Unmasked: Policing your Deployments with Open Policy Agents

Policy as Code

FOSSAsia Open Summit 2021, Tues 16th March 2021
@mjbright Consulting

https://linkedin.com/in/mjbright @mjbright @mjbright

@mjbright CONSULTING

$ who am i

- Trainer/consultant based in France
- Docker, Kubernetes, Terraform, Security
- Meetup Organizer

Grab a seat and let's get started …

Policy as Code / Policy Engines

Open Policy Agent & eco-system

Specialized tools: Kyverno

Summary

Why Policy Engines?

Why ?

We need ways of automatically checking policy adherence to secure the platform from attacks

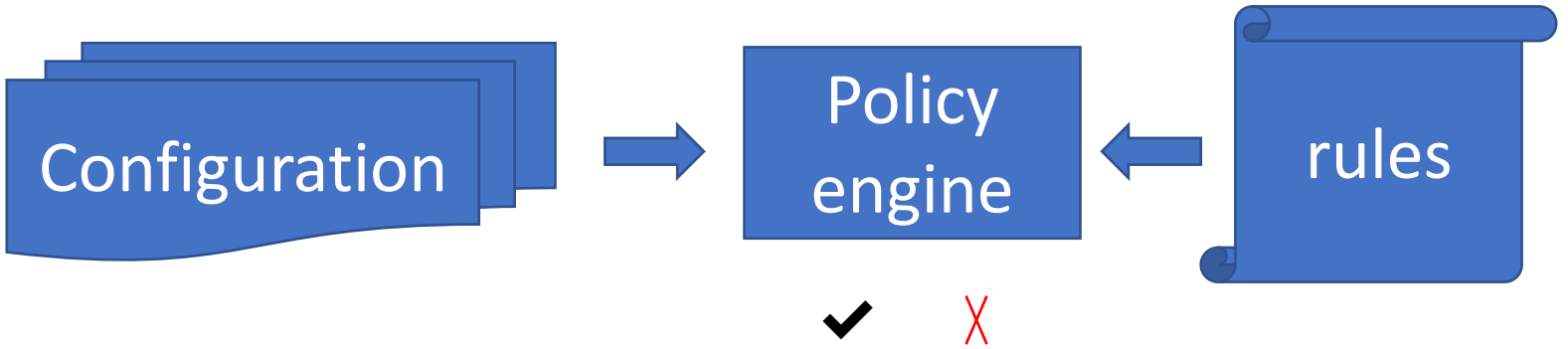
Policy engines allow to apply a set of rules, which can evolve with security requirements independently from the platform

Policy engines may be
- open source or not
- declarative or not
- platform-specific or not

Usage

Usage (static)

Static validation

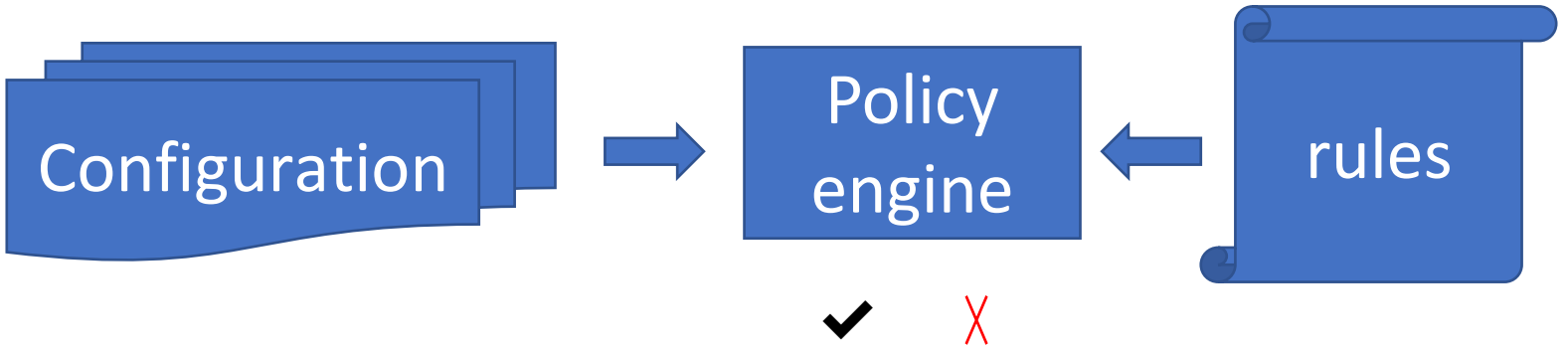


Can be used to validate

- Kubernetes YAML or JSON manifests
- Terraform configurations
- Many other types of configuration … via conftest

Usage (static)

