



Connect

Policy Management and Delivery at Scale with ACM

Mark Roberts

Principal Solution Architect, Red Hat

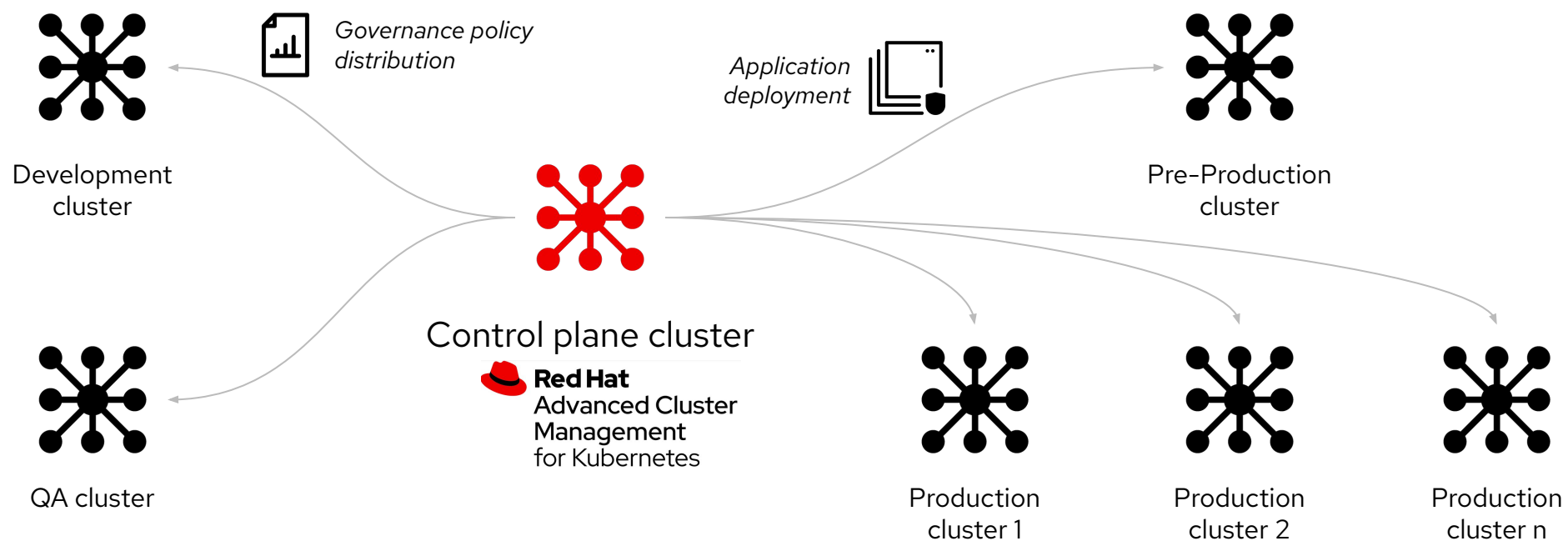
Agenda

- Brief review of ACM
- Policy management introduction
- Policy rollout - how to test and deploy policies to production
- Multitenancy policy management - because it's not all one team's responsibility
- Drift detection - control what can be updated and what needs to be replaced

In 25 minutes 😞



Security and governance in a multicluster environment



Policy Management

An introduction to ACM policies



Governance, Risk and Compliance

- Set and enforce policies for security, applications and infrastructure
- Visibility for the configuration of apps & clusters
- Categorize violations based on your standards
- Incorporate Gatekeeper policies and constraints

- Visualize results in ACM

- Policies are bound to a placement rule

- Identify the clusters to which the policies apply
- Various rules based on
 - Labels
 - Cluster type
 - Cloud etc.

Governance ?

Overview Policy sets **Policies** Discovered policies

☐ Search Filter Create policy Actions					
	Name ↑	Namespace	Remediation	Policy set	Cluster violations
> ☐	dev-spaces	rhacm-policies	enforce	-	✓ 1
> ☐	gatekeeper-config	rhacm-policies	enforce	-	✓ 1
> ☐	gatekeeper-policy	rhacm-policies	enforce	-	✓ 1
> ☐	gitops-operator	rhacm-policies	enforce	-	✓ 1
> ☐	ossm3-operator	rhacm-policies	enforce	-	✓ 1
> ☐	required-annotations-opa	rhacm-policies	inform	-	! 1
> ☐	web-terminal	rhacm-policies	enforce	-	✓ 1

