June 20 | Bologna | Track

Lang: 🇮🇹

How to leverage and extend CEL for your cluster security

Flavio Castelli

@flavio_castelli

Distinguished Engineer
SUSE

ITALY 2024 KCD
GROWING CLOUD NATIVE TOGETHER

Common Expression Language (CEL)

- Lightweight expression evaluation
- Memory safe
- Strong and dynamically typed
- Side-effect free
- Immutable

CEL syntax

```
"ghcr.io/flavio/kcd:latest".startsWith("ghcr.io/")

[
  "ghcr.io/flavio/foo:1",
  "ghcr.io/flavio/bar:1",
].all(img, img.startsWith("ghcr.io"))
```

CEL syntax

```
{
    "name": "test",
    "image": "ghcr.io/flavio/test",
}.image.startsWith("ghcr.io")
```

CEL syntax

```
[
    {
        "name": "foo",
        "image": "ghcr.io/flavio/foo",
    },
    {
        "name": "bar",
        "image": "ghcr.io/flavio/bar",
    },
].map(c, c.image).all(img, img.startsWith("ghcr.io"))
```

🍕 Kubernetes 💙 CEL 🍕

CEL adoption inside of Kubernetes

- CRD Validation Rules: since Kubernetes 1.25
- `matchConditions` attribute of Dynamic Admission Controllers
- `ValidatingAdmissionPolicy`

Kubernetes Admission Controllers

Use cases

- Security:
    - Deny privileged containers
    - Drop Linux capabilities
- Compliance:
    - Deny images from Docker Hub
    - Require readiness probes
- Efficiency:
    - Require memory/CPU limits

Types of admission controllers

- Validating
- Mutating

Admission Controllers bundled into Kubernetes

- AlwaysPullImages
- DefaultIngressClass
- DefaultStorageClass
- LimitRanger
- PodSecurity
- ...

Bring custom admission rules

- ValidatingAdmissionWebhook
- MutatingAdmissionWebhook
- ValidatingAdmissionPolicy

NEW

Kubernetes Policy Engines

- Focus: write policy business logic
- Policy as a Service platforms
- CNCF projects operating in this space:
    - Gatekeeper
    - Kubewarden
    - Kyverno

Admission Webhooks vs built-in controllers

- Pros:
  - Write custom rules
  - Access external data sources
  - No computation done by the Kubernetes API server
- Cons:
  - Uncertainty
  - Latency

Validating Admission Policy (VAP)

- New admission controller **built into** Kubernetes
  - Available since 1.26 (alpha)
  - Stable since 1.30
- Write custom rules using CEL language

A tour of Validating Admission Policy

ValidatingAdmissionPolicy CR

```
apiVersion: admissionregistration.k8s.io/v1
kind: ValidatingAdmissionPolicy
metadata:
  name: "demo.kcd.it"
spec:
  failurePolicy: Fail
  matchConstraints:
    resourceRules:
    - apiGroups:   ["apps"]
      apiVersions: ["v1"]
      operations:  ["CREATE", "UPDATE"]
      resources:   ["deployments"]
  validations:
  - expression: "object.spec.replicas <= 5"
    message: "cannot have more than 5 replicas"
```

Target resources

CEL code

ValidatingAdmissionPolicyBinding CR

```yaml
apiVersion: admissionregistration.k8s.io/v1
kind: ValidatingAdmissionPolicyBinding
metadata:
  name: "demo.kcd.it"
spec:
  policyName: "demo.kcd.it"
  validationActions: [Deny]
  matchResources:
    namespaceSelector:
      matchLabels:
        kubernetes.io/metadata.name: kcd-demo
```

What to do on violation

Where the policy is enforced

Enforce livenessProbe

```yaml
apiVersion: admissionregistration.k8s.io/v1alpha1
kind: ValidatingAdmissionPolicy
metadata:
  name: "force-liveness-probe"
spec:
  failurePolicy: Fail
  matchConstraints:
    resourceRules:
    - apiGroups:   ["apps"]
      apiVersions: ["v1"]
      operations:  ["CREATE", "UPDATE"]
      resources:   ["deployments"]
  validations:
    - expression: |
        object.spec.template.spec.containers.all(c, has(c.livenessProbe))
      message: "all the containers must have a livenessProbe defined"
```

Improve rejection message

```yaml
validations:
- expression: |
     object.spec.template.spec.containers.all(c, has(c.livenessProbe))
  messageExpression: |
     'These containers are missing a liveness probe: ' +
     object.spec.template.spec.containers
          .filter(c, !has(c.livenessProbe))
          .map(c, c.name).join(', ')
  reason: Invalid
```