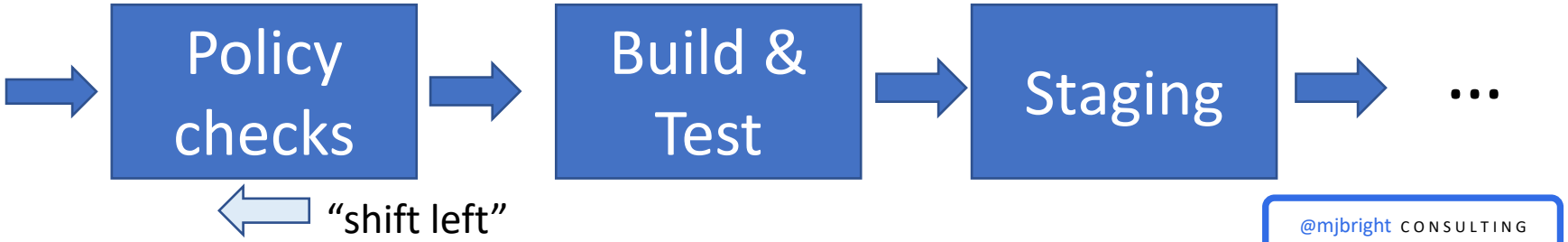
Static validation



Can be used to validate

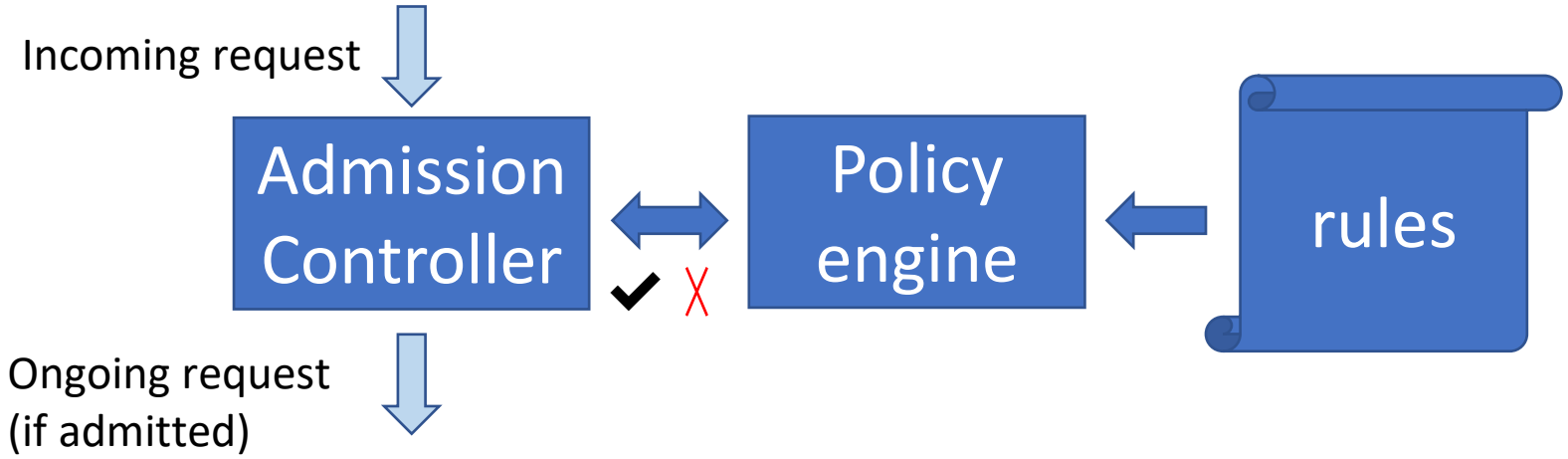
- Kubernetes YAML or JSON manifests
- Terraform configurations

e.g. as part of a CI/CD pipeline “shift left”



Usage (dynamic)

Dynamic checks, e.g. intercepting API requests



Can be used to validate

- [enforce] allow/deny API requests
- [audit] log API requests

Can be used to mutate

- Modify the incoming request, to adhere to Policy

Kubernetes Policy Enforcement

Kubernetes – *“pathologically extensible”*

Kubernetes Policy Enforcement

Kubernetes – *“pathologically extensible”*

So we need to be able to adapt to that

The good news is we can plugin a policy agent and update it’s rules independently of Kubernetes

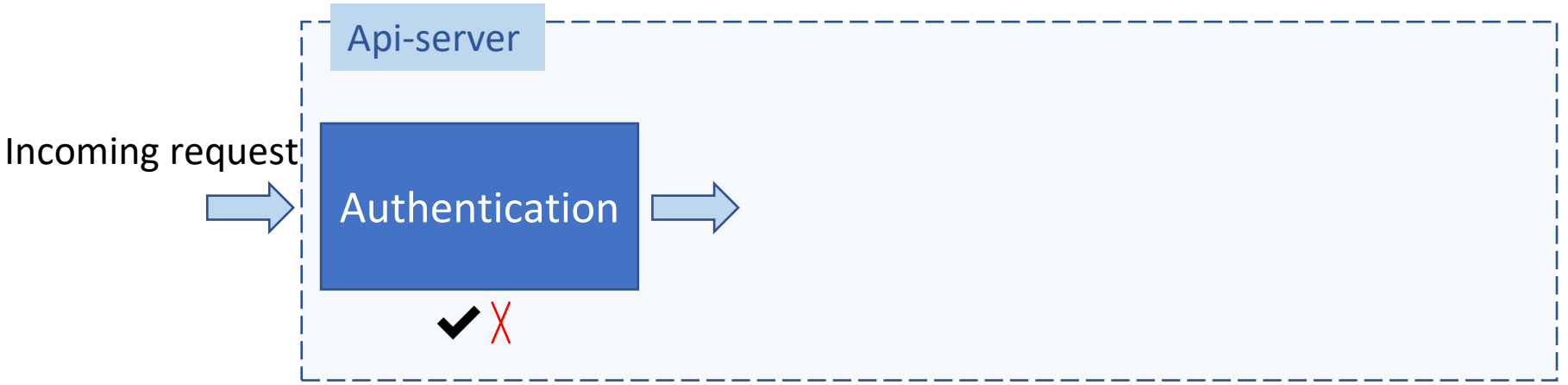
Kubernetes Access Control

3 stages of Kubernetes (API) Access control

Kubernetes Access Control

3 stages of Kubernetes (API) Access control

- Authentication: verify identity
- Authorisation: verify permissions
- Admission Control: verify request is allowed/valid

