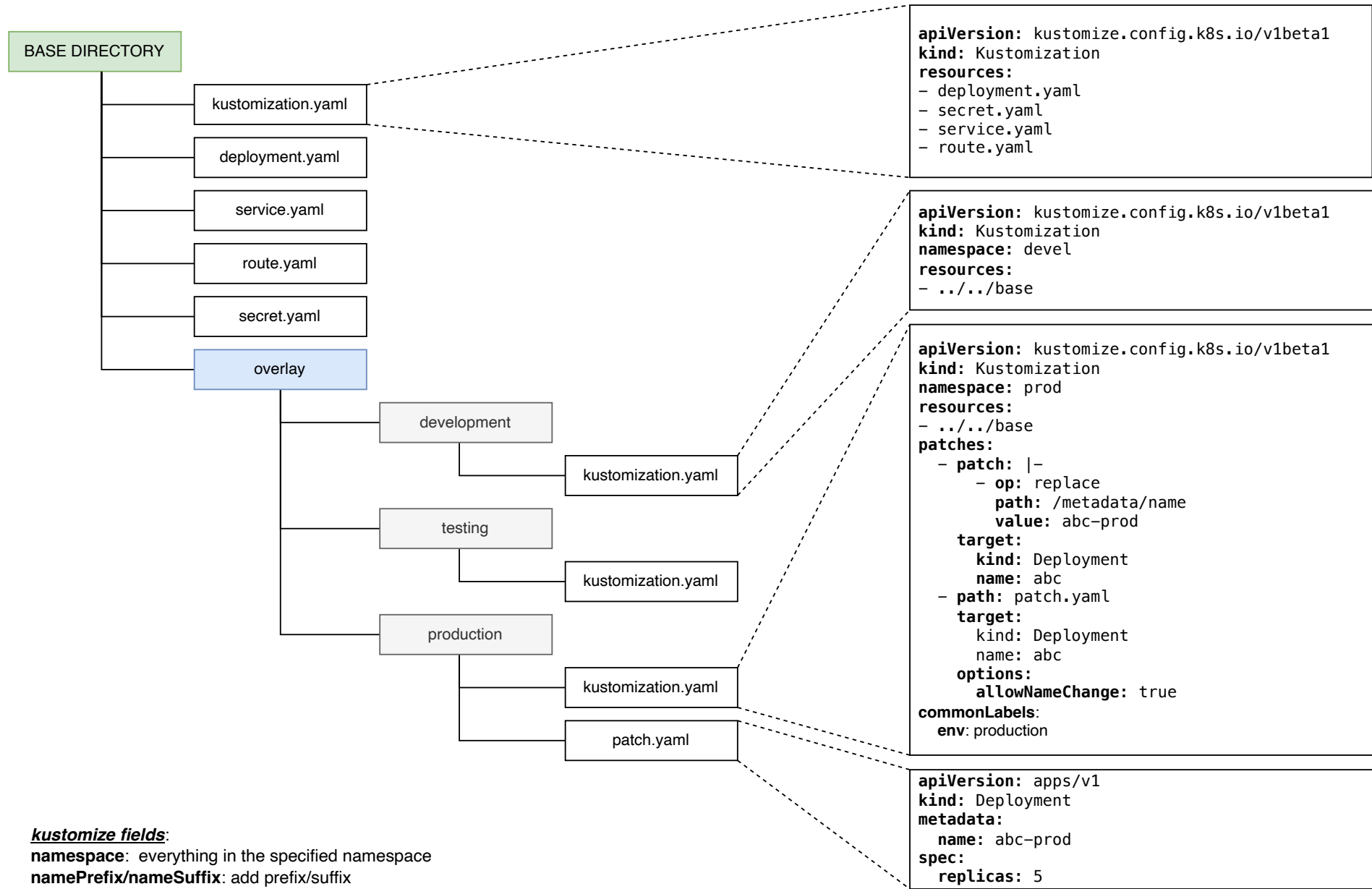


# Kustomize




## **kustomize fields:**

**namespace:** everything in the specified namespace


**namePrefix/nameSuffix:** add prefix/suffix

**commonLabels/commonAnnotations:** add Label/Annotations

 Base Directory

 Overlay Directory

 Directory

 File

## Commands

### View manifests without applying

```
kubectl kustomize overlay/development
```

### Apply/create resources

```
kubectl apply -k overlay/production
```

### Delete resources

```
kubectl delete -k overlay/production
```

### ConfigMap/Secret Generator

- generate CM/Secrets
- hash changes
- deployments can only detect CM *name* change