Optional, sets the HTTP response code

Keep the code DRY with variables

```yaml
apiVersion: admissionregistration.k8s.io/v1
kind: ValidatingAdmissionPolicy
metadata:
  name: "force-liveness-probe"
spec:
  failurePolicy: Fail
  variables:
  - name: ctrs_no_liveness_probe
    expression: |
     object.spec.template.spec.containers.filter(c, !has(c.livenessProbe)).map(c, c.name)
  matchConstraints:
    resourceRules:
    - apiGroups: ["apps"]
      apiVersions: ["v1"]
      operations: ["CREATE", "UPDATE"]
      resources: ["deployments"]
  validations:
  - expression: "size(variables.ctrs_no_liveness_probe) == 0"
    messageExpression: |
     'These containers are missing a liveness probe: ' +
     variables.ctrs_no_liveness_probe.join(', ')
```

Add multiple validation rules

```
apiVersion: admissionregistration.k8s.io/v1
kind: ValidatingAdmissionPolicy
metadata:
  name: "demo.kcd.it"
spec:
  failurePolicy: Fail
  matchConstraints:
    resourceRules:
    - apiGroups:   ["apps"]
      apiVersions: ["v1"]
      operations:  ["CREATE", "UPDATE"]
      resources:   ["deployments"]
  validations:
  - expression: "object.spec.replicas <= 10"
    message: "cannot have more than 10 replicas"
```

Add multiple validation rules

```yaml
validations:
  - expression: "object.spec.replicas > 2"
    message: "should have at least 2 replicas"
  - expression: "object.spec.replicas <= 10"
    message: "should have at most 10 replicas"
  - expression: "object.spec.replicas % 2 != 0"
    message: "should have an odd number of replicas"
```

Remove hardcoded values

```
validations:
  - expression: "object.spec.replicas > 2"
    message: "should have at least 2 replicas"
  - expression: "object.spec.replicas <= 10"
    message: "should have at most 10 replicas"
  - expression: "object.spec.replicas % 2 != 0"
    message: "should have an odd number of replicas"
```

Store settings somewhere

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: vap-replicasize-params
  namespace: default
data:
  maxReplicas: "10"
  minReplicas: "2"
```

Use “params”

```yaml
apiVersion: admissionregistration.k8s.io/v1
kind: ValidatingAdmissionPolicy
metadata:
  name: "demo.kcd.it"
spec:
  failurePolicy: Fail
  paramKind:
      apiVersion: v1
      kind: ConfigMap
  matchConstraints:
      resourceRules:
      - apiGroups: ["apps"]
        apiVersions: ["v1"]
      operations: ["CREATE", "UPDATE"]
      resources: ["deployments", "deployments/scale"]
  validations:
      - expression: "object.spec.replicas > int(params.data.minReplicas)"
        messageExpression: "'should have at least ' + params.data.minReplicas + ' replicas'"
      - expression: "object.spec.replicas <= int(params.data.maxReplicas)"
        messageExpression: "'should have at most ' + params.data.maxReplicas + ' replicas'"
      - expression: "object.spec.replicas % 2 != 0"
        message: "should have an odd number of replicas"
```

Bind policy and params together

```yaml
apiVersion: admissionregistration.k8s.io/v1
kind: ValidatingAdmissionPolicyBinding
metadata:
  name: "demo.kcd.it"
spec:
  policyName: "demo.kcd.it"
  validationActions: [Deny]
  paramRef:
     name: vap-replicasize-params
     namespace: default
     parameterNotFoundAction: Deny
  matchResources:
    namespaceSelector:
      matchLabels:
        kubernetes.io/metadata.name: default
```

Where to look for the parameters

Bind the policies to multiple namespaces

```yaml
apiVersion: admissionregistration.k8s.io/v1
kind: ValidatingAdmissionPolicyBinding
metadata:
  name: "demo.kcd.it"
spec:
  policyName: "demo.kcd.it"
  validationActions: [Deny]
  paramRef:
     name: vap-replicasize-params
     parameterNotFoundAction: Deny
  matchResources:
    namespaceSelector:
      matchLabels:
        cel.kcd.it/replica-size: enabled
```

Namespace: no longer defined

More generic match rule

Dynamic Admission Controllers and CEL

Kubewarden in a nutshell

- CNCF project
- Policies:
  - Written using WebAssembly (Go, Rust, Rego, …)
  - Distributed as OCI artifacts

Kubewarden's CEL policy

- Share VAP foundations
- No need to change VAP policies

From a VAP policy…

```yaml
apiVersion: admissionregistration.k8s.io/v1
kind: ValidatingAdmissionPolicy
metadata:
  name: "demo.kcd.it"
spec:
  failurePolicy: Fail
  matchConstraints:
    resourceRules:
    - apiGroups:   ["apps"]
      apiVersions: ["v1"]
      operations:  ["CREATE", "UPDATE"]
      resources:   ["deployments"]
  validations:
  - expression: "object.spec.replicas <= 5"
    message: "cannot have more than 5 replicas"
```

... into a Kubewarden policy

