

# ICP-MCM 201

**Mike Chang**  
**Cloud Adoption Leader**  
**IBM Cloud Labs**



IBM Cloud

# The path of multicloud can accelerate rapidly . . .

## **IBM Multicloud Manager** maintains your pace of innovation

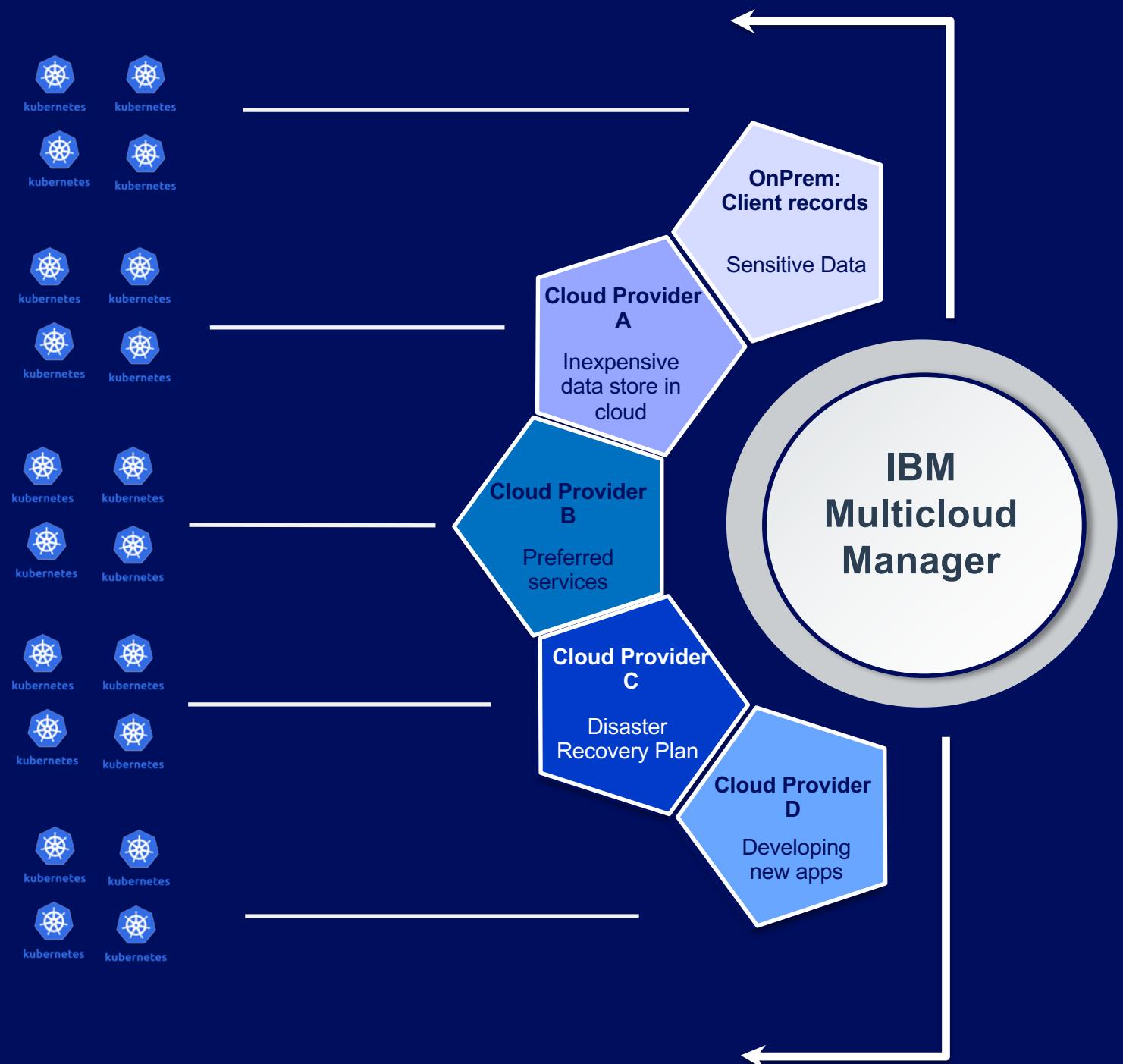
Multiple clusters to manage geographical data residency laws

Reduce on prem storage costs by storing non sensitive data in public cloud

Leverage most desired services in preferred cloud (AI/Analytics/object storage)

Load all applications on another provider maintain operations if one provider goes down (disaster recovery)

Build net new client experience cloud native apps on another cloud to posture against vendor lock-in



From 1 cluster on 1 cloud ....



to many clusters on numerous clouds ....



***IBM Multicloud Manager***

seamlessly moves you through the journey

*As organizations modernize and deploy **containerized** clusters, new challenges are introduced...*

I need  
broad  
Visibility

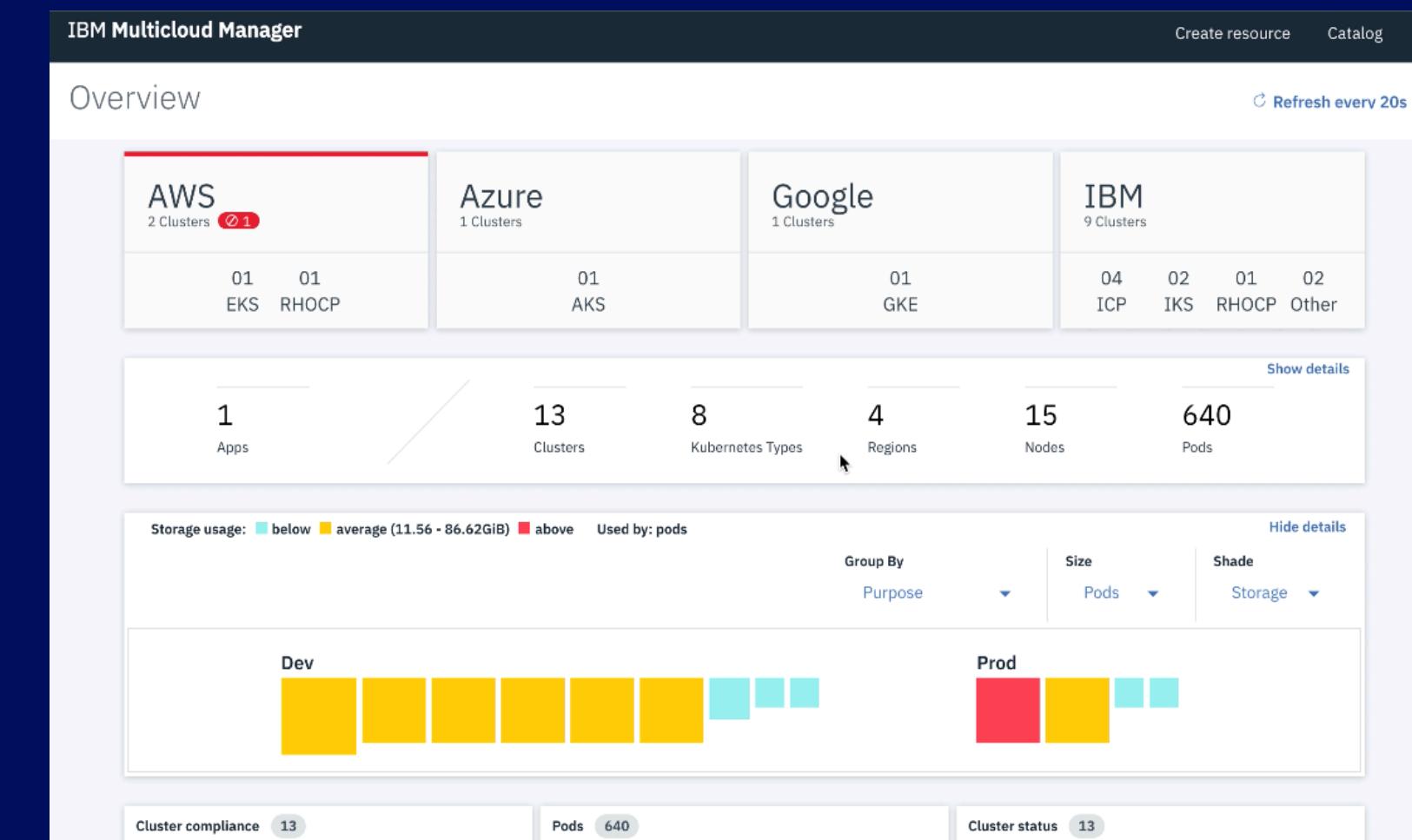
I need  
automated  
Governance

I need  
seamless  
**Application**  
Management

# **Visibility:** Clear insight into any environment, any application, any cluster

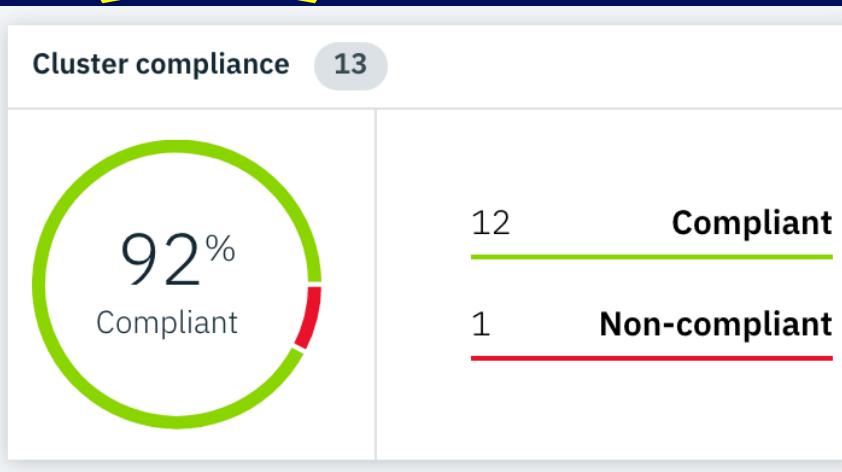
- Where are the failed components?
- Where are my services running?
- How can I monitor applications across clusters and clouds?
- How can I manage clusters as if they were one environment?
- How do I monitor usage across clouds?

**One Dashboard:** See health, usage, policy adherence on any cluster, any environment



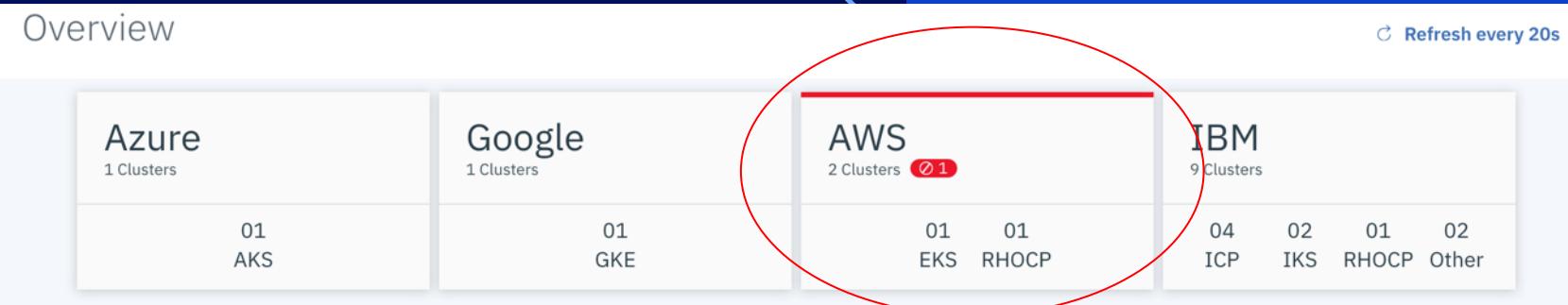
# **Governance:** Maintain controls across applications & clusters with policies

Something new in 3.2

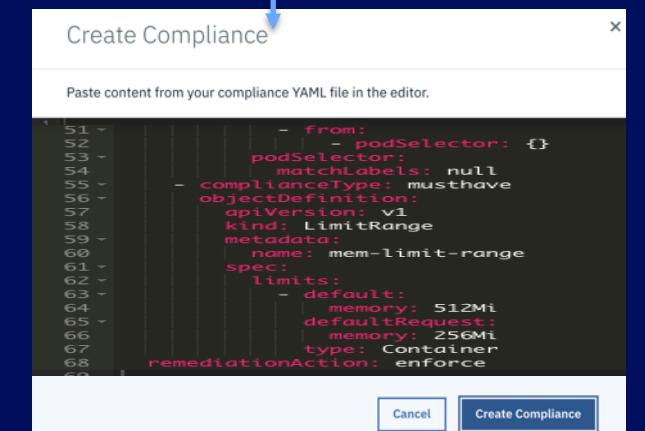


## I need automated Governance

- How do I set consistent security policies across environments?
- Which clusters are in compliance with our policies?
- How can I manage configuration across this large environment?
- How can I place workloads based on capacity, policy?

