
HELM
TiKV
KubeCon
CloudNativeCon
Europe 2023

KubeCon | CloudNativeCon

Europe 2023

The path to self contained CRDs

Cici Huang, Google

Things to be covered?

- The Journey of CRD in Kubernetes
- The **Common Expression Language** (CEL)
- Make CRD More self contained
- The Power to extend - Policy Enforcement in Admission Chain
- Q & A

KubeCon | CloudNativeCon

Europe 2023

The Journey of CRD

The Journey of CRD

- Start as ThirdPartyResource
- Beta in 1.7; Stable in 1.16
- Extension of the Kubernetes API

The Journey of CRD

- Trying to make CRD as good as built-in types
  - Versioning
  - Subresources
  - OpenAPI schema
  - Structural schema
  - Defaulting
  - Pruning
  - ……

Validation is critical

If you don't validate the data comes in, **things will break in a way hard to reason**!

- Immutability
- Cross field checks
- Mutually exclusive
- Specific format of a field
- ...

Validation is critical

Before CEL

- Build-in Validation
  - CRD structural schemas
  - OpenAPIV3 validation
    - Format, Regex, Range/Size limit
- Validating Admission Webhook

The Price of Webhooks

- Development complexity
  - Core logic + monitoring + alerting
  - Packaging/Releasing
- Operational complexity
  - Mis-configure
  - Latency added
  - Upgrade/rollback

Validation is critical

Things people wanna do with CRD validation are **simple!**

1. Ensure a CRD field is immutable.
2. Require that field x has a specific format.
3. Cross field validations.

Can we use something simpler than webhooks?

KubeCon | CloudNativeCon

Europe 2023

The **Common Expression Language** (CEL)

CEL Examples

CEL Expression	Purpose
self.minReplicas <= self.replicas && self.replicas <= self.maxReplicas	Validate that the three fields defining replicas are ordered appropriately
self.components['Widget'].parts[2].isEnabled	Access a deeply nested field.
self.created + self.ttl < self.expired	Validate that 'expired' date is after a 'create' date plus a 'ttl' duration

CEL Documentation

The **Common Expression Language (CEL)** implements common semantics for expression evaluation, enabling different applications to more easily interoperate.

Keep it small & fast.

CEL evaluates in linear time, is mutation free, and not Turing-complete. This limitation is a feature of the language design, which allows the implementation to evaluate orders of magnitude faster than equivalently sandboxed JavaScript.

Make it extensible.

CEL is designed to be embedded in applications, and allows for extensibility via its context which allows for functions and data to be provided by the software that embeds it.

Developer-friendly.

The language is approachable to developers. The initial spec was based on the experience of developing Firebase Rules and usability testing many prior iterations.
The library itself and accompanying toolings should be easy to adopt by teams that seek to integrate CEL into their platforms.

CEL Limitations

No `for`/`while`/...

Instead use comprehension "macros":

- all()
- exists()
- exists_one()
- map()
- filter()

No `if`/`else`..

Instead use the ternary operator:

<condition> ? <ifTrue> : <ifFalse>

CEL function libraries

- CEL "stdlib"
- CEL extended string function library(regex, contains, ...)
- Kubernetes CEL library
  - list library
  - more regex
  - URLs support

Learn more:

https://kubernetes.io/docs/reference/using-api/cel/

github.com/google/cel-spec

KubeCon | CloudNativeCon

Europe 2023

Make CRDs **more self contained**

CRD Validation Rules

x-kubernetes-validations

https://kubernetes.io/docs/tasks/extend-kubernetes/custom-resources/custom-resource-definitions/#validation-rules

CRD Validation Rules

```
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
spec:
...
  openAPIV3Schema:
    type: object
    properties:
      spec:
        type: object
        x-kubernetes-validations:
          - rule: "self.replicas <=
self.maxReplicas"
        properties:
          ...
          replicas:
            type: integer
          maxReplicas:
            type: integer
```

Can be put anywhere under openAPIV3Schema.

