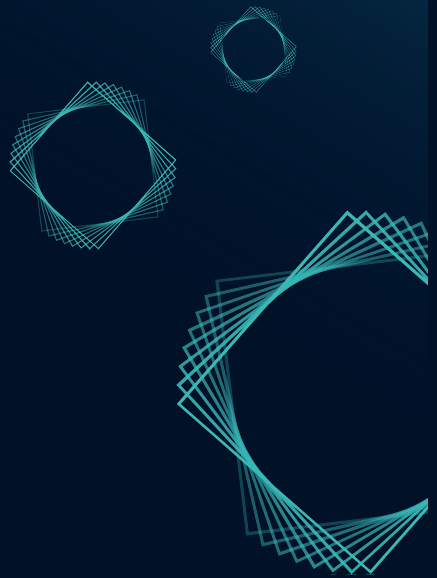
KUBERMATIC

Kyverno Gets Smarter: Writing Dynamic Policies with CEL

KyvernoCon Virtual Event 2025



Kyverno

KUBERMATIC

WHOAMI?

Koray Oksay

- Kubernetes Consultant and Trainer
- CNCF Ambassador & Kubestranout
- KCD and DevOpsDays Istanbul Org.
- Contributor at #sig-k8s-infra and apisnoop/verify-conformance

KUBERMATIC

Kubernetes 💙 CEL

So does Kyverno

What is CEL?

- Google's **`Common Expression Language`**
- https://cel.dev
- Fast, Portable, Extensible, and Safe
- Declarative expressions, evaluated at runtime
- Common Expression Language in Kubernetes
- Kubernetes support:
  - CRD Validation Rules (since k8s v1.25)
  - ValidatingAdmissionPolicy (since k8s v1.30)

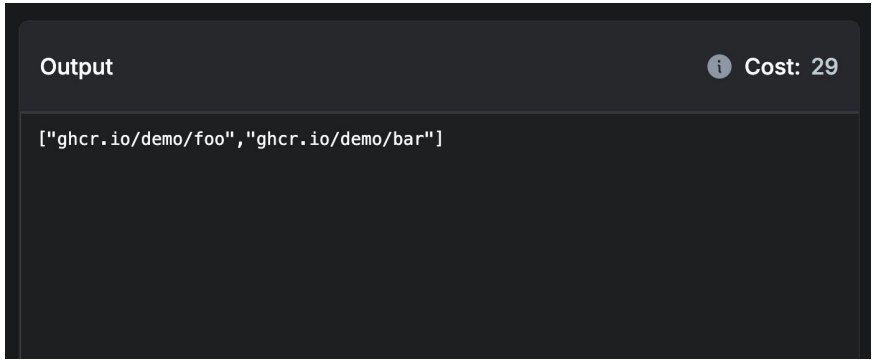
```
"ghcr.io/kyverno/kyverno".startsWith("ghcr.io") // true

"nginx:latest".endsWith(":latest") // true
```

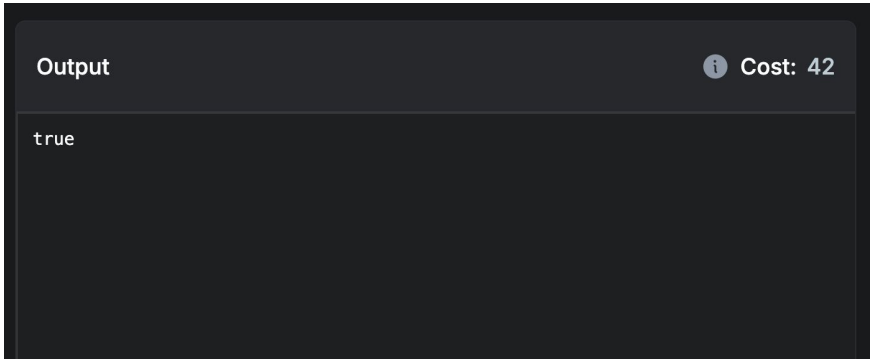
```
[
  "ghcr.io/demo/foo",
  "ghcr.io/demo/bar",
  "docker.io/demo/foobar"
].all(img, img.startsWith("ghcr.io"))  // false
```

```
[
  {
    "containerName": "foo",
    "containerImage": "ghcr.io/demo/foo"
  },
  {
    "containerName": "bar",
    "containerImage": "ghcr.io/demo/bar"
  }
].map(c, c.containerImage).all(img, img.startsWith("ghcr.io"))   // true
```

https://github.com/google/cel-spec/blob/master/doc/langdef.md

https://playcel.undistro.io/





CEL PLAYGROUND | CEL Expression | Modes | Share | Visit our GitHub

CEL Expression

Disallow HostPorts

```
// According the Pod Security Standards, HostPorts should be disallowed entirely.
// https://kubernetes.io/docs/concepts/security/pod-security-standards/#baseline

object.spec.template.spec.containers.all(container,
  !has(container.ports) ||
  container.ports.all(port,
    !has(port.hostPort) ||
    port.hostPort == 0
  )
)

```

Input

Run

```
object:
  apiVersion: apps/v1
  kind: Deployment
  metadata:
    name: nginx
  spec:
    template:
      metadata:
        name: nginx
        labels:
          app: nginx
      spec:
        containers:
          - name: nginx
            image: nginx
            ports:
              - containerPort: 80
                hostPort: 80 # the expression looks for this field
    selector:
      matchLabels:
        app: nginx
```

Output

Cost: 22

```
false
```

What about Kyverno?