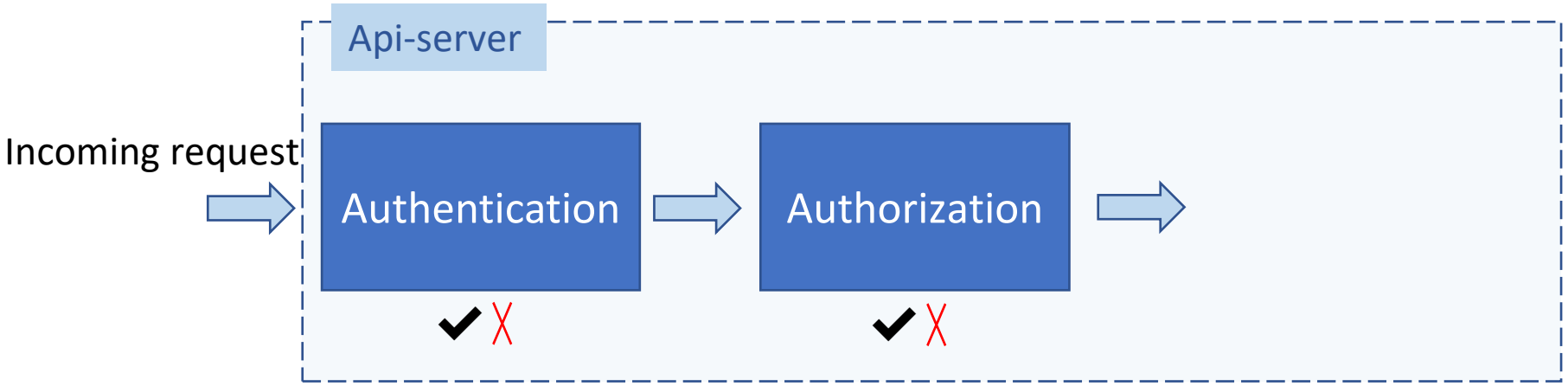


API requests from inside (Pods) or outside the cluster (kubectl, dashboard, …)

Kubernetes Access Control

3 stages of Kubernetes (API) Access control

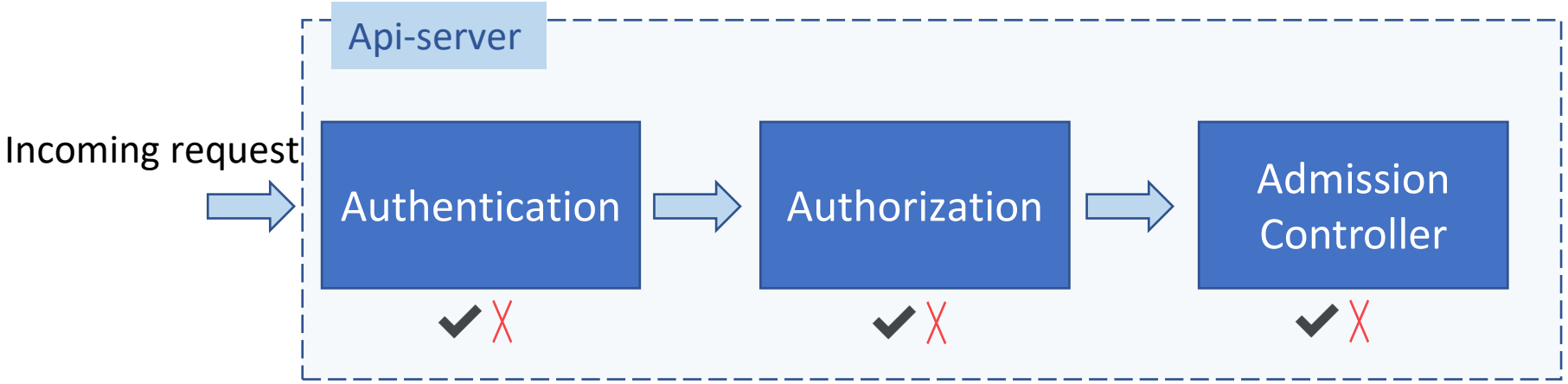
- Authentication: verify identity
- Authorisation: verify permissions
- Admission Control: verify request is allowed/valid



Kubernetes Access Control

3 stages of Kubernetes (API) Access control

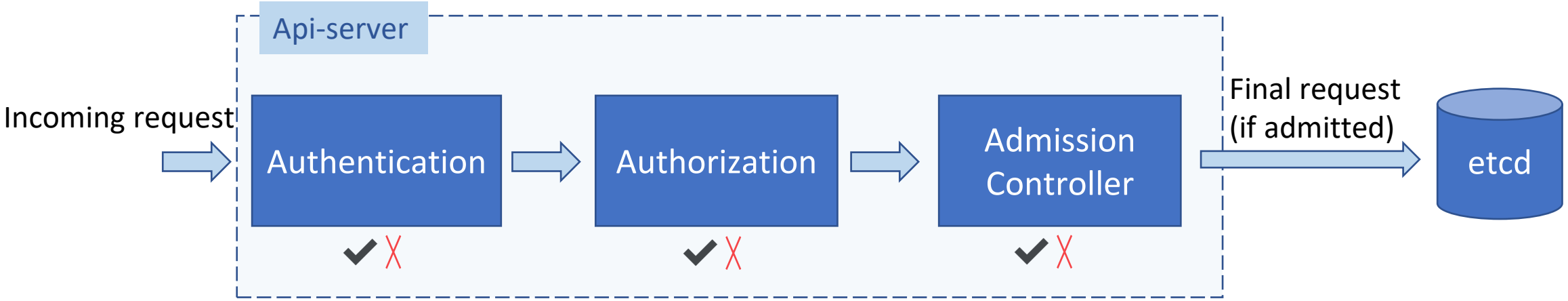
- Authentication: verify identity
- Authorisation: verify permissions
- Admission Control: verify request is allowed/valid



