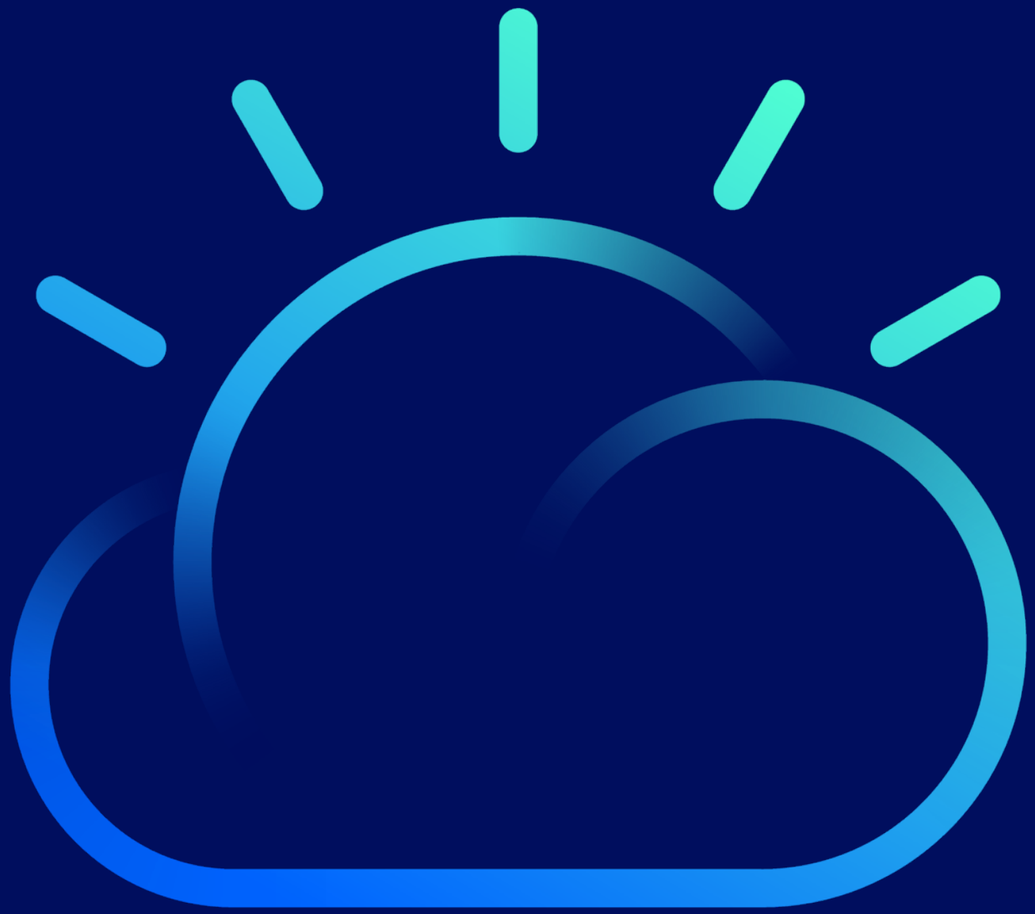
ICP-MCM 201

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

