K8S con OPA Gatekeeper

Daniel Pérez Gallo 2º ASIR

ÍNDICE

- Kubernetes
- Políticas
- RBAC
- Gatekeeper
- Rego
- Prueba práctica

¿Qué es Kubernetes?

Kubernetes es una plataforma portable y extensible de código abierto para administrar cargas de trabajo y servicios centrada en un entorno de contenedores.
Puede ser una plataforma de contenedores, microservicios y/o portable de nube.

¿Por qué usar contenedores?

- Se despliegan basados en virtualización a nivel del sistema operativo, en vez del hardware.
- Están aislados entre ellos y con el servidor anfitrión (propios sistemas de archivos, no ven los procesos de los demás, se limita el uso de recursos, etc).
- Son más fáciles de construir que una máquina virtual, ya que no están acoplados a la infraestructura y sistema de archivos del anfitrión.
- Podemos crear imágenes inmutables al momento de la compilación en vez del despliegue, esto permite tener un entorno consistente que va desde desarrollo hasta producción.
- El monitoreo y la administración son más fáciles que en máquinas virtuales.

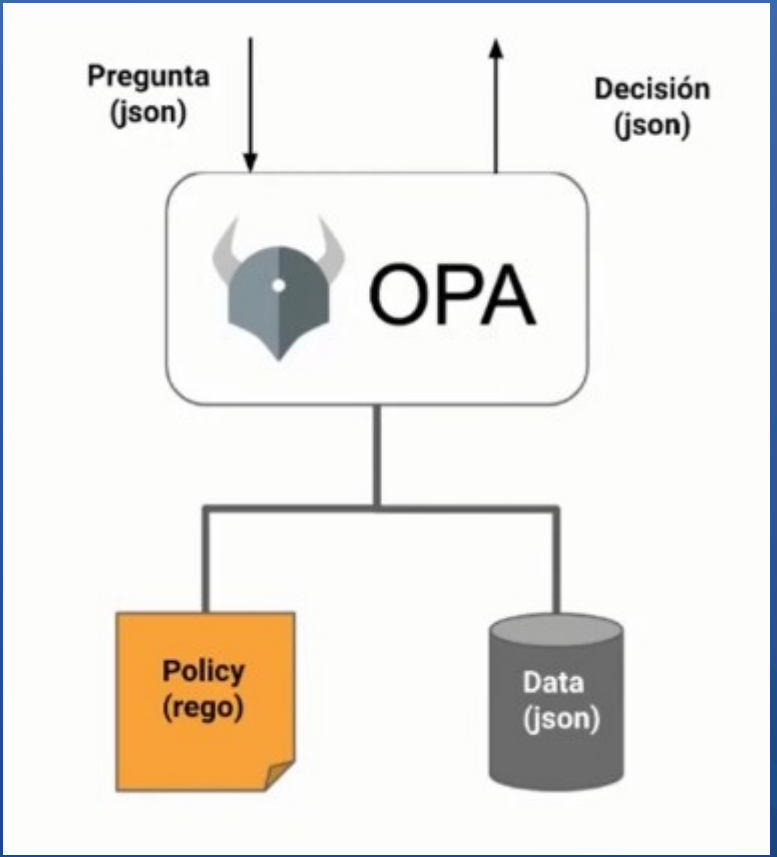
¿Qué es una política?

Una política es un conjunto de reglas que su autor declara para que un sistema cumpla con un conjunto determinado de requisitos. Estas se escriben con el lenguaje anteriormente mencionado, Rego.

```
apiVersion: templates.gatekeeper.sh/v1beta1
kind: ConstraintTemplate
metadata:
  name: k8srequiredlabels
spec:
  crd:
    spec:
      names:
        kind: K8sRequiredLabels
        listKind: K8sRequiredLabelsList
        plural: k8srequiredlabels singular: k8srequiredlabels
      validation: # Schema for the `parameters` field openAPIV3
      Schema:
      properties:
           labels:
           type: array items: string targets:
           - target: admission.k8s.gatekeeper.sh
           rego: |
             package k8srequiredlabels
             deny[{"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("you must provide labels: %v",
             [missing]) }
```

¿Qué es un gestor de políticas (OPA)?

Open Policy Agent es un motor de políticas de uso general y código abierto que permite la implementación de políticas unificadas



¿Qué es RBAC?

Es un tipo de gestión de identidades y acceso que involucra una serie de
permisos o plantillas que determinan quién (sujeto) puede realizar qué (verbo) y
en qué parte (recurso de la API de Kubernetes)

¿Qué son las funciones?

Las funciones pueden identificarse como funciones a secas o funciones del clúster.