Kubernetes Access Control

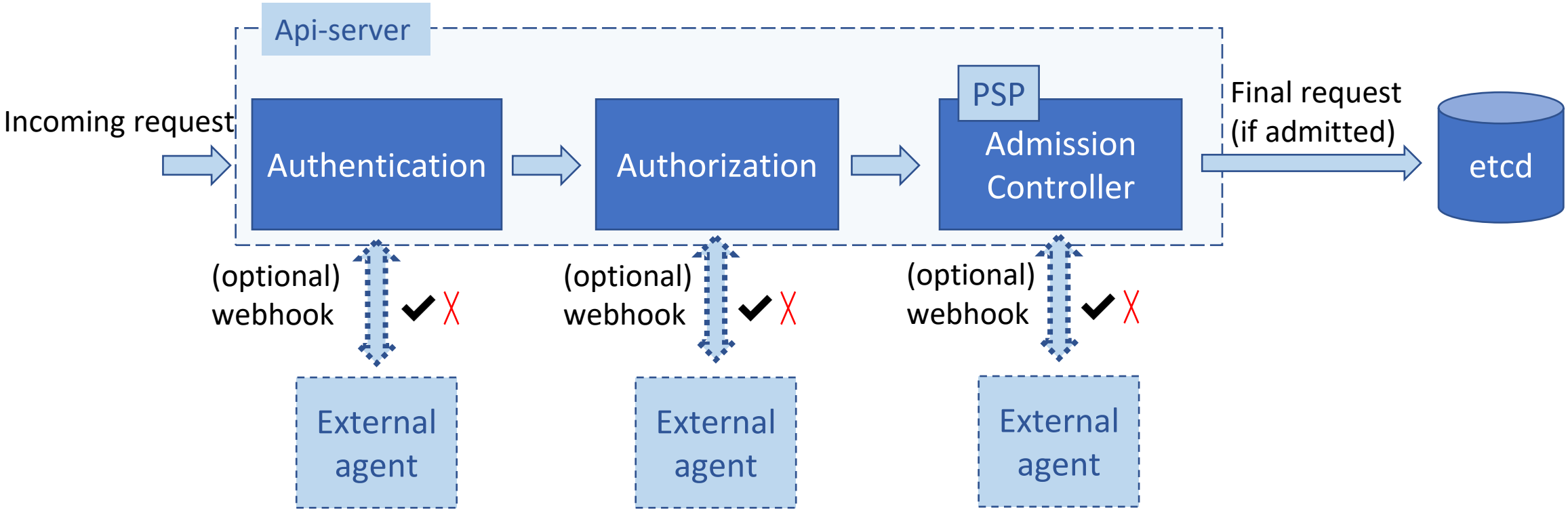
3 stages of Kubernetes (API) Access control

- Authentication: verify identity
- Authorisation: verify permissions
- Admission Control: verify request is allowed/valid



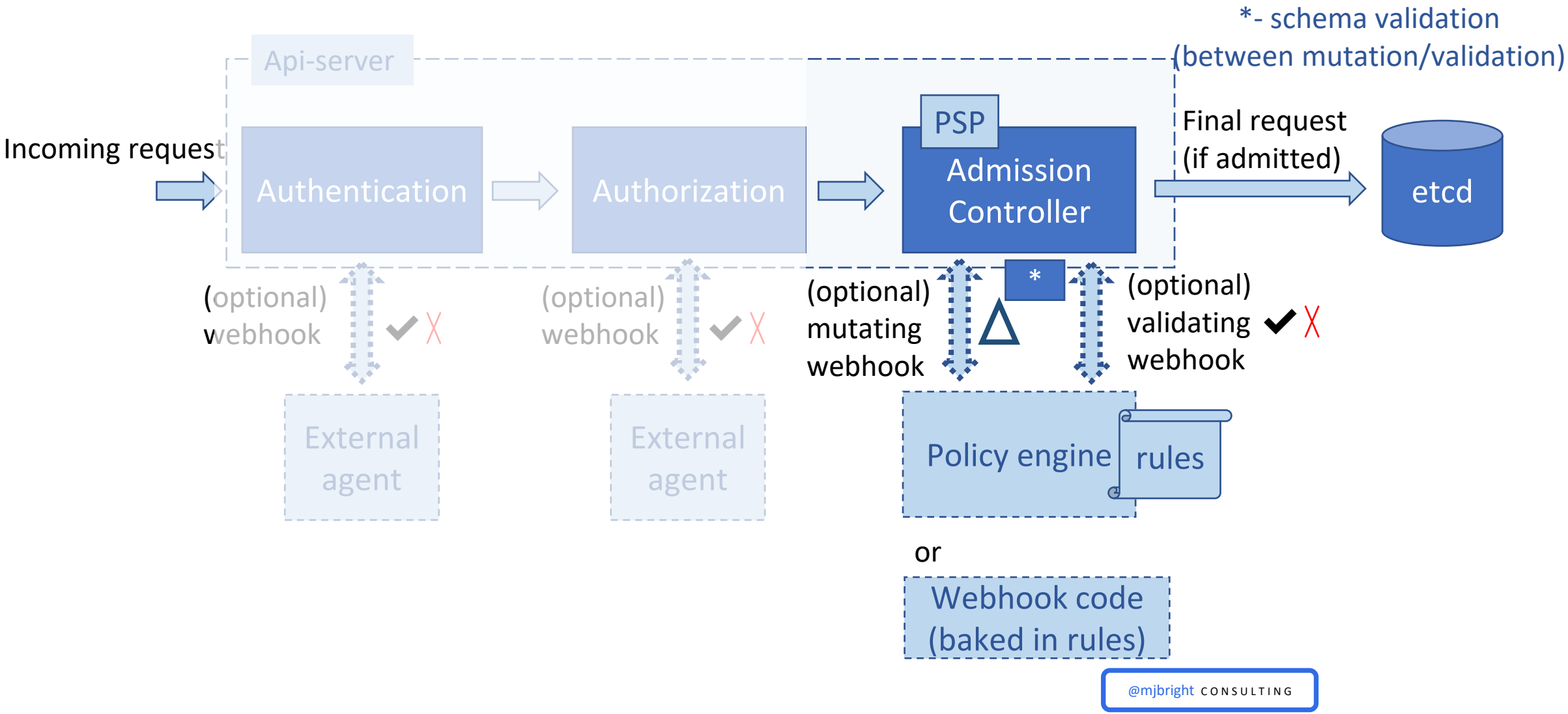
Kubernetes Access Control

Optional webhooks



Kubernetes Access Control

Optional webhooks



Example Policies

Example Kubernetes Policies

Validating webhook to enforce or log policy exceptions
Mutating webhook to coerce request to adhere to policy