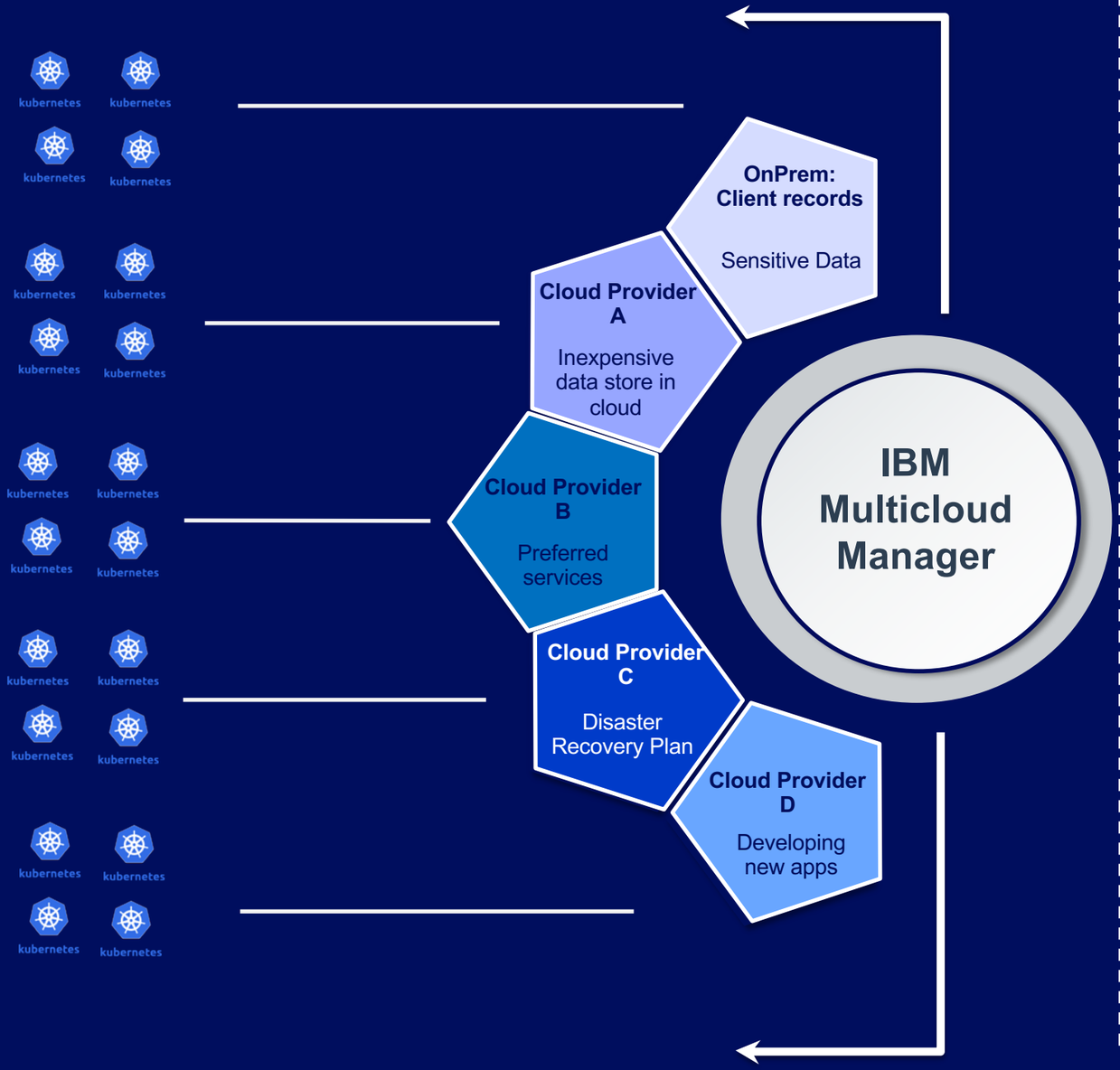
Multiple clusters to manage geographical data residency laws

