KubeCon | CloudNativeCon

North America 2024

KubeCon | CloudNativeCon

North America 2024

CEL-Ebrating Simplicity: Mastering Kubernetes Policy Enforcement

Speakers

Kevin Conner

Chief Engineer, Getup Cloud

@knrc

Anish Ramasekar

Software Engineer, Microsoft

@aramase

Agenda

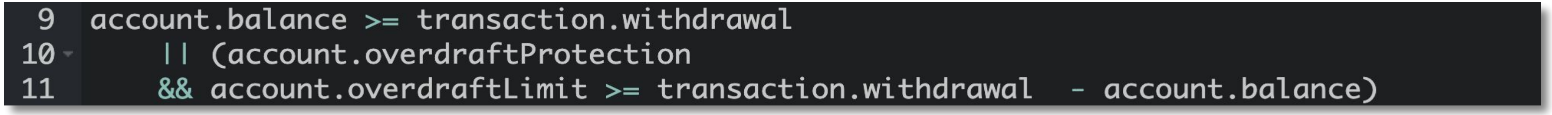
- Common Expression Language (CEL)
- CEL Playground
- Validating Admission Policies (VAPs)
- VAPs in Action
- Mutating Admission Policies (MAPs)
- MAPs in Action
- Questions

Agenda

- Common Expression Language (CEL)
- CEL Playground
- Validating Admission Policies (VAPs)
- VAPs in Action
- Mutating Admission Policies (MAPs)
- MAPs in Action
- Questions

What is CEL?

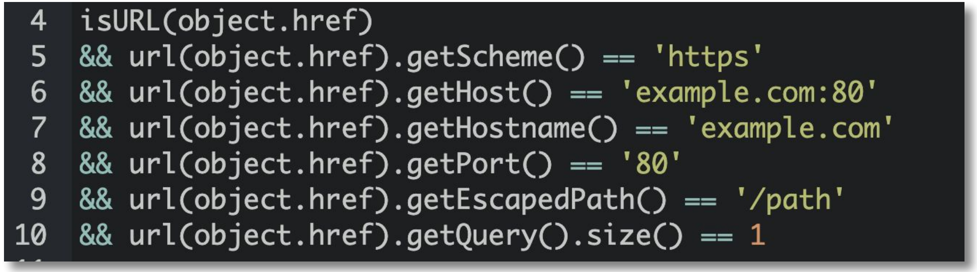
- Developed by Google
- Non-Turing complete
- Small, Fast & Predictable
- Embeddable
- Extensible
- Constrained Resource Usage



```
account.balance >= transaction.withdrawal
    || (account.overdraftProtection
    && account.overdraftLimit >= transaction.withdrawal  - account.balance)
```

What is CEL?

- Familiar language (C/Java/Go)
- Rich libraries
    - URL Manipulation
    - Quantities
    - Regex
    - Authorizer
    - etc.



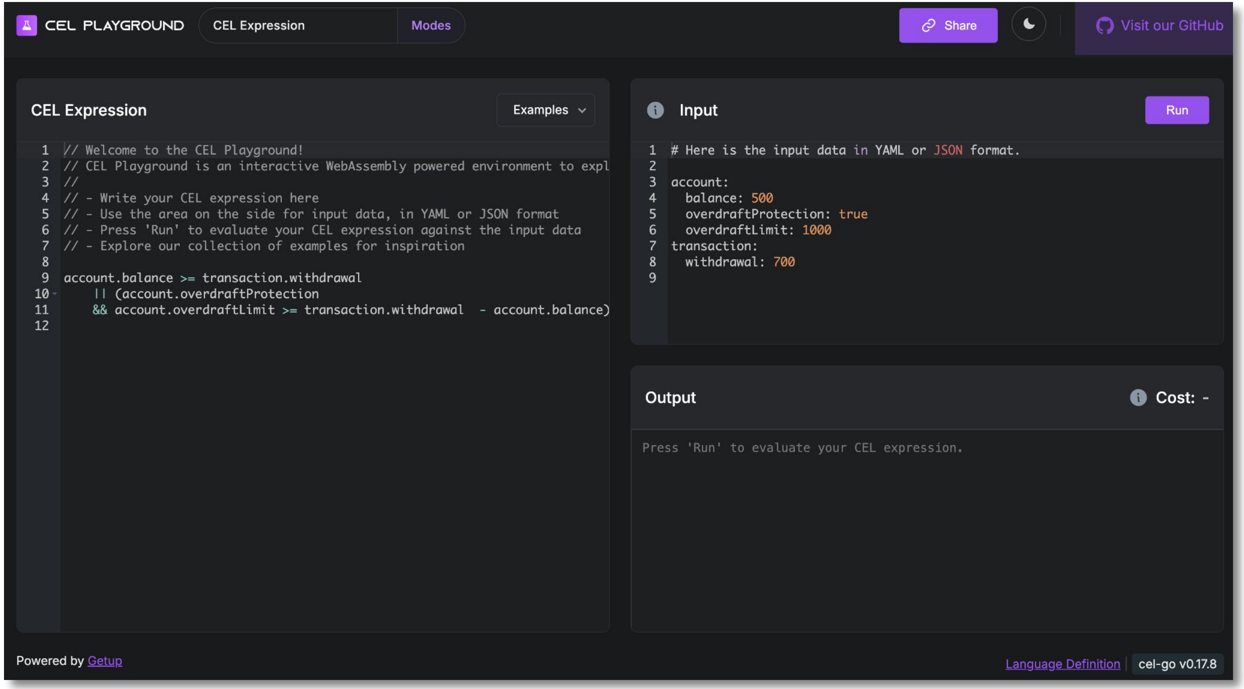
```
isURL(object.href)
&& url(object.href).getScheme() == 'https'
&& url(object.href).getHost() == 'example.com:80'
&& url(object.href).getHostname() == 'example.com'
&& url(object.href).getPort() == '80'
&& url(object.href).getEscapedPath() == '/path'
&& url(object.href).getQuery().size() == 1
```

