POLICY BASED CONTROL

FOR CLOUD NATIVE ENVIRONMENT WITH OPA

Docker DevTools Day 3.0

Aug 24, 2024

- Kunal Das
- Sr DevOps Engineer @ Cyncly
- HashiCorp Certified Terraform Associate
- Microsoft Certified DevOps Expert

POLICY ENFORCEMENT CHALLENGES

Traditional

Firewall

Access Control

Manual Approval

Cloud-Native

Granular checks

Dynamic Control

Automated Approval

OPA AND GATEKEEPER

Gatekeeper

OPA

Open Policy Agent (OPA)

- General-purpose policy engine
- Evaluates policies written in Rego language
- Can be used across various systems and platforms

Gatekeeper

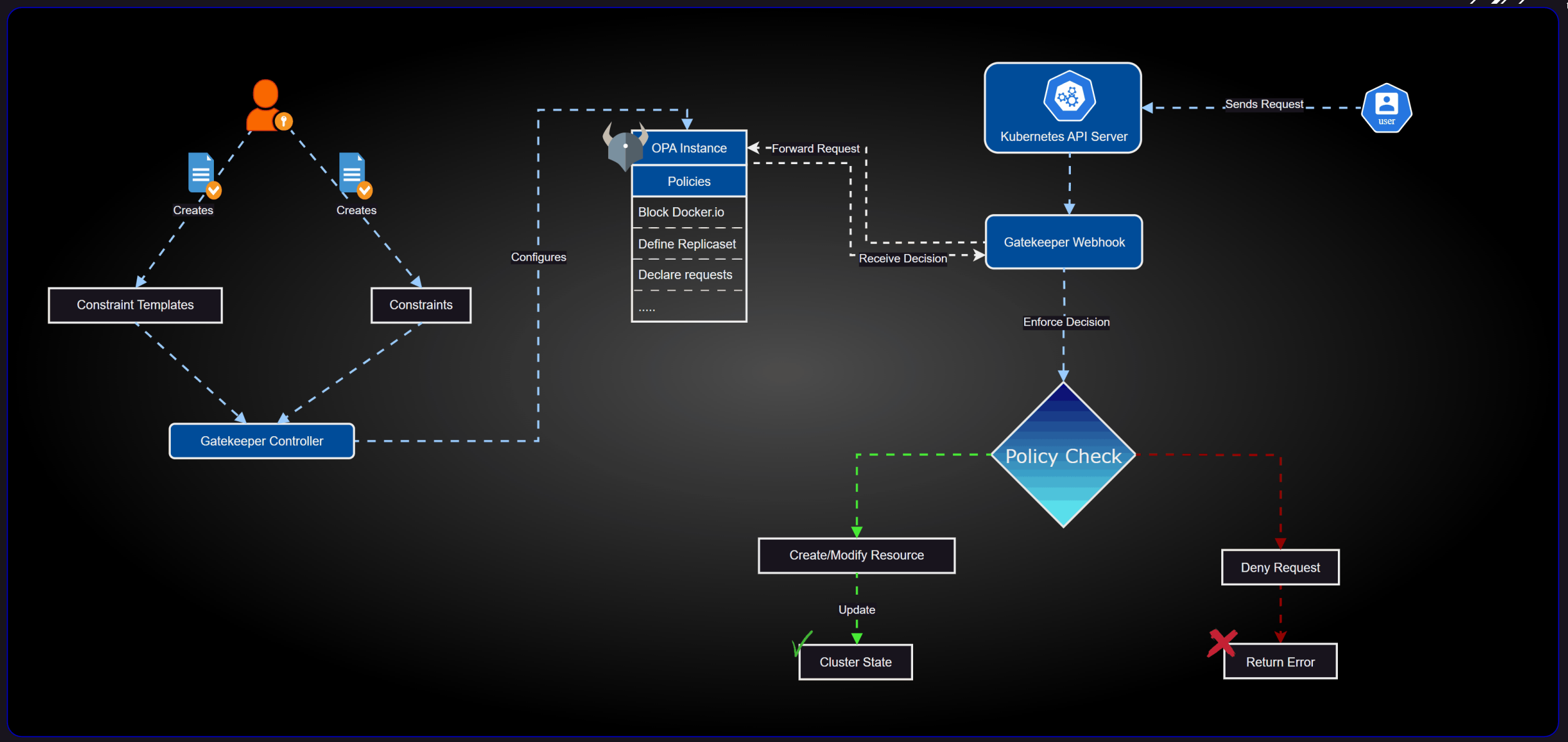
- Kubernetes-native policy management
- Implements OPA as an admission controller
- Extends Kubernetes API with custom resources