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

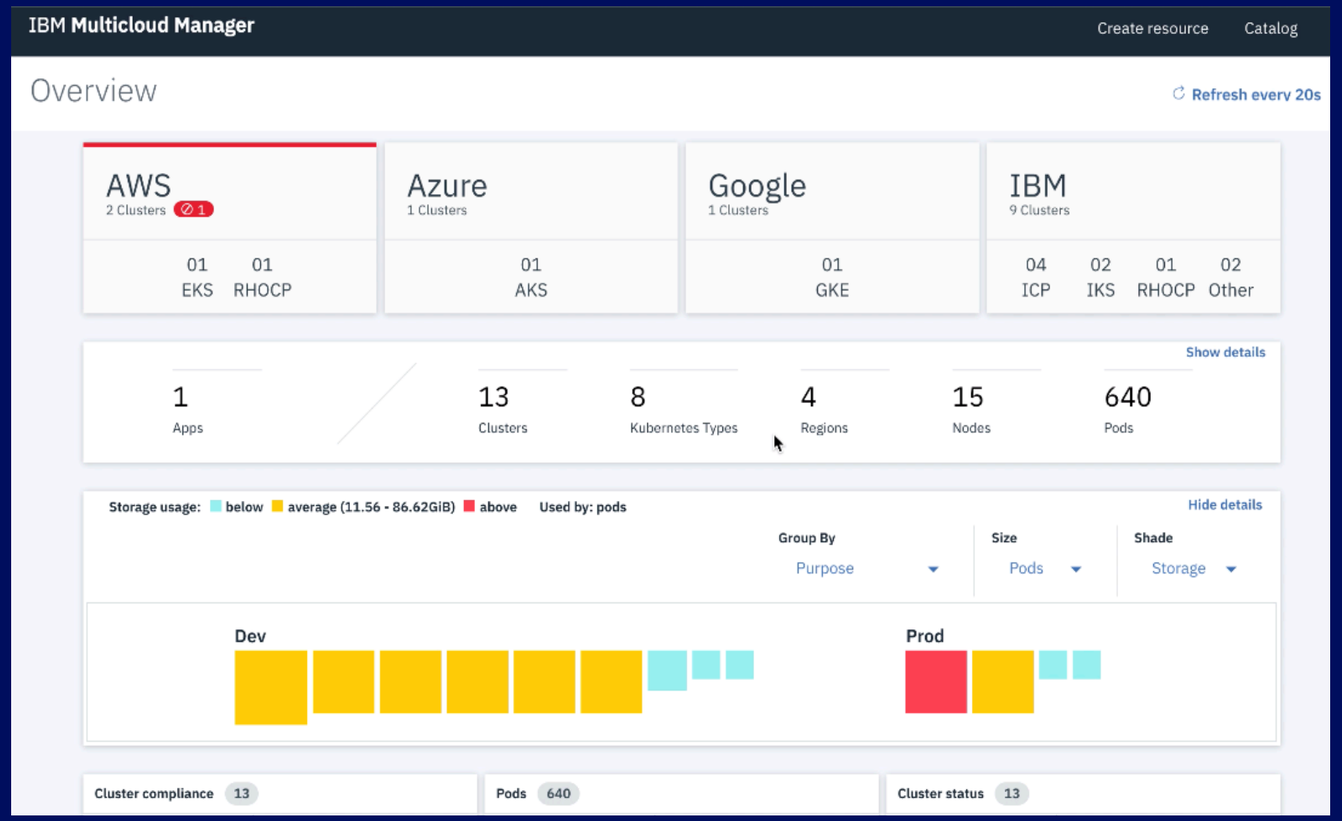
I need automated Governance

I need seamless Application Management

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

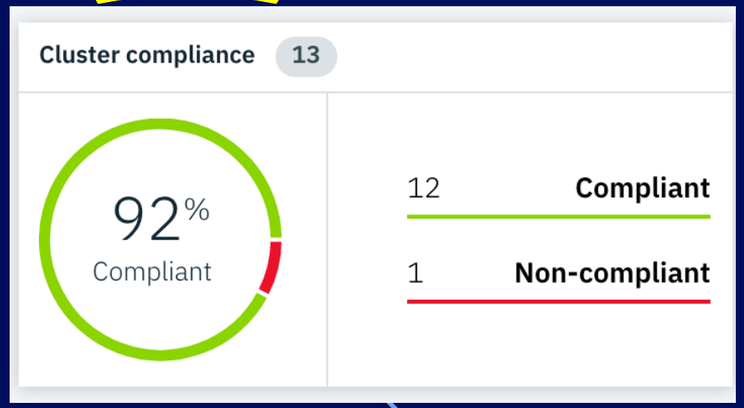
- Where are the failed components?
- Where are my services running?
- How can I monitor applications across clusters and clouds?
- How can I manage clusters as if they were one environment?
- How do I monitor usage across clouds?