```yaml
apiVersion: policies.kubewarden.io/v1
kind: ClusterAdmissionPolicy
metadata:
  name: "demo.kcd.it"
spec:
  failurePolicy: Fail
  rules:
  - apiGroups:   ["apps"]
    apiVersions: ["v1"]
    operations:  ["CREATE", "UPDATE"]
    resources:   ["deployments"]
  module: ghcr.io/kubewarden/policies/cel-policy:v1.0.0
  settings:
    validations:
    - expression: "object.spec.replicas <= 5"
      message: "cannot have more than 5 replicas"
```

Kubewarden extends VAP

- Context aware policy
- Sigstore integration
- Primitives to interact with:
  - x509 certificate
  - Network
  - OCI registries

Unique Ingress policy

- Prevent the creation of Ingress resources with duplicated hosts
- Must access Kubernetes to obtain information

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: example-ingress
spec:
  rules:
  - host: example.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: example-service
            port:
              number: 80
```

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: not-valid-ingress
spec:
  rules:
  - host: example.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: foobar
            port:
              number: 80
```

Should not be created

KW CEL extension: query Kubernetes - pt1

```yaml
apiVersion: policies.kubewarden.io/v1
kind: ClusterAdmissionPolicy
metadata:
  name: "unique-ingress"
spec:
  module: ghcr.io/kubewarden/policies/cel-policy
  failurePolicy: Fail
  rules:
    - apiGroups: ["networking.k8s.io"]
      apiVersions: ["v1"]
      resources: ["ingresses"]
      operations: ["CREATE", "UPDATE"]
  contextAwareResources:
    - apiVersion: networking.k8s.io/v1
      kind: Ingress
  settings:
    variables:
      - name: knownIngresses
        expression: "kw.k8s.apiVersion('networking.k8s.io/v1').kind('Ingress').list().items"
      - 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"
```

Grant read access to Ingress resources

KW CEL extension: query Kubernetes - pt2

```
variables:
- name: knownIngresses
  expression: |
    kw.k8s.apiVersion('networking.k8s.io/v1').kind('Ingress').list().items
- 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"
```

```
[{ ingress1 }, {}, ...]
```

```
[["host1","h2"],["h3"],…]
```

```
["host1","host2"]
```

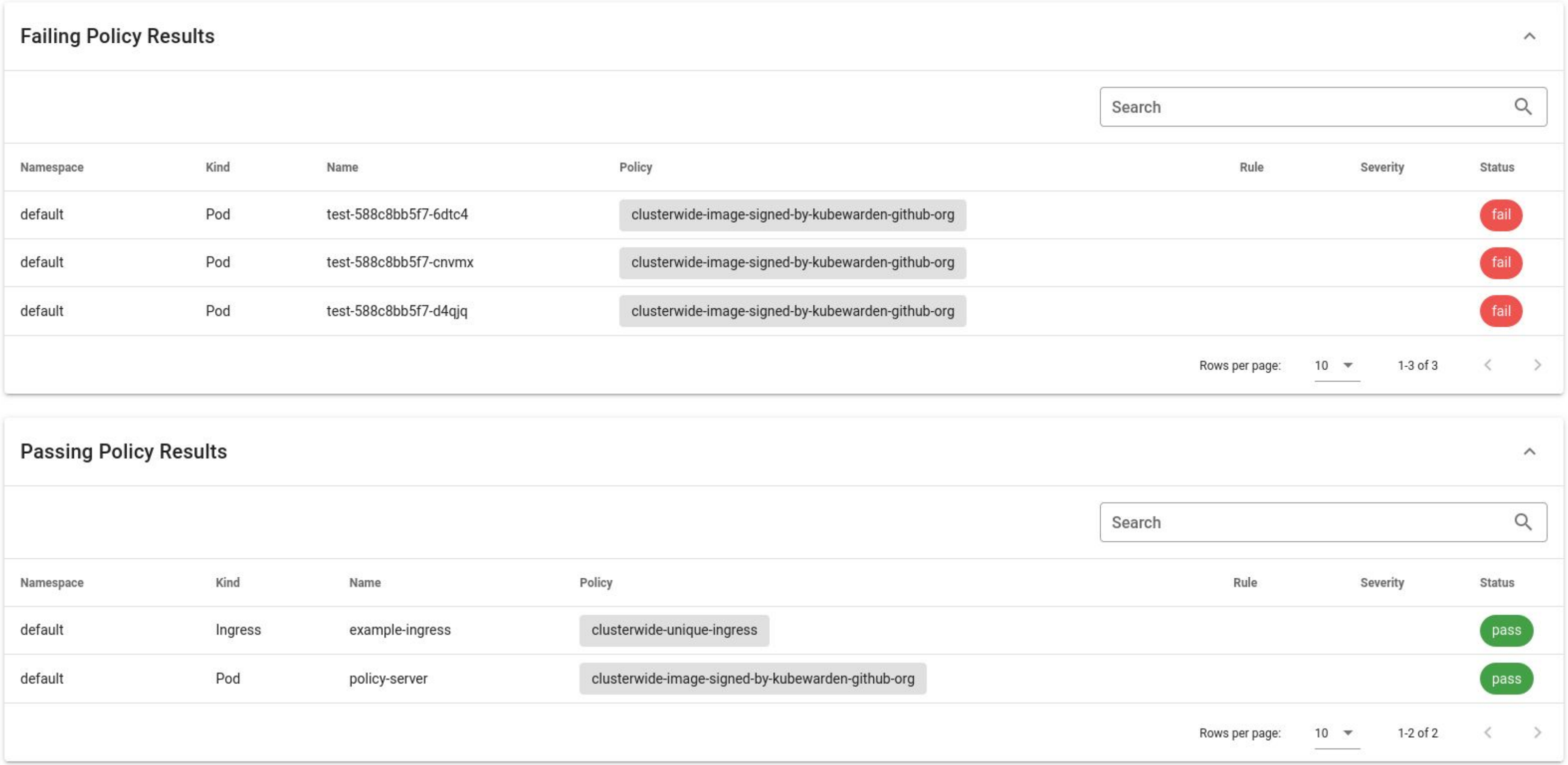
KW CEL extension: sigstore - pt 1