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: disallow-latest-tag
spec:
  validationFailureAction: Audit
  background: true
  rules:
  - name: require-image-tag
    match:
      any:
      - resources:
          kinds:
          - Pod
    validate:
      message: "An image tag is required."
      foreach:
        - list: "request.object.spec.containers"
          pattern:
            image: "*:*"
  - name: validate-image-tag
    match:
      any:
      - resources:
          kinds:
          - Pod
    validate:
      message: "Using image tag 'latest' is not allowed."
      foreach:
        - list: "request.object.spec.containers"
          pattern:
            image: "!*:latest"
```

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: disallow-latest-tag
spec:
  validationFailureAction: Audit
  background: true
  rules:
  - name: require-and-validate-image-tag
    match:
      any:
      - resources:
          kinds:
          - Pod
          operations:
          - CREATE
          - UPDATE
    validate:
      cel:
        expressions:
          - expression: "object.spec.containers.all(c, c.image.contains(':'))"
            message: "An image tag is required."
          - expression: "object.spec.containers.all(c, !c.image.endsWith(':latest'))"
            message: "Using a mutable image tag e.g. 'latest' is not allowed."
```

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: check-deployment-replicas
spec:
  validationFailureAction: Enforce
  background: false
  rules:
    - name: deployment-replicas
      match:
        any:
        - resources:
            kinds:
              - Deployment
      validate:
        cel:
          paramKind:
            apiVersion: v1
            kind: ConfigMap
          paramRef:
            name: replica-limit
            parameterNotFoundAction: "Deny"
          expressions:
            - expression: "object.spec.replicas <= int(params.data.maxReplicas)"
              messageExpression:  "'Deployment spec.replicas must be less than ' + string(params.data.maxReplicas)"
```

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: replica-limit
data:
  maxReplicas: 3
```

KUBERMATIC

Quick Demo

What actually we gain from CEL?

CEL Libraries

Extended CEL functions for complex policy logic and advanced features

⚠ **FEATURE STATE: Alpha** Kyverno v1.15

Kyverno enhances Kubernetes' CEL environment with libraries enabling complex policy logic and advanced features. These libraries are available in both ValidatingPolicy and MutatingPolicy.

Resource library

The **Resource library** provides functions like `resource.Get()` and `resource.List()` to retrieve Kubernetes resources from the cluster, either individually or as a list. These are useful for writing policies that depend on the state of other resources, such as checking existing ConfigMaps, Services, or Deployments before validating or mutating a new object.

CEL Expression	Purpose
`resource.Get("v1", "configmaps", "default", "clusterregistries").data["registries"]`	Fetch a ConfigMap value from a specific namespace
`resource.List("apps/v1", "deployments", "").items.size() > 0`	Check if there are any Deployments across all namespaces
`resource.Post("authorization.k8s.io/v1", "subjectaccessreviews", {…})`	Perform a live SubjectAccessReview (authz check) against the Kubernetes API
`resource.List("apps/v1", "deployments", object.metadata.namespace).items.exists(d, d.spec.replicas > 3)`	Ensure at least one Deployment in the same namespace has more than 3 replicas

```yaml
apiVersion: policies.kyverno.io/v1alpha1
kind: ValidatingPolicy
metadata:
  name: unique-ingress-host
spec:
  validationActions:
    - Deny
  evaluation:
    background:
      enabled: false
  matchConstraints:
    resourceRules:
      - apiGroups: ["networking.k8s.io"]
        apiVersions: ["v1"]
        operations: ["CREATE", "UPDATE"]
        resources: ["ingresses"]
  variables:
    - name: knownIngresses
      expression: resource.List("networking.k8s.io/v1", "ingresses", "").items.orValue([])
    - name: knownHosts
      expression: "variables.knownIngresses.map(i, i.spec.rules.map(r, r.host))"
    - name: desiredHosts
      expression: "object.spec.rules.map(r, r.host)"
  validations:
    - expression: "!variables.knownHosts.exists_one(hosts, sets.intersects(hosts, variables.desiredHosts))"
      message: "Cannot reuse a host across multiple ingresses"
```

```yaml
apiVersion: v1
items:
  - apiVersion: networking.k8s.io/v1
    kind: Ingress
    metadata:
      name: ingress-1
      namespace: default
    spec:
      rules:
        - host: foo.com
          http:
            paths:
              - backend:
                  service:
                    name: service1
                    port:
                      number: 80
                path: /foo
                pathType: Prefix
    status:
      loadBalancer: {}
kind: List
metadata:
  resourceVersion: ""
```

Conclusion

- CEL enables dynamic, context-aware policies
- Less repetition, more control
- Production-ready logic in YAML-native form
- Keep expressions simple and readable

THANK YOU!


koray@kubermatic.com


@koksay


linkedin.com/in/korayoksay