[psp:capabilities] Implementation of Pod Security Policies

[manageability] Enforce use of labels on Kubernetes resources

[manage costs] Prevent use of LoadBalancer service

[best practice] Require specification of a non-default namespace

[best practice] Enforce Requests/Limit specifications on PodSpec

[best practice] Enforce ResourceQuota/LimitRange on Namespaces

Policy as Code / Policy Engines

Open Policy Agent & eco-system

Specialized tools: Kyverno

Summary

Open Policy Agent (OPA)

Generalized Policy Agent

Open Policy Agent (OPA)

Generalized Policy Agent

Open source project created by Styra

Generalized policy engine - not Kubernetes-specific - can be applied to many components of your diverse IT stack

Policies defined in the Rego language

Styra provide their DAS product as well as many open source tools

3rd-party products and open source tools

https://www.openpolicyagent.org/

Open Policy Agent (OPA)

OPA Open Source & free tools

Open Source OPA Policy Engine usable as REST API server, cll tool or API

Or via free tools

- rego playground https://play.openpolicyagent.org/
- VSCode extension https://github.com/open-policy-agent/vscode-opa

3rd-party tools

- Conftest https://www.conftest.dev/
- Fregot [rego toolkit]
- regula[rego lib for Terraform] https://github.com/fugue/regula

Open Policy Agent (OPA)

Rego Playground: edit, test, share rules and input/output

The Rego Playground | Examples | Coverage | Evaluate | Publish

```
1  package play
2
3  # Welcome to the Rego playground! Rego (pronounced "ray-go") is OPA's po
4  #
5  # Try it out:
6  #
7  #   1. Click Evaluate. Note: 'hello' is 'true'
8  #   2. Change "world" to "hello" in the INPUT panel. Click Evaluate. Not
9  #   3. Change "world" to "hello" on line 25 in the editor. Click Evaluat
10 #
11 # Features:
12 #
13 #         Examples  browse a collection of example policies
14 #         Coverage  view the policy statements that were executed
15 #         Evaluate  execute the policy with INPUT and DATA
16 #          Publish  share your playground and experiment with local depl
17 #            INPUT  edit the JSON value your policy sees under the 'inpu
18 #    (resize) DATA  edit the JSON value your policy sees under the 'data
19 #           OUTPUT  view the result of policy execution
20
21 default hello = false
22
23 hello {
24     m := input.message
25     m == "world"
26 }
```

INPUT

```
1 {
2     "message": "world"
3 }
```

DATA

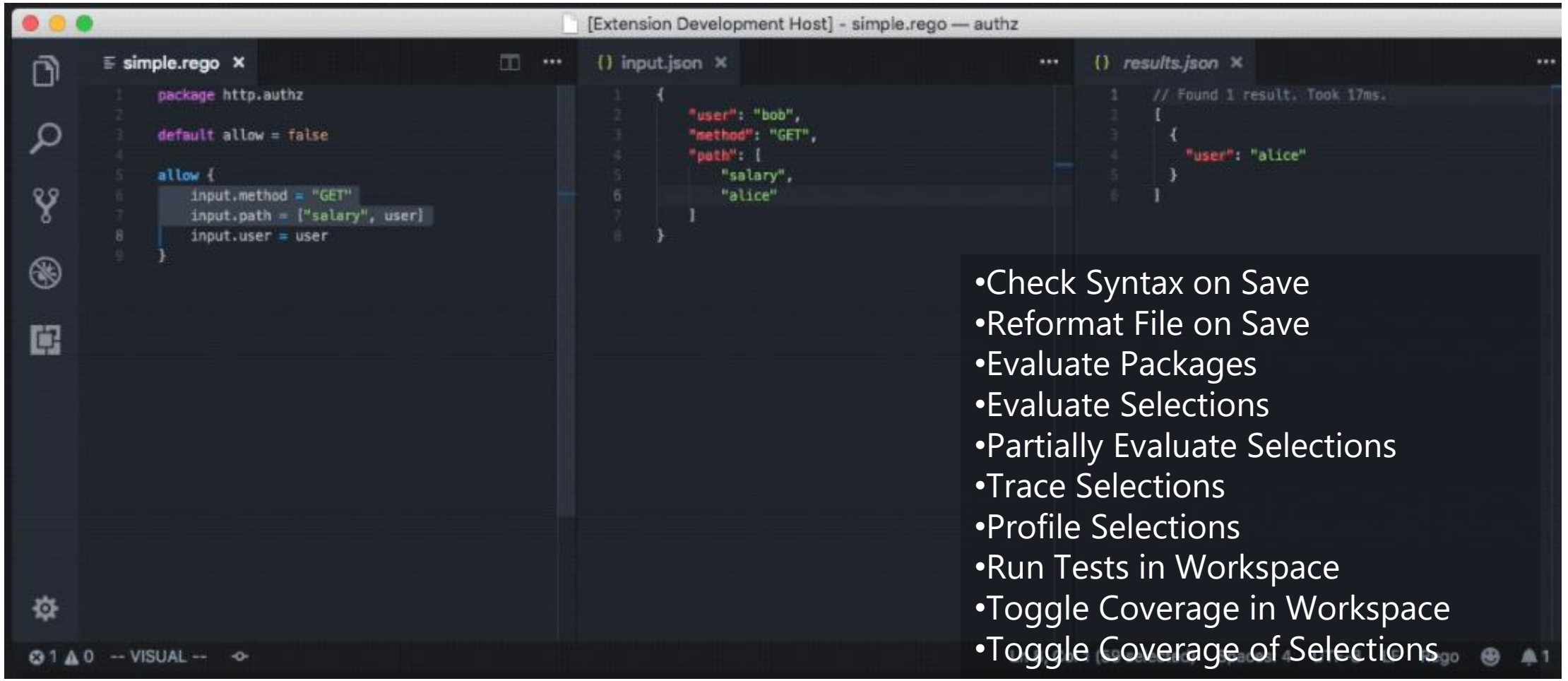
OUTPUT

```
1
```

https://play.openpolicyagent.org/

Open Policy Agent (OPA)