Las funciones permiten acceder a los grupos de clústeres vinculados virtualmente que se conocen como espacios de nombres. Dichas funciones son recursos con espacios de nombres. El enlace de funciones se utiliza para consolidar los usuarios, los grupos de usuarios o los nombres de las cuentas de servicio en una sola función.

Las funciones del clúster, las cuales abarcan varios espacios de nombres, permiten acceder a clústeres completos, que son grupos de nodos de hardware individuales. Los enlaces de funciones del clúster asocian una función a cada espacio de nombres.

Funcionamiento de RBAC

La API de Kubernetes es el frontend del plano de control de este sistema, la cual comunica las interacciones con una computadora o un sistema con el fin de recuperar información o realizar una función.

El RBAC de Kubernetes recopila las solicitudes de funciones relacionadas en grupos de API, que se comunican con los servidores de API cuando se conectan ciertas funciones a los extremos de la misma.

No puede comprobar todas las etiquetas (labels) que tiene un pod, es decir, nos permite crear un pod, pero le resulta indiferente lo que haya dentro del mismo o de cualquier otro objeto.

¿Qué es Gatekeeper?

Gatekeeper es un plugin de OPA para Kubernetes.
Se caracteriza por forzar una decisión que tomamos para un entorno de Kubernetes.

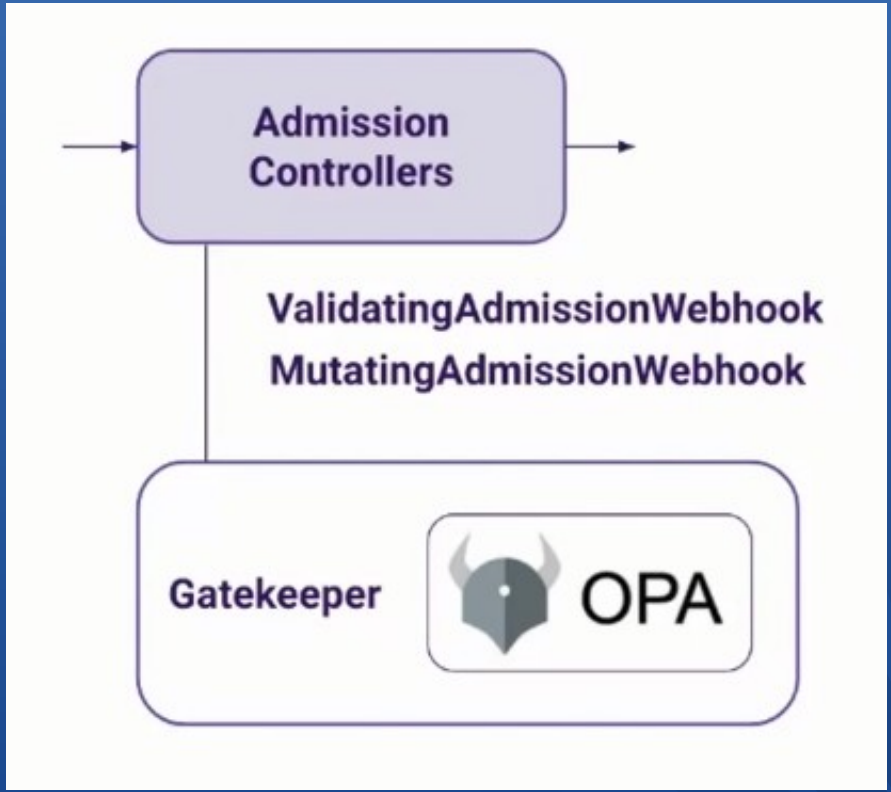
¿Cómo funciona Gatekeeper?

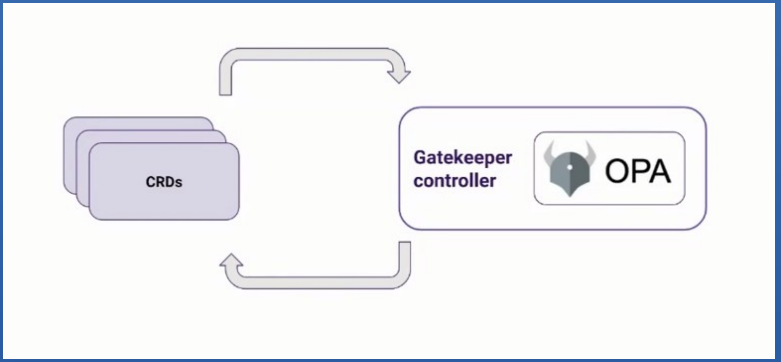
Gatekeeper utiliza un webhook mediante una API que posee Kubernetes, este verifica si las políticas se cumplen o no para cargar dicha solicitud. También nos proporciona un lenguaje llamado Rego, el cual nos permite especificar que políticas queremos aplicar.