1 - 7 of 7 items



Governance, Risk and Compliance

Policy structure

Policy header

Remediation action

Object definition
Configuration or operator policy

Where to apply within the cluster

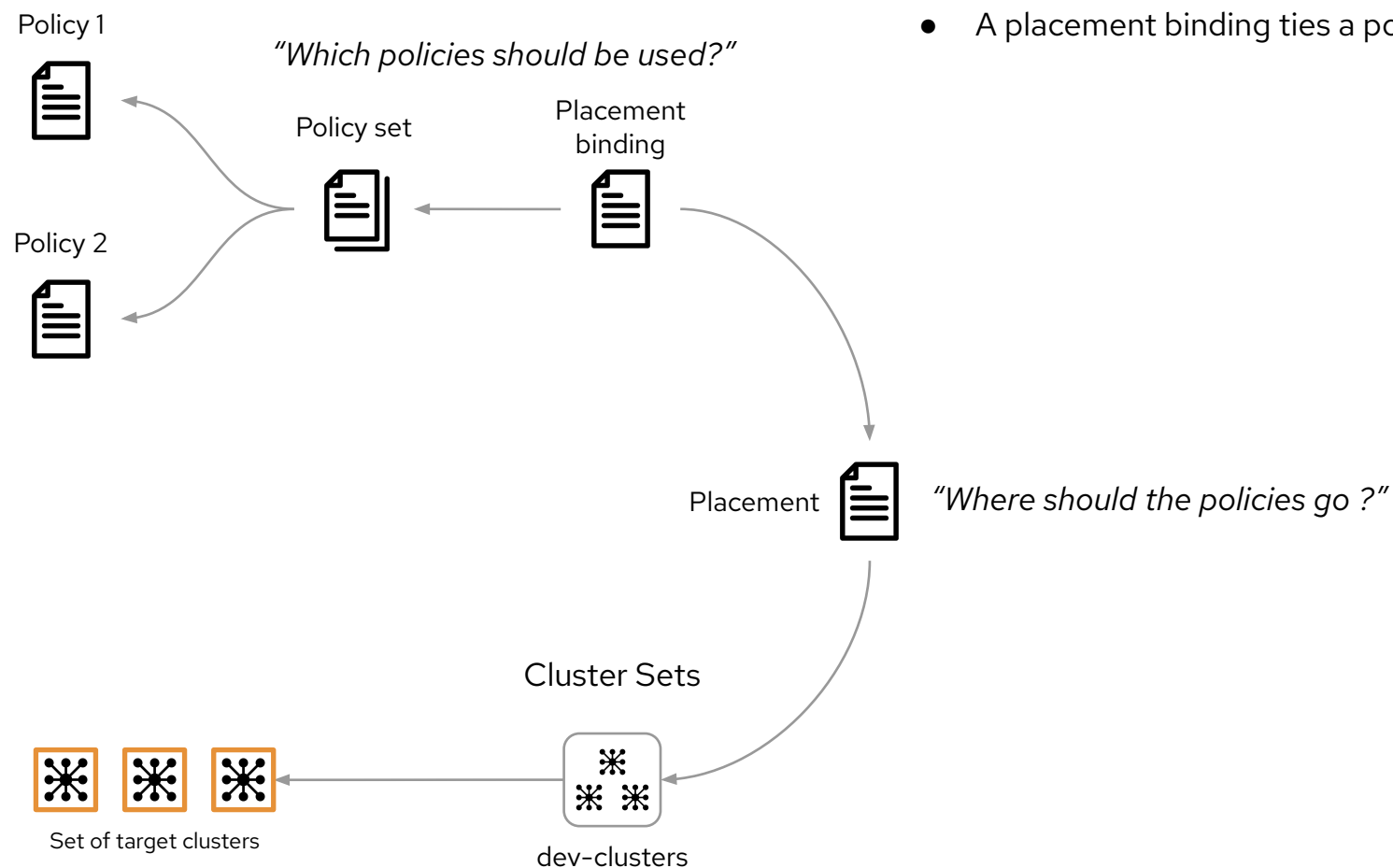
Object template
YAML resource to be applied

Repeated for multiple (related) objects



Policy object relationships

- Policy objects are grouped into policy sets to simplify the management as a group
- A placement resource identifies the cluster set or subset of clusters in a cluster set
- A placement binding ties a policy set to a specific placement resource

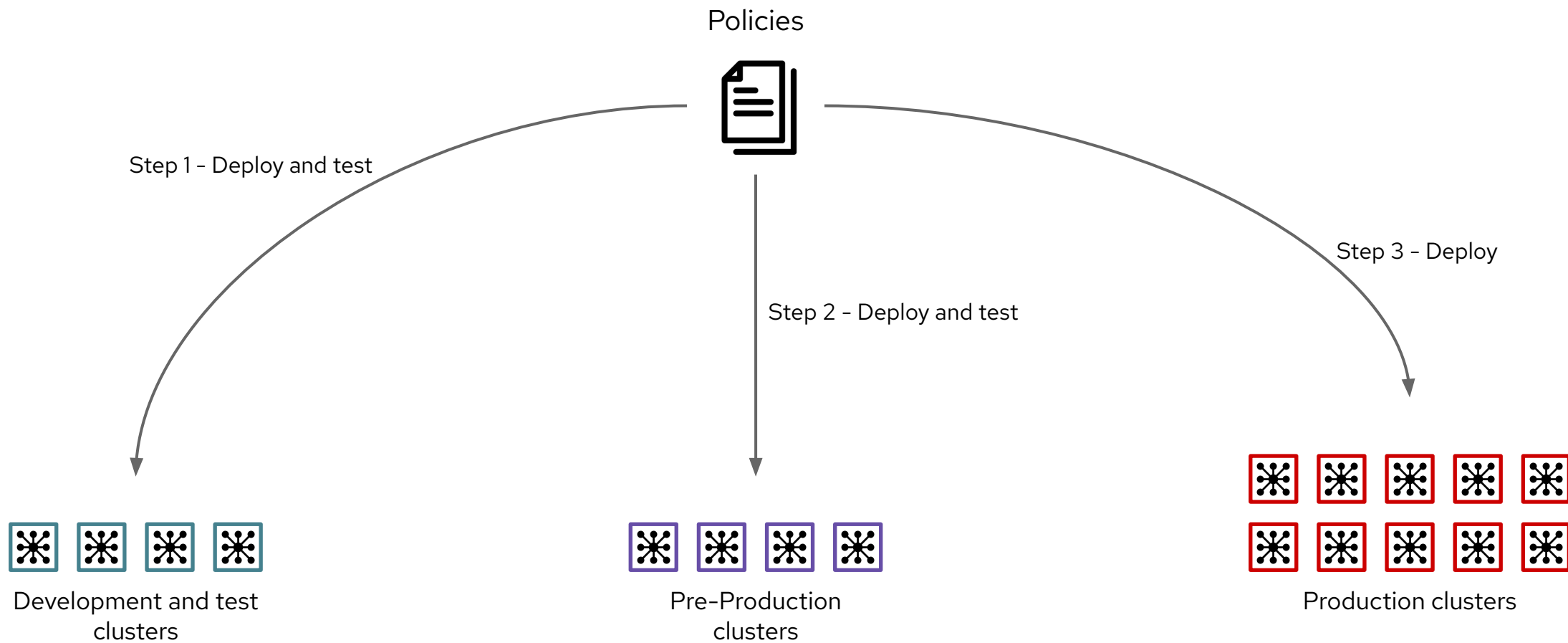


Policy Rollout

How to manage the deployment of new and updated policies without going straight to production