Agenda

- Common Expression Language (CEL)
- ➡ CEL Playground
- Validating Admission Policies (VAPs)
- VAPs in Action
- Mutating Admission Policies (MAPs)
- MAPs in Action
- Questions

CEL Playground

playcel.undistro.io



CEL Expression

```
// Welcome to the CEL Playground!
// CEL Playground is an interactive WebAssembly powered environment to expl
//
// - Write your CEL expression here
// - Use the area on the side for input data, in YAML or JSON format
// - Press 'Run' to evaluate your CEL expression against the input data
// - Explore our collection of examples for inspiration

account.balance >= transaction.withdrawal
    || (account.overdraftProtection
    && account.overdraftLimit >= transaction.withdrawal  - account.balance)

```

Input

```
# Here is the input data in YAML or JSON format.

account:
  balance: 500
  overdraftProtection: true
  overdraftLimit: 1000
transaction:
  withdrawal: 700

```

Output

```
Press 'Run' to evaluate your CEL expression.
```

Powered by Getup

Language Definition | cel-go v0.17.8

CEL Playground

- Safe space for playing with CEL
- WASM application
- CEL support for
    - Plain CEL Expressions
    - Web Hooks
    - Validating Admission Policies

CEL Playground

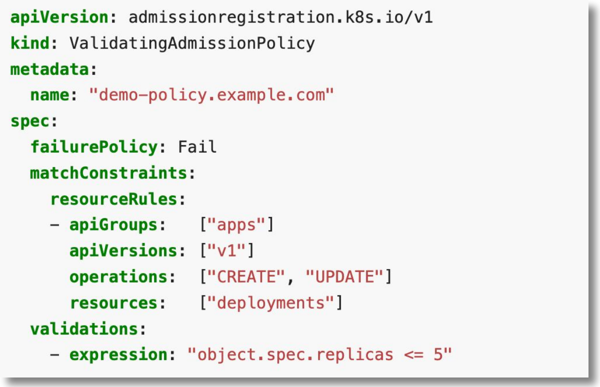
Demo Time

Agenda

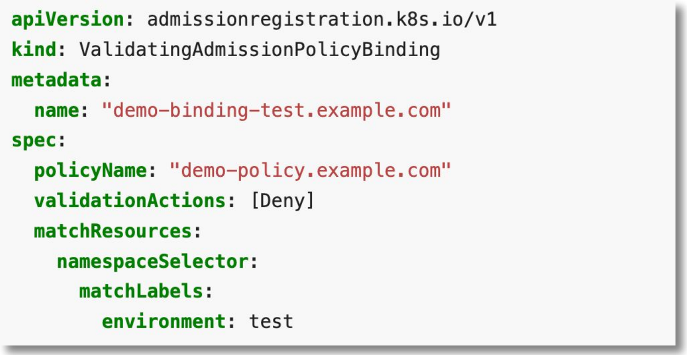
- Common Expression Language (CEL)
- CEL Playground
- ➡ Validating Admission Policies (VAPs)
- VAPs in Action
- Mutating Admission Policies (MAPs)
- MAPs in Action
- Questions

