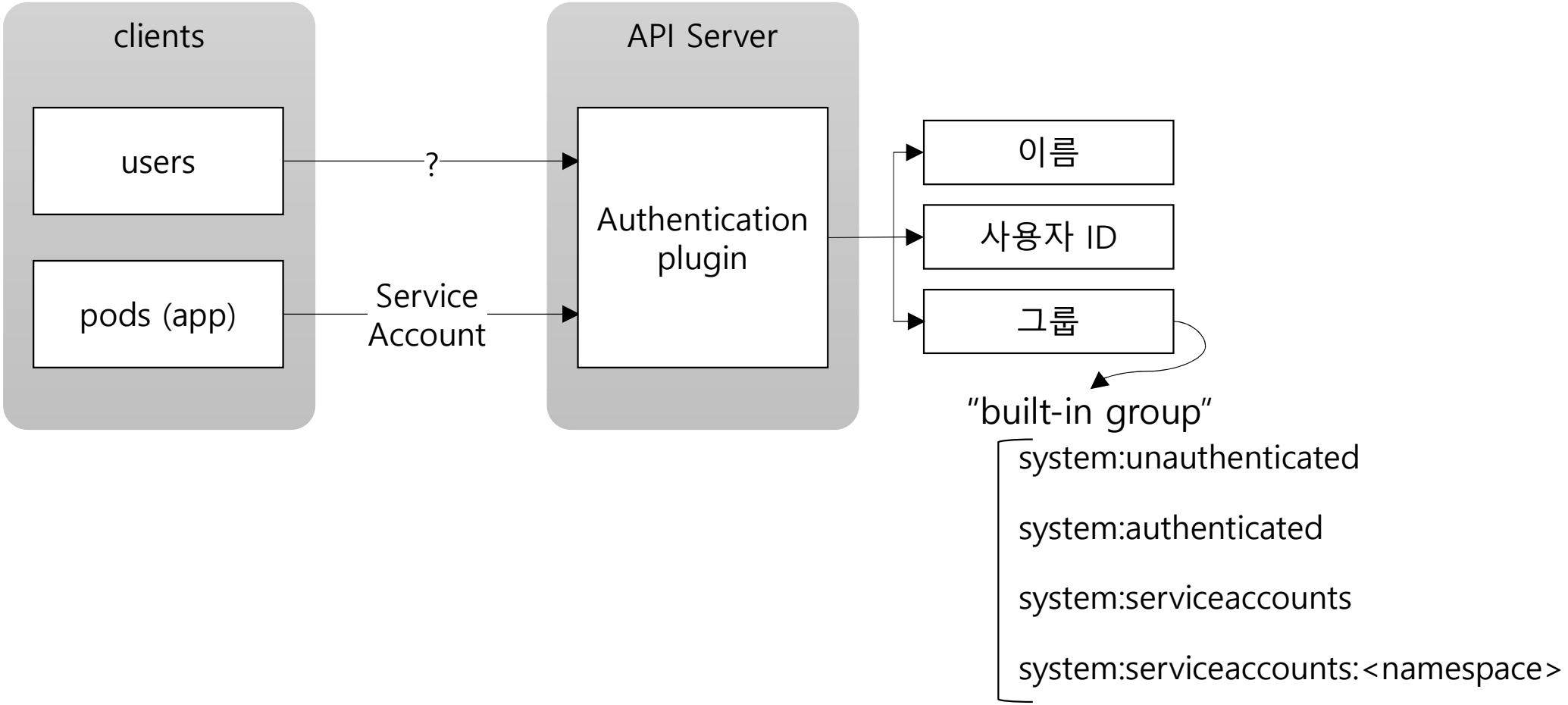
If the authentication was successful, the credentials are taken as one input of the authorization module.

If the user or application is permitted to execute a certain action on a certain resource, the request is passed on further to the next component in the chain, the admission controller.