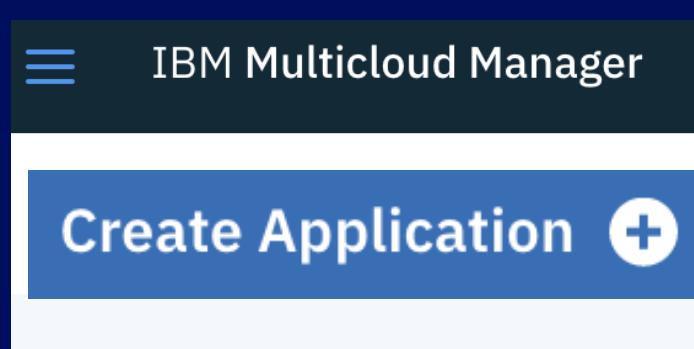


Create Placement & Security policies directly from MCM console and push to all clusters **with a click**

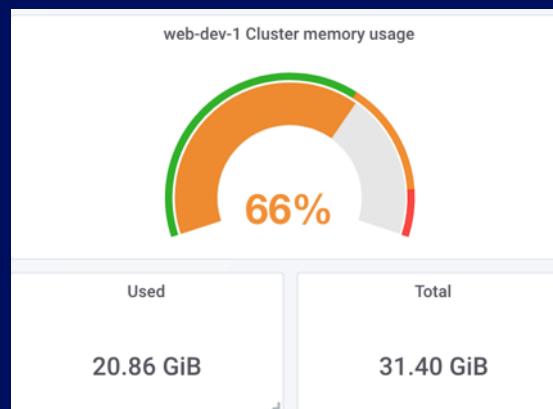


# ***Application Management:*** Create, Monitor, Manage and Backup

Something  
new in 3.2

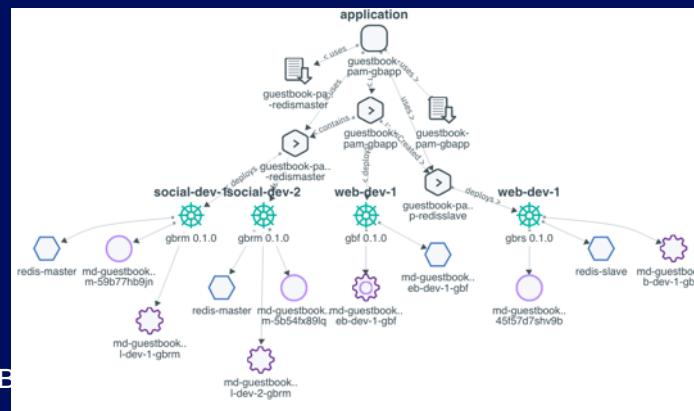


Create **an Application** across environments **all in one place**



**Monitor** your application with **Grafana/Prometheus**

**View all relevant information, deployments and placement policies for your applications**

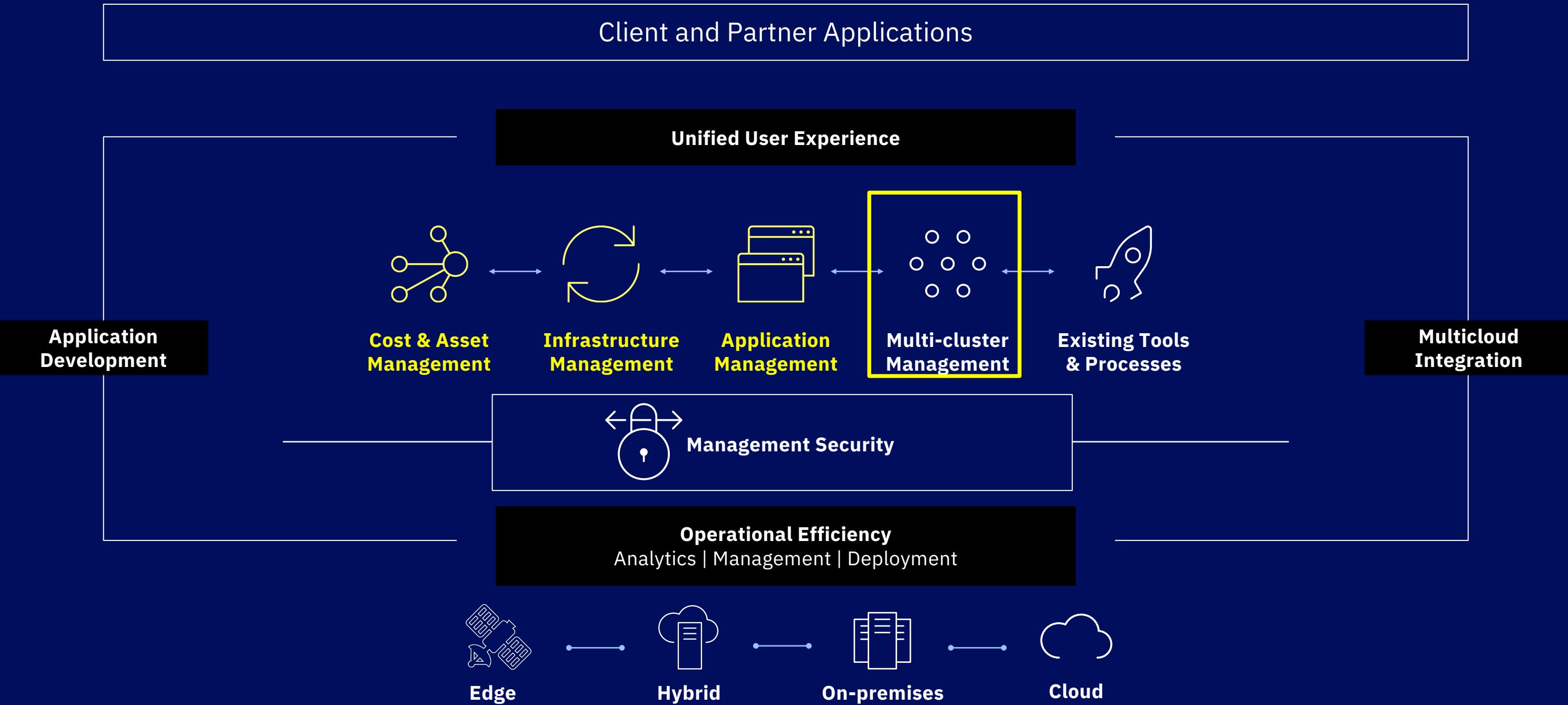


I need help with **Application Management**

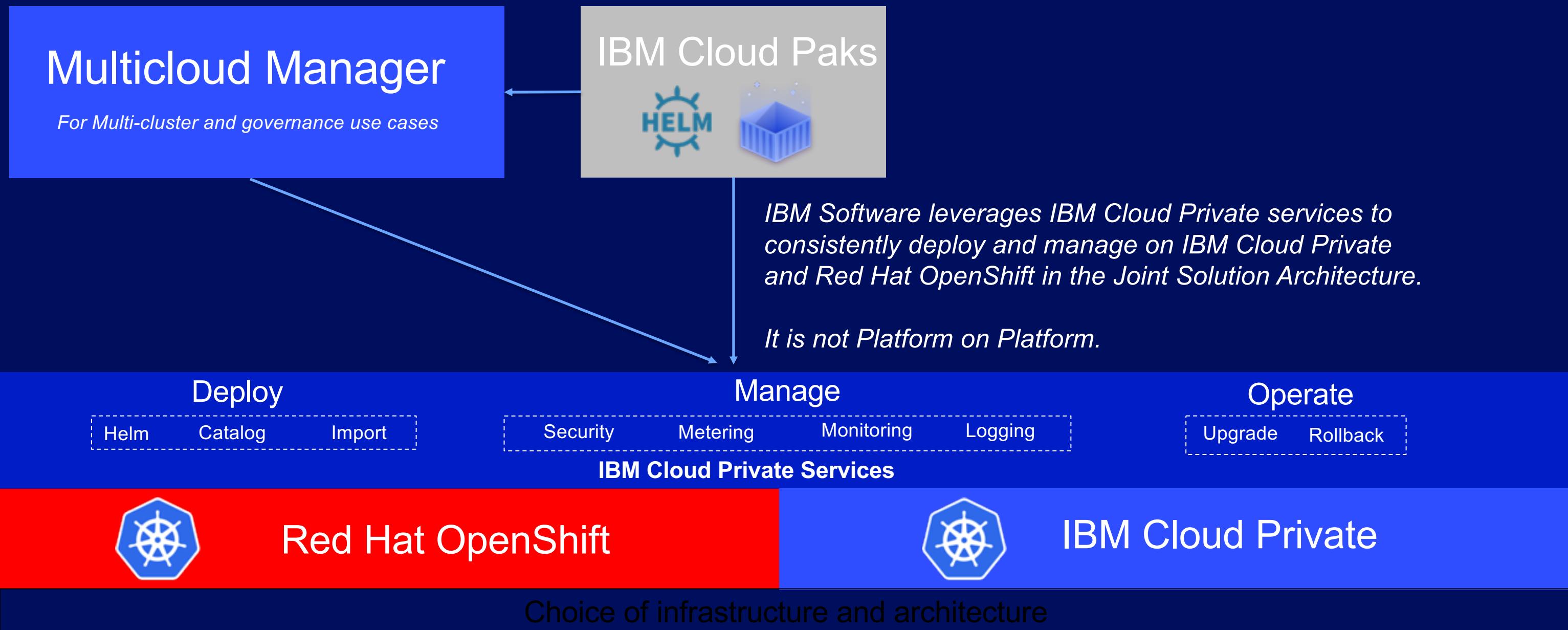
- How do I deploy applications across these environments?
- How do I move workloads across environments?
- How can I backup my applications?
- How do I do Business Continuity?

# IBM Cloud Pak for Multicloud Management

Formal announce in June 2019  
Will have one part # covering what is inside the box



# How it works today



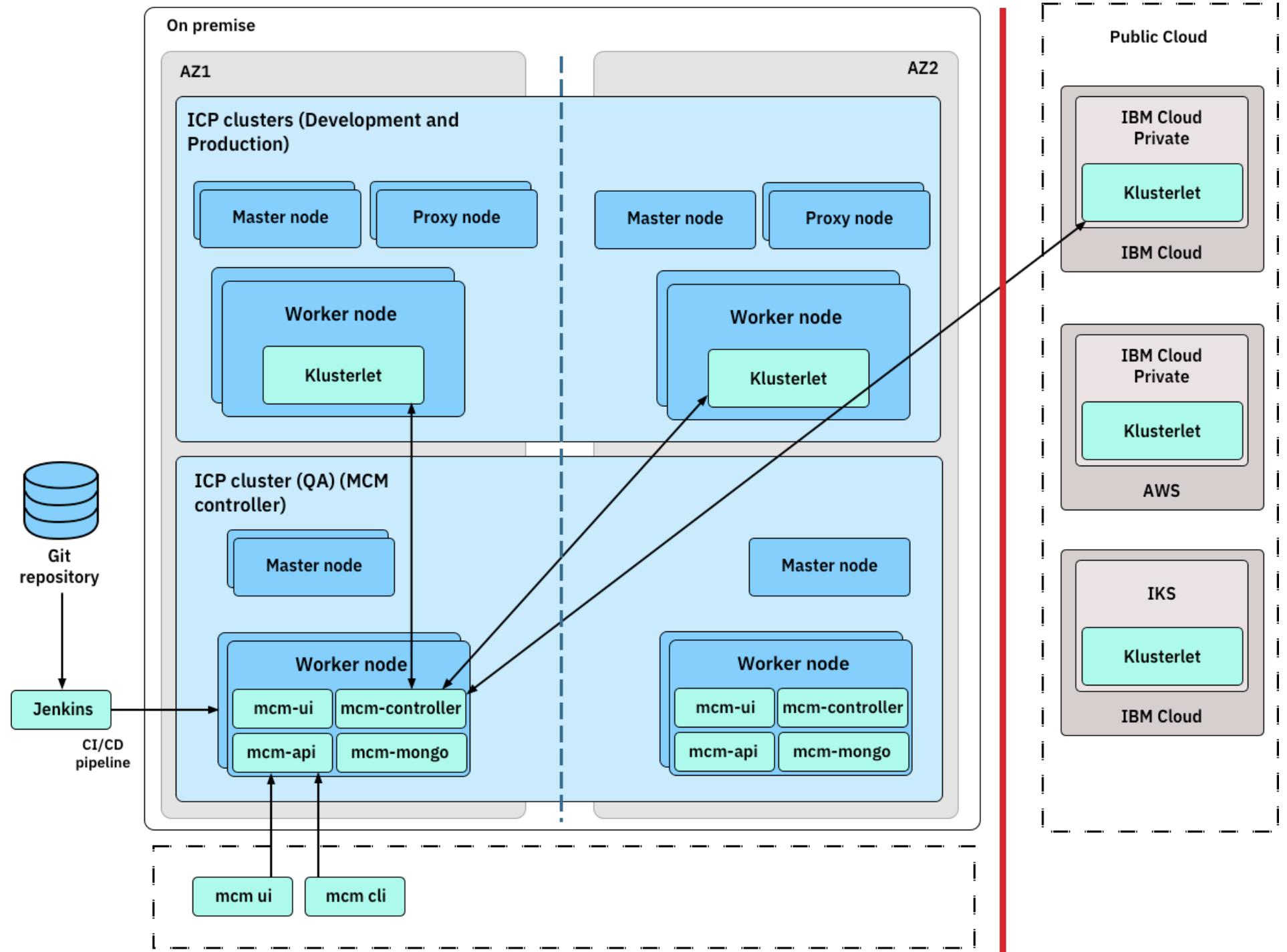
# **MCM components and architecture**

# MCM components and architecture

MCM Relies on ICP as the underlying platform and runs on an ICP worker node

## MCM Components:

- MCM **Controller**
- MCM **Klusterlet**
- MCM CLI
- MCM console (UI)
- MCM configuration info
- MCM API