#### kustomize.yaml

```
kind: Kustomization
configMapGenerator:
- name: hello-app-configmap
  literals:
  - msg="Welcome!"
  - enable="true"

- name: cm-prop
  files:
  - abc.properties

- name: cm-env
  envs:
  - abc.env

secretGenerator:
- name: secret-literals
  literals:
  - ABC_USER=albert
  - ABC_PASS=einstein

- name: secret-file
  files:
  - abc.env

- name: secret-env
  envs:
  - abc.env

generatorOptions:
  disableNameSuffixHash: true
  labels:
    MYLABEL: hahaha
  annotations:
    MYNOTES: testing123
```

#### pod.yaml

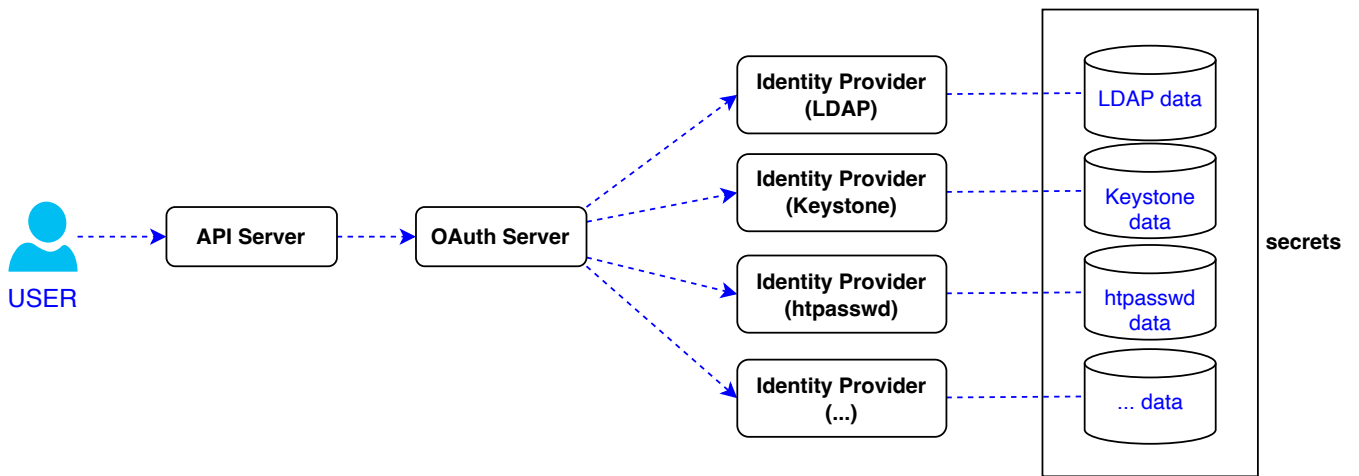
```
apiVersion: v1
kind: Pod
metadata:
  name: myPod
spec:
  containers:
  - name: hello
    image: ...
    env:
    - name: ABC_USER
      valueFrom:
        configMapKeyRef:
          name: cm-env
          key: user
```

#### abc.env

```
user=albert
pass=einstein
```

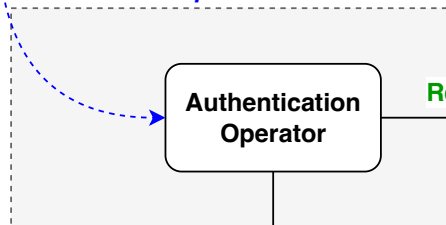
#### abc.properties

```
color=blue
language=english
```