Policy Rollout Objective



Objective

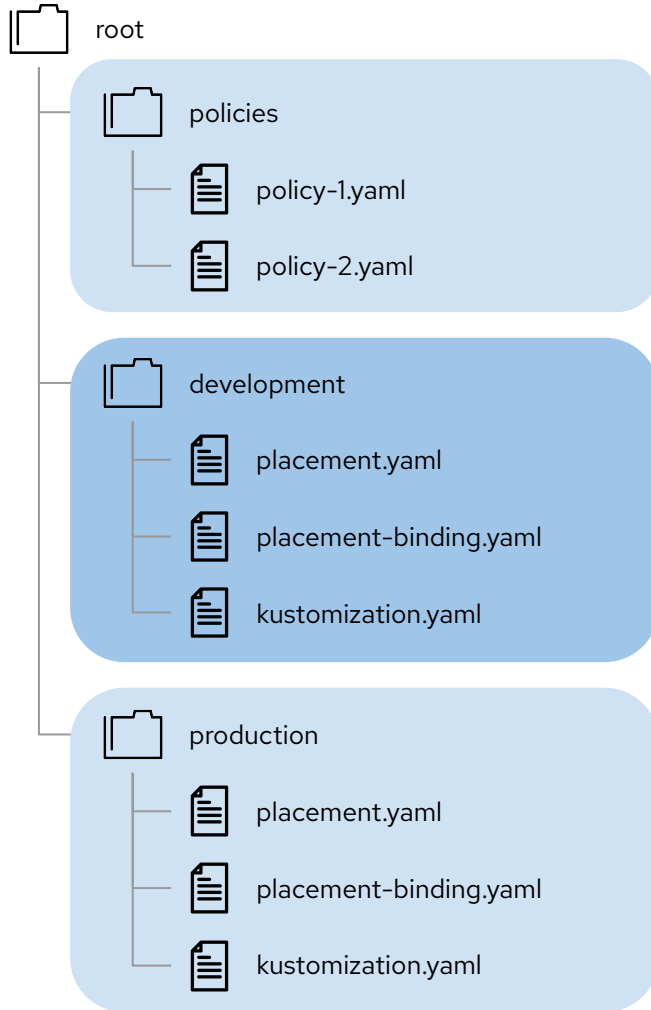
To create a test and deployment process for policies

- Requirement
 - To be able to test new policies and policy changes in a development cluster before deployment to production
- Constraint
 - Must only have a single variant of each policy
 - No 'development policies' and 'production policies' as separate entities



