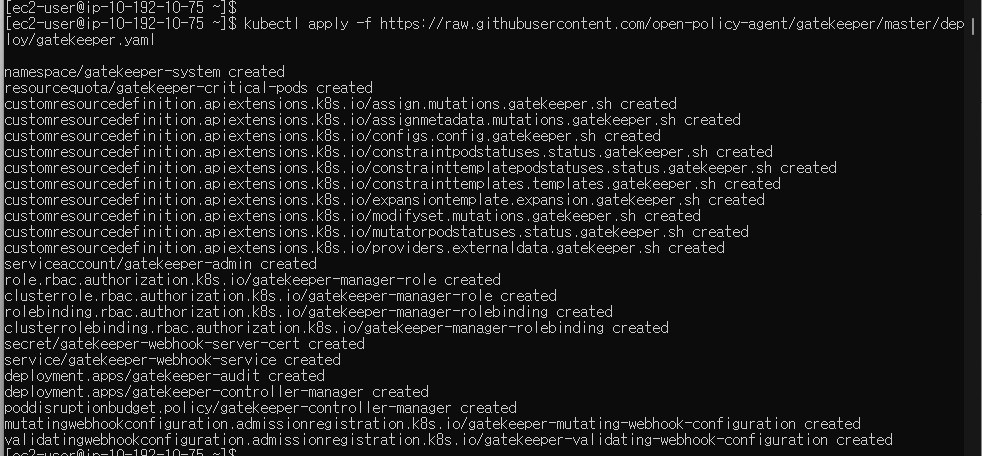
우선 게이트키퍼를 설치해준다.

|  |
| --- |
| kubectl apply -f https://raw.githubusercontent.com/open-policy-agent/gatekeeper  /master/deploy/gatekeeper.yaml |

‘

이제 템플릿을 생성해준다.

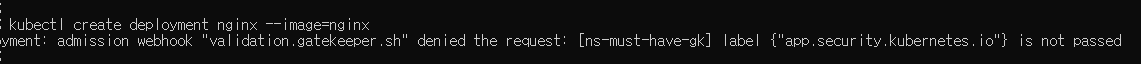
|  |
| --- |
| apiVersion: templates.gatekeeper.sh/v1  kind: ConstraintTemplate  metadata:    name: k8srequiredlabels  spec:    crd:      spec:        names:          kind: K8sRequiredLabels        validation:          # Schema for the `parameters` field          openAPIV3Schema:            type: object            properties:              labels:                type: array                items:                  type: string    targets:      - target: admission.k8s.gatekeeper.sh        rego: |          package k8srequiredlabels          violation[{"msg": msg }] {            provided := {label | input.review.object.metadata.labels[label]}            required := {label | label := input.parameters.labels[\_]}            missing := required - provided            count(missing) > 0            msg := sprintf("label %s is not passed", [missing])          } |

이제 deployment 기반의 제약조건을 하나 생성해준다.

|  |
| --- |
| apiVersion: constraints.gatekeeper.sh/v1beta1  kind: K8sRequiredLabels  metadata:  name: ns-must-have-gk  spec:  match:  kinds:  - apiGroups: ["apps"]  kinds: ["Deployment"]  parameters:  labels: ["app.security.kubernetes.io"] |

모든 조건을 적용하면 출력은 다음과 같다.

kubectl create deployment nginx --image=nginx



Deployment.yaml 파일을 생성한다.

|  |
| --- |
| apiVersion: apps/v1  kind: Deployment  metadata:  name: nginx-deployment  labels:  app.security.kubernetes.io: pass  spec:  replicas: 2  selector:  matchLabels:  app.security.kubernetes.io: pass  template:  metadata:  labels:  app.security.kubernetes.io: pass  spec:  containers:  - name: nginx  image: nginx:1.14.2  ports:  - containerPort: 80 |

Kubectl apply –f ./deployment.yaml

정상적으로 생성된 걸 볼수 있다.