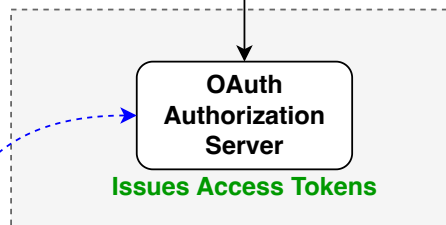


```
$ oc get co | grep auth
$ oc get pods -n openshift-authentication-operator
```

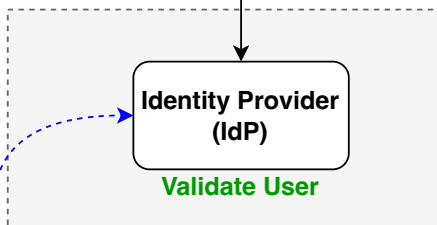
Project: *openshift-authentication-operator*



Manages/Configures

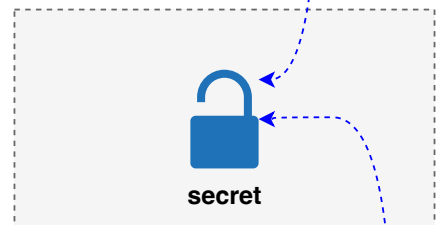


Project: *openshift-oauth-apiserver*



Project: *openshift-authentication*

```
$ oc get oauth cluster -o yaml
apiVersion: config.openshift.io/v1
kind: OAuth
metadata:
  name: cluster
  ...
spec:
  identityProviders:
  - htpasswd:
    fileName:
      name: htpasswd-secret
    mappingMethod: claim
    name: htpasswd_provider
    type: HTPasswd
```



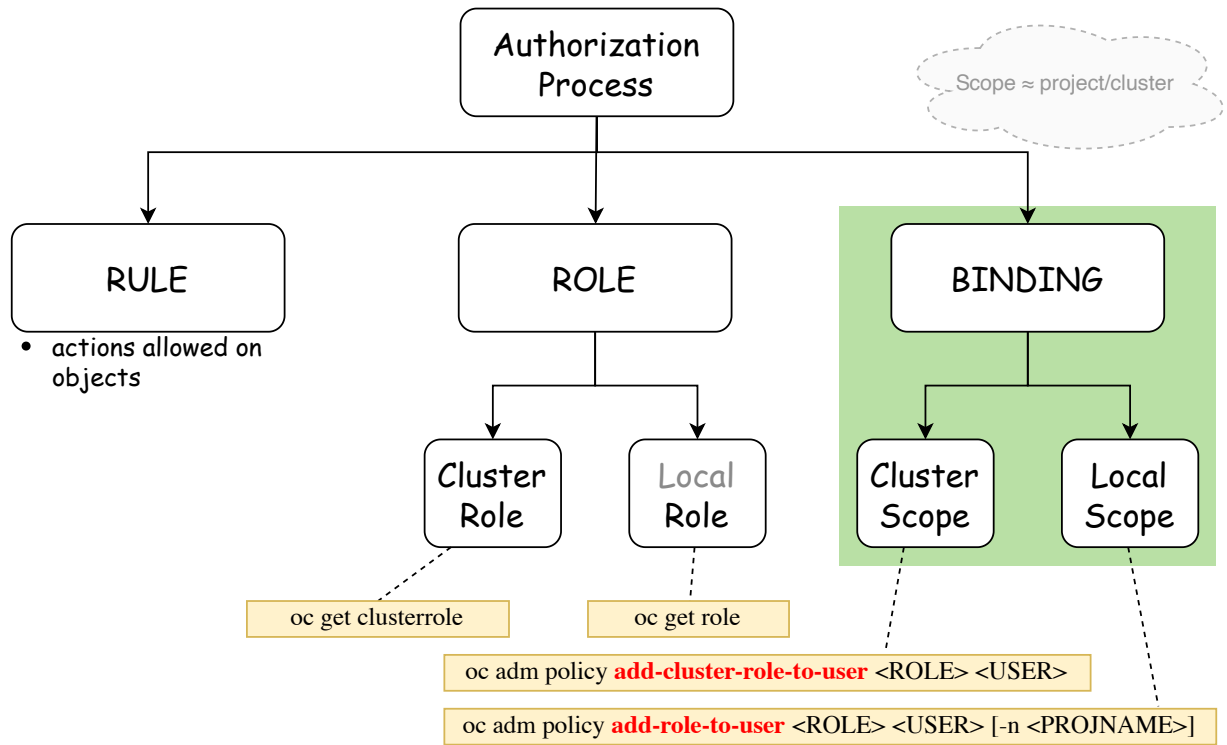
Project: *openshift-config*

```
$ oc get pods -n openshift-oauth-apiserver
```