Content structure

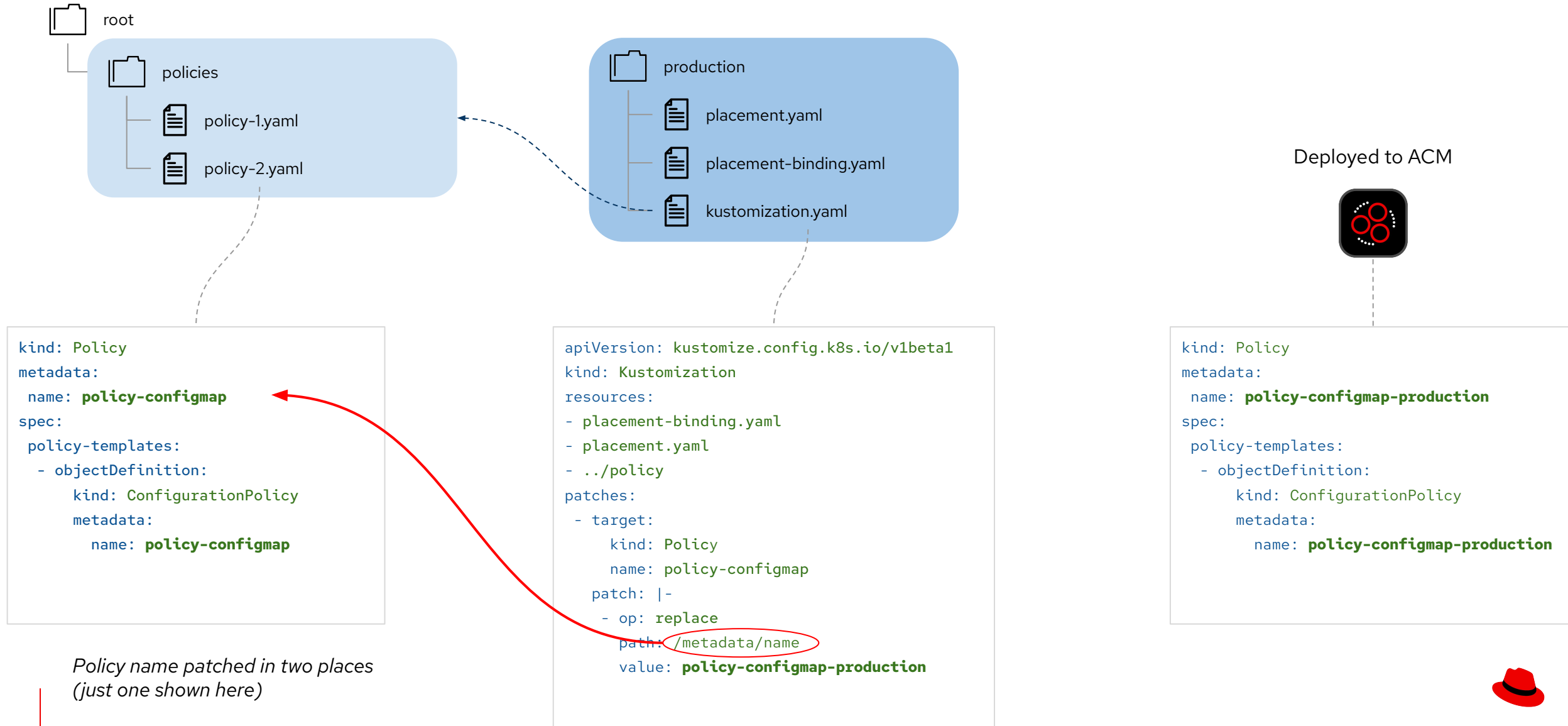
Managing policies and placements



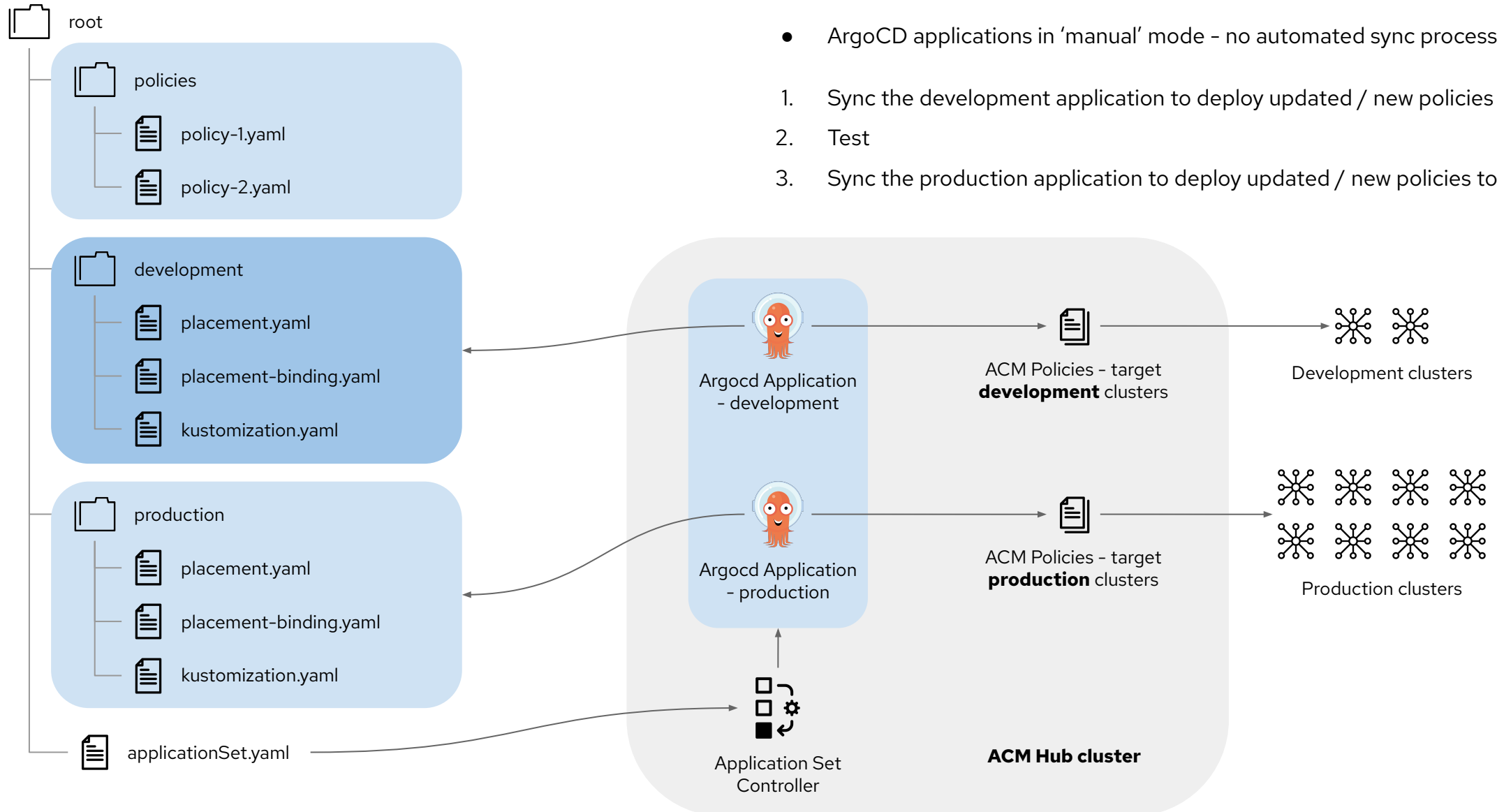
- Single set of policies
- Environment specific directories
 - Specific placement rules for the target clusters
 - Kustomization file to reference the placement content and ../policies directory
 - Kustomization file patch to change the policy names to be environment specific



Content patching



Content delivery



- ArgoCD applications in 'manual' mode - no automated sync process
1. Sync the development application to deploy updated / new policies to development
 2. Test
 3. Sync the production application to deploy updated / new policies to production



ApplicationSet resource

```
kind: Policy
apiVersion: argoproj.io/v1alpha1
kind: ApplicationSet
metadata:
  name: policy-mgmt
  namespace: openshift-gitops
spec:
  generators:
  - git:
      repoURL : https://github.com/marrober/acm-content.git
      revision: HEAD
      directories:
        - path: policies/policy-rollout/development
        - path: policies/policy-rollout/production
  template:
    metadata:
      name: 'acm-{{path.basename}}'
    spec:
      project: default
      destination:
        namespace: rhacm-policies
        server: https://kubernetes.default.svc
      source:
        repoURL: https://github.com/marrober/acm-content.git
        path: '{{path}}'
        targetRevision: HEAD
      syncPolicy:
        syncOptions:
          - CreateNamespace=true
```

Reference to environment specific paths in the repository

*New ArgoCD applications called **acm-development** and **acm-production***

Crucial automated sync policy is missing which creates manual mode applications

```
syncPolicy:
  automated: {}
```