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



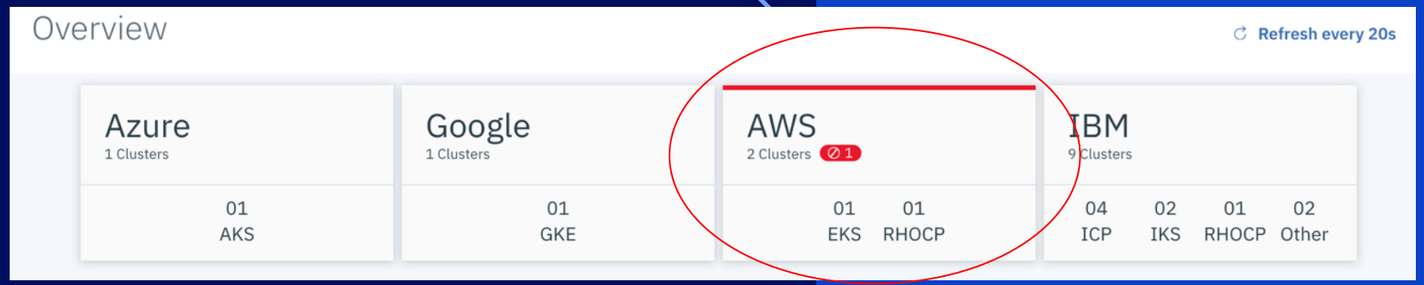
Governance: Maintain controls across applications & clusters with policies

Something new in 3.2

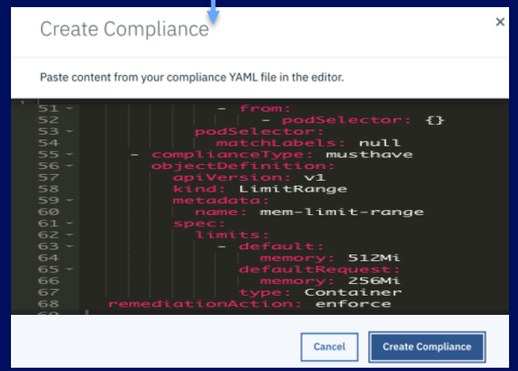


I need automated Governance

- 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?

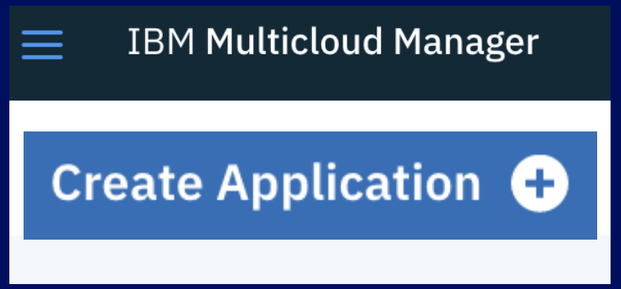


Create **Placement** & **Security policies** directly from MCM console and push to all clusters **with a click**