Validating Admission Policies

- Replace Validating Webhooks for many use cases
- Run in kube-apiserver
- Consist of
  - ValidatingAdmissionPolicy
  - ValidatingAdmissionPolicyBinding
  - Parameter Resources
- Can Deny, Warn or Audit



```yaml
apiVersion: admissionregistration.k8s.io/v1
kind: ValidatingAdmissionPolicy
metadata:
  name: "demo-policy.example.com"
spec:
  failurePolicy: Fail
  matchConstraints:
    resourceRules:
    - apiGroups:   ["apps"]
      apiVersions: ["v1"]
      operations:  ["CREATE", "UPDATE"]
      resources:   ["deployments"]
  validations:
    - expression: "object.spec.replicas <= 5"
```



```yaml
apiVersion: admissionregistration.k8s.io/v1
kind: ValidatingAdmissionPolicyBinding
metadata:
  name: "demo-binding-test.example.com"
spec:
  policyName: "demo-policy.example.com"
  validationActions: [Deny]
  matchResources:
    namespaceSelector:
      matchLabels:
        environment: test
```

Validating Admission Policies

- CEL Expressions used in
  - matchConditions
  - variables
  - validations
  - auditAnnotations

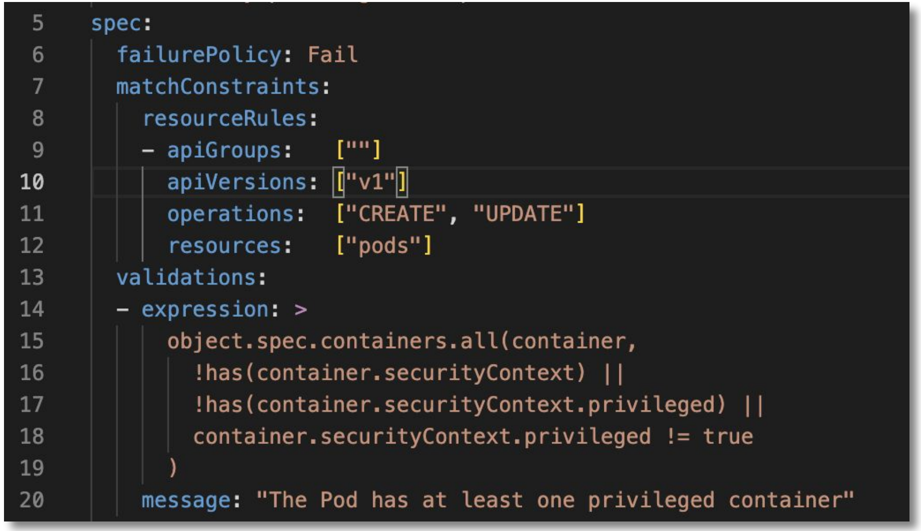
- CEL expressions have access to
  - ‘object’
  - ‘oldObject’
  - ‘request’
  - ‘params’
  - ‘namespaceObject’
  - ‘variables’
  - ‘authorizer’
  - ‘authorizer.requestResource’

Agenda

- Common Expression Language (CEL)
- CEL Playground
- Validating Admission Policies (VAPs)
- VAPs in Action
- Mutating Admission Policies (MAPs)
- MAPs in Action
- Questions

First We Crawl

- Validate pods
- Prevent privileged containers

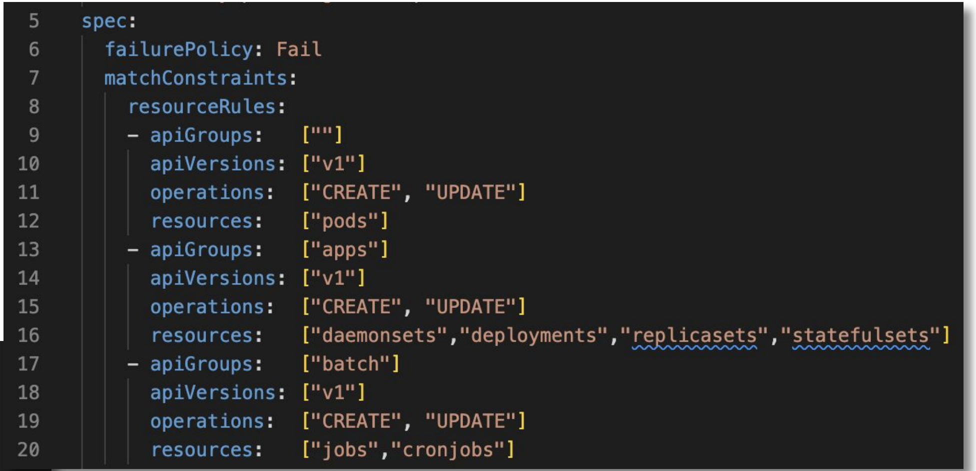


```
spec:
  failurePolicy: Fail
  matchConstraints:
    resourceRules:
    - apiGroups:   [""]
      apiVersions: ["v1"]
      operations:  ["CREATE", "UPDATE"]
      resources:   ["pods"]
  validations:
  - expression: >
      object.spec.containers.all(container,
        !has(container.securityContext) ||
        !has(container.securityContext.privileged) ||
        container.securityContext.privileged != true
      )
    message: "The Pod has at least one privileged container"
```