CRD Validation Rules

```
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
spec:
...
  openAPIV3Schema:
    type: object
    properties:
      spec:
        type: object
        x-kubernetes-validations:
          - rule: "self.replicas <= self.maxReplicas"
        properties:
          ...
          replicas:
            type: integer
          maxReplicas:
            type: integer
```

"self" is a CEL variable. It provides access to values, scoped to current schema.

CRD Validation Rules

```
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
spec:
...
  openAPIV3Schema:
    type: object
    properties:
      spec:
        type: object
        required: [foo]
        properties:
          foo:
            type: integer
            x-kubernetes-validations:
              - rule: "self == oldSelf"
```

"oldSelf" is another CEL variable.

If you use it, the CEL expression is called a "transition rule".

This transition rule enforces immutability of the foo field.

Learn More: https://kubernetes.io/blog/2022/09/29/enforce-immutability-using-cel

CRD Validation Rules

```
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
spec:
...
  openAPIV3Schema:
    type: object
    properties:
      spec:
        type: object
        x-kubernetes-validations:
          - rule: "!has(self.foo) ||
self.foo.startsWith('kube')"
        properties:
          ...
          foo:
            type: string
            x-kubernetes-validations:
              - rule: "self.startsWith('kube')"
```

Same validation but scoped differently.

CRD Validation Rules

```
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
spec:
...
  openAPIV3Schema:
    type: object
    properties:
      spec:
        type: object
        x-kubernetes-validations:
          - rule: "self.replikas <= self.maxReplicas"
        properties:
          ...
          replicas:
            type: integer
          maxReplicas:
            type: integer
```

Type checking while creating or updating your CRDs:

compilation failed: ERROR: <input>:1:5: undefined field 'replikas'

CRD Validation Rules

```
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
spec:
...
  openAPIV3Schema:
    type: object
    properties:
      spec:
        type: object
        x-kubernetes-validations:
          - rule: "self.replicas <= self.maxReplicas"
            message: “replicas must be no greater than 3”
            messageExpression: "'replicas must be no greater
than ' + string(self.maxReplicas)"
        properties:
          ...
          replicas:
            type: integer
          maxReplicas:
            type: integer
```

Support CEL expression in validation failure message.

CRD Validation Rules

Runtime Constraints

- Estimated Cost Limit
- Runtime Cost Limit
- Context Cancelation

Learn More:
https://kubernetes.io/docs/tasks/extend-kubernetes/custom-resources/custom-resource-definitions/#resource-use-by-validation-functions

One more step…

CRD Version Conversion

Conversion is required when

- custom resource is requested in a different version than stored version.
- Watch is created in one version but the changed object is stored in another version.
- custom resource PUT request is in a different version than storage version.

Webhook Conversion is in Stable since 1.16, but...

Webhook conversion

```
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
spec:
...
  conversion:
    # a Webhook strategy instruct API server to call an external
webhook for any conversion between custom resources.
    strategy: Webhook
    webhook:
      # conversionReviewVersions indicates what
ConversionReview versions are understood/preferred by the
webhook.
      # The first version in the list understood by the API server is
sent to the webhook.
      # The webhook must respond with a ConversionReview
object in the same version it received.
      conversionReviewVersions: ["v1","v1beta1"]
      clientConfig:
        service:
          namespace: default
          name: example-conversion-webhook-server
          path: /crdconvert
        caBundle: "Ci0tLS0tQk...<base64-encoded PEM
bundle>...tLS0K"
```

- Hard to configure
- Involve Webhook
- Perform bad

CRD Conversion with CEL

Next Step: CRD Conversion with CEL

- Self Contained
- Fast
- Sufficient subs
  conversion we

mentation
8s.io/v1
ition

s

Name
toField: .spec.name.first
transformation: self

Learn More: [KEP-3945 CRD conversion with CEL]

CRD VS Built-in Types

CRD is **more self contained** than built-in types

	built-in	CRD
API declaration	🟢	🟢
OpenAPI schema	🟢	🟢
Versioning	🟢	🟢
Subresources	🟢	🟢
Declarative validation	🟡	🟢
Declarative conversion	🟡	🟢

Declarative Validation

Catch up on Native Types: Declarative Validation

Learn More: [KEP-3937 Declarative Validation]

KubeCon | CloudNativeCon

Europe 2023

Extend the power to a larger and more impactful area: **policy enforcement**.