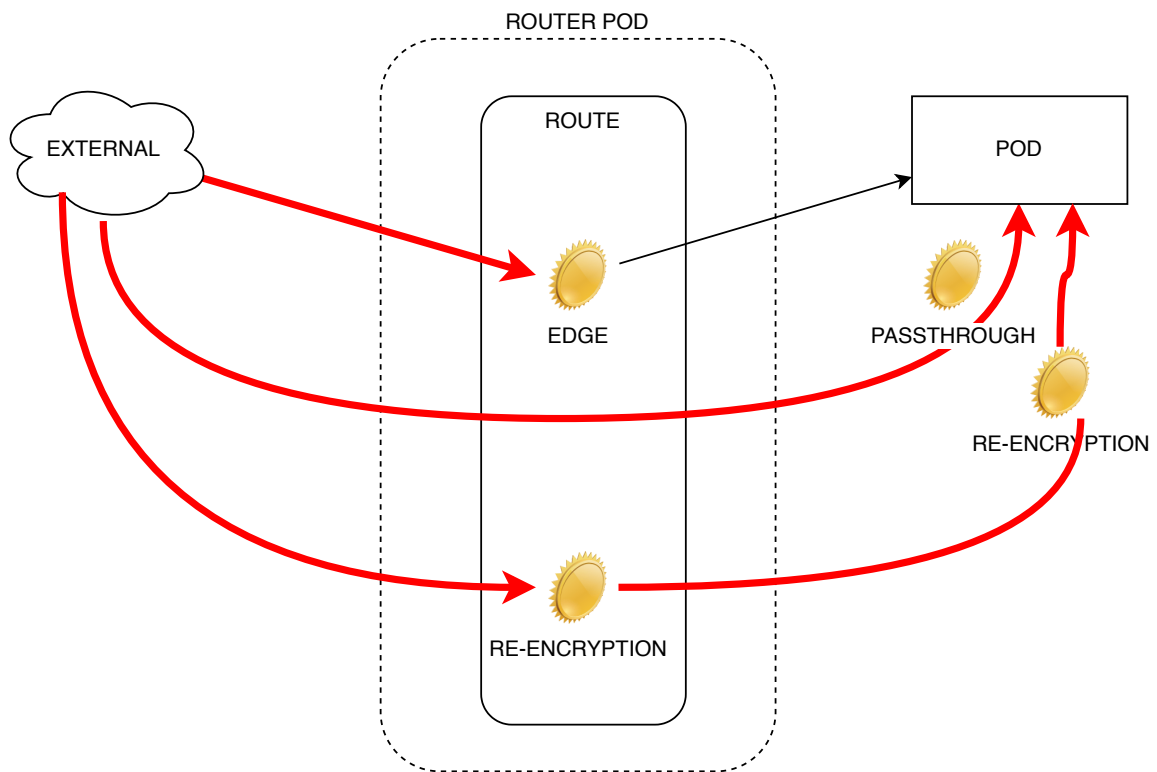
```
$ oc get pods -n openshift-authentication
```

```
$ htpasswd -cBb ./myusers <USER> <PASSWORD>
$ oc create secret generic htpasswd-secret --from-file htpasswd=./myhtpasswd -n openshift-config
```