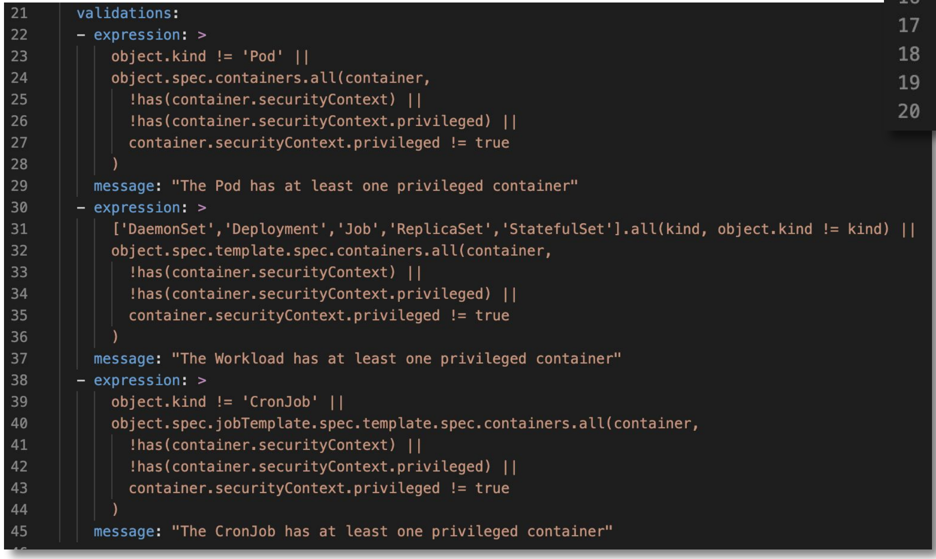
Then We Walk

KubeCon CloudNativeCon North America 2024

- Expand to other resources



```
spec:
  failurePolicy: Fail
  matchConstraints:
    resourceRules:
    - apiGroups:   [""]
      apiVersions: ["v1"]
      operations:  ["CREATE", "UPDATE"]
      resources:   ["pods"]
    - apiGroups:   ["apps"]
      apiVersions: ["v1"]
      operations:  ["CREATE", "UPDATE"]
      resources:   ["daemonsets","deployments","replicasets","statefulsets"]
    - apiGroups:   ["batch"]
      apiVersions: ["v1"]
      operations:  ["CREATE", "UPDATE"]
      resources:   ["jobs","cronjobs"]
```

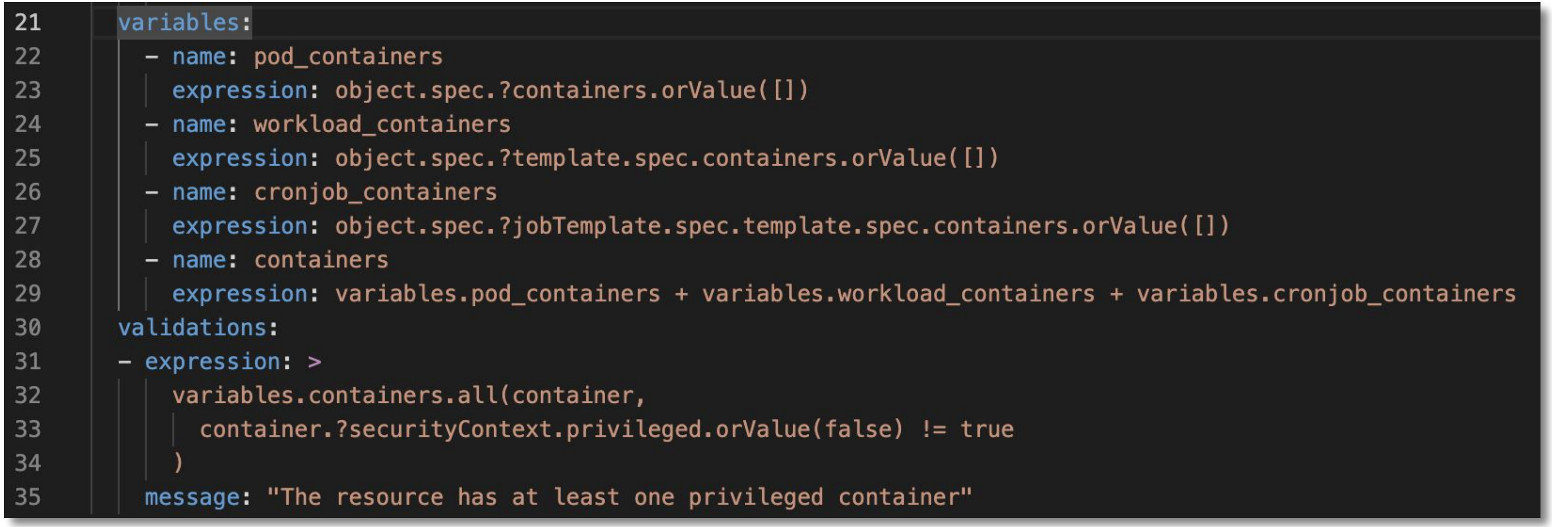


```
validations:
- expression: >
    object.kind != 'Pod' ||
    object.spec.containers.all(container,
      !has(container.securityContext) ||
      !has(container.securityContext.privileged) ||
      container.securityContext.privileged != true
    )
  message: "The Pod has at least one privileged container"
- expression: >
    ['DaemonSet','Deployment','Job','ReplicaSet','StatefulSet'].all(kind, object.kind != kind) ||
    object.spec.template.spec.containers.all(container,
      !has(container.securityContext) ||
      !has(container.securityContext.privileged) ||
      container.securityContext.privileged != true
    )
  message: "The Workload has at least one privileged container"
- expression: >
    object.kind != 'CronJob' ||
    object.spec.jobTemplate.spec.template.spec.containers.all(container,
      !has(container.securityContext) ||
      !has(container.securityContext.privileged) ||
      container.securityContext.privileged != true
    )
  message: "The CronJob has at least one privileged container"
```