Policy Enforcement in Kubernetes

- Internal support such like Pod Security Admission
- Polic Gat
- Buil

Open Policy Agent

ValidatingAdmissionPolicy

Kyverno

Learn More: https://kubernetes.io/docs/reference/access-authn-authz/validating-admission-policy/

Policy Management

Roles involved in policy management

Usually not the same person

Often in different organizations

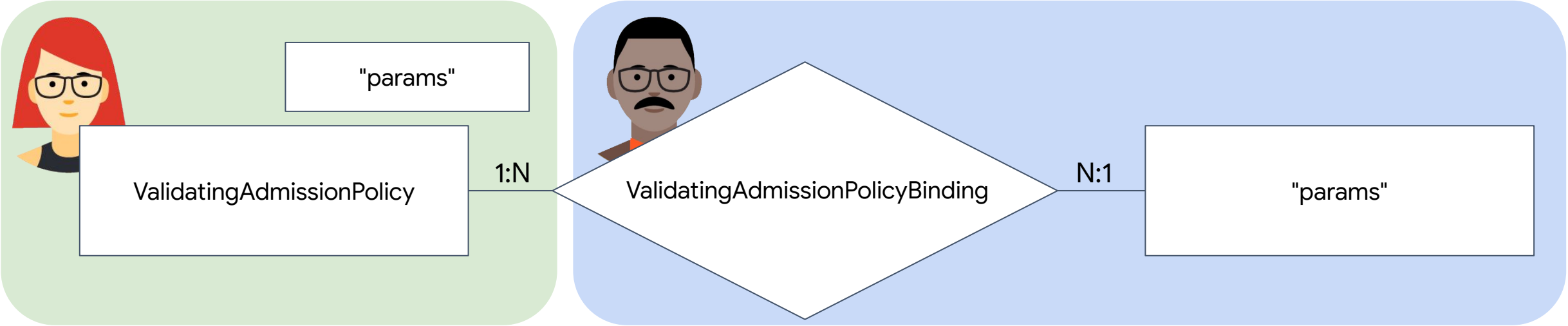
Policy Author

Concerns:
- Correctness of policy
- Reuse/Configurability

Cluster Administrator

Concerns:
- Configuring policy for organization
- Operability, esp. Safe rollouts