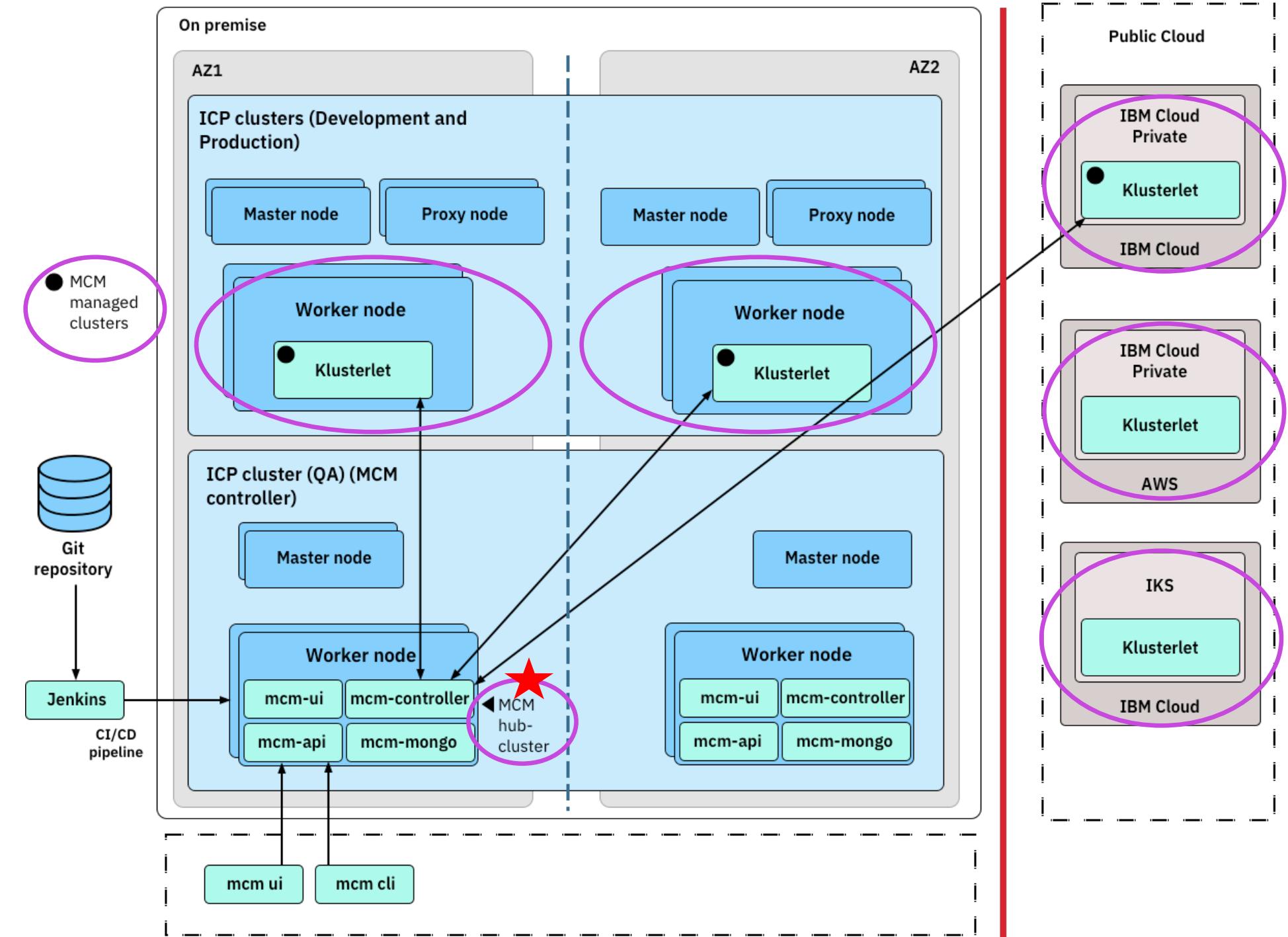
# Cluster communication

MCM Relies on ICP as the underlying platform and runs on an ICP worker node

ICP HA will add to resilience of the platform

- MCM hub-cluster
- MCM managed clusters

**Klusterlets** communicate with the **hub-cluster**

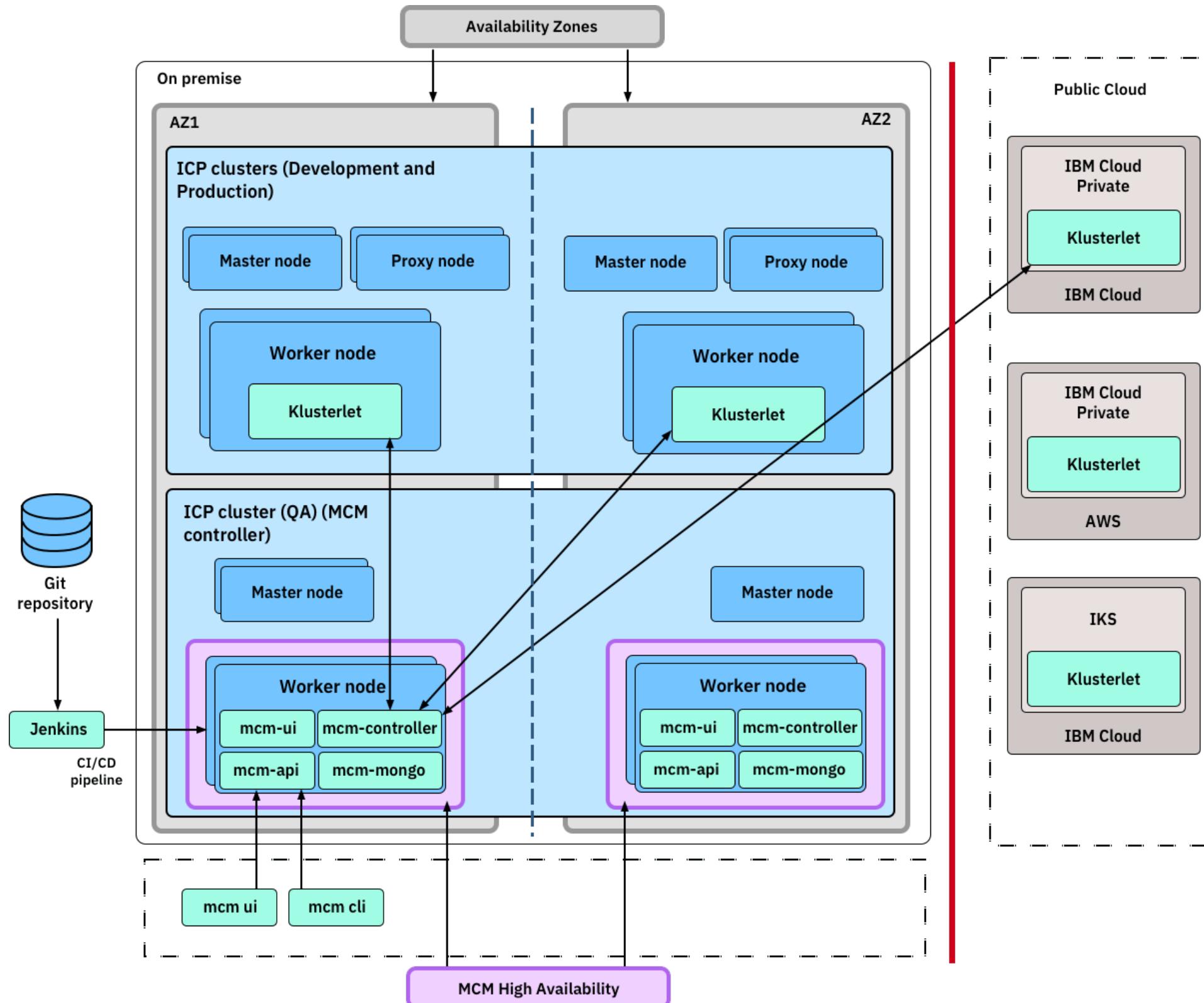


# Topology

In this diagram, two IBM Cloud Private on-premises availability zones, AZ1 and AZ2, are horizontally arranged for development and production workloads.

The IBM Cloud Private cluster with master node is also horizontally replicated for IBM Cloud Private HA across the AZ1 and AZ2 availability zones.

The Multicloud Manager hub-cluster is also configured in two availability zones to **support HA for the Multicloud Manager Controller**.



# **IBM Multicloud Manager Administration and CLI**

# IBM Multicloud Manager roles and access control

Topic page	Cluster Administrator	Administrator	Operator	Editor	Viewer
Overview	x	x	x		
Clusters	x	x	x		
Policies	x				
Applications	x	x	x		
Helm Releases	x	x	x		
Pods	x	x	x		
Nodes	x	x			
Storage	x	x			
Topology	x	x	x		
Event Management	x	x	x		
Identity and Access	x				
Launch to ICP	x				

# IBM Multicloud Manager console

The screenshot illustrates the integration between the IBM Cloud Private dashboard and the IBM Multicloud Manager console.

**Left Panel (IBM Cloud Private Dashboard):**

- Header: IBM Cloud Private, CLUSTER dmzcluster
- Navigation:
  - Dashboard
  - Container Images
  - Multicloud Manager (highlighted with a green oval)
  - Workloads
  - Network Access
  - Configuration
  - Platform
  - Manage
  - Command Line Tools
  - Getting started
- Side panel:
  - Active
  - Inactive

**Middle Panel (IBM Multicloud Manager - Mozilla Firefox):**

- Header: IBM Multicloud Manager - Mozilla Firefox, https://10.0.0.1:8443/multicloud/overview
- Content:
  - Header: IBM Multicloud Manager, Create resource, Catalog, Docs, Refresh every 20s
  - Section: Overview
  - Metrics: VCPU usage (yellow), Used by: nodes (cyan)
  - Widgets:
    - Cluster compliance: 100% Compliant (green circle)
    - Pods: 168 (100% Running, 0 Pending, 0 Failed)
    - Cluster status: 100% Ready (green circle)
  - Metrics: VCPU 17.65/40 | 44%, Memory 41.82 / 94.18GiB | 44%, Storage 183 / 243GiB | 75%

**Right Panel (IBM Multicloud Manager Sidebar):**

- Header: IBM Multicloud Manager
- Navigation:
  - Overview
  - Search
  - Clusters
  - Policies
  - Applications
  - Helm Releases
  - Pods
  - Nodes
  - Storage
  - Topology
  - Identity & Access
  - Manage Local Cluster
  - Getting Started

# IBM Multicloud Manager command-line interface

- **mcmctl** is the IBM Multicloud Manager command-line tool which is used to
  - manage and operate Multicloud Manager clusters.
  - script and automate many tasks, you use the CLI instead of the Multicloud Manager console.
- It's important to prepare the computer that you intend to use to run Multicloud Manager CLI commands
  - You need to first install the prerequisites packages, which include:
    - Docker
    - Kubernetes command-line tool
    - IBM Cloud Private command-line tool, and Helm CLI for ICP
  - You can then install the **mcmctl** tool
  - [https://www.ibm.com/support/knowledgecenter/SSBS6K\\_3.1.2/mcm/getting\\_started/start\\_guide.html](https://www.ibm.com/support/knowledgecenter/SSBS6K_3.1.2/mcm/getting_started/start_guide.html)

# IBM Multicloud Manager command-line interface

- General IBM Multicloud Manager commands
  - mcmctl create
  - mcctl delete
  - mcmctl deploy
  - mcmctl describe
  - mcmctl get

```
skytap@icpc11master:~$ cloudctl login -a https://10.0.0.1:8443 --skip-ssl-validation -u admin -p admin
...
skytap@icpc11master:~$ mcmctl get pods
CLUSTER      NAME            READY   STATUS    RESTARTS   AGE
dmzcluster   st-db2-ibm-db2oltp-dev-0  1/1     Running   4          57d
dmzcluster   st-mq-ibm-mq-0        1/1     Running   8          58d
dmzcluster   st-odm-ibm-odm-dev-7fd7f6488d-mnnph  1/1     Running   6          58d
dmzcluster   st-redis-master-0      1/1     Running   5          58d
dmzcluster   st-redis-slave-5b5bcd4bc7-bpqx8       0/1     CrashLoopBackOff 336      58d
skytap@icpc11master:~$
```

# IBM Multicloud Manager command-line

- IBM Multicloud Manager commands options

--alsologtostderr	log to standard error as well as files
--cluster-namespace string	namespace of the cluster (default "default")
-c, --cluster-selector string	Selector (label query) to filter on clusters, supports '=', '==', and '!='.(e.g. -l key1=value1,key2=value2)
-h, --help	help for mcmctl
-kubeconfig string	path to kubeconfig file. Overrides \$KUBECONFIG (default "/home/ubuntu/.kube/config")
--log-backtrace-at traceLocation	when logging hits line file:N, emit a stack trace (default :0)
--log-dir string	If non-empty, write log files in this directory
--log-flush-frequency duration	Maximum number of seconds between log flushes (default 5s)
--logtostderr	log to standard error instead of files (default true)
--stderrthreshold severity	logs at or above this threshold go to stderr (default 2)
-v, --v Level	log level for V logs
--vmodule moduleSpec	comma-separated list of pattern=N settings for file-filtered logging

# IBM Multicloud Manager command-line

```
skytap@icpcl1master:~$ cloudctl login -a https://10.0.0.1:8443 --skip-ssl-validation -u admin -p admin
adminAuthenticating...
OK

Targeted account dmzcluster Account (id-dmzcluster-account)
...

Configuring kubectl ...
Property "clusters.dmzcluster" unset.
Property "users.dmzcluster-user" unset.
Property "contexts.dmzcluster-context" unset.
Cluster "dmzcluster" set.
User "dmzcluster-user" set.
Context "dmzcluster-context" created.
Switched to context "dmzcluster-context".
OK

Configuring helm: /home/skytap/.helm
OK

skytap@icpcl1master:~$ mcmctl get clusters --all-namespaces
NAMESPACE      NAME           ENDPOINTS        STATUS       AGE
dmzns          dmzcluster     10.0.0.1:8001    Ready        57d
nondmzns       nondmzcluster 10.0.0.3:8001    Ready        58d
```

```
skytap@icpcl1master:~$ mcmctl describe clusters --all-namespaces
```

NAMESPACE	NAME	ENDPOINTS	STATUS	AGE
dmzns	dmzcluster	10.0.0.1:8001	Ready	57d
nondmzns	nondmzcluster	10.0.0.3:8001	Ready	58d

```
skytap@icpcl1master:~$ mcmctl describe clusters sedev -n dmzns
```

```
error: clusters.clusterregistry.k8s.io "sedev" not found
```

```
skytap@icpcl1master:~$ mcmctl describe clusters dmzcluster -n dmzns
```

Name: dmzcluster

Namespace: dmzns

Labels: cloud=IBM

datacenter=Raleigh

environment=Prod

name=dmzcluster

owner=marketing

region=US

vendor=ICP

Annotations: mcm.ibm.com/deployer-prefix=md

mcm.ibm.com/secretRef=dmzcluster-federation-secret

mcm.ibm.com/user-group=aGNtOmNsdXN0ZXJzLHN5c3RlbTphdXRoZW50aNhdGVk

mcm.ibm.com/user-identity=aGNtOmNsdXN0ZXJzOmRtem5zOmRtemNsdXN0ZXI=  
seed-generation=1

Metadata:

Creation Timestamp: 2019-02-20T20:27:06Z

Finalizers:

propagator.finalizer.mcm.ibm.com

Resource Version: 26529

Self Link: /apis/clusterregistry.k8s.io/v1alpha1/namespaces/dmzns/clusters/dmzcluster

UID: e4098560-354d-11e9-8c8d-f251166d50c1

Spec:

Auth Info:

Kubernetes API Endpoints:

Server Endpoints:

Client CIDR: 0.0.0.0/0

Server Address: 10.0.0.1:8001

Status:

Conditions:

Last Heartbeat Time: 2019-04-19T20:07:14Z

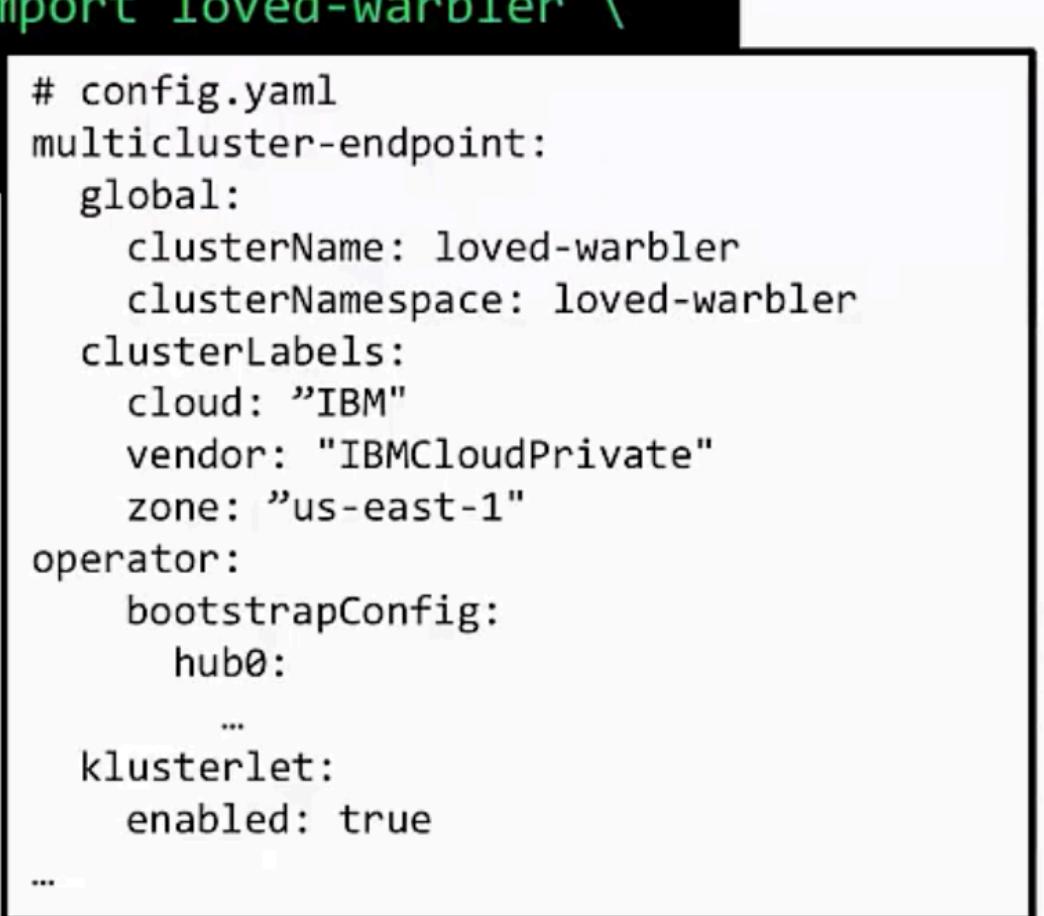
Last Transition Time: 2019-03-13T00:55:50Z