※ 참고 : https://livebook.manning.com/book/kubernetes-in-action/chapter-12/114

Authentication – Users & Groups

- master node = Control Plane



Create ServiceAccount

- Why? Secret!

```
❯ kubectl create serviceaccount foo
serviceaccount/foo created

❯ kubectl describe serviceaccounts foo
Name:                foo
Namespace:           default
Labels:              <none>
Annotations:         <none>
Image pull secrets:  <none>
Mountable secrets:   foo-token-f6ld5
Tokens:              foo-token-f6ld5
Events:              <none>

❯ kubectl get serviceaccounts
NAME      SECRETS   AGE
default   1         45d
foo       1         32m
```

이 SA를 사용하는 Pod의 기본값으로 설정됨

```
❯ kubectl describe secrets foo-token-f6ld5
Name:         foo-token-f6ld5
Namespace:    default
Labels:       <none>
Annotations:  kubernetes.io/service-account.name: foo
              kubernetes.io/service-account.uid: 5d0a4ef2-2423-4a9a-96ed-
ba2c203cf0b8

Type:  kubernetes.io/service-account-token

Data
====
ca.crt:     1066 bytes
namespace:  7 bytes
token:
eyJhbGciOiJSUzI1NiIsImtpZCI6Img4LVYtQ20wRVhYOVNXak5tMENpUnRueHlGU05CLUJVczZhY...
```

ServiceAccount – ImagePullSecrets 1/2

- private image를 사용하려면 별도 secret 설정 필요

09-pull-fail-pod.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: private-nginx
  labels:
    app: simple

spec:
  containers:
  - name: simple
    image: whatwant/simple-nginx:v0.1
    imagePullPolicy: Always
```

private image 지정

```
❯ kubectl create -f 09-pull-fail-pod.yaml
pod/private-nginx created

❯ kubectl get pods
NAME            READY   STATUS         RESTARTS   AGE
private-nginx   0/1     ErrImagePull   0          39s

❯ kubectl delete pods private-nginx
pod "private-nginx" deleted
```

권한이 없기에 Pod 생성 실패

ServiceAccount – ImagePullSecrets 2/2

- DockerHub 인증을 위한 Secret 생성 후 ServiceAccount에서 지정

```
❯ kubectl create secret docker-registry <secret-name> --docker-server=<your-registry-server> \
              --docker-username=<your-name> --docker-password=<your-password> --docker-email=<your-email>
```

```
❯ kubectl create secret docker-registry docker-credential --docker-username=whatwant --docker-password='xxx' --docker-email='whatwant@gmail.com'
```

09-serviceaccount.yaml

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: my-service-account

imagePullSecrets:
- name: dockerhub-credential
```

```
❯ kubectl create -f 09-serviceaccount.yaml
serviceaccount/my-service-account created

❯ kubectl get serviceaccounts
NAME                 SECRETS   AGE
default              1         45d
my-service-account   1         12s
```