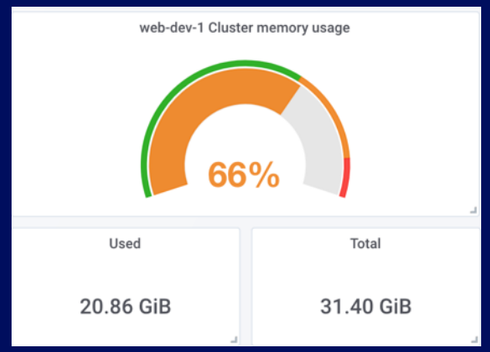


Application Management: Create, Monitor, Manage and Backup

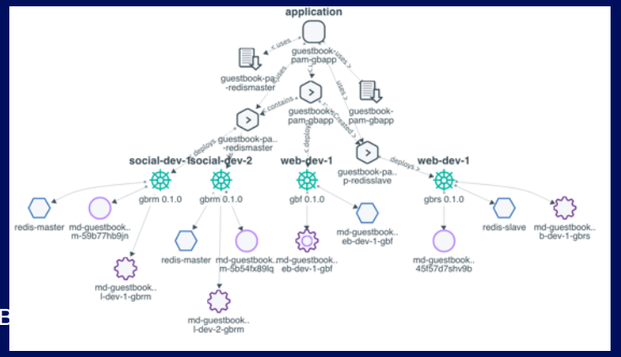
Something new in 3.2



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



Monitor your application with **Grafana/Prometheus**



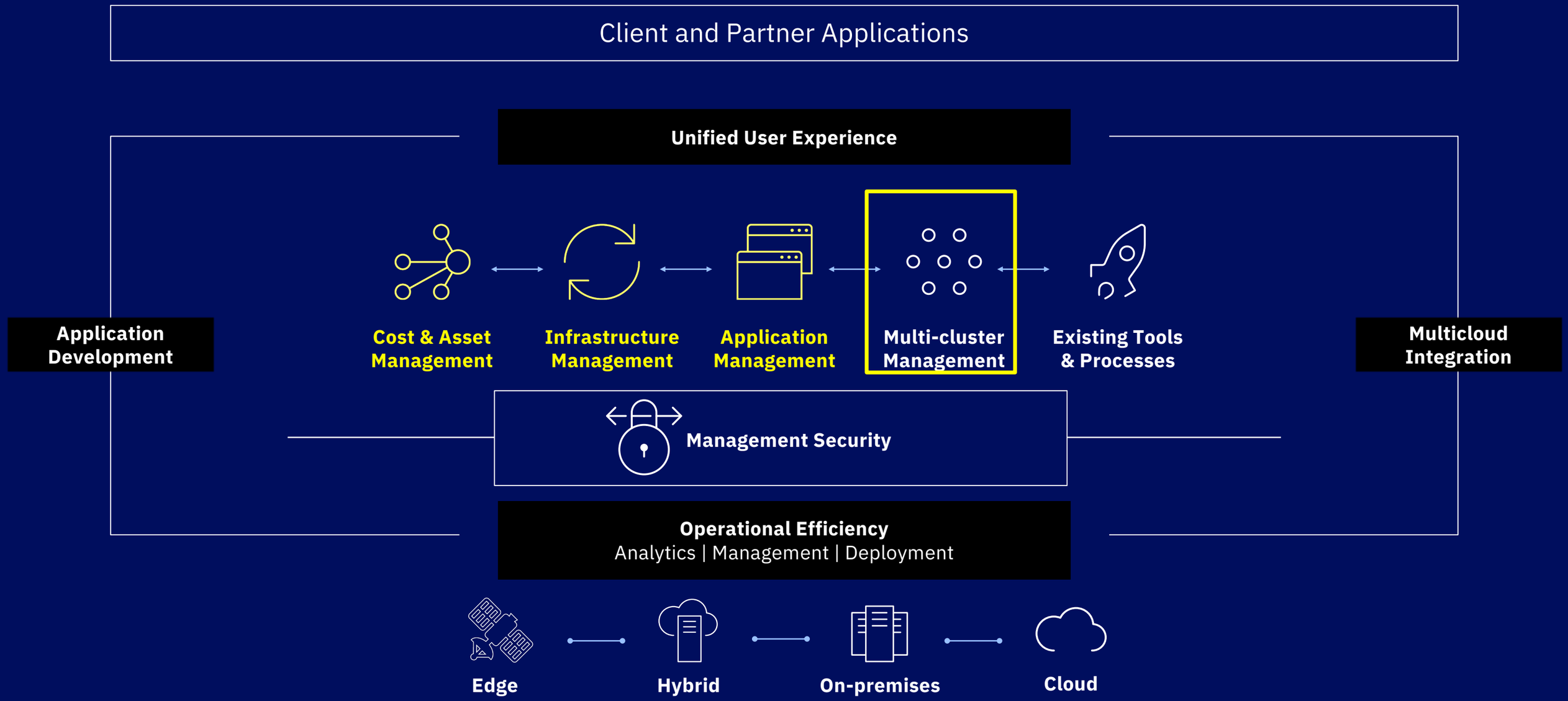
View all relevant information, deployments and placement policies for your applications

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

Multicloud Manager

For Multi-cluster and governance use cases

IBM Cloud Paks

IBM Software leverages IBM Cloud Private services to consistently deploy and manage on IBM Cloud Private and Red Hat OpenShift in the Joint Solution Architecture.

It is not Platform on Platform.