ValidatingAdmissionPolicy



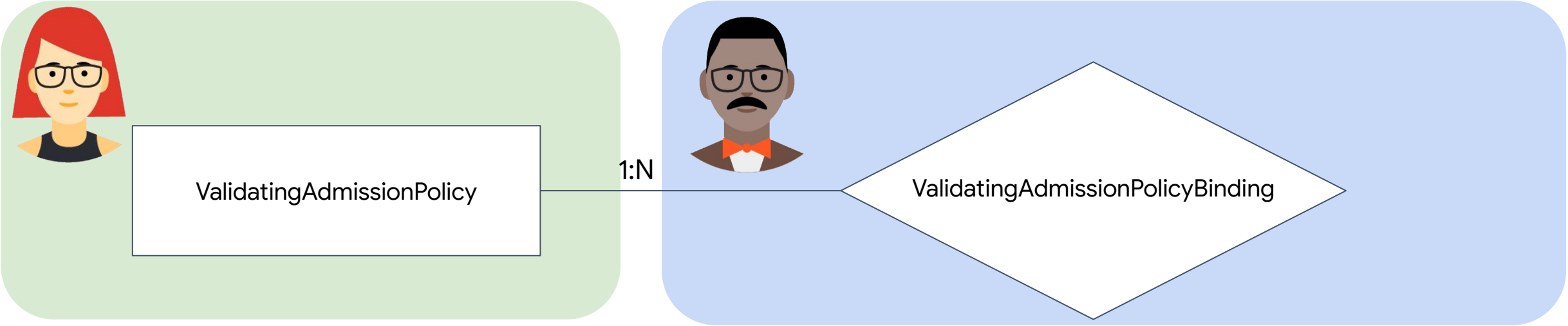

```
apiVersion: admissionregistration.k8s.io/v1alpha1
kind: ValidatingAdmissionPolicy
metadata:
  name: "replicalimit-policy.example.com"
spec:
  matchConstraints:
    resourceRules:
    - apiGroups:   ["apps"]
      apiVersions: ["v1"]
      operations:  ["CREATE", "UPDATE"]
      resources:   ["deployments"]
  validations:
    - expression: "object.spec.replicas <=
int(params.data.maxReplicas)"
  paramKind:
    apiVersion: v1
    kind: ConfigMap
```

```
apiVersion: admissionregistration.k8s.io/v1alpha1
kind: ValidatingAdmissionPolicyBinding
metadata:
  name: "replicalimit-binding-test.example.com"
spec:
  policyName: "replicalimit-policy.example.com"
  paramRef:
    name: "replica-limit-test.example.com"
    namespace: "default"
  matchResources:
    namespaceSelector:
      matchLabels:
        environment: test
  validationActions: [Deny]
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: "replica-limit-test.example.com"
data:
  maxReplicas: “3”
```

ValidatingAdmissionPolicy

No parameterization needed?




```yaml
apiVersion: admissionregistration.k8s.io/v1alpha1
kind: ValidatingAdmissionPolicy
metadata:
  name: "replicalimit-policy.example.com"
spec:
  matchConstraints:
    resourceRules:
    - apiGroups:   ["apps"]
      apiVersions: ["v1"]
      operations:  ["CREATE", "UPDATE"]
      resources:   ["deployments"]
  validations:
    - expression: "object.spec.replicas <= 3"
```

```yaml
apiVersion: admissionregistration.k8s.io/v1alpha1
kind: ValidatingAdmissionPolicyBinding
metadata:
  name: "replicalimit-binding-test.example.com"
spec:
  policyName: "replicalimit-policy.example.com"
  validationActions: [Deny]
```

Best Practices

- **Parameterization**
- policy.matchConstraints VS binding.matchResources VS policy.matchConditions
- failurePolicy VS validationActions
- Leverage authz check

Best Practices

- Parameterization
- **policy.matchConstraints VS binding.matchResources VS policy.matchConditions**
- failurePolicy VS validationActions
- Leverage authz check

Best Practices - matchConditions

```
kind: ValidatingAdmissionPolicy
spec:
  ...
  matchConditions:
   - name: 'exclude-leases' # Each match condition must have a unique name
     expression: '!(request.resource.group == "coordination.k8s.io" && request.resource.resource == "leases")' # Match non-lease resources.
    - name: 'exclude-kubelet-requests'
     expression: '!("system:nodes" in request.userInfo.groups)' # Match requests made by non-node users.
    - name: 'rbac' # Skip RBAC requests.
     expression: 'request.resource.group != "rbac.authorization.k8s.io"'
  ...
```

MatchConditions to fine-grained request filtering:

- Are CEL expressions
- Must evaluate to true
- Has been added into Webhook as alpha feature in 1.27

Learn More:
https://kubernetes.io/docs/reference/access-authn-authz/validating-admission-policy/#matching-requests-matchconditions

Best Practices

- Parameterization
- policy.matchConstraints VS binding.matchResources VS policy.matchConditions
- **failurePolicy VS validationActions**
- Leverage authz check

failurePolicy VS validationActions

policy.failurePolicy

- Define how to handle failures
  - Compilation errors
  - Runtime errors
  - Mis-configuration
  - ...
- Fail
- Ignore

binding.validationActions

Define how `validations` are enforced:

- **Deny** - Validation failure results in a denied request.
- **Warn** - Validation failure is reported to the request client as a warning
- **Audit** - Validation failure is included in the audit event for the API request.