09-pull-success-pod.yaml

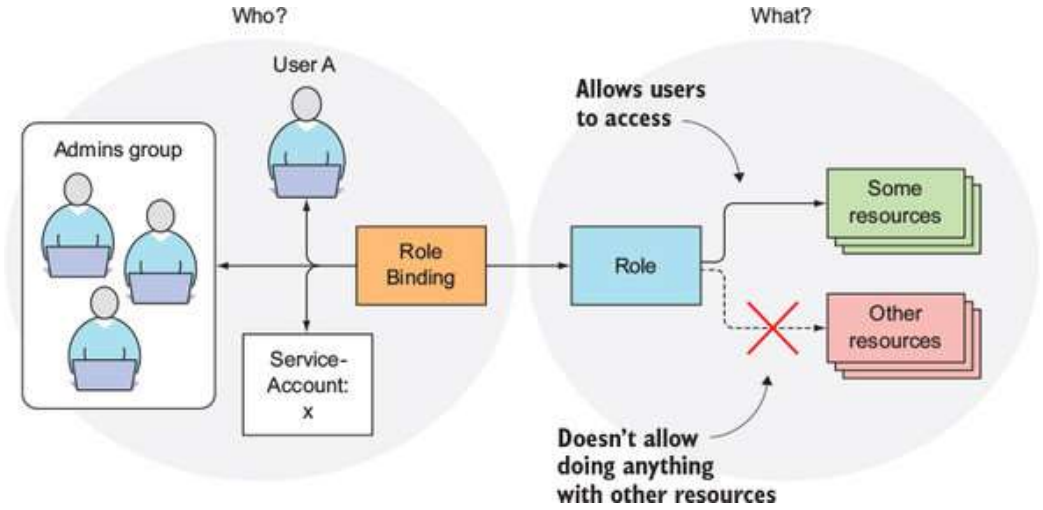
```
apiVersion: v1
kind: Pod
metadata:
  name: private-nginx
  labels:
    app: simple

spec:
  serviceAccountName: my-service-account
  containers:
  - name: simple
    image: whatwant/simple-nginx:v0.1
    imagePullPolicy: Always
```

```
❯ kubectl get pods
NAME            READY   STATUS    RESTARTS   AGE
private-nginx   1/1     Running   0          18s
```

RBAC (Role-Based Access Control, 역할 기반 접근 제어)

- Role / ClusterRole : 리소스에 수행할 수 있는 동사 지정
- RoleBinding / ClusterRoleBinding : 특정 사용자, 그룹, ServiceAccount 바인딩



※ 참고 : https://livebook.manning.com/book/kubernetes-in-action/chapter-12/114

RBAC enable/disable

- 보안을 위해 최근 kubernetes에서는 RBAC이 활성화 되어 있다
- 공부를 위해 앞에서 이를 비활성화 시켰었다.

```
❯ kubectl create clusterrolebinding permissive-binding --clusterrole=cluster-admin --group=system:serviceaccounts
clusterrolebinding.rbac.authorization.k8s.io/permissive-binding created
```

역할 기반 액세스 제어(RBAC) 비활성화

- 이제는 RBAC 공부를 위해 다시 활성화 하자

```
❯ kubectl delete clusterrolebindings permissive-binding
clusterrolebinding.rbac.authorization.k8s.io "permissive-binding" deleted
```

기본 정보 확인

- RBAC을 테스트 하기위해 기본 정보 확인을 해보자

```
❯ kubectl config view
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data: DATA+OMITTED
    server: https://192.168.100.111:6443
  name: cluster.local
contexts:
- context:
    cluster: cluster.local
    user: kubernetes-admin
  name: kubernetes-admin@cluster.local
current-context: kubernetes-admin@cluster.local
kind: Config
preferences: {}
users:
- name: kubernetes-admin
  user:
    client-certificate-data: REDACTED
    client-key-data: REDACTED

❯ APISERVER=https://192.168.100.111:6443
```

```
❯ kubectl get secrets
NAME                  TYPE                                  DATA   AGE
default-token-4b9h2   kubernetes.io/service-account-token   3      45d
```

decode 해서 사용해야 함

```
❯ kubectl get secrets default-token-4b9h2 -o jsonpath='{$.data.token}' | base64 --decode
eyJhbGciOiJSUzI1NiIsImtpZCI6Img4LVYtQ20wRVhYOVNXak5tMENpUnRueHlGU05CLUJVczZhYXJwX3pRZmMifQ.eyJ ...

❯ TOKEN=eyJhbGciOiJSUzI1NiIsImtpZCI6Img4LVYtQ20wRVhYOVNXak5tMENpUnRueHlGU05CLUJVczZhYXJwX3p ...
```

마지막 %는 제외

```
❯ curl -D - --insecure --header "Authorization: Bearer $TOKEN" $APISERVER/api/v1
HTTP/2 200
cache-control: no-cache, private
content-type: application/json
x-kubernetes-pf-flowschema-uid: b390439f-a87f-4657-aa9c-593d3200192d
x-kubernetes-pf-prioritylevel-uid: 9777a9d3-3be9-4453-803c-2eeb5c194865
date: Fri, 18 Jun 2021 22:34:20 GMT

{
  "kind": "APIResourceList",
  "groupVersion": "v1",

...
```

