우선 OPA를 설치해주기 전에 환경 설정을 해주자.

|  |
| --- |
| $ kubectl create ns opa |

이제 인증서를 생성해주자. 왜냐하면 OPA와 Kubernetes 같의 통신은 TLS로 보호 되어야하기 때문에 CA와 OPA용 Certificate/Key를 생성해주자.

|  |
| --- |
| $ openssl genrsa -out ca.key 2048  $ openssl req -x509 -new -nodes -sha256 -key ca.key -days 100000 -out ca.crt -subj "/CN=admission\_ca" |

|  |
| --- |
| $ cat >server.conf <<EOF  [ req ]  prompt = no  req\_extensions = v3\_ext  distinguished\_name = dn  [ dn ]  CN = opa.opa.svc  [ v3\_ext ]  basicConstraints = CA:FALSE  keyUsage = nonRepudiation, digitalSignature, keyEncipherment  extendedKeyUsage = clientAuth, serverAuth  subjectAltName = DNS:opa.opa.svc,DNS:opa.opa.svc.cluster,DNS:opa.opa.svc.cluster.local  EOF |

|  |
| --- |
| $ openssl genrsa -out server.key 2048  $ openssl req -new -key server.key -sha256 -out server.csr -extensions v3\_ext -config server.conf  $ openssl x509 -req -in server.csr -sha256 -CA ca.crt -CAkey ca.key -CAcreateserial -out server.crt -days 100000 -extensions v3\_ext -extfile server.conf |

그리고 TLS 정보를 Secret에 저장해주자.

|  |
| --- |
| $ kubectl create secret tls opa-server --cert=server.crt --key=server.key --namespace opa |

그리고 Deny Policy를 작성해주자.

|  |
| --- |
| $ vim main.rego |
| package system  main = {  "apiVersion": "admission.k8s.io/v1beta1",  "kind": "AdmissionReview",  "response": response  }  default response = { "allowed": true }  response = {  "allowed": false,  "status": {  "reason": reason  }  } {  reason = concat(", ", deny)  reason != ""  }  deny[msg] {  input.request.operation == "CREATE"  input.request.kind.kind == "Pod"  msg := sprintf("Pod is not allowed to be created by %s", [input.request.userInfo.username])  } |

이제 OPA를 배포해보자

|  |
| --- |
| $ vim admission\_controller.yaml |
| # Grant OPA/kube-mgmt read-only access to resources. This lets kube-mgmt  # replicate resources into OPA so they can be used in policies.  kind: ClusterRoleBinding  apiVersion: rbac.authorization.k8s.io/v1  metadata:  name: opa-viewer  roleRef:  kind: ClusterRole  name: view  apiGroup: rbac.authorization.k8s.io  subjects:  - kind: Group  name: system:serviceaccounts:opa  apiGroup: rbac.authorization.k8s.io  ---  # Define role for OPA/kube-mgmt to update configmaps with policy status.  kind: Role  apiVersion: rbac.authorization.k8s.io/v1  metadata:  namespace: opa  name: configmap-modifier  rules:  - apiGroups: [""]  resources: ["configmaps"]  verbs: ["update", "patch"]  ---  # Grant OPA/kube-mgmt role defined above.  kind: RoleBinding  apiVersion: rbac.authorization.k8s.io/v1  metadata:  namespace: opa  name: opa-configmap-modifier  roleRef:  kind: Role  name: configmap-modifier  apiGroup: rbac.authorization.k8s.io  subjects:  - kind: Group  name: system:serviceaccounts:opa  apiGroup: rbac.authorization.k8s.io  ---  kind: Service  apiVersion: v1  metadata:  name: opa  namespace: opa  spec:  selector:  app: opa  ports:  - name: https  protocol: TCP  port: 443  targetPort: 8443  ---  apiVersion: apps/v1  kind: Deployment  metadata:  labels:  app: opa  namespace: opa  name: opa  spec:  replicas: 1  selector:  matchLabels:  app: opa  template:  metadata:  labels:  app: opa  name: opa  spec:  containers:  # WARNING: OPA is NOT running with an authorization policy configured. This  # means that clients can read and write policies in OPA. If you are  # deploying OPA in an insecure environment, be sure to configure  # authentication and authorization on the daemon. See the Security page for  # details: https://www.openpolicyagent.org/docs/security.html.  - name: opa  image: openpolicyagent/opa:0.49.0-rootless  args:  - "run"  - "--server"  - "--tls-cert-file=/certs/tls.crt"  - "--tls-private-key-file=/certs/tls.key"  - "--addr=0.0.0.0:8443"  - "--addr=http://127.0.0.1:8181"  - "--set=services.default.url=http://host.minikube.internal:8888"  - "--set=bundles.default.resource=bundle.tar.gz"  - "--log-format=json-pretty"  - "--set=status.console=true"  - "--set=decision\_logs.console=true"  volumeMounts:  - readOnly: true  mountPath: /certs  name: opa-server  readinessProbe:  httpGet:  path: /health?plugins&bundle  scheme: HTTPS  port: 8443  initialDelaySeconds: 3  periodSeconds: 5  livenessProbe:  httpGet:  path: /health  scheme: HTTPS  port: 8443  initialDelaySeconds: 3  periodSeconds: 5  - name: kube-mgmt  image: openpolicyagent/kube-mgmt:2.0.1  args:  - "--replicate-cluster=v1/namespaces"  - "--replicate=networking.k8s.io/v1/ingresses"  volumes:  - name: opa-server  secret:  secretName: opa-server |
| $ kubectl apply -f admission\_controller.yaml |

이제 Webhook을 생성해주자.

|  |
| --- |
| $ cat > webhook-configuration.yaml <<EOF  kind: ValidatingWebhookConfiguration  apiVersion: admissionregistration.k8s.io/v1  metadata:  name: opa-validating-webhook  webhooks:  - name: validating-webhook.openpolicyagent.org  namespaceSelector:  matchExpressions:  - key: openpolicyagent.org/webhook  operator: NotIn  values:  - ignore  rules:  - operations: ["CREATE", "UPDATE"]  apiGroups: ["\*"]  apiVersions: ["\*"]  resources: ["\*"]  clientConfig:  caBundle: $(cat ca.crt | base64 | tr -d '\n')  service:  namespace: opa  name: opa  admissionReviewVersions: ["v1"]  sideEffects: None  EOF  $ kubectl label ns kube-system openpolicyagent.org/webhook=ignore  $ kubectl label ns opa openpolicyagent.org/webhook=ignore  $ kubectl apply -f webhook-configuration.yaml |

OPA를 설치해주자.

|  |
| --- |
| $ helm repo add openpolicyagent https://openpolicyagent.org/helm |