```yaml
apiVersion: policies.kubewarden.io/v1
kind: ClusterAdmissionPolicy
metadata:
  name: "image-signed-by-kubewarden-github-org"
spec:
  module: ghcr.io/kubewarden/policies/cel-policy
  namespaceSelector:
    matchLabels:
      kubernetes.io/metadata.name: default
  rules:
  - apiGroups: [""]
    apiVersions: ["v1"]
    resources: ["pods"]
    operations: ["CREATE", "UPDATE"]
```

KW CEL extension: sigstore - pt 2

```
settings:
  variables:
  - name: containerImages
    expression: "object.spec.containers.map(c, c.image)"
  - name: containerImagesNotVerified
    expression: |
      variables.containerImages.filter(
        image,
        !kw.sigstore.image(image).githubAction("kubewarden").verify().isTrusted()
      )
  validations:
    - expression: "size(variables.containerImagesNotVerified) == 0"
      messageExpression: "'These container images are not signed by the kubewarden GitHub organization: ' +
variables.containerImagesNotVerified.join(', ')"
```

Compliance report of CEL policies



Failing Policy Results

Search

Namespace	Kind	Name	Policy	Rule	Severity	Status
default	Pod	test-588c8bb5f7-6dtc4	clusterwide-image-signed-by-kubewarden-github-org			fail
default	Pod	test-588c8bb5f7-cnvmx	clusterwide-image-signed-by-kubewarden-github-org			fail
default	Pod	test-588c8bb5f7-d4qjq	clusterwide-image-signed-by-kubewarden-github-org			fail

Rows per page: 10 1-3 of 3

Passing Policy Results

Search

Namespace	Kind	Name	Policy	Rule	Severity	Status
default	Ingress	example-ingress	clusterwide-unique-ingress			pass
default	Pod	policy-server	clusterwide-image-signed-by-kubewarden-github-org			pass

Rows per page: 10 1-2 of 2

Conclusions: CEL

- Pros:
  - Easy to learn
  - Actively developed
  - “Limited” language
- Cons:
  - Documentation should be improved
  - Information is scattered
  - Core language is missing some functionalities
  - Testing story should be improved

Conclusions: VAP

- Pros:
  - Built into Kubernetes
  - Stable feature since 1.30
  - Write your own admission controller
- Cons:
  - Only validation, no mutation
  - No access to external data source
  - No way to find non-compliant resources already inside of the cluster
  - Testing story should be improved

Conclusions: Kubewarden and CEL

- Pros:
    - 1:1 mapping with VAP
    - Brings new capabilities to CEL
    - Leverage Kubewarden ecosystem: tracing, monitoring, compliance report, testing, ...
    - Conversion and testing tooling
- Cons:
    - Runs outside of the API server
    - Only validation, no mutation

Learning resources

- CEL website
- CEL-go extensions
- VAP official docs
- Kubernetes changes to CEL: more extensions, runtime cost budget
- CEL playground
- Kubewarden and CEL
- Kubewarden’s CEL policy

ITALY 2024

KCD

GROWING CLOUD NATIVE TOGETHER

kcd.pizza