- Expand validations
- Use Guards

Then We Walk

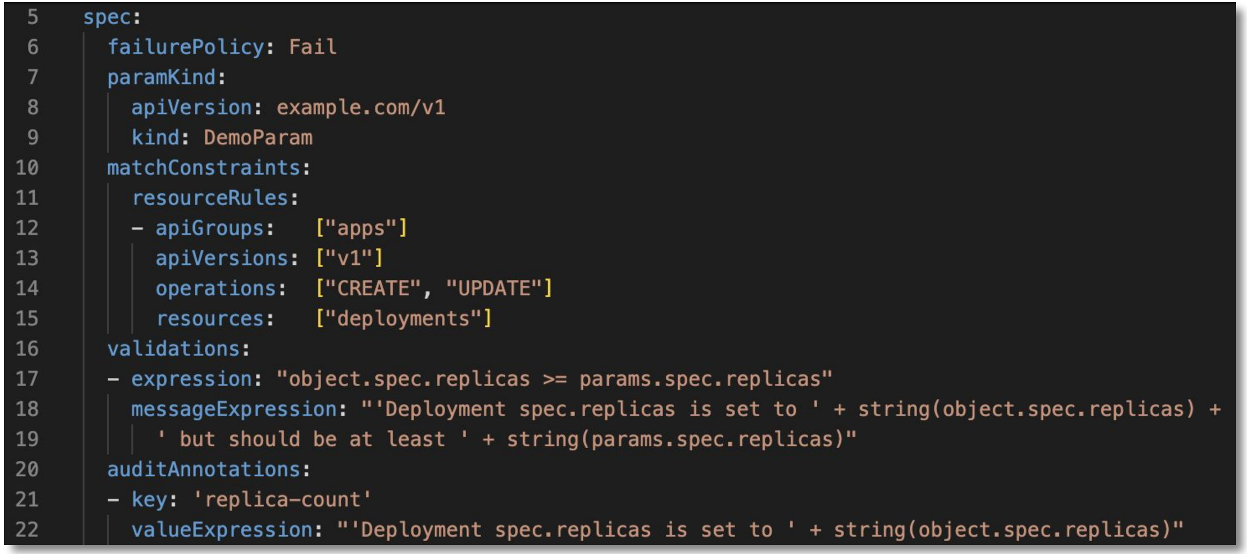
- Use optional operator! .? and **orValue(...)**
- Use variables!



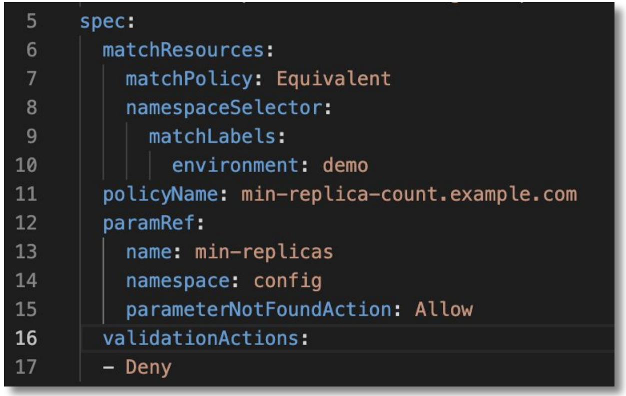
```
variables:
  - name: pod_containers
    expression: object.spec.?containers.orValue([])
  - name: workload_containers
    expression: object.spec.?template.spec.containers.orValue([])
  - name: cronjob_containers
    expression: object.spec.?jobTemplate.spec.template.spec.containers.orValue([])
  - name: containers
    expression: variables.pod_containers + variables.workload_containers + variables.cronjob_containers
validations:
- expression: >
    variables.containers.all(container,
      container.?securityContext.privileged.orValue(false) != true
    )
  message: "The resource has at least one privileged container"
```