Status:

Type: OK

# Adding Clusters

```
cloudctl mc cluster import loved-warbler \
  -k ./kubeconfig \
  -f config.yaml
```



```
# config.yaml
multicloud-endpoint:
global:
  clusterName: loved-warbler
  clusterNamespace: loved-warbler
clusterLabels:
  cloud: "IBM"
  vendor: "IBMCLOUDPrivate"
  zone: "us-east-1"
operator:
  bootstrapConfig:
    hub0:
      ...
    klusterlet:
      enabled: true
    ...
```

Simplified cluster import via CLI

Adding managed clusters only requires a kubeconfig file (for authorization) and a config.yaml file (for installation options)

Add labels via config.yaml (or post install)

Optionally configure all controllers (app, security, search, etc)

# Adding Clusters

[https://www.ibm.com/support/knowledgecenter/en/SSBS6K\\_3.2.0/mcm/installing/klusterlet.html#post](https://www.ibm.com/support/knowledgecenter/en/SSBS6K_3.2.0/mcm/installing/klusterlet.html#post)

## cluster-import.yaml

```
# Licensed Materials - Property of IBM
# IBM Cloud private
# © Copyright IBM Corp. 2019 All Rights Reserved
# US Government Users Restricted Rights - Use, duplication o

default_admin_user: admin
container_runtime: docker      # options: docker, containerd

inception_image: ibmcom/icp-inception:latest
# image_repo:
# private_registry_enabled: true
# docker_username: admin
# docker_password: token

multicloud-endpoint:
  global:
    clusterName: mycluster
    clusterNamespace: default
  clusterLabels:
    environment: "Dev"
    region: "US"
    datacenter: "us-east"
    owner: "owner"
  operator:
    bootstrapConfig:
      hub0:
        name: hub0
        secret: kube-system/klusterlet-bootstrap
      hub1:
        name: null
        secret: null
```

## kubeconfig

```
apiVersion: v1
clusters:
- cluster:
    insecure-skip-tls-verify: true
    server: https://119.81.151.156:8001
    name: mycluster
contexts:
- context:
    cluster: mycluster
    namespace: default
    user: admin
    name: mycluster-context
  current-context: mycluster-context
  kind: Config
  preferences: {}
users:
- name: admin
  user:
    token: eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJhdF9oYXNoIj0
```

**cloudctl mc cluster import -f cluster-import.yaml  
--cluster-kubeconfig kubeconfig**

# Removing a remote-cluster

[https://www.ibm.com/support/knowledgecenter/en/SSBS6K\\_3.2.0/mcm/installing/uninstall\\_k8s.html](https://www.ibm.com/support/knowledgecenter/en/SSBS6K_3.2.0/mcm/installing/uninstall_k8s.html)

1. From a terminal, go to the <import\_config\_directory> directory, which you created during the import process. Log in to the hub-cluster with the following command:

```
cloudctl login -a https://<hub_cluster_host_name>:8443 --skip-ssl-validation
```

2. Run the following command to remove the targeted managed-cluster:

```
cloudctl mc cluster remove {cluster-name} -K kubeconfig | tee cluster-remove.log
```

3. Verify that your cluster is *Offline*.

- Log in to your IBM Multicloud Manager hub-cluster.
- From the navigation bar, click **Clusters**.
- Find your managed-cluster in the list of clusters.
- Ensure the the status is *Offline*.

4. Remove the cluster resource with the following command:

```
kubectl delete cluster {cluster-name} -n {cluster-namespace}
```

5. Run the following command to verify that the status of the cluster that you want to remove changed to *Offline*.

```
kubectl get clusters --all-namespaces
```

# Removing a remote-cluster -Tips

```
*****
```

```
* Get all cluster
```

```
*****
```

```
kubectl get clusters --all-namespaces
```

```
*****
```

When Run the following command to remove the targeted managed-cluster but get error

```
kubectl get clusters --all-namespaces
```

```
*****
```

```
cloudctl mc cluster remove {cluster-name} -K kubeconfig | tee cluster-remove.log
```

```
*****
```

```
* go to the <import_config_directory> directory
```

```
*****
```

```
cloudctl login -a https://<hub_cluster_host_name>:8443 --skip-ssl-validation
```

```
*****
```

```
* Login user id?
```

```
*****
```

what user did you use to login cloudctl? Please use admin.

```
*****
```

```
* How to choose right namespace?
```

```
*****
```

verify configmap via command "kubectl get cm --all-namespaces". In the meanwhile, confirm which namespace your kubectl is configured, switch to the correct namespace that "{cluster-name}-bootstrap-config

```
*****
```

```
* How to switch right namespace?
```

```
*****
```

Please take either of the steps below to switch to default namespace.

run command cloudctl login again and choose right namespace.

run command cloudctl target -n {namespace}

# Demo

IBM Multicloud Manager

Create resource Catalog   

Overview 

IBM

2 Clusters

02 ICP

1 Apps

2 Clusters

1 Kubernetes Types

1 Regions

Cluster: nodes VCPU usage (CPU):  above (none)  average (8.7 - 5.57)  below (none)

Filters

Condition

None

Non-compliant

High VCPU

High storage

High memory

Cloud providers

IBM

Purpose

Dev

Regions

US

Kubernetes type

ICP

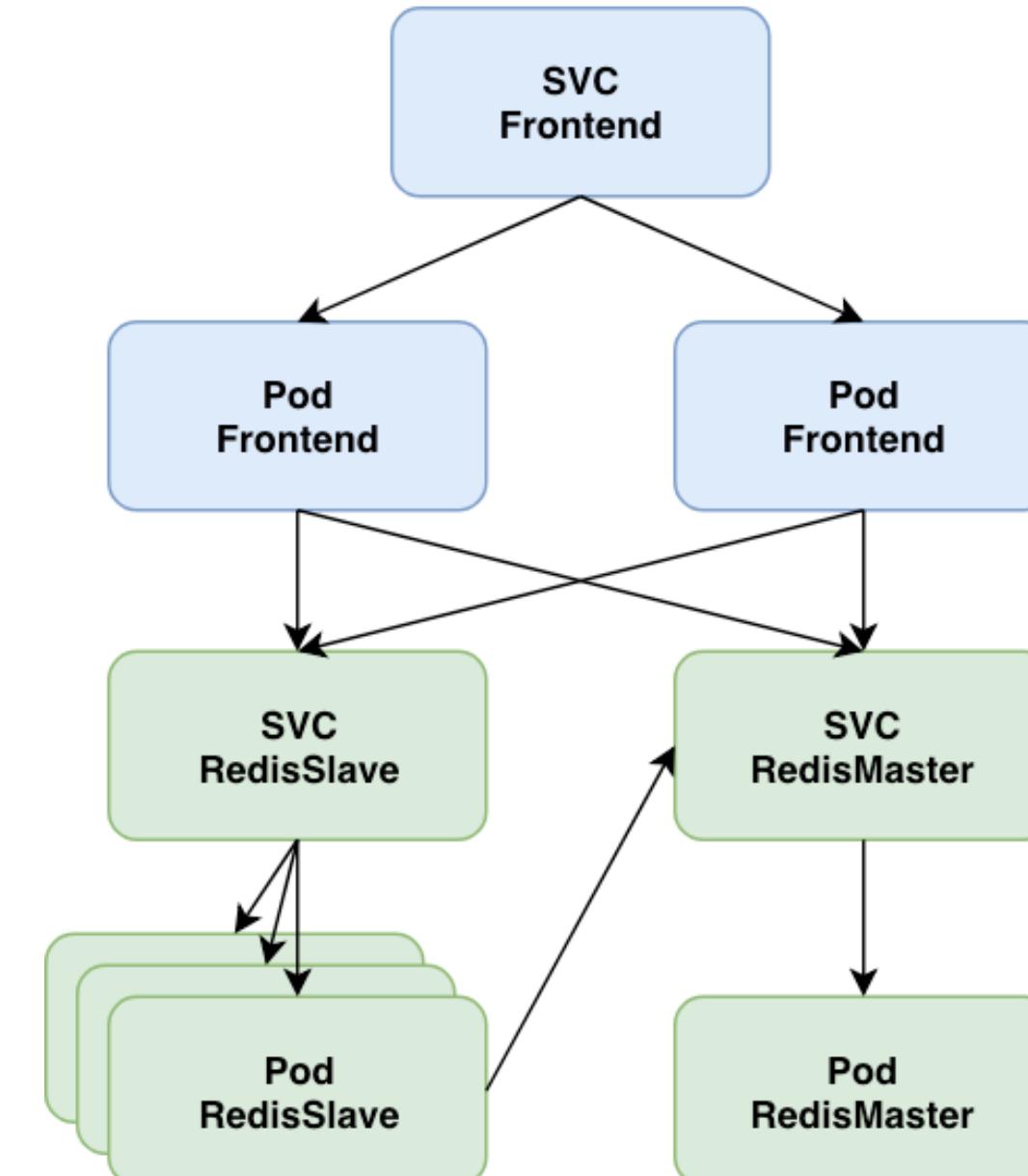
# MCM Application

# Guestbook

basic **Application** concepts we learned in [Core Multicloud Manager Resources to Learn](#),

- **Redis Master:** A Redis Master instance.
- **Redis Slave:** A Redis Slave instance, which replicates the contents of the Redis Master.
- **Frontend:** An Angular.js application that uses PHP script to communicate with Redis Master and Slave.
  - In the application, you can submit Guestbook names, which get persisted in Redos.
  - It sends new names to the Redis Master, which then get replicated to the Redis Slave.
  - It then retrieves the names from the Redis Slave and prints them on the screen.

## Guestbook Architecture



# Guestbook - MCM

This example is trying to convert k8s guest book example

[<https://kubernetes.io/docs/tutorials/stateless-application/guestbook>] into mcm application

Following 3 charts are wrapper for deployment and service in k8s example, already loaded into public place

1.gbf

2.gbrm

3.gbrs

Following 1 chart is for mcm-application

1.gbapp

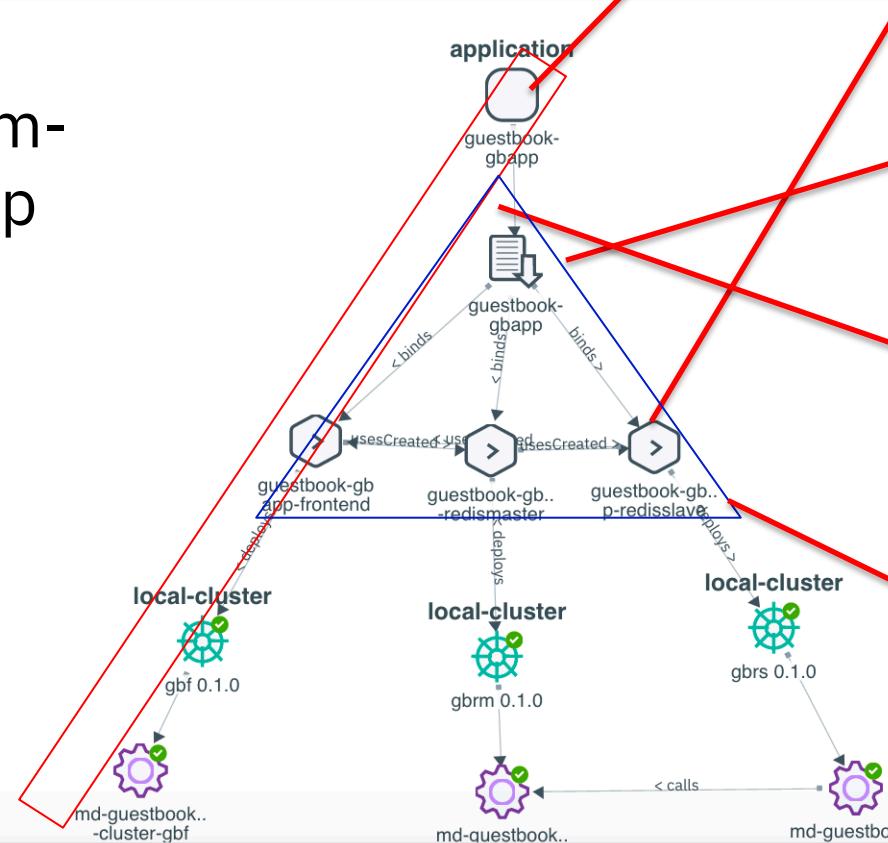
# MCM Application

**5 components** that are required to **define an application** in IBM Multicloud Manager - **Application, Deployable, PlacementPolicy, ApplicationRelationship and PlacementBinding.**

application placement deployable chart service deployment pod

1 chart is for mcm-application gbapp

- gbf
- gbrm
- gbrs



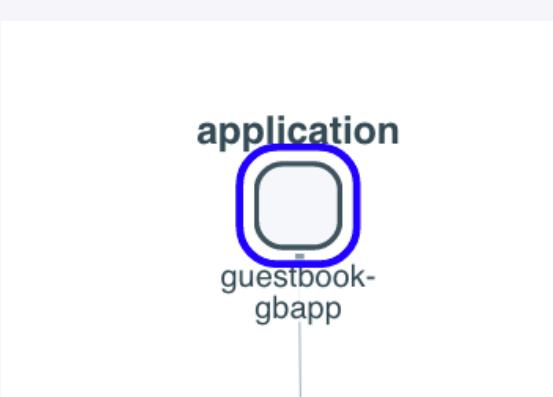
```
apiVersion: app.k8s.io/v1beta1
kind: Application
metadata:
spec:
---
apiVersion: mcm.ibm.com/v1alpha1
kind: Deployable
metadata:
spec:
---
apiVersion: mcm.ibm.com/v1alpha1
kind: PlacementPolicy
metadata:
spec:
---
apiVersion: mcm.ibm.com/v1alpha1
kind: ApplicationRelationship
metadata:
spec:
---
apiVersion: mcm.ibm.com/v1alpha1
kind: PlacementBinding
metadata:
placementRef:
subjects:
---
```