Learn More: https://kubernetes.io/docs/reference/access-authn-authz/validating-admission-policy/#validation-actions

Best Practices

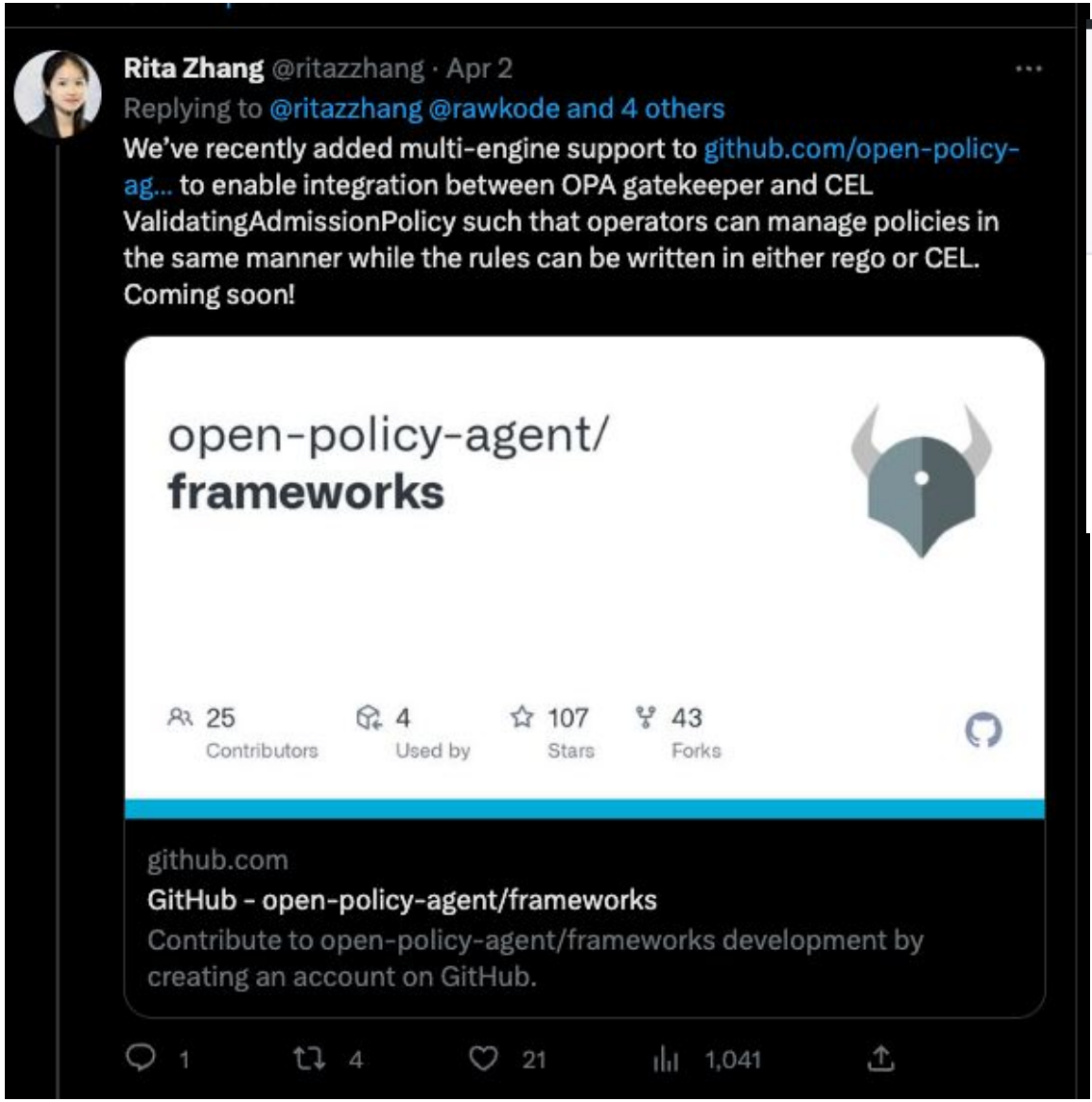
- Parameterization
- policy.matchConstraints VS binding.matchResources VS policy.matchConditions
- failurePolicy VS validationActions
- **Leverage authz check**