Next We Run

- Time for a Parameter



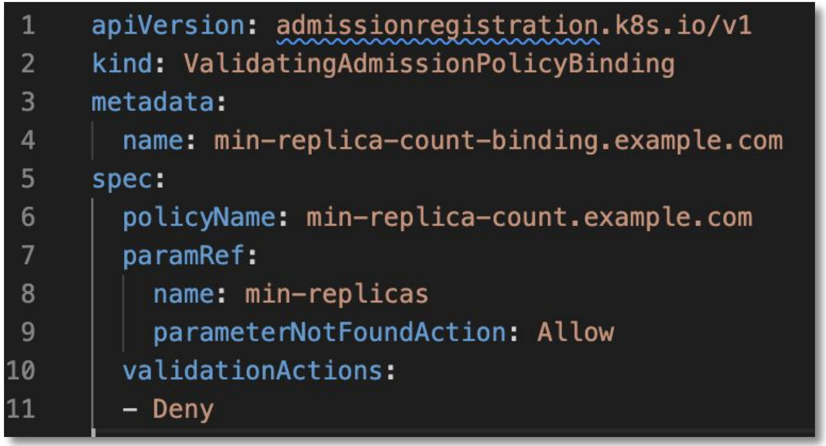
```yaml
spec:
  failurePolicy: Fail
  paramKind:
    apiVersion: example.com/v1
    kind: DemoParam
  matchConstraints:
    resourceRules:
    - apiGroups:   ["apps"]
      apiVersions: ["v1"]
      operations:  ["CREATE", "UPDATE"]
      resources:   ["deployments"]
  validations:
  - expression: "object.spec.replicas >= params.spec.replicas"
    messageExpression: "'Deployment spec.replicas is set to ' + string(object.spec.replicas) +
      ' but should be at least ' + string(params.spec.replicas)"
  auditAnnotations:
  - key: 'replica-count'
    valueExpression: "'Deployment spec.replicas is set to ' + string(object.spec.replicas)"
```



```yaml
spec:
  matchResources:
    matchPolicy: Equivalent
    namespaceSelector:
      matchLabels:
        environment: demo
  policyName: min-replica-count.example.com
  paramRef:
    name: min-replicas
    namespace: config
    parameterNotFoundAction: Allow
  validationActions:
  - Deny
```

… and Fly

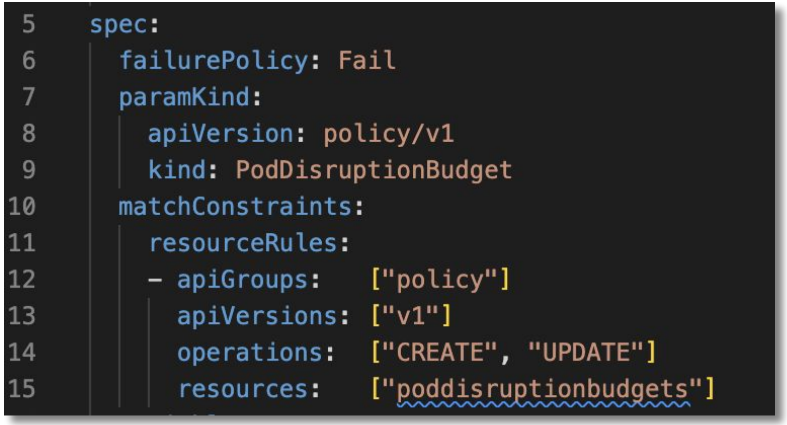
- Parameter per Namespace!



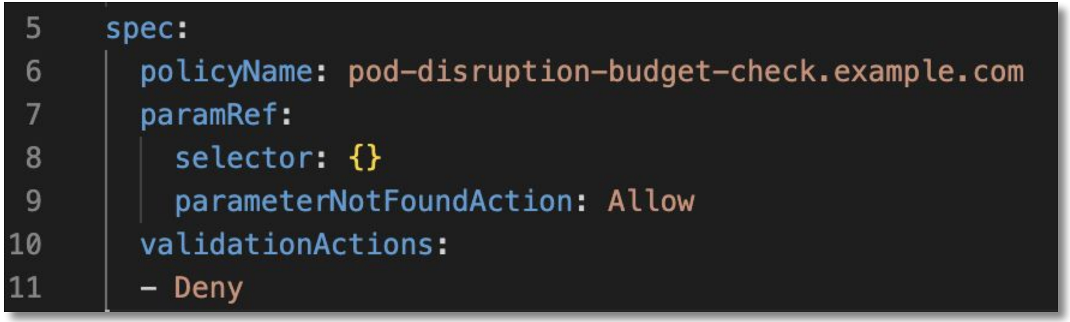
```yaml
apiVersion: admissionregistration.k8s.io/v1
kind: ValidatingAdmissionPolicyBinding
metadata:
  name: min-replica-count-binding.example.com
spec:
  policyName: min-replica-count.example.com
  paramRef:
    name: min-replicas
    parameterNotFoundAction: Allow
  validationActions:
  - Deny
```