RBAC 확인

```
❯ curl -D - --insecure --header "Authorization: Bearer $TOKEN" $APISERVER/api/v1/pods
HTTP/2 403
cache-control: no-cache, private
content-type: application/json
x-content-type-options: nosniff
x-kubernetes-pf-flowschema-uid: b390439f-a87f-4657-aa9c-593d3200192d
x-kubernetes-pf-prioritylevel-uid: 9777a9d3-3be9-4453-803c-2eeb5c194865
content-length: 323
date: Fri, 18 Jun 2021 22:43:14 GMT

{
  "kind": "Status",
  "apiVersion": "v1",
  "metadata": {

  },
  "status": "Failure",
  "message": "pods is forbidden: User \"system:serviceaccount:default:default\" cannot list resource \"pods\" in API group \"\" at the cluster scope",
  "reason": "Forbidden",
  "details": {
    "kind": "pods"
  },
  "code": 403
}%
```

pods 목록을 확인하는 것은 실패

kubectl get pods 사용은 가능한데, api 통신만 안된다

Kube-dns Role – 1/2

- system:kube-dns 권한으로 해보자

```
❯ kubectl describe clusterroles system:kube-dns
Name:         system:kube-dns
Labels:       kubernetes.io/bootstrapping=rbac-defaults
Annotations:  rbac.authorization.kubernetes.io/autoupdate: true
PolicyRule:
  Resources  Non-Resource URLs  Resource Names  Verbs
  ---------  -----------------  --------------  -----
  endpoints  []                 []              [list watch]
  services   []                 []              [list watch]

❯ kubectl get secrets --namespace kube-system
NAME                                          TYPE                                  DATA   AGE
attachdetach-controller-token-ww7w5           kubernetes.io/service-account-token   3      45d
bootstrap-signer-token-b229f                  kubernetes.io/service-account-token   3      45d
calico-kube-controllers-token-76g72           kubernetes.io/service-account-token   3      45d
calico-node-token-wv5hd                       kubernetes.io/service-account-token   3      45d
certificate-controller-token-dx6zg            kubernetes.io/service-account-token   3      45d
clusterrole-aggregation-controller-token-kt4l6   kubernetes.io/service-account-token   3      45d
coredns-token-vrvps                           kubernetes.io/service-account-token   3      45d
...

❯ kubectl get secrets --namespace kube-system coredns-token-vrvps -o jsonpath='{$.data.token}' ¦ base64 --decode
eyJhbGciOiJSUzI1NiIsImtpZCI6Img4LVYtQ20wRVhYOVNXak5tMENpUnRueHlGU05CLUJVczZhYXJwX3pRZmMifQ.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ ...
```

Kube-dns Role – 2/2

- system:kube-dns 권한으로 해보자

```
❯ TOKEN=eyJhbGciOiJSUzI1NiIsImtpZCI6Img4LVYtQ20wRVhYOVNXak5tMENpUnRueHlGU05CLUJVczZhYXJwX3p ...

❯ curl -D - --insecure --header "Authorization: Bearer $TOKEN" $APISERVER/api/v1/pods
HTTP/2 200
cache-control: no-cache, private
content-type: application/json
x-kubernetes-pf-flowschema-uid: 9429da0e-417a-4055-bd63-aa667a08a4ec
x-kubernetes-pf-prioritylevel-uid: 168d2ac4-af9e-4fb5-adfc-41354a0feab8
date: Fri, 18 Jun 2021 23:11:53 GMT

{
  "kind": "PodList",
  "apiVersion": "v1",
  "metadata": {
    "resourceVersion": "1405319"
 },
  "items": [
    {
      "metadata": {
        "name": "node-web-78f578d65c-dk6dp",
        "generateName": "node-web-78f578d65c-",
---
```