VSCode Extension



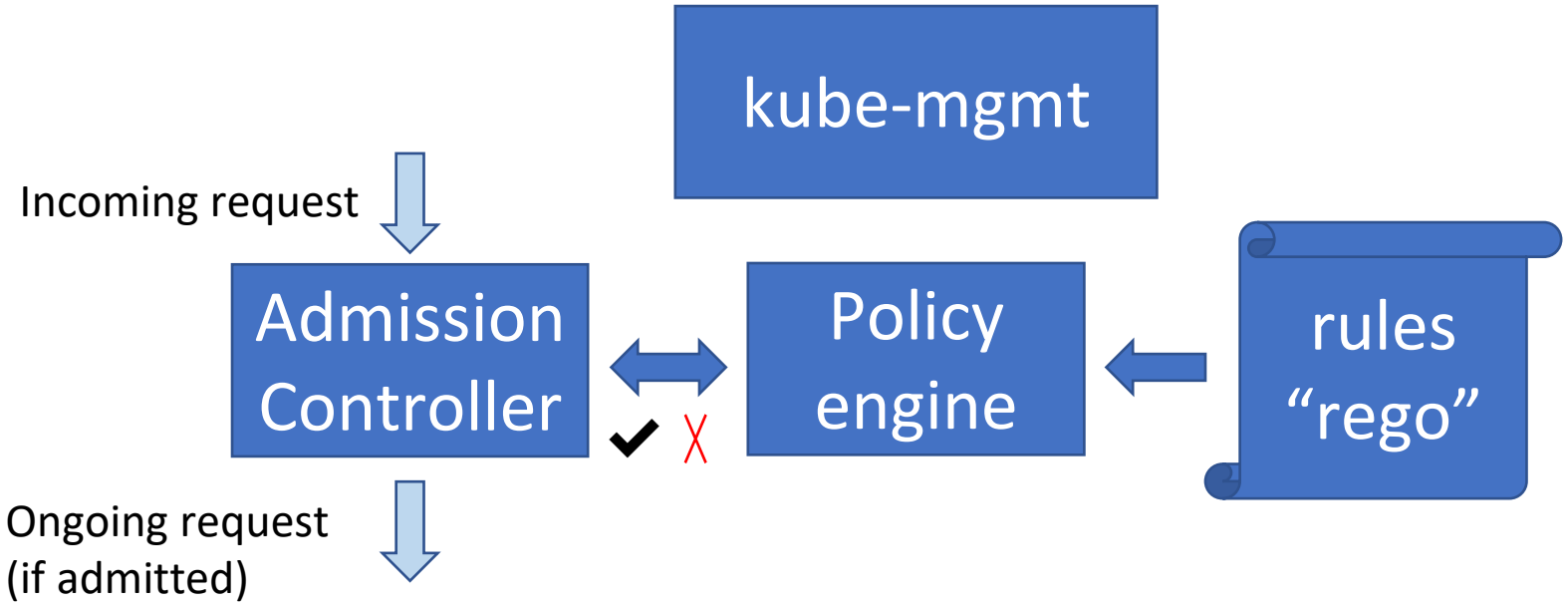
https://github.com/open-policy-agent/vscode-opa

Open Policy Agent (OPA)

OPA & Kubernetes

Open Policy Agent (OPA)

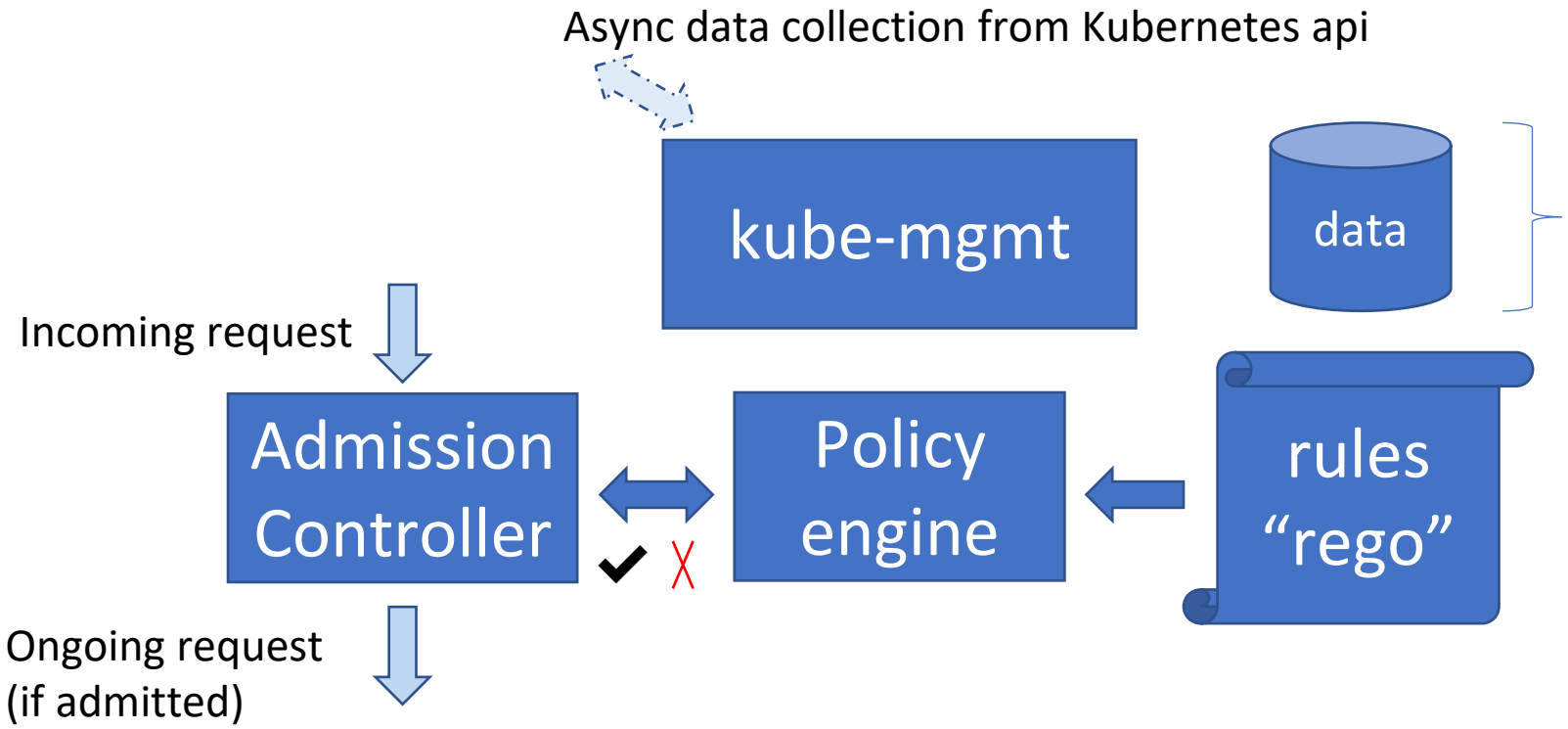
OPA & Kubernetes (GK v1)