Aim for the Moon

- Why use **One** when you can use **Many**



```yaml
spec:
  failurePolicy: Fail
  paramKind:
    apiVersion: policy/v1
    kind: PodDisruptionBudget
  matchConstraints:
    resourceRules:
    - apiGroups:   ["policy"]
      apiVersions: ["v1"]
      operations:  ["CREATE", "UPDATE"]
      resources:   ["poddisruptionbudgets"]
```



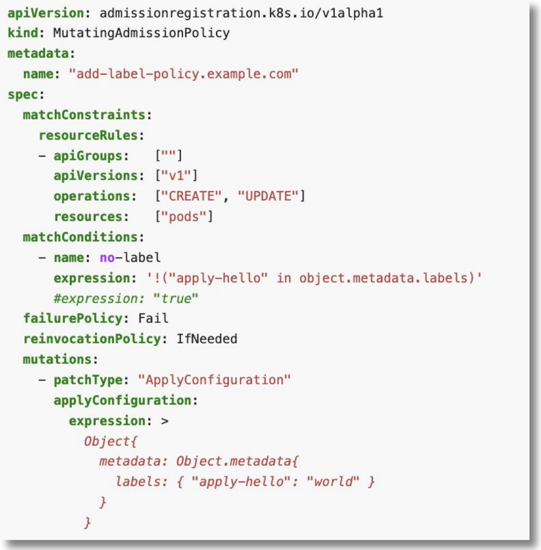
```yaml
spec:
  policyName: pod-disruption-budget-check.example.com
  paramRef:
    selector: {}
    parameterNotFoundAction: Allow
  validationActions:
  - Deny
```

Agenda

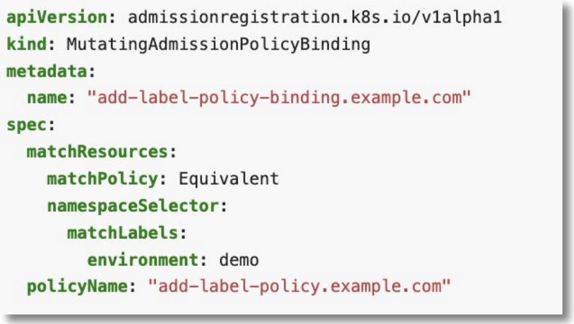
- Common Expression Language (CEL)
- CEL Playground
- Validating Admission Policies (VAPs)
- VAPs in Action
- ➡ Mutating Admission Policies (MAPs)
- MAPs in Action
- Questions

Mutating Admission Policies

- Replace Mutating Webhooks for many use cases
- Run in kube-apiserver
- Consist of
  - MutatingAdmissionPolicy
  - MutatingAdmissionPolicyBinding
  - Parameter Resources
- Generates
  - ApplyConfiguration
  - JSONPatch



```yaml
apiVersion: admissionregistration.k8s.io/v1alpha1
kind: MutatingAdmissionPolicy
metadata:
  name: "add-label-policy.example.com"
spec:
  matchConstraints:
    resourceRules:
    - apiGroups:   [""]
      apiVersions: ["v1"]
      operations:  ["CREATE", "UPDATE"]
      resources:   ["pods"]
  matchConditions:
    - name: no-label
      expression: '!("apply-hello" in object.metadata.labels)'
      #expression: "true"
  failurePolicy: Fail
  reinvocationPolicy: IfNeeded
  mutations:
    - patchType: "ApplyConfiguration"
      applyConfiguration:
        expression: >
          Object{
            metadata: Object.metadata{
              labels: { "apply-hello": "world" }
            }
          }
```