# Application Resource - MCM

## Application Resource

you have to create an Application resource. The Application resource lets you specify criteria to **associate Kubernetes resources to your application** so that MCM can manage and monitor them. Let's look at the gbapp Application resource from the Helm Chart below:

```
apiVersion: app.k8s.io/v1beta1
kind: Application
metadata:
  name: {{ template "guestbookapplication.fullname" . }}
  labels:
    app: {{ template "guestbookapplication.name" . }}
    chart: {{ .Chart.Name }}-{{ .Chart.Version | replace "+" "_" }}
    release: {{ .Release.Name }}
    heritage: {{ .Release.Service }}
    name: {{ template "guestbookapplication.fullname" . }}
spec:
  selector:
    matchExpressions:
      - key: app
        operator: In
        values:
          - gbapp
          - gbf
          - gbrm
          - gbrs
  componentKinds:
    - group: core
      kind: Pods
```



```
targetCluster:
  labelSelector:
    matchLabels:
      #vendor: "ICP"
      #region: "US"
      #environment: Dev
      #datacenter: "toronto"
      #owner: "marketing"
  resourceSelector:
    type: cpu
  compliances: ""
```

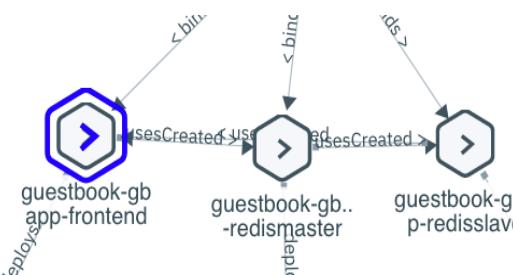
```
1  apiVersion: app.k8s.io/v1beta1
2  kind: Application
3  metadata:
4    name: guestbook-gbapp
5    namespace: default
6    generation: 2
7    labels:
8      name: guestbook-gbapp
9      app: gbapp
10     chart: gbapp-0.1.0
11     heritage: Tiller
12     release: guestbook
13   spec:
14     componentKinds:
15       - group: core
16         kind: Pods
17     descriptor: {}
18   selector:
19     matchExpressions:
20       - key: app
21         operator: In
22         values:
23           - gbapp
24           - gbf
25           - gbrm
26           - gbrs
```

# Deployable Resource - MCM

## Deployable Resource

The way to do that is by creating a Deployable object, where you can specify what Helm chart **to deploy and what namespace to deploy the chart into once the PlacementPolicy** (explained in later sections) decides what clusters to deploy the Helm Charts into. Let's look at the gbapp-frontend Deployable resource from the Helm Chart below:

```
apiVersion: mcm.ibm.com/v1alpha1
kind: Deployable
metadata:
  name: {{ template "guestbookapplication.fullname" . }}-frontend
  labels:
    app: {{ template "guestbookapplication.name" . }}
    chart: {{ .Chart.Name }}-{{ .Chart.Version | replace "+" "_" }}
    release: {{ .Release.Name }}
    heritage: {{ .Release.Service }}
  name: {{ template "guestbookapplication.fullname" . }}-frontend
  servicekind: ApplicationService
spec:
  deployer:
    kind: helm
    helm:
      chartURL: {{ .Values.chartRepoURL }}/gbf-0.1.0.tgz
      namespace: {{ .Values.appInClusterNamespace }}
```



```
apiVersion: mcm.ibm.com/v1alpha1
kind: Deployable
metadata:
  name: guestbook-gbapp-frontend
  namespace: default
  labels:
    name: guestbook-gbapp-frontend
    app: gbapp
    chart: gbapp-0.1.0
    heritage: Tiller
    release: guestbook
    servicekind: ApplicationService
spec:
  deployer:
    helm:
      namespace: default
      chartURL: >-
        https://raw.githubusercontent.com
        /ibm-cloud-architecture
        /kubernetes-multicloud
        -management/v312/demos/guestbook
        /chart-packages/gbf-0.1.0.tgz
      kubeKind: helm
      kubeName: ''
      kind: helm
```

# Relationship Resource - MCM

## Relationship Resource

MCM provides application topology features via [Weave Scope](#) out of the box. However, to explicitly specify dependencies between certain components (Deployment, Application, etc), you will need to create an ApplicationRelationship resource. Let's look at the gbapp-frontend-app ApplicationRelationship resource from the Helm Chart below:

```
apiVersion: mcm.ibm.com/v1alpha1
kind: ApplicationRelationship
metadata:
  name: {{ template "guestbookapplication.fullname" . }}-app-frontend
  labels:
    app: {{ template "guestbookapplication.name" . }}
    chart: {{ .Chart.Name }}-{{ .Chart.Version | replace "+" "_" }}
    release: {{ .Release.Name }}
    heritage: {{ .Release.Service }}
spec:
  type: contains
  source:
    kind: Application
    name: {{ template "guestbookapplication.fullname" . }}
  destination:
    kind: Deployable
    name: {{ template "guestbookapplication.fullname" . }}-frontend
```

```
apiVersion: mcm.ibm.com/v1alpha1
kind: ApplicationRelationship
metadata:
  name: guestbook-gbapp-app-frontend
  namespace: default
  labels:
    app: gbapp
    chart: gbapp-0.1.0
    destinationKind: Deployable
    destinationName: guestbook-gbapp
    -frontend
    heritage: Tiller
    release: guestbook
    sourceKind: Application
    sourceName: guestbook-gbapp
spec:
  destination:
    name: guestbook-gbapp-frontend
    kind: Deployable
  source:
    name: guestbook-gbapp
    kind: Application
  type: contains
```

# Placement Policy Resource - MCM

## Placement Policy Resource

MCM expands on this capability with Placement Policies by allowing you to deploy workloads through it by specifying cluster labels, which are much easier to remember and manage. By adopting MCM into your CI/CD pipelines, you only have to remember the credentials for the MCM Hub cluster. To deploy workloads to multiple clusters, you only have to provide the number of application replicas and the cluster labels that match the clusters that you would like to deploy them to.

```
apiVersion: mcm.ibm.com/v1alpha1
kind: PlacementPolicy
metadata:
  name: {{ template "guestbookapplication.fullname" . }}
  labels:
    app: {{ template "guestbookapplication.name" . }}
    chart: {{ .Chart.Name }}-{{ .Chart.Version | replace "+" "-" }}
    release: {{ .Release.Name }}
    heritage: {{ .Release.Service }}
    name: {{ template "guestbookapplication.fullname" . }}
    servicekind: CacheService
spec:
  clusterReplicas: {{ .Values.replicaCount }}
  clusterLabels:
    matchLabels:
      {{ toYaml .Values.targetCluster.labelSelector.matchLabels | indent 6 }}
    resourceHint:
      {{ toYaml .Values.targetCluster.resourceSelector | indent 4 }}
  compliances: {{ .Values.targetCluster.compliances }}
```



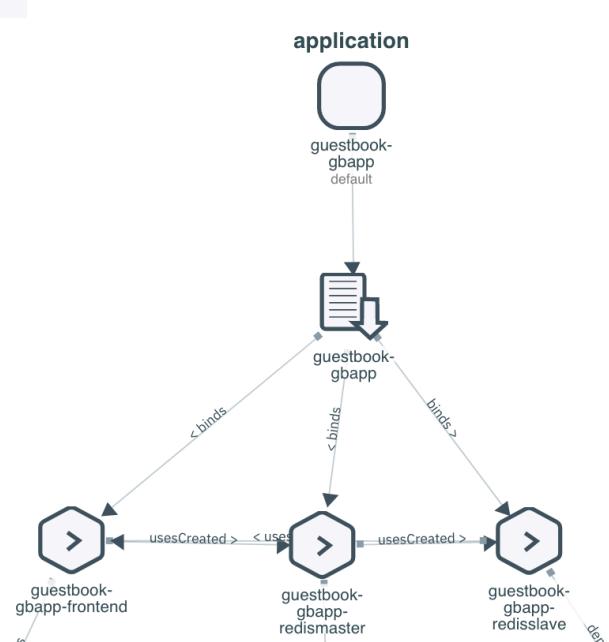
```
apiVersion: mcm.ibm.com/v1alpha1
kind: PlacementPolicy
metadata:
  name: guestbook-gbapp
  namespace: default
  labels:
    name: guestbook-gbapp
    app: gbapp
    chart: gbapp-0.1.0
    heritage: Tiller
    release: guestbook
    servicekind: CacheService
spec:
  clusterLabels:
    matchLabels:
      environment: Dev
  clusterReplicas: 1
  resourceHint:
    type: cpu
  resourceSelector: {}
```

# Placement Binding Resource - MCM

## Placement Binding Resource

Just like how a Kubernetes Role *requires RoleBinding to attach it to a specific ServiceAccount*, a PlacementPolicy requires another resource that binds its rules to specific resources. This MCM resource is the PlacementBinding, which can bind the PlacementPolicy rules to specific Deployables. Let's look at the gbapp PlacementBinding resource from the Helm Chart below:

```
apiVersion: mcm.ibm.com/v1alpha1
kind: PlacementBinding
metadata:
  name: {{ template "guestbookapplication.fullname" . }}
  labels:
    app: {{ template "guestbookapplication.name" . }}
    chart: {{ .Chart.Name }}-{{ .Chart.Version | replace "+" "_" }}
    release: {{ .Release.Name }}
    heritage: {{ .Release.Service }}
    name: {{ template "guestbookapplication.fullname" . }}
    servicekind: CacheService
placementRef:
  apiGroup: mcm.ibm.com
  kind: PlacementPolicy
  name: {{ template "guestbookapplication.fullname" . }}
subjects:
- apiGroup: mcm.ibm.com
  kind: Deployable
  name: {{ template "guestbookapplication.fullname" . }}-frontend
- apiGroup: mcm.ibm.com
  kind: Deployable
  name: {{ template "guestbookapplication.fullname" . }}-redismaster
- apiGroup: mcm.ibm.com
  kind: Deployable
  name: {{ template "guestbookapplication.fullname" . }}-redisslave
```



```
apiVersion: mcm.ibm.com/v1alpha1
kind: PlacementBinding
metadata:
  name: guestbook-gbapp
  namespace: default
  labels:
    name: guestbook-gbapp
    app: gbapp
    chart: gbapp-0.1.0
    heritage: Tiller
    placementPolicy: guestbook-gbapp
    release: guestbook
    servicekind: CacheService
placementRef:
  name: guestbook-gbapp
  apiGroup: mcm.ibm.com
  kind: PlacementPolicy
subjects:
- name: guestbook-gbapp-frontend
  apiGroup: mcm.ibm.com
  kind: Deployable
- name: guestbook-gbapp-redismaster
  apiGroup: mcm.ibm.com
  kind: Deployable
- name: guestbook-gbapp-redisslave
  apiGroup: mcm.ibm.com
  kind: Deployable
```

# Lab Guestbook

# Guestbook - MCM

# Clone the Repository

```
git clone https://github.com/ibm-cloud-architecture/kubernetes-multicloud-management.git
```

Also, open a new browser tab and go to the Applications page on the MCM Dashboard, as shown below:

The screenshot shows the IBM Multicloud Manager Overview dashboard. On the left, a sidebar lists navigation options: Overview (selected), Search, Clusters, Policies, Applications (selected), Helm Releases, Pods, Nodes, Storage, and Topology. The main content area displays cluster statistics: 3 Clusters, 1 Kubernetes Types, 1 Regions, 37 Nodes, and 75 Pods. Below these stats is a section titled 'Used by: nodes' with a note '(19) 19 above'. At the bottom, there's a 'Topology' section with a grid of colored dots (red, yellow, green) representing node connections.

You should then be greeted by an empty Applications page, which we will be populating later:

The screenshot shows the IBM Multicloud Manager Applications dashboard. The title bar says 'IBM Multicloud Manager'. The main content area has a heading 'Applications' and a message 'No Applications found.' followed by the subtext 'Your cluster does not contain any Applications.' A large bee icon is centered on the page. At the bottom right, there is a blue button labeled 'Create Application' with a plus sign.

# Guestbook - MCM

## 0. Create Image Policies on Both Clusters

Since ICP version 3.1, you are required to create Image Policies that allow you to pull Docker images from specific Docker registries (gcr.io in our case). To do so, let's run the following commands on EACH ICP CLUSTER:

```
# Login to the ICP Cluster  
cloudctl login -a https://ICP_MASTER_IP:8443 -n default --skip-ssl-validation  
# Go to Helm Charts directory  
cd demos/guestbook  
# Create the Image Policy in the ICP Cluster  
kubectl apply -f demos/guestbook/guestbook-cluster-image-policy.yaml
```

# Guestbook - MCM

## 1. Deploy to Dev Cluster

The process of deploying an MCM Application is the same as deploying a Helm Chart. To deploy the gbapp MCM Application Helm chart, run the commands below:

```
# Log into MCM HUB Cluster
```

```
cloudctl login -a https://HUB_CLUSTER_MASTER_IP:8443 -n default --skip-ssl-validation
```

```
# If not already there, go to the Helm Charts directory
```

```
cd demos/guestbook
```

```
# Deploy MCM Application Helm Chart
```

```
helm upgrade --install guestbook --set replicaCount=1 --set  
targetCluster.labelSelector.matchLabels.environment=Dev ./gbapp --tls
```

- **replicaCount**: is a field that indicates the number of application instances we are deploying, which is 1.

- **targetCluster.labelSelector.matchLabels.environment**: means we are using the environment field with a value of Dev, which tells the MCM Controller to find a cluster that has that value for that field.

- Checkout the [demos/guestbook/gbapp/values.yaml](#) to learn the different fields we can use as cluster selectors.

# Guestbook - MCM

Assuming that everything went well, you should now see the guestbook-gbapp MCM Application show up in the Applications page as shown below:

The screenshot shows the IBM Multicloud Manager Applications page. The top navigation bar includes links for Create resource, Catalog, Docs, Support, and a user icon. The main title is "Applications". A search bar is at the top left, and a "Create Application" button is at the top right. The main content area displays a table of applications. The table has columns for Name, Namespace, Labels, Created, and Dashboard. One row is visible, showing "guestbook-gbapp" in the Name column, "default" in the Namespace column, and "app=gbapp,chart=gbapp-0.1.0,heritage=Tiller,name=guestbook-gbapp,release=guestbook" in the Labels column. The Created column shows "22 minutes ago". The Dashboard column contains "Launch" and "Health View" links. At the bottom, there are pagination controls for items per page (set to 20) and page navigation.