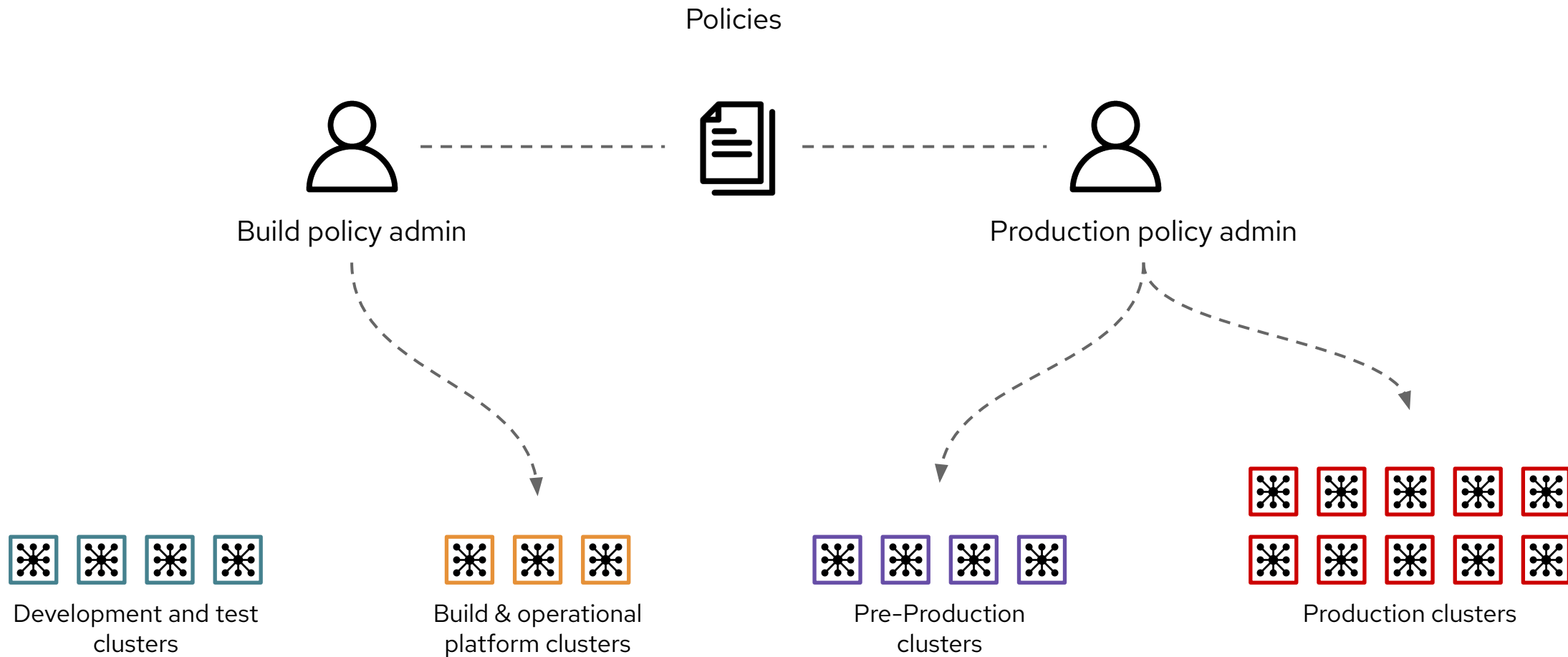


Multi Tenancy policy management

Delegated policy management and shared responsibility

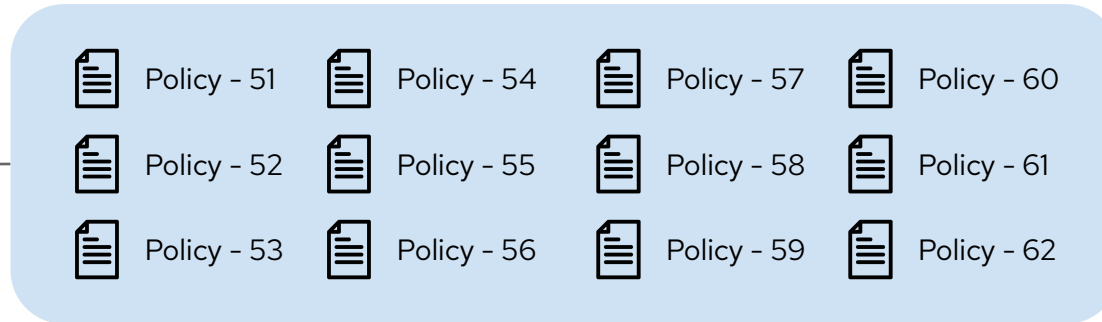
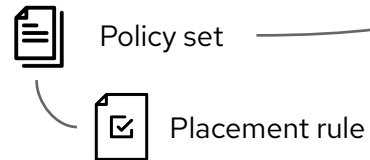