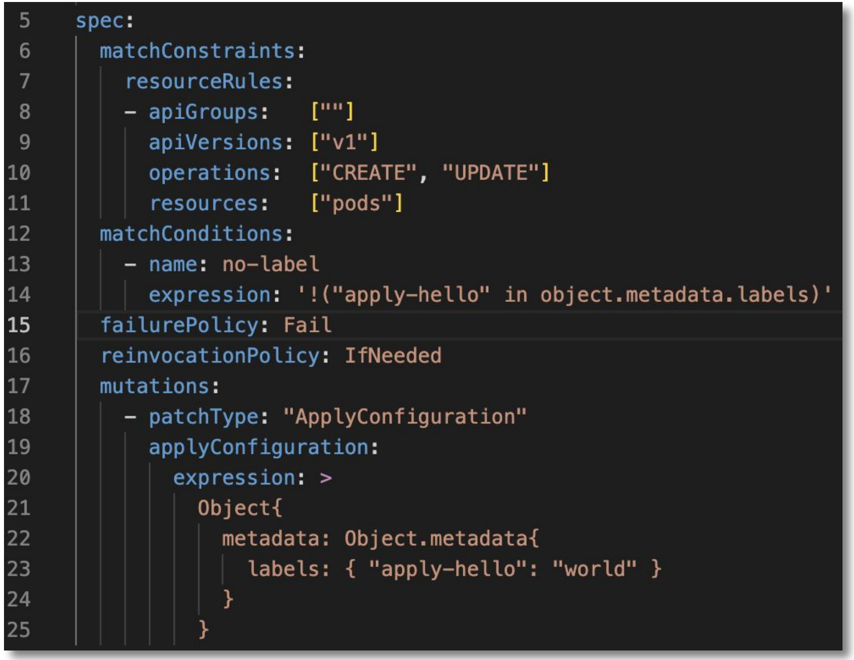
```yaml
apiVersion: admissionregistration.k8s.io/v1alpha1
kind: MutatingAdmissionPolicyBinding
metadata:
  name: "add-label-policy-binding.example.com"
spec:
  matchResources:
    matchPolicy: Equivalent
    namespaceSelector:
      matchLabels:
        environment: demo
  policyName: "add-label-policy.example.com"
```

Agenda

- Common Expression Language (CEL)
- CEL Playground
- Validating Admission Policies (VAPs)
- VAPs in Action
- Mutating Admission Policies (MAPs)
- MAPs in Action
- Questions

Server Side Apply mutations

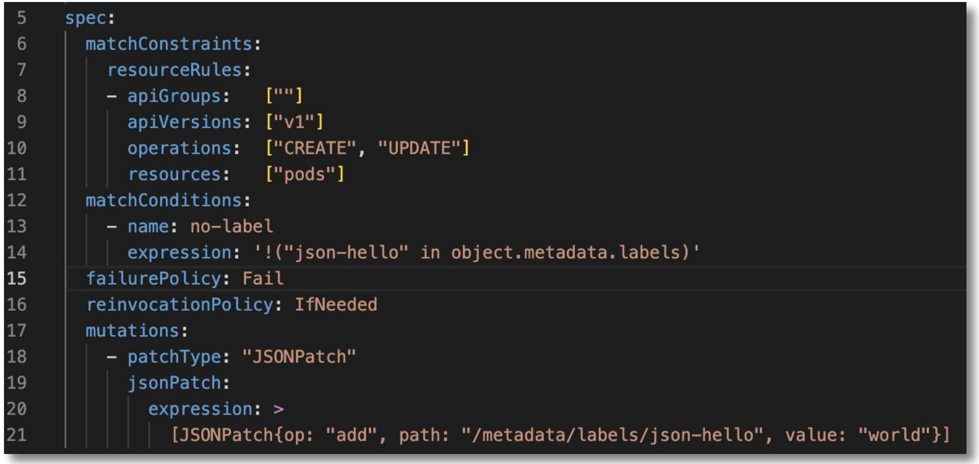
- ApplyConfiguration
- Object represents resource type



```yaml
spec:
  matchConstraints:
    resourceRules:
    - apiGroups:   [""]
      apiVersions: ["v1"]
      operations:  ["CREATE", "UPDATE"]
      resources:   ["pods"]
  matchConditions:
    - name: no-label
      expression: '!("apply-hello" in object.metadata.labels)'
  failurePolicy: Fail
  reinvocationPolicy: IfNeeded
  mutations:
    - patchType: "ApplyConfiguration"
      applyConfiguration:
        expression: >
          Object{
            metadata: Object.metadata{
              labels: { "apply-hello": "world" }
            }
          }
```

JSONPatch mutations

- JSONPatch
- Operations
  - Add
  - Remove
  - Replace
  - Copy
  - Move
  - Test



```
spec:
  matchConstraints:
    resourceRules:
    - apiGroups:   [""]
      apiVersions: ["v1"]
      operations:  ["CREATE", "UPDATE"]
      resources:   ["pods"]
  matchConditions:
    - name: no-label
      expression: '!("json-hello" in object.metadata.labels)'
  failurePolicy: Fail
  reinvocationPolicy: IfNeeded
  mutations:
    - patchType: "JSONPatch"
      jsonPatch:
        expression: >
          [JSONPatch{op: "add", path: "/metadata/labels/json-hello", value: "world"}]
```

Agenda

- Common Expression Language (CEL)
- CEL Playground
- Validating Admission Policies (VAPs)
- VAPs in Action
- Mutating Admission Policies (MAPs)
- MAPs in Action
- Questions

Give us feedback