https://www.openpolicyagent.org/

Open Policy Agent (OPA)

OPA & Kubernetes (GK v1)



Caching of resource data e.g. rules may act on pods but may require information about namespaces

Information about namespaces will not be in the incoming request but can be cached by OPA

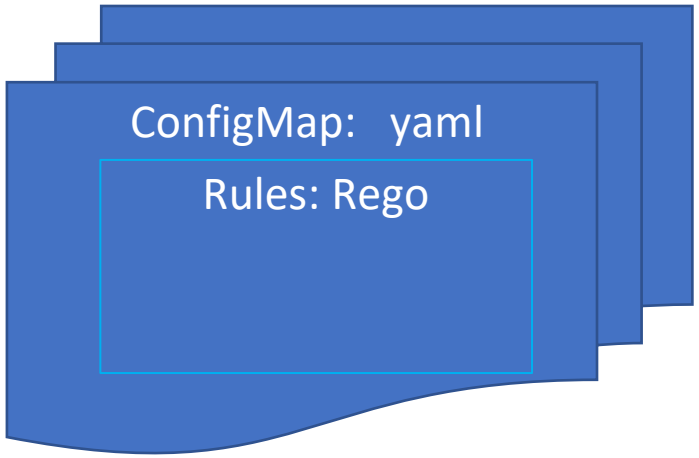
https://www.openpolicyagent.org/

OPA – Policy Configuration

Create ConfigMaps containing Policies (Rego), e.g.

```
kubectl create configmap enforce-labels --from-file=enforce_labels.rego
```

Check status of ConfigMap
- annotation: status OK??



Open Policy Agent Rules (Rego)

Example Rego policy rules

Open Policy Agent Rules (Rego)

Enforce label Best Practices

Open Policy Agent Rules (Rego)

Enforce label Best Practices – Rego policy

```
operations={ "CREATE", "UPDATE" }

deny[msg] {
    input.request.kind.kind == “Deployment"

    # Check that this is one of the operations to apply Policy to:
    operations[ input.request.operation ]

    # Create the Sets of 'provided' and 'required' labels, then calculate
    provided := {label | input.request.object.metadata.labels[label]}
    required  := {label | label := enforce_labels[_]}
    missing := required - provided
    count(missing) > 0
    msg := sprintf("Required labels are missing: %v", [missing])
}
```

```
enforce_labels = {
    "app.kubernetes.io/name",
    "app.kubernetes.io/instance",
    "app.kubernetes.io/version",
    "app.kubernetes.io/component",
    "app.kubernetes.io/part-of",
    "app.kubernetes.io/managed-by“
}
```

Adapted from example here:
https://medium.com/@bouwe.ceunen/admission-control-on-kubernetes-9a1667b7e322

Open Policy Agent Rules (Rego)

Enforce label Best Practices – input / output

INPUT (derived from OPA decision logs, here reduced):

```
{       "apiVersion": "admission.k8s.io/v1beta1",
       "kind": "AdmissionReview",
       "request": {
          "kind": { " kind ": " Deployment “ },
          "operation": "CREATE",
          "object": {
             "apiVersion": "apps/v1",
             "kind": "Deployment",
             "metadata": {
                "labels": {
                   "app": "ckad-demo-no-labels"
                },
   ....
```

```
enforce_labels={
   "app.kubernetes.io/name",
}
```

OUTPUT (reduced):

```
{  "result": {
    "deny": [
     "Deployment/ckad-demo-no-labels: Required labels are missing:
{\"app.kubernetes.io/name\"}"
    ],
    "enforce_labels": [
     "app.kubernetes.io/name“,
      …
    ],
```

Playground example: missing label - https://play.openpolicyagent.org/p/lbbB2MfyXD

Open Policy Agent Rules (Rego)

Enforce label Best Practices – input / output

INPUT (derived from OPA decision logs, here reduced):

```
{       "apiVersion": "admission.k8s.io/v1beta1",
        "kind": "AdmissionReview",
        "request": {
           "kind": { " kind ": " Deployment “ },
           "operation": "CREATE",
           "object": {
              "apiVersion": "apps/v1",
              "kind": "Deployment",
              "metadata": {
                 "labels": {
                    "app": "ckad-demo-with-labels“,
                    "app.kubernetes.io/name“: "ckad-demo-with-labels”
                 },
   ….
```

```
enforce_labels={
   "app.kubernetes.io/name",
}
```