Forma de forzar las políticas

La autenticación seríamos nosotros, mientras que la autorización sería RBAC.
Mientras que los Controles de acceso son una serie de reglas específicas que están en el binario del servidor de la API de kubernetes, nosotros podemos habilitarlos o desactivarlos cuando arrancamos el servidor de kubernetes.

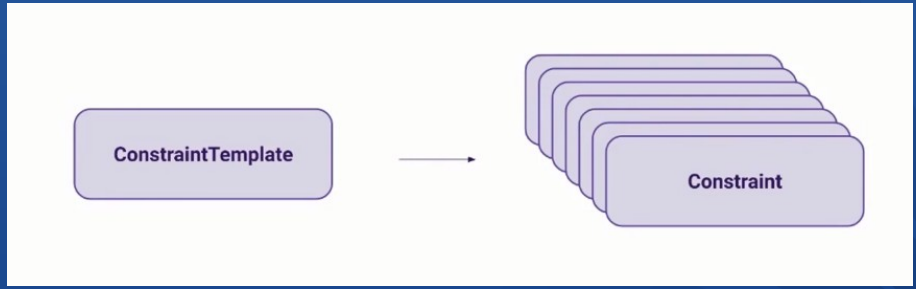
Tenemos dos binarios: ValidatingAdmissionWebhook y MutatingAdmissionWebhook Gatekeeper lo que hace es un ValidatingAdmissionWebhook que va a hacer que cuando una request está pasando por los pasos mencionados previamente, concretamente en el AdmissionControllers va a comprobar que cumple con las políticas de nuestro clúster.





Gatekeeper está completamente diseñado para kubernetes, ya que crea nuevos objetos para la API de kubernetes mediante CRDs y un controller que hace el “Reconciliation Loop” (Esto sirve para conducir el estado actual hacia el estado deseado). De los CRDs que se crean, los principales son el ConstraintTemplate y los Constraint.

En el primero es donde vamos a definir las políticas a nivel genérico usando el lenguaje Rego. A partir de dichas políticas vamos a poder hacer distintas instancias para cosas particulares de nuestro clúster.