Deploy

Helm | Catalog | Import

Manage

Security | Metering | Monitoring | Logging

Operate

Upgrade | Rollback

IBM Cloud Private Services

Red Hat OpenShift

IBM Cloud Private

Choice of infrastructure and architecture

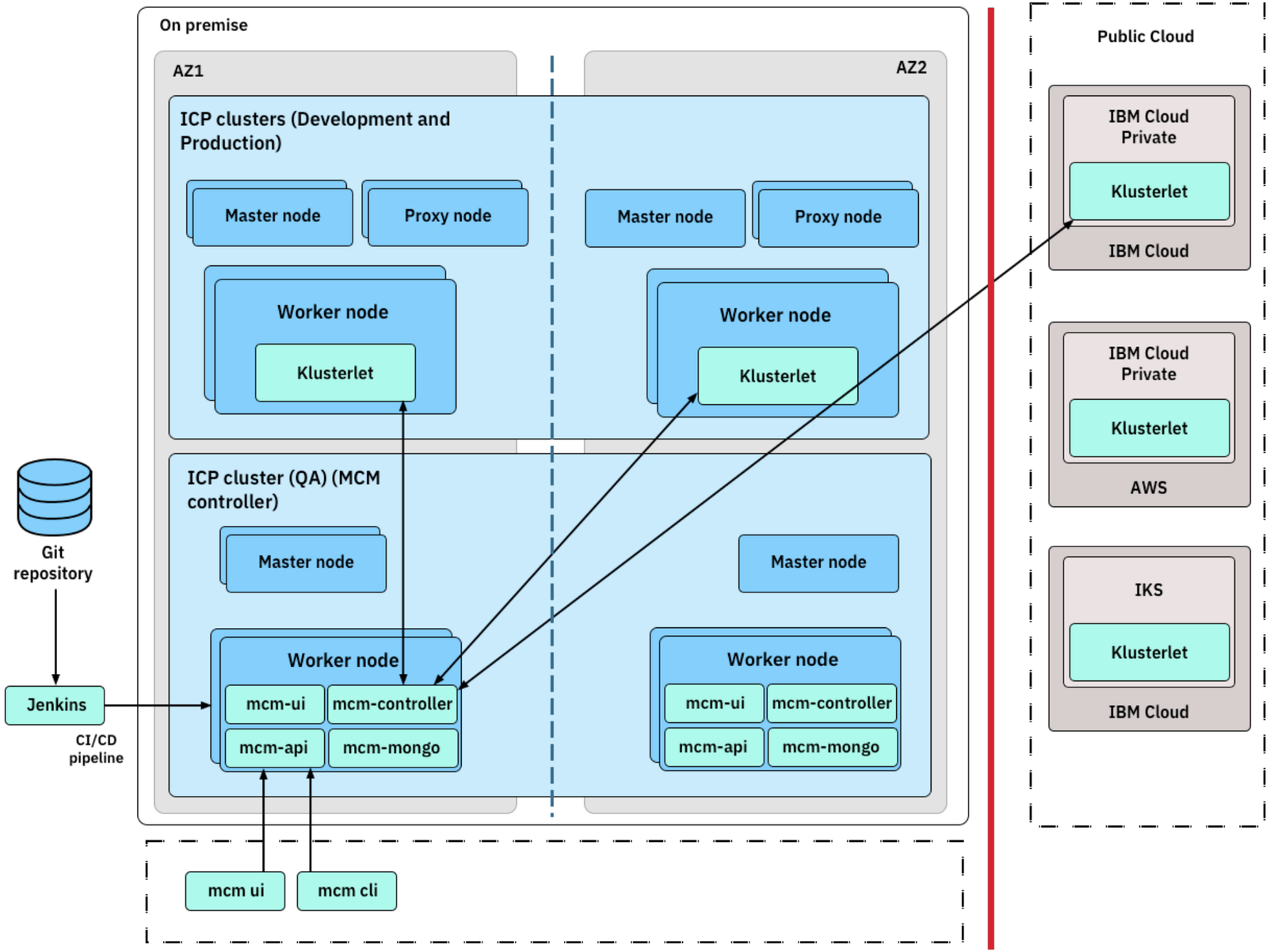
MCM components and architecture

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

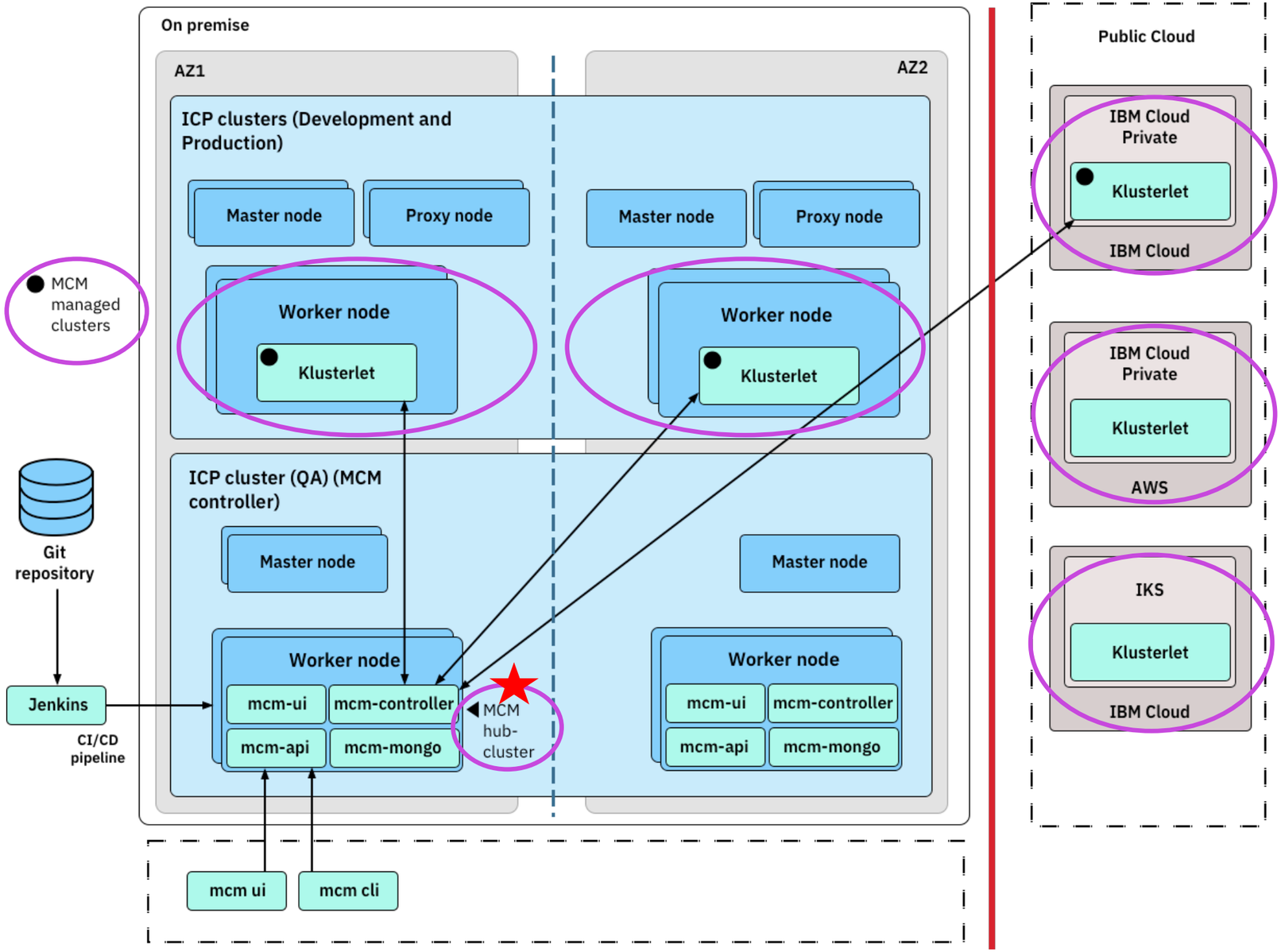
MCM Relies on ICP as the underlying platform and runs on an ICP worker node