Policy Management - Different cluster classifications



Policy Management - Different cluster classifications

Common policies



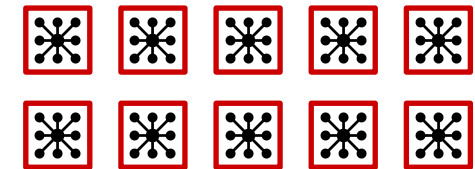
Development and test clusters



Build & operational platform clusters



Pre-Production clusters

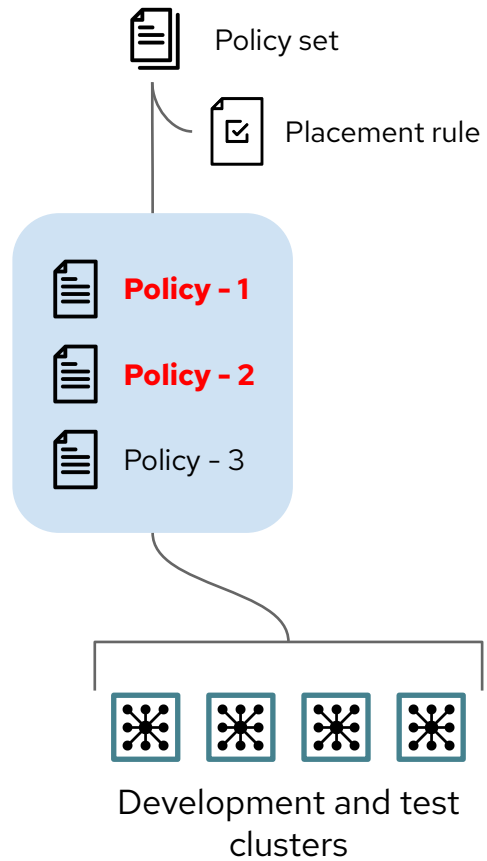


Production clusters

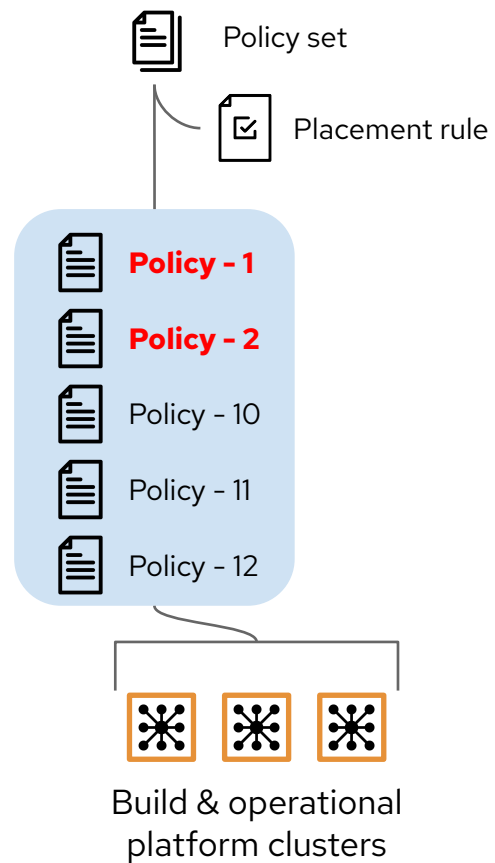


Policy Management - Different cluster classifications

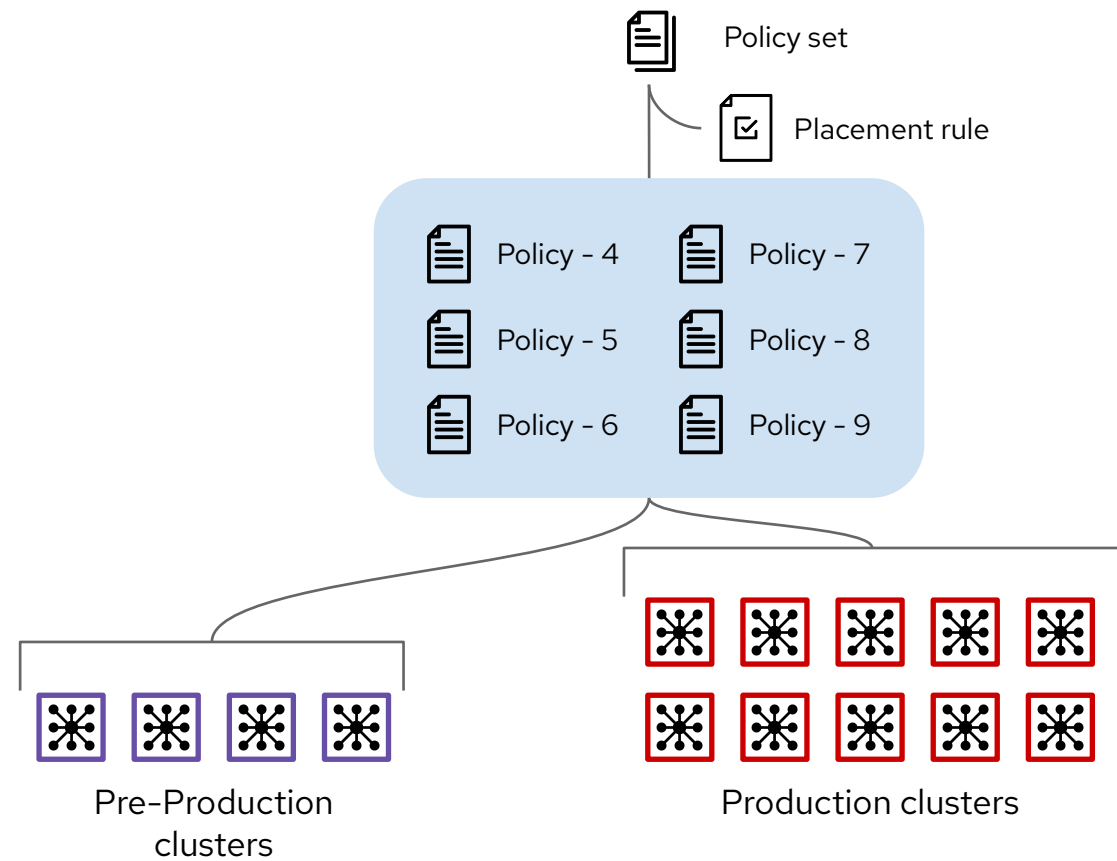
Development policies



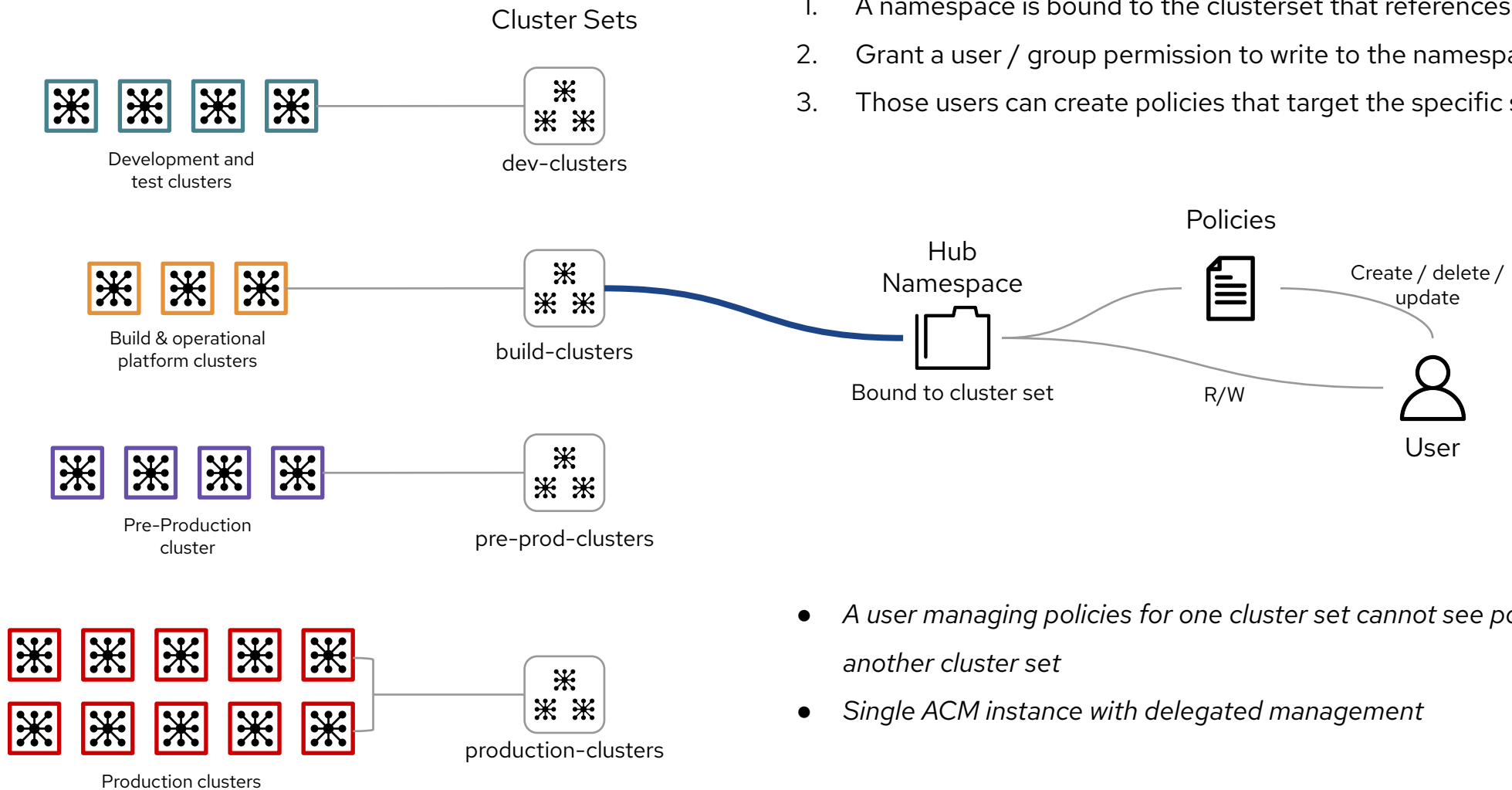
Build policies



Production policies



Multitenant policy management



1. A namespace is bound to the clusterset that references the target clusters
2. Grant a user / group permission to write to the namespace
3. Those users can create policies that target the specific set of clusters

- A user managing policies for one cluster set cannot see policies (and results) for another cluster set
- Single ACM instance with delegated management



Multitenant policy hub cluster tasks

- To operate a multitenant policy management process the ACM cluster administration team need to :
 - Create a namespace in which to store the policies for a specific team - example : **policies-build**
 - Create a cluster set in Advanced Cluster Management and allocate clusters to it - example below
 - Manage the resource assignment of the cluster set to add clusters to it

Clusters

Cluster list Cluster sets Cluster pools Discovered clusters

[> Learn more about the terminology](#)

☐  [Create cluster set](#) [Actions](#) 