OUTPUT (full):

```
{  "result": {
     "deny": [],
     "enforce_labels": [
       "app.kubernetes.io/name"
     ],
     "operations": [
       "CREATE",    "UPDATE"
     ]
   }
}
```

Playground example: missing label - https://play.openpolicyagent.org/p/lbbB2MfyXD

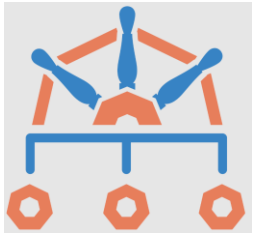
Policy as Code / Policy Engines

Open Policy Agent & eco-system

Specialized tools: Kyverno

Summary

Kyverno



Kyverno - Configuration

Policies in Kyverno are defined completely as YAML resources
They can be

- Namespaced of kind *Policy*
- or cluster-wide of kind *ClusterPolicy*

A policies may be:

- Validating or auditing
- Mutating
- Generating

Kyverno - Installation

Helm:

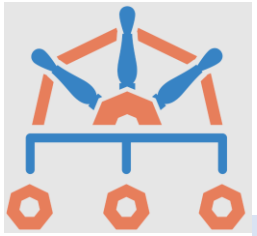
```
helm repo add kyverno https://kyverno.github.io/kyverno/
helm repo update
helm install kyverno kyverno/kyverno --namespace kyverno --create-namespace
```

Or directly from YAML:

```
kubectl create -f https://raw.githubusercontent.com/kyverno/kyverno/main/definitions/release/install.yaml
```

Kyverno - Examples

```
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: scenario1-require-labels
spec:
  validationFailureAction: enforce
  rules:
  - name: check-for-labels
    match:
      resources:
        kinds:
        - Pod
    validate:
      message: "[enforce] kyverno says -label `app.kubernetes.io/name` is required"
      pattern:
        metadata:
          labels:
            app.kubernetes.io/name: "?*"
```



Kyverno - Examples

```
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: scenario1-require-labels
spec:
  validationFailureAction: enforce
  rules:
  - name: check-for-labels
    match:
      resources:
        kinds:
        - Pod
    validate:
      message: "[enforce] kyverno says -label `app.kubernetes.io/name` is required"
      pattern:
        metadata:
          labels:
            app.kubernetes.io/name: "?*"
```

Policy as Code / Policy Engines

Open Policy Agent & eco-system

Specialized tools: Kyverno

Summary

Summary – Comparison OPA-GK/Kyverno

Both target Kubernetes using native resources and can replace PSP

OPA/GK: since 2017

- Rego allows complex rules – but steeper learning curve (Rego)
- "*OPA*" Allows to manage full IT stack, beyond Kubernetes
- HA

Kyverno: since 2019

- More familiar YAML for rules
- Generation capabilities

Choice is great …

Summary – Comparison OPA-GK/Kyverno

Features/Capabilities	Gatekeeper	Kyverno
Validation	✓	✓
Mutation	✓* (alpha)	✓
Generation	X	✓
Policy as native resources	✓	✓
Metrics exposed	✓	X
OpenAPI validation schema (kubectl explain)	X	✓
High Availability	✓	X
API object lookup	✓	✓* (alpha)
CLI with test ability	✓** (separate CLI)	✓
Policy audit ability	✓	✓

https://neonmirrors.net/post/2021-02/kubernetes-policy-comparison-opa-gatekeeper-vs-kyverno/

Summary

- Policy engines provide a way to dynamically provide policy rules
- Policy engines may be
  - open source or not
  - declarative or not
  - platform-specific or not
- Kubernetes can be extended using policy engines (needed due to impending PSP deprecation)

OPA/GK and Kyverno are useful options, other engines exist.

Thank you !

https://linkedin.com/in/mjbright @mjbright @mjbright

@mjbright CONSULTING

Resources

Resource	URL
	Kubernetes Admission Control
Kubernetes Blog	A Guide to Kubernetes Admission Controllers | Kubernetes https://kubernetes.io/blog/2019/03/21/a-guide-to-kubernetes-admission-controllers/
Banzai Cloud Blog	In-depth introduction to Kubernetes admission webhooks · Banzai Cloud https://banzaicloud.com/blog/k8s-admission-webhooks/
Kubernetes Documentation	Dynamic Admission Control https://kubernetes.io/docs/reference/access-authn-authz/extensible-admission-controllers/

Resources

Resource	URL
	Open Policy Agent
OPA website	https://www.openpolicyagent.org/
OPA docs	https://www.openpolicyagent.org/docs/latest/
Styra Academy	https://academy.styra.com/
OPA blog	https://blog.openpolicyagent.org/
Rego Playground	https://play.openpolicyagent.org/

Resources

Resource	URL
	Open Policy Agent
OPA github	https://github.com/open-policy-agent/opa
OPA gatekeeper github	https://github.com/open-policy-agent/gatekeeper
Conftest github	https://github.com/open-policy-agent/conftest
VScode Extension	https://github.com/open-policy-agent/vscode-opa

Resources

Resource	URL
	Kyverno
Kyverno website	https://kyverno.io/
Kyverno github	https://github.com/kyverno/kyverno/

Resources

Resource	URL
	OPA & Kyverno
2021-Feb Chip Zoller	Kubernetes Policy Comparison: OPA/Gatekeeper vs Kyverno https://neonmirrors.net/post/2021-02/kubernetes-policy-comparison-opa-gatekeeper-vs-kyverno/

Thank you !

 https://linkedin.com/in/mjbright

 @mjbright

 @mjbright

@mjbright CONSULTING