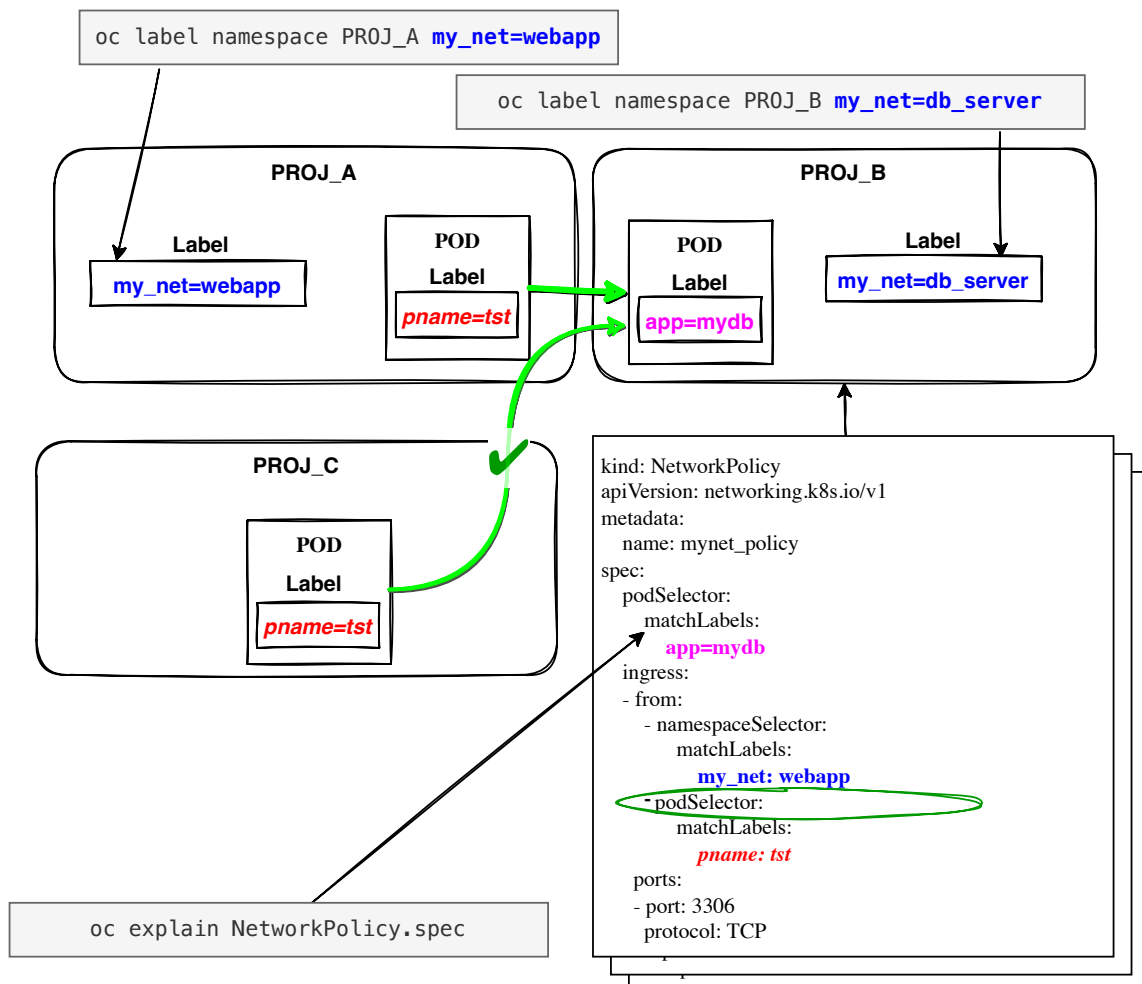
# Role Based Access Control (RBAC)