Custom Resource Definitions (CRDs)

ConstraintTemplate	•Defines the policy logic using Rego •Creates a new custom resource type
Constraint	•Instance of a ConstraintTemplate •Specifies which resources the policy applies to

```yaml
apiVersion: constraints.gatekeeper.sh/v1beta1
kind: K8sRequiredLabels
metadata:
  name: ns-must-have-gk
spec:
  match:
    kinds:
      - apiGroups: [""]
        kinds: ["Namespace"]
    excludedNamespaces:
    - kube-system
    - kube-public
    - kube-node-lease
    - default
    - gatekeeper-system
    - kured
  parameters:
    labels: ["createdby"]
```

```yaml
apiVersion: templates.gatekeeper.sh/v1beta1
kind: ConstraintTemplate
metadata:
  name: k8srequiredlabels
spec:
  crd:
    spec:
      names:
        kind: K8sRequiredLabels
      validation:
        openAPIV3Schema:
          properties:
            labels:
              type: array
              items: string
  targets:
    - target: admission.k8s.gatekeeper.sh
      rego: |
        package k8srequiredlabels
        violation[{"msg": msg, "details": {"missing_labels": missing}}] {
          provided := {label | input.review.object.metadata.labels[label]}
          required := {label | label := input.parameters.labels[_]}
          missing := required - provided
          count(missing) > 0
          msg := sprintf("\n%v is missing from your defination 😠😓", [missing])
        }
```


Creates
Creates
Constraint Templates
Constraints
Gatekeeper Controller
Configures
OPA Instance
Policies
Block Docker.io
Define Replicaset
Declare requests
.....
Forward Request
Receive Decision
Kubernetes API Server
Sends Request
user
Gatekeeper Webhook
Enforce Decision
Policy Check
Create/Modify Resource
Update
Cluster State
Deny Request
Return Error

USAGE SCENARIOS

Resource Constraints	• Enforcing CPU/memory limits on pods • Restricting storage class usage
Security Enforcement	• Ensuring pods run as non-root • Enforcing network policies
Compliance Requirements	• Mandatory labeling of resources • Restricting use of latest image tags
Cost Optimization	• Enforcing resource quotas • Limiting expensive cloud services usage
Multi-tenancy	• Namespace-based restrictions • Enforcing isolation between tenants
Best Practices	• Enforcing liveness and readiness probes • Ensuring high availability configurations

DEMO

```yaml
apiVersion: constraints.gatekeeper.sh/v1beta1
kind: K8sRequiredLabels
metadata:
  name: ns-must-have-gk
spec:
  match:
    kinds:
      - apiGroups: [""]
        kinds: ["Namespace"]
    excludedNamespaces:
    - kube-system
    - kube-public
    - kube-node-lease
    - default
    - gatekeeper-system
    - kured
  parameters:
    labels: ["createdby"]
```

1

```yaml
apiVersion: templates.gatekeeper.sh/v1beta1
kind: ConstraintTemplate
metadata:
  name: k8srequiredlabels
spec:
  crd:
    spec:
      names:
        kind: K8sRequiredLabels
      validation:
        openAPIV3Schema:
          properties:
            labels:
              type: array
              items: string
  targets:
    - target: admission.k8s.gatekeeper.sh
      rego: |
        package k8srequiredlabels
        violation[{"msg": msg, "details": {"missing_labels": missing}}] {
          provided := {label | input.review.object.metadata.labels[label]}
          required := {label | label := input.parameters.labels[_]}
          missing := required - provided
          count(missing) > 0
          msg := sprintf("\n%v is missing from your defination 👴👵", [missing])
        }
```

2

```yaml
apiVersion: v1
kind: Namespace
metadata:
  name: mynamespace
  labels:
    createdby: kunal
```

4

```yaml
apiVersion: v1
kind: Namespace
metadata:
  name: badns
```

3

CHALLENGES

Learning Curve	• Policies are written on Rego.
Performance Impact	• Potential latency in admission requests • Resource consumption in large-scale deployments
Policy Management	• Maintaining consistency across multiple clusters • Version control and change management of polices.
Integration Complexity	• Incorporating with existing CI/CD pipelines • Aligning with other security tools and practices

REFERENCES

https://github.com/open-policy-agent/gatekeeper

https://open-policy-agent.github.io/gatekeeper/website/

https://kubernetes.io/blog/2019/08/06/opa-gatekeeper-policy-and-governance-for-kubernetes/

https://www.openpolicyagent.org/

Thank You

Reach out to me in case any doubts, suggestions or feedback !!