Best Practices - Authz Check

CEL Expression	Purpose
authorizer.path('/healthz').check('get').allowed()	Check if the principal (user or service account) submitting the request is authorized to perform a HTTP GET for the non-resource `/healthz` path.
authorizer.serviceAccount('default', 'myserviceaccount')	Checks if the principal (user or service account) submitting the request is authorized the service account with namespace 'default' and name 'myserviceaccount'.
authorizer.requestResource.check('custom-verb').allowed()	Checks if the principal (user or service account) submitting the request is authorized for `custom-verb` on the resource being handled.

Learn More: https://pkg.go.dev/k8s.io/apiserver/pkg/cel/library#Authz

Adoption



Kubernetes Validating Admission Policies: A Practical Example

Thursday, March 30, 2023

Authors: Craig Box (ARMO), Ben Hirschberg (ARMO)

Adoption

Wanna try the feature early?

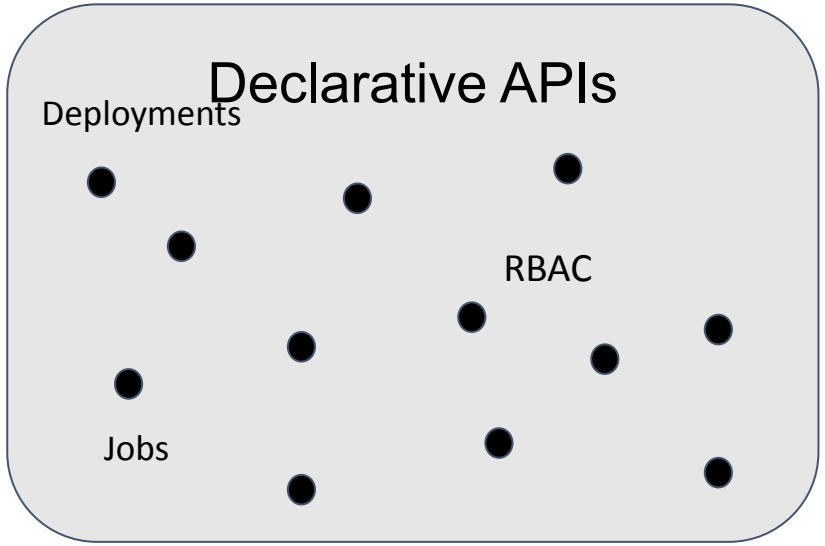
kubernetes/cel-admission-webhook

Takeaway

Takeaway

CRD advanced validation

Policy Enforcement

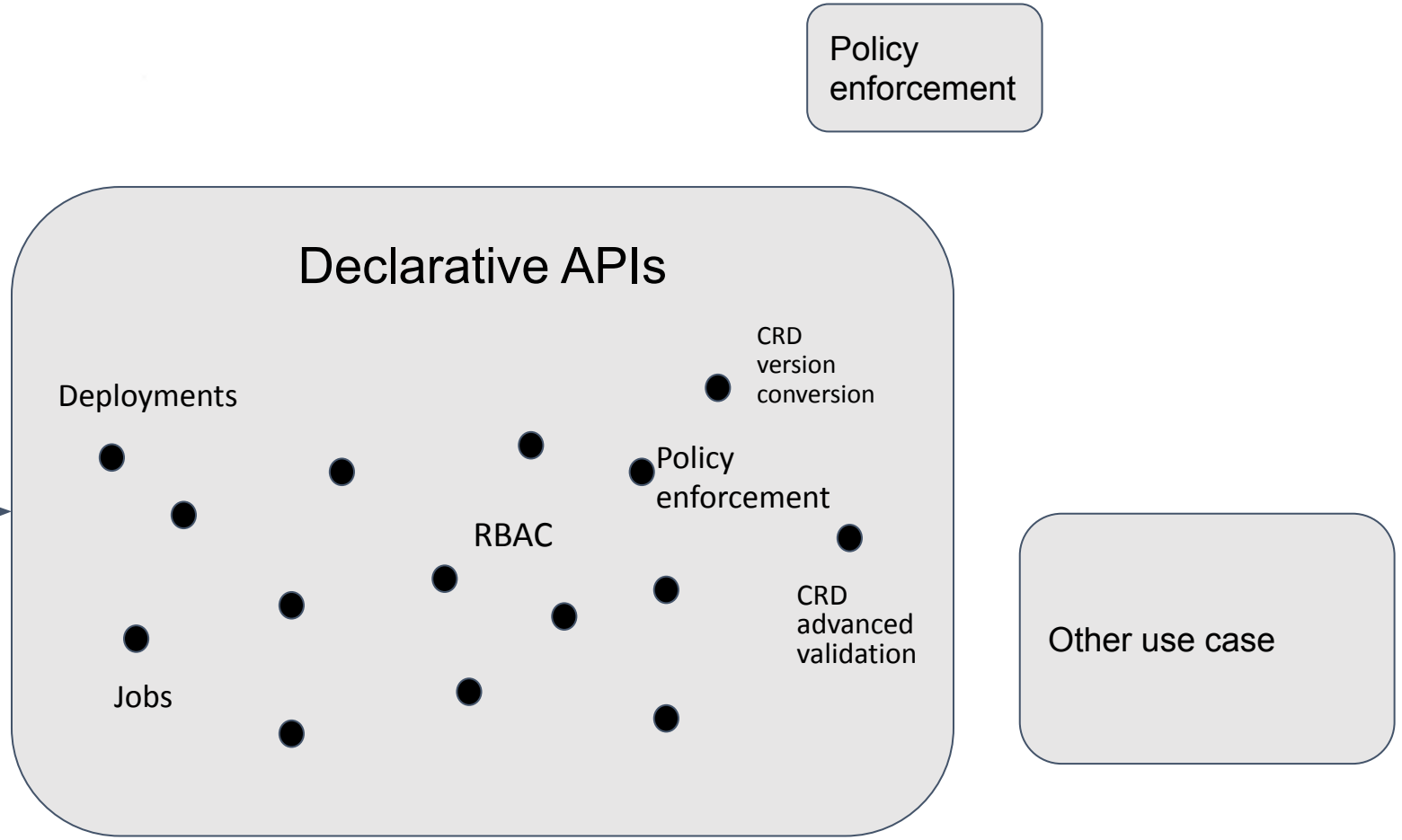


CRD version conversion

Other use cases

Takeaway

CEL helps to expand the power of declarative APIs



The Next

- CRD Conversion with CEL
- CEL in native types: declarative validation
- MutatingAdmissionPolicy
- Client Side Validation Tool

To Learn More:

KubeCon | CloudNativeCon Europe 2023

SIG API Machinery CEL group:

- Mailing list:
  - kubernetes-sig-api-machinery@googlegroups.com
  - wg-sig-api-machinery-cel-dev-external
- Slack channel: #sig-api-machinery-cel-dev
- Bi-weekly community meetings

Please scan the QR Code above
to leave feedback on this session