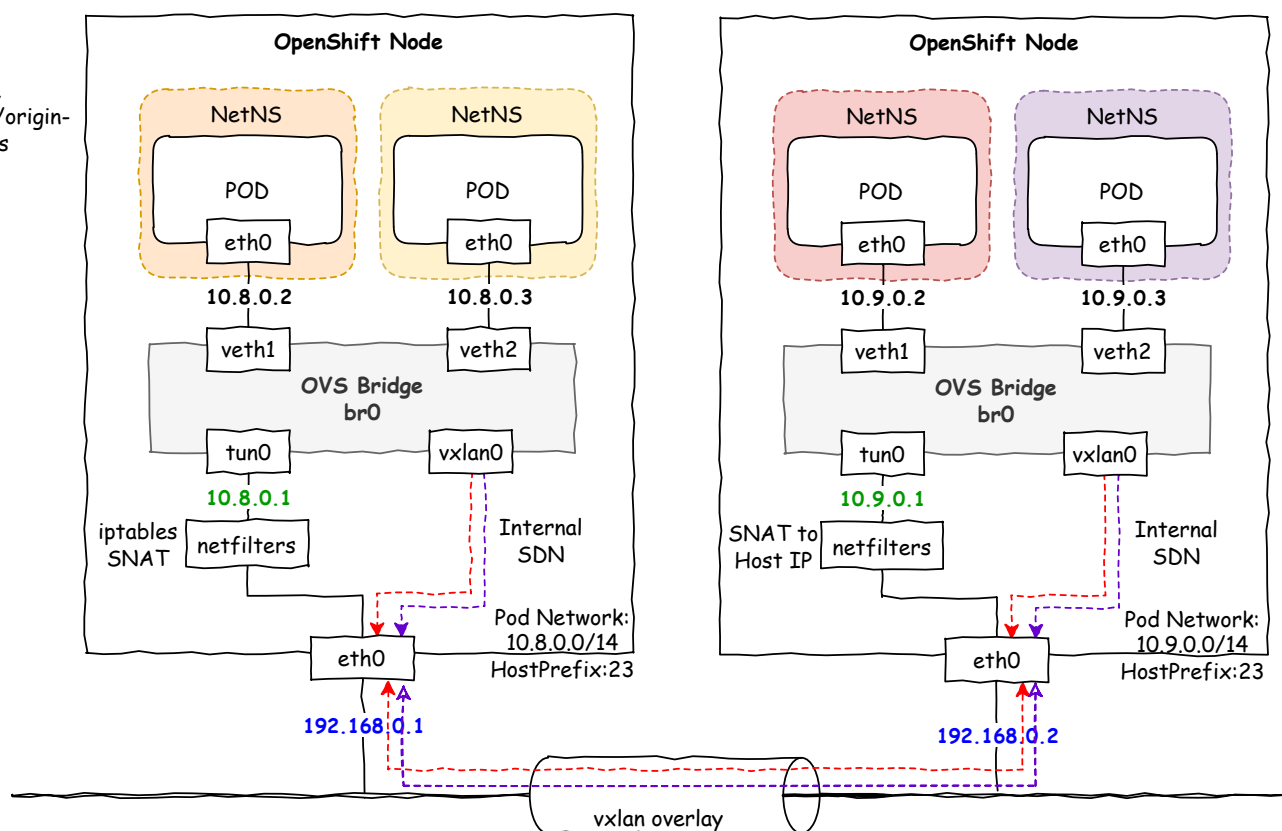
## TLS Termination Types



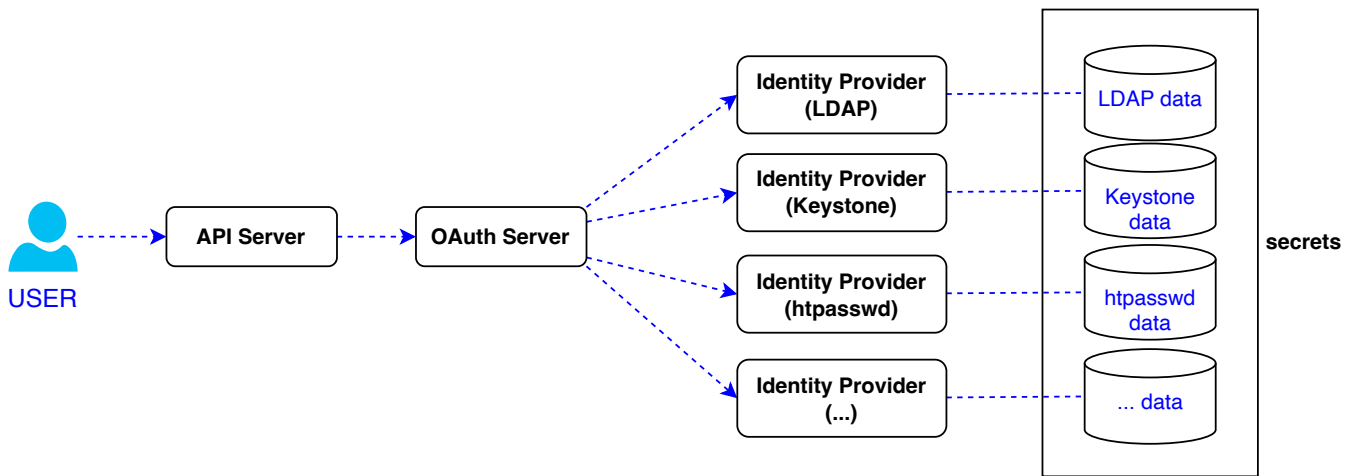
## Network Policy



use  
openshift/origin-  
tools

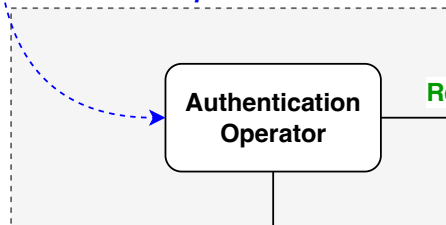


nnnnnnnn . nnnnnnxx . xxxxxxxx . xxxxxxxx

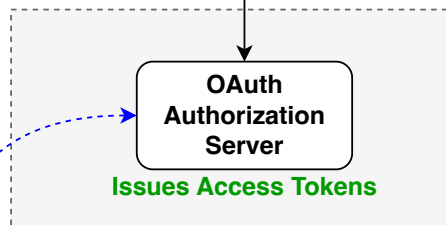


```
$ oc get co | grep auth
$ oc get pods -n openshift-authentication-operator
```

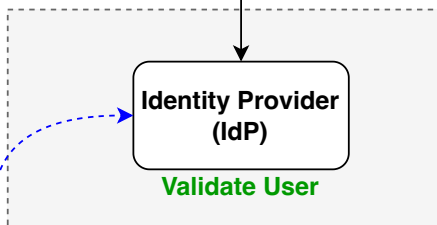
Project: *openshift-authentication-operator*



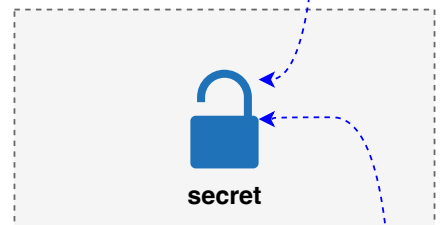
Manages/Configures



Project: *openshift-oauth-apiserver*



Project: *openshift-authentication*



Project: *openshift-config*

```
$ oc get oauth cluster -o yaml
apiVersion: config.openshift.io/v1
kind: OAuth
metadata:
  name: cluster
  ...
spec:
  identityProviders:
  - htpasswd:
      fileName:
        name: htpasswd-secret
      mappingMethod: claim
      name: htpasswd_provider
      type: HTPasswd
```

```
$ oc get pods -n openshift-oauth-apiserver
```

```
$ oc get pods -n openshift-authentication
```

```
$ htpasswd -cBb ./myusers <USER> <PASSWORD>
$ oc create secret generic htpasswd-secret --from-file htpasswd=./myhtpasswd -n openshift-config
```

# Role Based Access Control (RBAC)