	Name 	Cluster status
☐	global 	 2
☐	build-clusters	 2
☐	default	-

[Cluster sets](#) > [build-clusters](#)

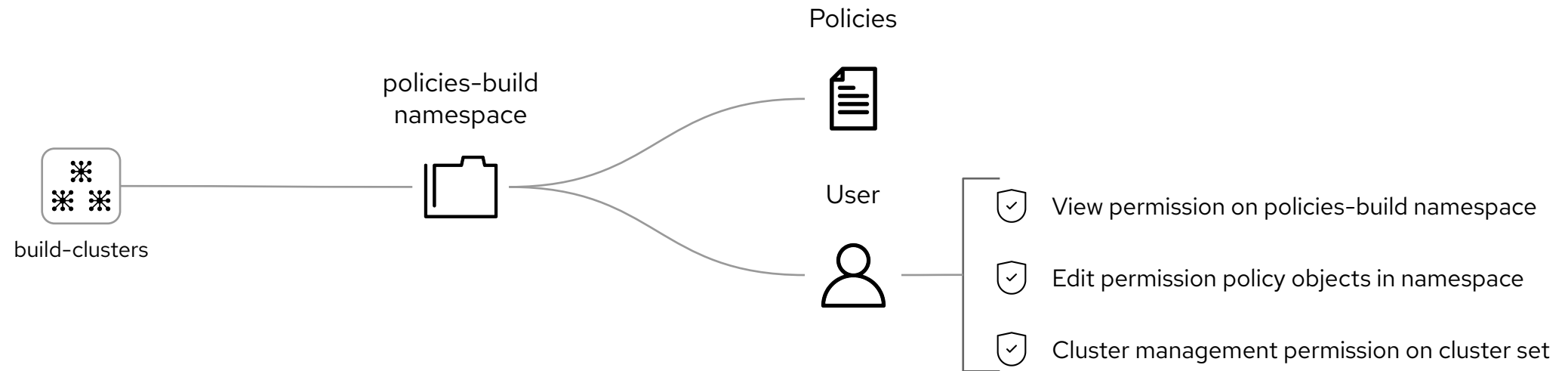
build-clusters

Overview Submariner add-ons Cluster list Cluster pools User management

☐ 	Search		 Filter 	 Labels 	Actions 	
	Name 	Namespace 	Status 	Infrastructure 	Control plane type	Distribution version 
☐	cluster-1	cluster-1	 Ready	 Amazon Web Services	Standalone	OpenShift 4.21.16
☐	local-cluster	local-cluster	 Ready	 Other	Hub	OpenShift 4.20.22  Upgrade available

Create a namespace for multitenant policies

- Administrator function to create a namespace for each policy tenant
- Apply relevant permissions to control access and distribute roles



Multitenant policy hub cluster tasks - 1

- Grant the user basic 'view' permission on the namespace - example below (left)
- Grant users who will manage the policies additional permission - example below (right)
 - Restrict the user from creating deployment workloads etc. in the policies namespace
- Group permission can be used

```
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: policies-build-view
  namespace: policies-build
subjects:
  - kind: User
    apiGroup: rbac.authorization.k8s.io
    name: greg
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: view
```

```
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: rhacm-edit-policy-greg
  namespace: policies-build
subjects:
  - kind: User
    apiGroup: rbac.authorization.k8s.io
    name: greg
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: rhacm-edit-policy
```