Name	Namespace	Labels	Created	Dashboard
guestbook-gbapp	default	app=gbapp,chart=gbapp-0.1.0,heritage=Tiller,name=guestbook-gbapp,release=guestbook	22 minutes ago	<a href="#">Launch</a> <a href="#">Health View</a>

items per page **20** | 1-1 of 1 items

1 of 1 pages < >

# Guestbook - MCM

## 2. Exploring the Application Details Page

IBM Multicloud Manager

Create resource Catalog Docs Support

Applications / guestbook-gbapp / guestbook-gbapp

[Overview](#) [Diagram](#)

**Application Details**

Type	Detail
Name	guestbook-gbapp
Namespace	default
Created	23 minutes ago
Labels	app = gbapp chart = gbapp-0.1.0 heritage = Tiller name = guestbook-gbapp release = guestbook
Selector	<b>matchExpressions =</b> <pre>[{"key": "app", "operator": "In", "values": ["guestbook", "gbf", "gbm", "gbrs"]}]</pre>

IBM Multicloud Manager

Create resource Catalog Docs Support

guestbook-gbapp

**Application Deployments**

Name	Namespace	Cluster	Helm Release	Status	Reason	Created
guestbook-gbapp-frontend-se-dev-312-ubuntu	se-dev-312-ubuntu	se-dev-312-ubuntu	md-guestbook-gbapp-frontend-se-dev-312-ubuntu	Completed	-	2 minutes ago
guestbook-gbapp-redismaster-se-dev-312-ubuntu	se-dev-312-ubuntu	se-dev-312-ubuntu	md-guestbook-gbapp-redismaster-se-dev-312-ubuntu	Completed	-	2 minutes ago
guestbook-gbapp-redisslave-se-dev-312-ubuntu	se-dev-312-ubuntu	se-dev-312-ubuntu	md-guestbook-gbapp-redisslave-se-dev-312-ubuntu	Completed	-	2 minutes ago

Items per page: 20 | 1-3 of 3 items

1 of 1 pages | < 1 >

**Placement Policies**

Name	Namespace	Replicas	Cluster Selector	Resource Selector	Decisions	Created
guestbook-gbapp-app-frontend	default	1	matchLabels = { "environment": "Dev" }	-	se-dev-312-ubuntu	3 minutes ago
guestbook-gbapp-app-redismaster	default	1	matchLabels = { "environment": "Dev" }	-	se-dev-312-ubuntu	3 minutes ago

Items per page: 20 | 1-2 of 2 items

1 of 1 pages | < 1 >

IBM Multicloud Manager

Create resource Catalog Docs Support

Placement Bindings

Name	Namespace	Placement Policy	Subjects	Created
guestbook-gbapp	default	guestbook-gbapp	guestbook-gbapp-frontend(Deployable), guestbook-gbapp-redismaster(Deployable), guestbook-gbapp-redisslave(Deployable)	an hour ago

Items per page: 20 | 1-1 of 1 items

1 of 1 pages | < 1 >

Deployable Objects

Name	Namespace	Deployer Details	Dependencies	Created
guestbook-gbapp-frontend	default	Chart URL = https://raw.githubusercontent.com/ibm-cloud-architecture/kubernetes-multicloud-management/v312/demos/guestbook/chart-packages/gbf-0.1.0.tgz Namespace = default	-	an hour ago
guestbook-gbapp-redismaster	default	Chart URL = https://raw.githubusercontent.com/ibm-cloud-architecture/kubernetes-multicloud-management/v312/demos/guestbook/chart-packages/gbm-0.1.0.tgz Namespace = default	-	an hour ago
guestbook-gbapp-redisslave	default	Chart URL = https://raw.githubusercontent.com/ibm-cloud-architecture/kubernetes-multicloud-management/v312/demos/guestbook/chart-packages/gbs-0.1.0.tgz Namespace = default	-	an hour ago

Items per page: 20 | 1-3 of 3 items

1 of 1 pages | < 1 >

IBM Multicloud Manager

Create resource Catalog Docs Support

guestbook-gbapp

**Application Relationships**

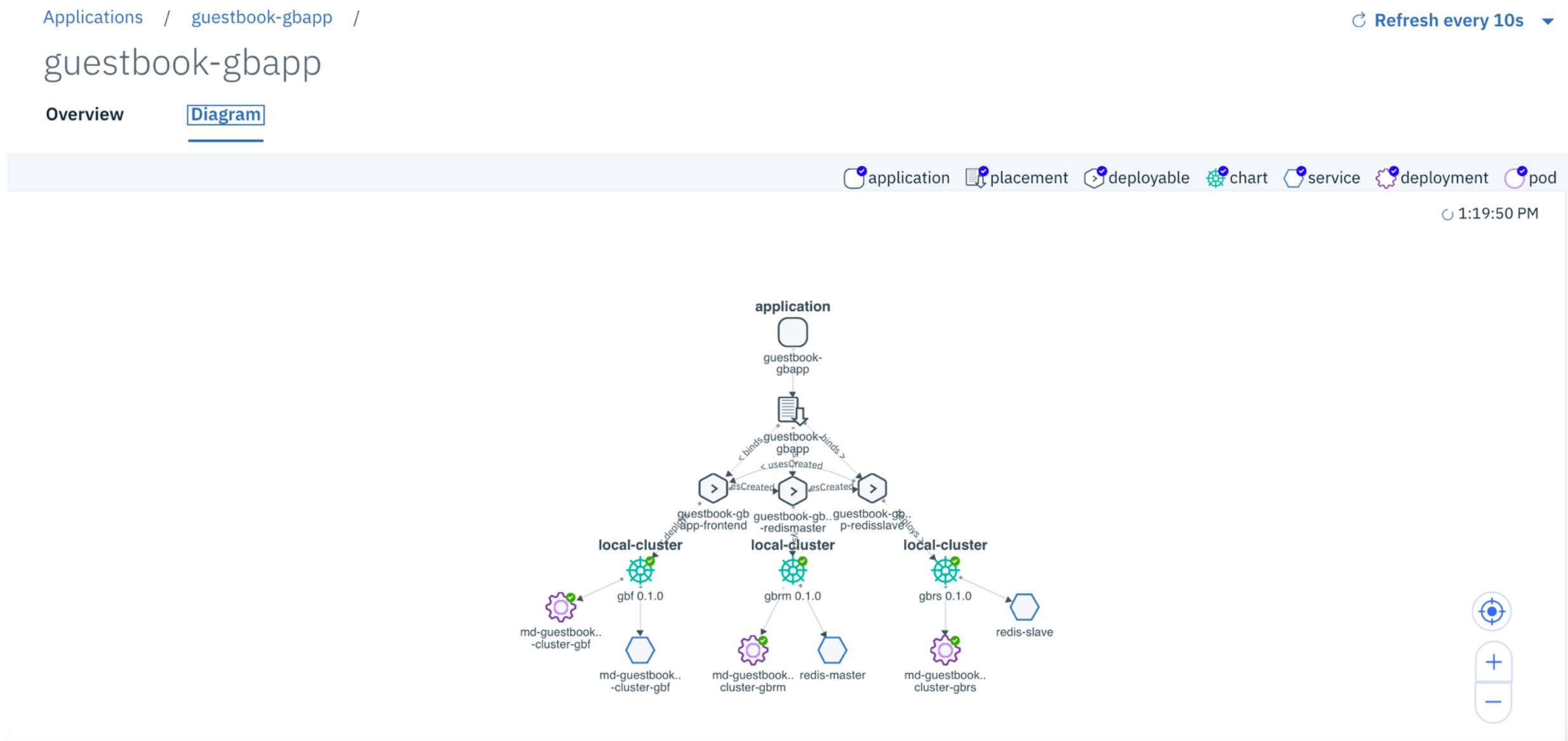
Name	Namespace	Source	Destination	Type	Created
guestbook-gbapp-frontend	default	name = guestbook-gbapp-kind = guestbook-gbapp	name = guestbook-gbapp-frontendl kind = guestbook-gbapp-frontendl	contains	20 minutes ago
guestbook-gbapp-app-redismaster	default	name = guestbook-gbapp-kind = guestbook-gbapp	name = guestbook-gbapp-redismaster-kind = guestbook-gbapp-redismaster	contains	20 minutes ago
guestbook-gbapp-app-redisslave	default	name = guestbook-gbapp-kind = guestbook-gbapp	name = guestbook-gbapp-redisslave-kind = guestbook-gbapp-redisslave	contains	20 minutes ago
guestbook-gbapp-frontend-redismaster	default	name = guestbook-gbapp-kind = guestbook-gbapp-frontendl	name = guestbook-gbapp-redismaster-kind = guestbook-gbapp-redismaster	usesCreated	20 minutes ago
guestbook-gbapp-redismaster-redisslave	default	name = guestbook-gbapp-kind = guestbook-gbapp-redismaster	name = guestbook-gbapp-redisslave-kind = guestbook-gbapp-redisslave	usesCreated	20 minutes ago
guestbook-gbapp-redisslave-frontend	default	name = guestbook-gbapp-kind = guestbook-gbapp-redisslave	name = guestbook-gbapp-frontendl kind = guestbook-gbapp-frontendl	usesCreated	20 minutes ago

Items per page: 20 | 1-6 of 6 items

1 of 1 pages | < 1 >

# Guestbook - MCM

On the Diagram tab, you will see a visual representation of the Application and all of the resources we inspected above, including the Helm Charts resources (Pods, Deployments, and Services). This diagram is available thanks to the [Weave Scope](#) component in MCM, which takes the Application Relationship objects and its own algorithm to deduce and reduce the application topology on the screen. If you click on any of the components you will open a small view with either a short description or the YAML resource file that was used to create the resource.



# Guestbook - MCM

## 3. Redeploy to Staging Cluster

To redeploy the application to the se-stg-31 cluster, all we have to do is change the value of the environment field to Staging, and MCM will take care of uninstalling the application from the se-dev-31 cluster and then redeploying it to the se-stg-31 cluster.

# Deploy MCM Application Helm Chart

```
helm upgrade --install guestbook --set replicaCount=1 --set  
targetCluster.labelSelector.matchLabels.environment=Staging ./gbapp --tls
```



se-dev-31 cluster



se-stg-31 cluster

```
targetCluster:  
  labelSelector:  
    matchLabels:  
      #vendor: "ICP"  
      #region: "US"  
      #environment: Dev  
      #datacenter: "toronto"  
      #owner: "marketing"
```

# Guestbook - MCM

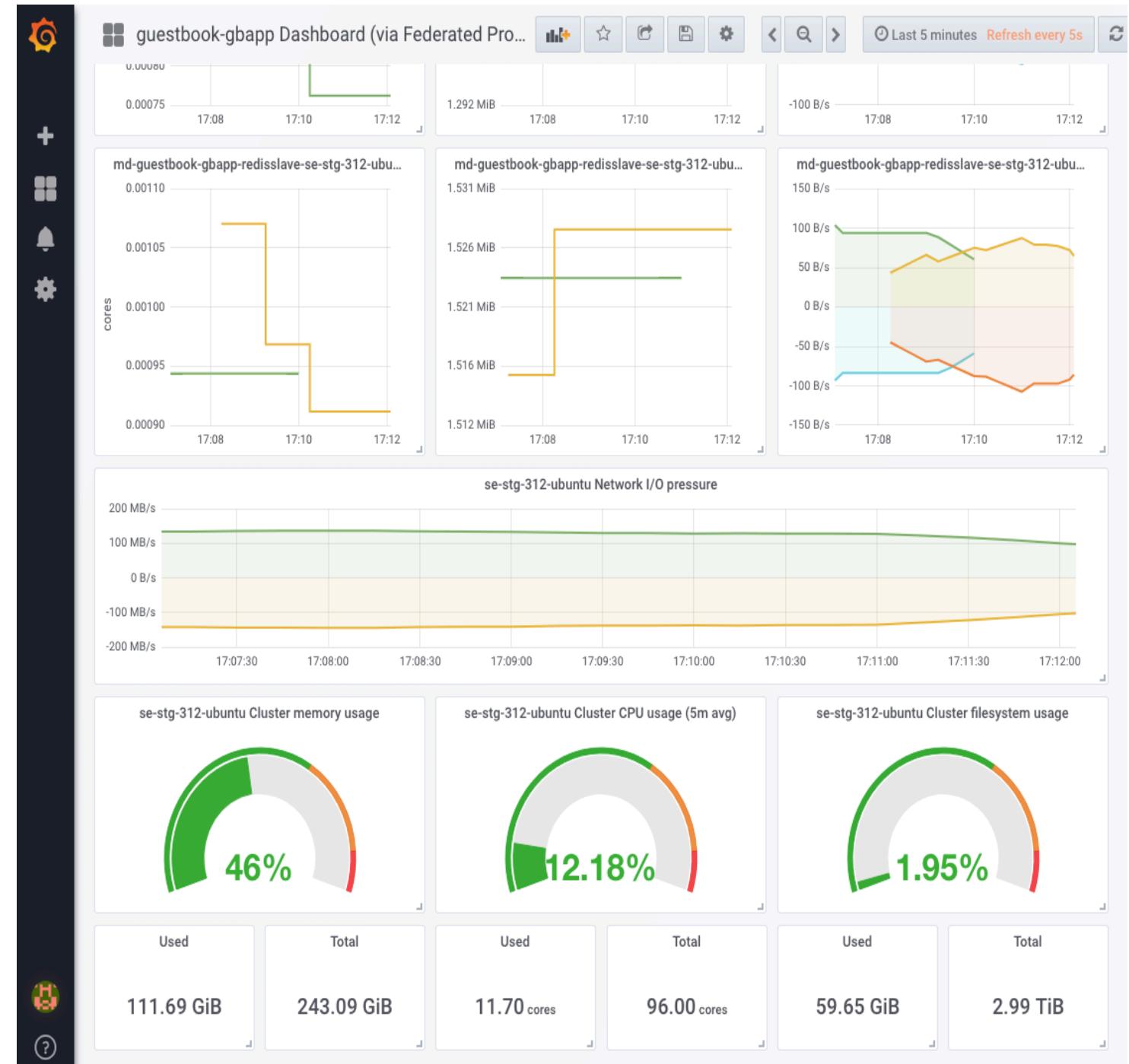
## 4. Health View Dashboard

Now click the Launch Health View link under the Dashboard column and it should open a Grafana dashboard in a new page that looks similar to the following:

The screenshot shows the IBM Multicloud Manager Applications dashboard. At the top, there is a search bar labeled "Search" and a "Create Application" button. Below the search bar is a table with the following columns: Name, Namespace, Labels, Created, and Dashboard. There is one item listed:

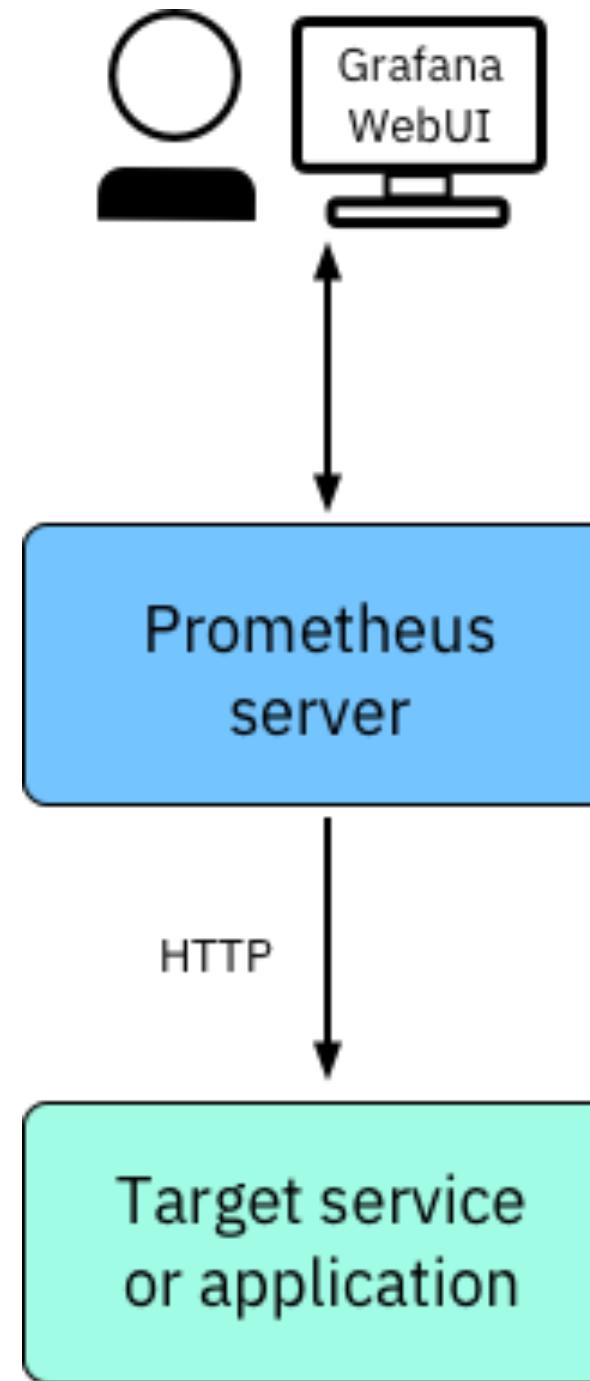
Name	Namespace	Labels	Created	Dashboard
<a href="#">guestbook-gbapp</a>	default	app=gbapp,chart=gbapp-0.1.0,heritage=Tiller,name=guestbook-gbapp,release=guestbook	22 minutes ago	<a href="#">Launch Health View</a>

At the bottom of the table, there are pagination controls: "items per page" set to 20, "1-1 of 1 items", and navigation arrows.



# Build-to-Manage

- Prometheus collects metrics from monitored systems by "scraping" metrics on the HTTP endpoints of those systems
- the things that Prometheus monitors are *targets*
- The Prometheus server scrapes targets at a defined interval and stores the information in a time-series database
- The targets and the time interval for scraping metrics is defined in the `prometheus.yml` configuration file



Grafana dashboards are used to visualize the information collected by Prometheus server

At pre-determined intervals, the Prometheus server scrapes build-to-manage target services or applications

# Helm Chart from Catalog - MCM

## 4. Deploy sample chart services to remote cluster from catalog

The screenshot shows the MCM (Multi-Cluster Management) interface for deploying a Helm chart. On the left, the 'sample-chart V 1.1.2' details page is visible, including sections for Overview, Configuration, Sample chart for contribution guidelines, ibm-community-charts, Licenses, Release Notes, Chart Version (1.1.2), Details & Links (Type: Helm Chart, Published: December 6, 2018), and Source & Tar Files. A 'Configure' button is at the bottom right of this panel.

The main area shows the deployment configuration:

- Helm release name \***: Enter value
- Target namespace \***: multicluster-endpoint
- Target Cluster \**i***: Select a cluster (dropdown menu open, showing local-cluster and mycluster)
- License \***:  I have read and agreed to the [License agreement](#)

At the bottom, a terminal window displays the output of kubectl commands:

```
nginx-7cddb8cdc9-wzpgt 1/1 Running 0 2d1h
[mikes-mbp:sample mikechang$ kubectl get pods -n default
NAME        READY   STATUS    RESTARTS   AGE
nginx-7cddb8cdc9-f5f5t  1/1    Running   0          2d1h
nginx-7cddb8cdc9-wzpgt  1/1    Running   0          2d1h
[mikes-mbp:sample mikechang$ kubectl describe pod nginx-7cddb8cdc9-f5f5t
Name:           nginx-7cddb8cdc9-f5f5t
Namespace:      default
Priority:       0
PriorityClassName: <none>
Node:           119.81.151.156/119.81.151.156
Start Time:     Sat, 29 Jun 2019 14:45:53 +0800
Labels:         pod-template-hash=7cddb8cdc9
Annotations:   kubernetes.io/psp: ibm-privileged-psp
Status:         Running
IP:             192.168.110.157
Controlled By: ReplicaSet/nginx-7cddb8cdc9
Containers:
  nginx:
    Container ID: docker://7cd1e6e764c2575066e7d9fc34f9ed2acb33e4211aa0336e1d0161fddb5a970c
    Image:         nginx
    Image ID:     docker-pullable://nginx@sha256:bdbf36b7f1f77ffe7bd2a32e59235dff6ecf131e3b6b5b96061c6
    Port:          <none>
    Host Port:    <none>
    State:        Running
      Started:   Sat, 29 Jun 2019 14:46:06 +0800
    Ready:        True
    Restart Count: 0
    Environment:  <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-nbhx8 (ro)
Conditions:  ...
```

# Another Sample - jpetstore

<https://medium.com/ibm-cloud/creating-and-deploying-applications-to-ibm-multicloud-manager-410e71604c9b>

git clone https://github.com/ibm-cloud/jpetstore-kubernetes

The screenshot shows the IBM Cloud Catalog interface. At the top, there is a navigation bar with 'IBM Cloud Private' (selected), 'mycluster', 'Create resource', 'Catalog', 'Docs', 'Support', and a user icon. Below the navigation bar is a search bar containing the text 'pet'. To the left of the search bar is a sidebar titled 'Catalog' with a list of categories: All Categories (selected), DevOps, Operations, Security, Blockchain, Network, Data Science & Analytics, Data, Integration, Storage, Tools, and IoT. To the right of the search bar is a main content area titled 'Helm Charts' which displays two chart entries: 'mmssearch' (local-charts) and 'modernpets' (local-charts). Both entries include a brief description and a small icon.

# **MCM GRC dashboard**

# **IBM Multicloud Manager**

# **policy framework**

# GRC dashboard overview

A single page to view the overall status of policy across all clusters

Cards are customizable and can be organized by categories and standards

Clickable to show details

Filterable

The screenshot shows the IBM Multicloud Manager GRC dashboard. At the top, there's a navigation bar with 'IBM Multicloud Manager' and links for 'Create resource', 'Catalog', and user icons. To the right, there are refresh and filter buttons, and the timestamp '8:45:23 PM'.

The main area is titled 'Policies' with an information icon. It has two tabs: 'Overview' (which is selected) and 'All policies'. Below this, there's a section titled 'Top violations' with a 'Clusters' dropdown.

Two cards are displayed under 'Top violations':

- cluster1**: 3 Policy Violations, including 'compliance-all', 'policy-ma', and 'policy-pod'.
- clusterhub**: 2 Policy Violations, including 'compliance-all' and 'policy-pod'.

Below this, there's a 'Policy overview' section with a 'Standards' dropdown. It shows three categories: NIST, HIPAA, and Other.

For NIST:

Cluster violations	Policy violations
2 /2	2 /3

For HIPAA:

Cluster violations	Policy violations
2 /2	1 /2

For Other:

Cluster violations	Policy violations
2 /2	1 /1

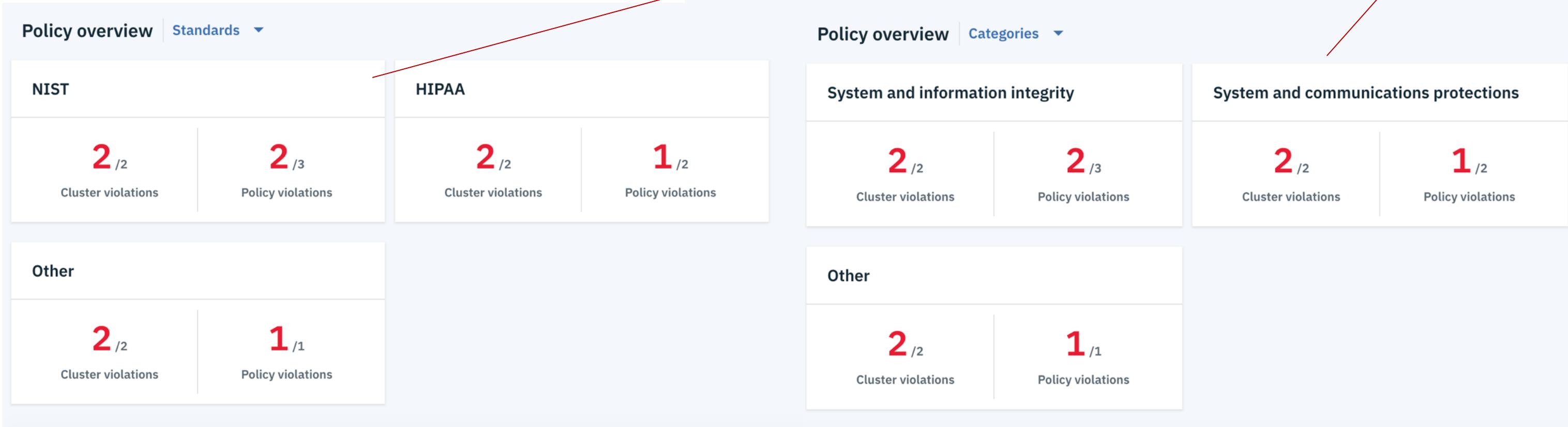
# Customizing dashboard cards

## Annotations metadata

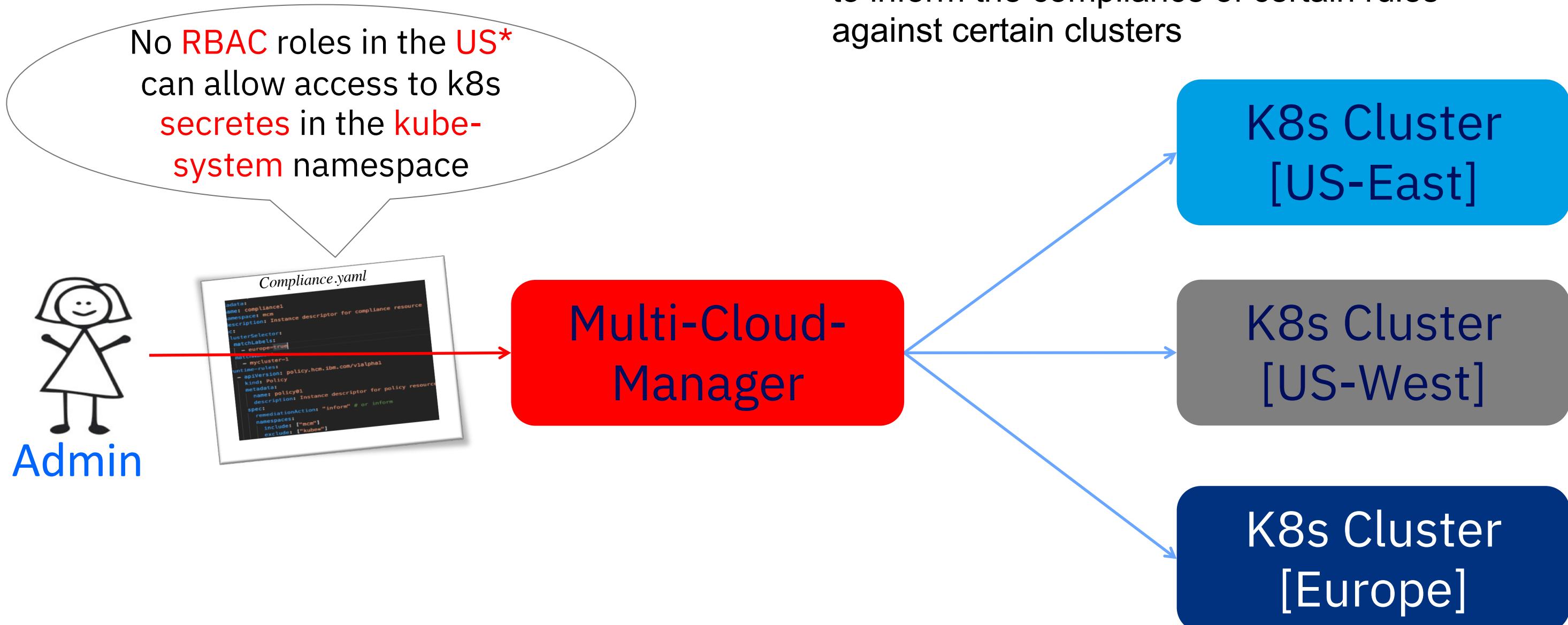
- policy.mcm.ibm.com/categories
- policy.mcm.ibm.com/standards
- policy.mcm.ibm.com/controls

Search							Create policy +
Policy name	Namespace	Remediation	Cluster violations	Controls	Standards	Categories	
> <a href="#">compliance-all (Deprecated)</a>	mcm	-	2/2	-	-	-	
> <a href="#">policy-ma</a>	mcm	enforce	1/2	Mutation Advisor	NIST	System And Information Integrity	
> <a href="#">policy-pod</a>	mcm	inform	2/2	Mutation Advisor, VA	NIST, HIPAA	System And Communications Protections, System And Information Integrity	
> <a href="#">policy-role</a>	mcm	enforce	0/2	Mutation Advisor, VA	NIST, HIPAA	System And Communications Protections, System And Information Integrity	

items per page 10 | 1-4 of 4 items



# IBM Multicloud Manager Policy overview



# Multicloud Manager

Multicloud Manager Compliance Manager provides a state-based management approach to inform and enforce the policy compliance of a set of clusters that are managed by Multicloud Manager.

MCM addresses these challenges with its Compliance Manager

- MCM Compliance Manager works based on three attributes:
  - Template
  - Policy
  - Compliance.

# Multicloud Manager - Template

Captured in a YML file, template is very important

- Defines a **list of policyRules** that determine whether a Kubernetes resource is compliant

- These rules are a list of attributes that the Kubernetes resource must have or not have to be compliant

```
apiVersion: roletemplate.mcm.ibm.com/v1alpha1
kind: RoleTemplate
complianceType: "musthave" # at this level, it means the role must exist with the rules that it
metadata:
  namespace: "" # will be inferred
  name: operator
rules:
  - complianceType: "musthave" # at this level, it means if the role exists the rule is a musthave
    policyRule:
      apiGroups: ["extensions", "apps"]
      resources: ["deployments"]
      verbs: ["get", "list", "watch", "delete"]
  - complianceType: "mustnothave" # at this level, it means if the role exists the rule is a mustnothave
    policyRule:
      apiGroups: ["core"]
      resources: ["pods"]
      verbs: ["create", "update", "patch"]
  - policyRule:
      apiGroups: ["core"]
      resources: ["secrets"]
      verbs: ["get", "watch", "list", "create", "delete", "update", "patch"]
```

Defines the template kind, which is RoleTemplate in this case.