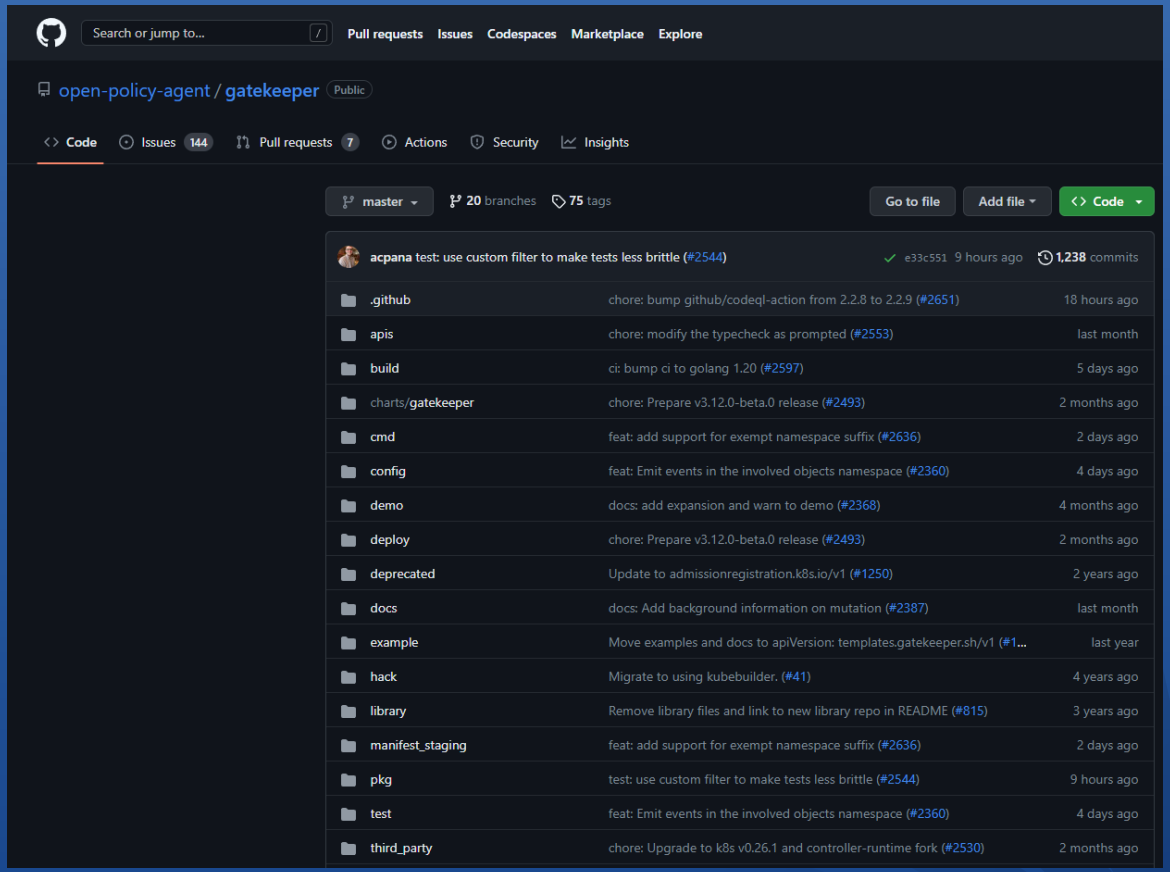
Reutilización de políticas

Gatekeeper facilita mucho la reutilización de las políticas, esto debido a varios factores:

- Las imágenes proceden únicamente de repositorios conocidos
- Los deployment tienen que tener una serie de etiquetas
- Las imágenes requieren un digest (identificador inmutable para la imágen de un contenedor)
- Los contenedores tienen que establecer límites para CPU y memoria dentro de unos valores

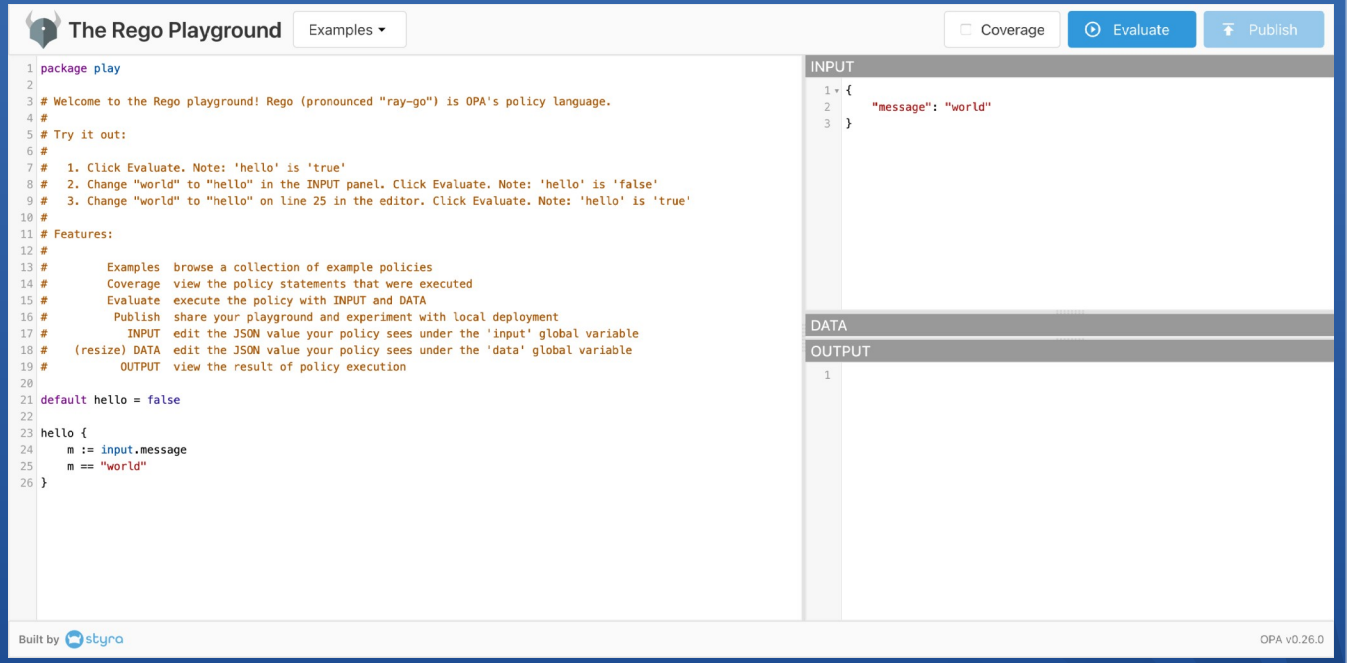
Gatekeeper Library

En GitHub existe un repositorio/librería open source llamado Gatekeeper Library, el cuál se puede reutilizar para muchas tareas comunes.
Este repositorio no solo nos proporciona la plantilla y el código en rego, sino que también nos proporciona varios ejemplos de como serían unas instancias mostrándonos que tipo de objetos se permiten y otra la cuál va a fallar dicha política.



Rego

Anteriormente hemos mencionado un lenguaje llamado Rego, este es un lenguaje de políticas declarativo creado específicamente para gatekeeper. Esta creado para expresar políticas sobre estructuras jerárquicas complejas de datos.



Prueba práctica