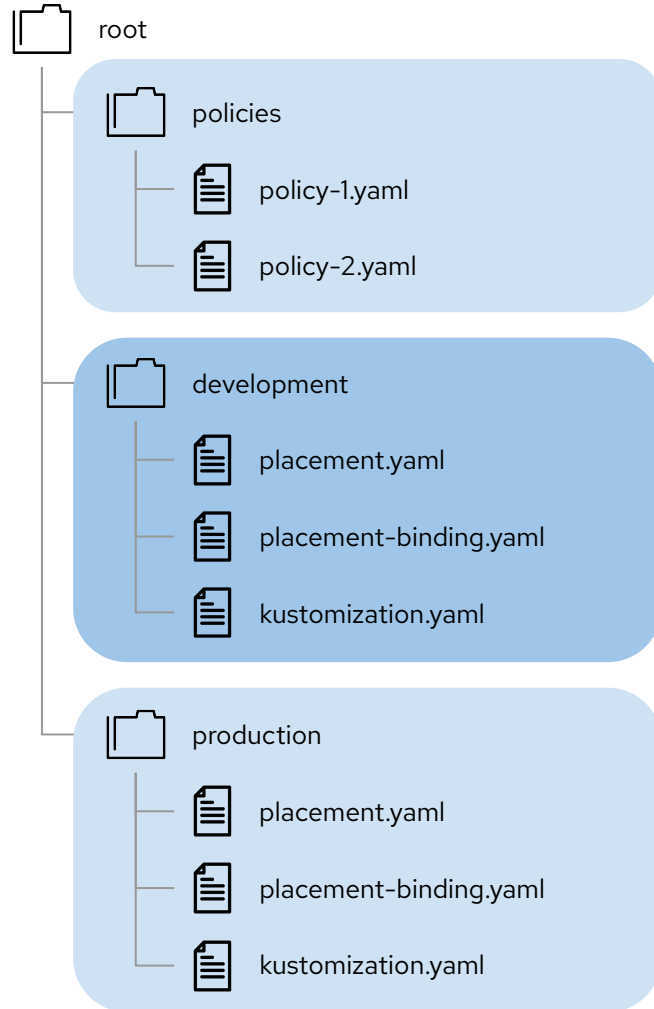
Multitenant policy hub cluster tasks - 2

- Each cluster has a namespace on the hub with a name that matches the cluster name
 - Users need view access to this namespace to view policy results and status
- Grant the user view permission to the managed cluster resource
 - This ensures that the user can see results of policy analysis on the policy-managed cluster

```
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: greg-view-build-cluster-01-namespace
  namespace: build-cluster-01
subjects:
- kind: User
  apiGroup: rbac.authorization.k8s.io
  name: greg
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: view
```



Combine policy rollout and multitenancy



- Manage the permissions on directory structures
- Manage the rollout of new versions using ArgoCD sync policy
- Manage the access to different policies for different cluster roles using placement rules
 - Select the policies and select the target clusters
- Use wide range of cluster selectors to pick a specific subset of a cluster set if necessary



Drift Detection

How to manage the deployment of new and updated policies without going straight to production



Changes to immutable resources

```
apiVersion: policy.open-cluster-management.io/v1
kind: Policy
metadata:
  name: policy-limitrange
  labels:
    policy: limit-range
spec:
  remediationAction: enforce
  disabled: false
  policy-templates:
    - objectDefinition:
        apiVersion: policy.open-cluster-management.io/v1
        kind: ConfigurationPolicy
        metadata:
          name: policy-limitrange
        spec:
          severity: medium
          namespaceSelector:
            Include: ['test-*']
```

```
object-templates:
  - complianceType: musthave
    objectDefinition:
      apiVersion: v1
      kind: LimitRange
      metadata:
        name: default-limit-range
      spec:
        limits:
          - type: Container
            default:
              cpu: 500m
              memory: 512Mi
```

Simple limit range policy for namespaces beginning with 'test-'



Update the limitrange in the policy

- Policy change of the default CPU limit from 500m to 600m
- Results in a policy error and the limit range is not updated
- The error indicates that the policy template should have the field `recreateOption` added
 - `recreateOption: ifRequired`
 - `recreateOption: always`

policy-limitrange

Details Results Actions ▾

Clusters						
Find clusters						
Cluster	Violations	Template	Message	Remediation	Last report	History
cluster-1	❗ Violations	policy-limitrange	violation - limitranges [default-limit-range] in namespace test-1 cannot be updated, likely due to immutable fields not matching, you may set spec["object-templates"][].recreateOption to recreate the object - View details	enforce	8 minutes ago	View history
local-cluster	❗ Violations	policy-limitrange	violation - limitranges [default-limit-range] in namespace test-1 cannot be updated, likely due to immutable fields not matching, you may set spec["object-templates"][].recreateOption to recreate the object - View details	enforce	8 minutes ago	View history

1 - 2 of 2 items

1 of 1 page



Changes to immutable resources

- Add the recreateOption field and the limit ranges can be altered by a policy update
- This setting also ensures that the field cannot be changed on the target cluster by a user who has enough permissions
 - Because the policy can update the resource and change it back

object-templates:

- complianceType: **musthave**
recreateOption: IfRequired

objectDefinition:

apiVersion: v1

kind: LimitRange

metadata:

name: default-limit-range

spec:

limits:

- type: Container

default:

cpu: 600m

memory: 512Mi



Changes to immutable resources

- Drift problem - without the `recreateOption` field set
- Example - Manual change to the limit range is allowed by RBAC
- User changes the limit range - `default.cpu = 800m`

Difference for the LimitRange test-1/default-limit-range

```
--- test-1/default-limit-range : existing
+++ test-1/default-limit-range : updated
@@ -7,10 +7,14 @@
  resourceVersion: "309638"
  uid: 537ffabe-4a58-4509-8813-8e0f3431ae3d
  spec:
    limits:
      - default:
+       cpu: 500m
+       memory: 512Mi
+       type: Container
+ - default:
      cpu: 800m
      memory: 512Mi
    defaultRequest:
      cpu: 500m
      memory: 512Mi
```

Close

[Policies](#) > [policy-limitrange](#) > policy-limitrange

CP **policy-limitrange** Violations

[Details](#) [YAML](#) [History](#)

ConfigurationPolicy details

Name	policy-limitrange	Kind	ConfigurationPolicy
Engine	Open Cluster Management	API version	policy.open-cluster-management.io/v1
Cluster	cluster-1	Labels	-

Related resources

1 - 1 of 1 < >

Name ↑	Namespace ↓	Kind ↓	API version ↓	Violations ↓	Reason
default-limit-range	test-01	LimitRange	v1	Violations - View diff	K8s update template error

1 - 1 of 1 items << < 1 of 1 page > >>



Changes to immutable resources – Summary

- Optionally add the field :

`object-templates:`

- `complianceType:` `musthave`
`recreateOption:` `IfRequired`

- If added
 - The resources will maintain compliance and cannot be altered by a user modifying the resource on cluster
 - The resource can be replaced (updated) by an update to the policy
- If not added
 - The user can change the resource value on the cluster and the ACM policy will highlight the out of compliance
 - The resource cannot be updated by an update to the policy





Connect

Thank you



linkedin.com/company/red-hat



facebook.com/redhatinc



youtube.com/user/RedHatVideos



x.com/RedHat