The path where the custom API is defined

Determines whether the resource (Role in this case) that matches the given rules must exist or not.

Array of API Groups to apply this rule against

List of rules (policyRule in this case) that will determine the compliance of a resource (Role in this case) as "musthave" and "mustnothave"

Array of Resources from API Groups mentioned above to apply this rule against.

Array of verbs that apply to the API Groups and Resources mentioned above the verbs

# Multicloud Manager - Policy

A policy is responsible for these actions:

- Take a list of templates and enforce their rules in a cluster
- Determine which namespaces to enforce the rules into
- Determine what remediation action to take when compliance isn't met

This is where you specify what you would like the Compliance Manager to do when there is a non-compliance for the given Policy, "enforce" in this case

```
apiVersion: policy.mcm.ibm.com/v1alpha1
kind: Policy
metadata:
  name: policy02
  description: Instance descriptor for policy resource
spec:
  remediationAction: "enforce" # or inform
  namespaces:
    include: ["default"]
    exclude: ["kube*"]
```

The path where the custom API is defined

Defines the template kind, which is *Policy* in this case.

Namespaces where the policies will be enforced in

# Multicloud Manager - Compliance

The Multicloud Manager Compliance Manager can manage policies for you by using a compliance object.

A compliance object takes a list of policy objects and their respective templates and determines what clusters to apply them to based on cluster selectors.

A compliance is the final piece that ties the policies into the actual infrastructure (clusters) and fully implements compliance.

This is the name of the namespace that was created specifically for MCM to put Compliance objects during MCM installation

```
apiVersion: compliance.mcm.ibm.com/v1alpha1
kind: Compliance
metadata:
  name: compliance1
  namespace: mcm
  description: Instance descriptor for compliance resource
spec:
  clusterSelector:
    matchNames:
      - "se-stg-31"
      - "se-dev-31"
    #   matchLabels:
    #     cloud: "IBM"
    #     hipaa: "true"
    #   matchExpressions:
    #     - key: key1
    #       operator: "NotIn"
    #       values:
    #         - "c13"
    #         - "c14"
    #   matchConditions:
    #     - type: "OK"
    #       status: "True"
  runtime-rules:
    - apiVersion: policy.mcm.ibm.com/v1alpha1
      kind: Policy
      metadata:
        name: policy01
        description: Instance descriptor for policy resource
```

The path where the custom API is defined

Defines the template kind, which is *Compliance* in this case.

Description of the Compliance object, which will be displayed in the MCM Dashboard.

**clusterSelector:** Here is where we define the criteria to select the clusters where the policies will be applied and/or enforced. You can use a combination of the following filters to select clusters:

**matchNames:** List of cluster names to match against.

**matchLabels:** List of cluster labels to match against. These are the labels which fields you for a cluster during Klusterlet installation.

**matchExpressions:** List of expressions to match against.

**matchConditions:** List of cluster conditions to match against, usually involving cluster health status.

**runtime-rules:** This is a list of **Policy** objects that will be applied to the clusters that match the criteria in **clusterSelector**

# IBM Multicloud Manager Policy sample

```
apiVersion: policy.mcm.ibm.com/v1alpha1          API-group/version
kind: Policy
metadata:
  name: policy-all
spec:
  remediationAction: "enforce" # enforce or inform
  namespaces:          where
    include: ["default"]
    exclude: ["kube*"]
  role-templates:      template
    - apiVersion: roletemplate.mcm.ibm.com/v1alpha1
      metadata:
        name: default
      complianceType: "musthave" # three types: "musthave" , "mustnothave" , "mustonlyhave"
      rules:
        - complianceType: "mustnothave"           how
          policyRule:
            apiGroups: ["core"]                 what
            resources: ["secrets"]
            verbs: ["get", "list", "watch", "create", "delete", "patch"]
        - complianceType: "musthave"
          policyRule:
            apiGroups: ["extensions", "apps"]
            resources: ["deployments"]
            verbs: ["get", "list", "watch", "create", "delete", "patch"]
```

if the name resolves to a set of RBAC roles, or if the RBAC role already exists, then the controller will perform a deep-compare to check if the existing RBAC roles that are selected based on their names and labels satisfy the rule conditions.

Translates to:  
deployments.extensions musthave [get list watch create delete]  
deployments.apps musthave [get list watch create delete]

## Supported templates

A **role-template** is used to list RBAC roles that must be evaluated or applied to the managed-clusters. Role templates are treated as a special category of templates, as they have rules inside that can be analyzed and compared to evaluate the compliance of a cluster.

- <https://github.ibm.com/IBMPublicCloud/hcm-compliance/blob/master/artifacts/policies-v2/policy-role-v2.yaml>

An **object-template** is used to list any other Kubernetes object that must be evaluated or applied to the managed-clusters. An example of object can be a pod security policy, an image policy, or a limit range.

- <https://github.ibm.com/IBMPublicCloud/hcm-compliance/blob/master/artifacts/policies-v2/policy-pod-v2.yaml>

A **policy-template** is used to create one or more policies for third party or external security controls. For example, you can create a mutation policy with the mutation policy controller. For more information, see the [Technology preview](#) for the *Mutation policy controller*.

- <https://github.ibm.com/IBMPublicCloud/hcm-compliance/blob/master/artifacts/policies-v2/policy-ma-v2.yaml>

# **Lab**

# **Compliance Manager**

# 1. Apply the Compliance

A **Policy** that enforces the creation of the policy's **RoleTemplate** if the cluster is found non-compliant. This means that even if the cluster was found compliant and, for some reason, a cluster admin deletes the role, the Compliance Manager will recreate it.

```
# Login against MCM Hub Cluster  
cloudctl login -a https://${MCM_HUB_CLUSTER_MASTER_IP}:8443 -u ${USERNAME} -p ${PASSWORD} -n mcm --skip-ssl-validation;
```

```
# Clone the Reference Architecture Repository  
https://github.ibm.com/IBMPublicCloud/hcm-compliance/tree/master/artifacts/policies-v2
```

```
# Apply the Compliance  
kubectl apply -f demos/compliance
```

Where:

- \${MCM\_HUB\_CLUSTER\_MASTER\_IP} is the IP Address of the master node in the MCM Hub Cluster.
- \${USERNAME} is admin user.
- \${PASSWORD} is the admin password.
- mcm is the namespace in which the Compliance object will be created.

# Policy-role

```
apiVersion: policy.mcm.ibm.com/v1alpha1
kind: Policy
metadata:
  name: policy-role
  namespace: kube-system
  annotations:
    policy.mcm.ibm.com/standards: HIPAA
    policy.mcm.ibm.com/categories: SystemAndCommunicationsProtections, SystemAndInformationIntegrity
    policy.mcm.ibm.com/controls: MutationAdvisor, VA
spec:
  remediationAction: "enforce" # enforce or inform
  namespaces:
    include: ["default"]
    exclude: ["kube*"]
  role-templates:
    - apiVersion: roletemplate.mcm.ibm.com/v1alpha1
      metadata:
        namespace: "" # will be inferred
        name: operator-role-policy
      selector:
        matchLabels:
          dev: "true"
      complianceType: "musthave" # at this level, it means the role must exist with the rules that it musthave
      rules:
        - complianceType: "musthave" # at this level, it means if the role exists the rule is a musthave
          policyRule:
            apiGroups: ["extensions", "apps"]
            resources: ["deployments"]
            verbs: ["get", "list", "watch", "create", "delete", "patch"]
```

```
---  
apiVersion: mcm.ibm.com/v1alpha1
kind: PlacementPolicy
metadata:
  name: placement-role
  namespace: kube-system
spec:
  clusterLabels:
    matchLabels:
  cloud: "IBM"
```

```
---  
apiVersion: mcm.ibm.com/v1alpha1
kind: PlacementBinding
metadata:
  name: binding-role
  namespace: kube-system
placementRef:
  name: placement-role
  kind: PlacementPolicy
  apiGroup: mcm.ibm.com
subjects:
  - name: policy-role
    kind: Policy
    apiGroup: policy.mcm.ibm.com
```

# Policy-pod

```
apiVersion: policy.mcm.ibm.com/v1alpha1
kind: Policy
metadata:
  name: policy-pod
  namespace: kube-system
  annotations:
    policy.mcm.ibm.com/standards: NIST,HIPAA
    policy.mcm.ibm.com/categories: SystemAndCommunicationsProtections, SystemAndInformationIntegrity
    policy.mcm.ibm.com/controls: MutationAdvisor,VA
spec:
  complianceType: musthave
  namespaces:
    exclude:
      - kube*
    include:
      - default
  object-templates:
    - complianceType: musthave
      objectDefinition:
        apiVersion: v1
        kind: Pod
        metadata:
          name: nginx-pod
        spec:
          containers:
            - image: nginx:1.7.9
              name: nginx
              ports:
                - containerPort: 80
  remediationAction: inform
```

```
---
apiVersion: mcm.ibm.com/v1alpha1
kind: PlacementBinding
metadata:
  name: binding-pod
  namespace: kube-system
placementRef:
  name: placement-pod
  kind: PlacementPolicy
  apiGroup: mcm.ibm.com
subjects:
  - name: policy-pod
    kind: Policy
    apiGroup: policy.mcm.ibm.com
---
apiVersion: mcm.ibm.com/v1alpha1
kind: PlacementPolicy
metadata:
  name: placement-pod
  namespace: kube-system
spec:
  clusterLabels:
    matchLabels:
  cloud: "IBM"
```

## 2. View Compliance Status on the Dashboard

To access the Policies view for the Compliance Manager, open a new browser window and go to [https://\\${MCM\\_HUB\\_CLUSTER\\_MASTER\\_IP}:8443/multicloud/policies](https://${MCM_HUB_CLUSTER_MASTER_IP}:8443/multicloud/policies), where \${MCM\_HUB\_CLUSTER\_MASTER\_IP} is the IP Address of the master node in the MCM Hub Cluster.

The screenshot shows the IBM Multicloud Manager Policies dashboard. At the top, there's a navigation bar with 'IBM Multicloud Manager' on the left, 'Create resource', 'Catalog', and user icons on the right. Below the navigation bar, the title 'Policies' is displayed with a refresh button ('Refresh every 10s') and a filter icon ('Filter'). The date '1:39:19 PM' is also shown. There are two tabs: 'Overview' (which is selected) and 'All policies'. The main area is divided into two sections: 'Top violations' and 'Policy overview'. In the 'Top violations' section, there are two items: 'local-cluster policy-pod' with 1 Policy Violation and 'mycluster policy-pod' with 1 Policy Violation. In the 'Policy overview' section, there are three categories: 'HIPAA', 'NIST', and 'Other'. For HIPAA, there are 2 Cluster violations and 1 Policy violation. For NIST, there are 2 Cluster violations and 1 Policy violation. For Other, there are 0 Cluster violations and 0 Policy violations.

IBM Multicloud Manager

Create resource Catalog

Policies

Refresh every 10s

Filter

1:39:19 PM

Overview All policies

Top violations Clusters

1 local-cluster policy-pod

1 mycluster policy-pod

Policy overview Standards

HIPAA

2 /2 Cluster violations

1 /2 Policy violations

NIST

2 /2 Cluster violations

1 /1 Policy violations

Other

0/0 Cluster violations

0/0 Policy violations

IBM Cloud

Overview

All policies

Policy Overview						
Policy name	Namespace	Remediation	Cluster violations	Controls	Standards	Categories
> <a href="#">policy-role</a>	kube-system	enforce	0/2	Mutation Advisor, VA	HIPAA	System And Communications Protections, System And Information Integrity
> <a href="#">policy2</a>	kube-system	enforce	0/0	-	-	-
> <a href="#">policy3</a>	kube-system	enforce	0/0	Mutation Advisor	NIST	System And Information Integrity, RBAC

items per page [10](#) ▾ | 1-3 of 3 items

Policies / policy-role /

policy-role

Overview

## Policy details

Type	Detail
Name	policy-role
Violation status	0/2
Created	10 minutes ago
Resource version	1390799
Self link	/apis/policy.mcm.ibm.com/v1alpha1/namespaces/kube-system/policies/policy-role
UID	96ea6974-9d51-11e9-b102-06f873ce3092

## Policy YAML template

[Edit](#) +[Submit](#) +

```
1 apiVersion: policy.mcm.ibm.com/v1alpha1
2 kind: Policy
3 metadata:
4   name: policy-role
5   namespace: kube-system
6   annotations:
7     policy.mcm.ibm.com/categories: 'SystemAndCommunicationsProtections,SystemAndInformationIntegrity'
8     policy.mcm.ibm.com/controls: 'MutationAdvisor,VA'
9     policy.mcm.ibm.com/standards: HIPAA
10    seed-generation: '1'
11    finalizers:
12      - propagator.finalizer.mcm.ibm.com
13    generation: 7
14    resourceVersion: '1390799'
15    spec:
16      namespaces:
17        exclude:
18          - kube*
19        include:
20          - default
21        remediationAction: enforce
```

# Policy-pod (Policy Compliant)

## Policy status

Q Search

Cluster name	Policy compliant	Policy valid
local-cluster	1/1	1/1
mycluster	1/1	1/1
items per page <b>10</b> ▾   1-2 of 2 items		1 of 1 pages < <b>1</b> ▾ >

## Policies

Q Search

Compliant	Name	Cluster compliant	Cluster not compliant
<input checked="" type="checkbox"/> Compliant	<a href="#">policy-role</a>	<a href="#">local-cluster</a> <a href="#">mycluster</a>	-
items per page <b>10</b> ▾   1-1 of 1 items			
1 of 1 pages < <b>1</b> ▾ >			

# Policy-pod (Placement policy and bindings)

## Placement policies

Q Search

Name	Namespace	Replicas	Cluster selector	Resource selector	Decisions	Created
placement-role	mcm	-	-	-	local-cluster, mycluster	18 minutes ago
items per page <b>10</b> ▾   1-1 of 1 items						1 of 1 pages < 1 ▾ >

## Placement bindings

Q Search

Name	Namespace	Placement policy	Subjects	Created
binding-role	mcm	placement-role	policy-role(Policy)	18 minutes ago

# MultiCloud Manager – useful links



Manage an IBM Cloud Private on Red Hat OpenShift cluster by using Multicloud Manager  
<https://www.ibm.com/cloud/garage/tutorials/manage-icp-on-openshift-with-mcm>

Tutorial on Garage Method

<https://www.ibm.com/cloud/garage/content/course/ibm-multicloud-manager/0>

Mulitcloud Manager home page

<https://www.ibm.com/cloud/multicloud-manager>

Multicloud Manager on IBM Blue Demos

<https://bluedemos.com/show/1719>

Cookbook for IBM Multicloud Manager Solution Architects

<https://pages.github.ibm.com/CASE/refarch-mcm/>