ICP HA will add to resilience of the platform

- MCM hub-cluster

- MCM managed clusters

Klusterlets communicate with the **hub-cluster**

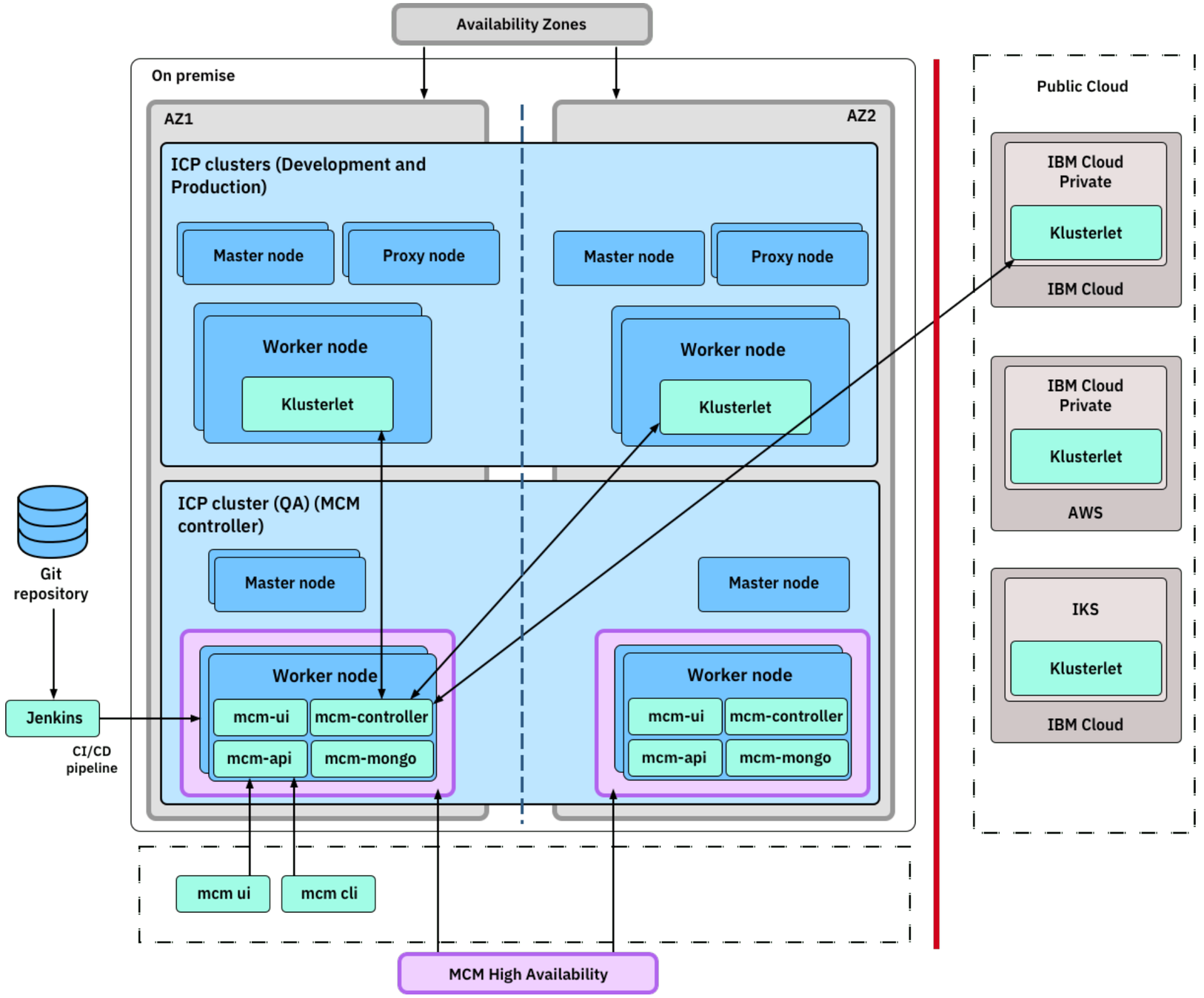


Topology

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.**

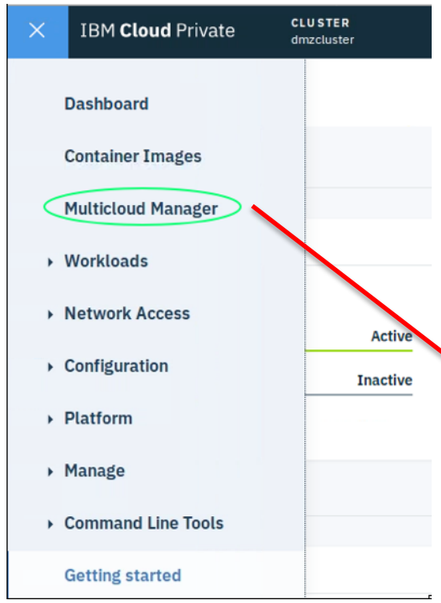


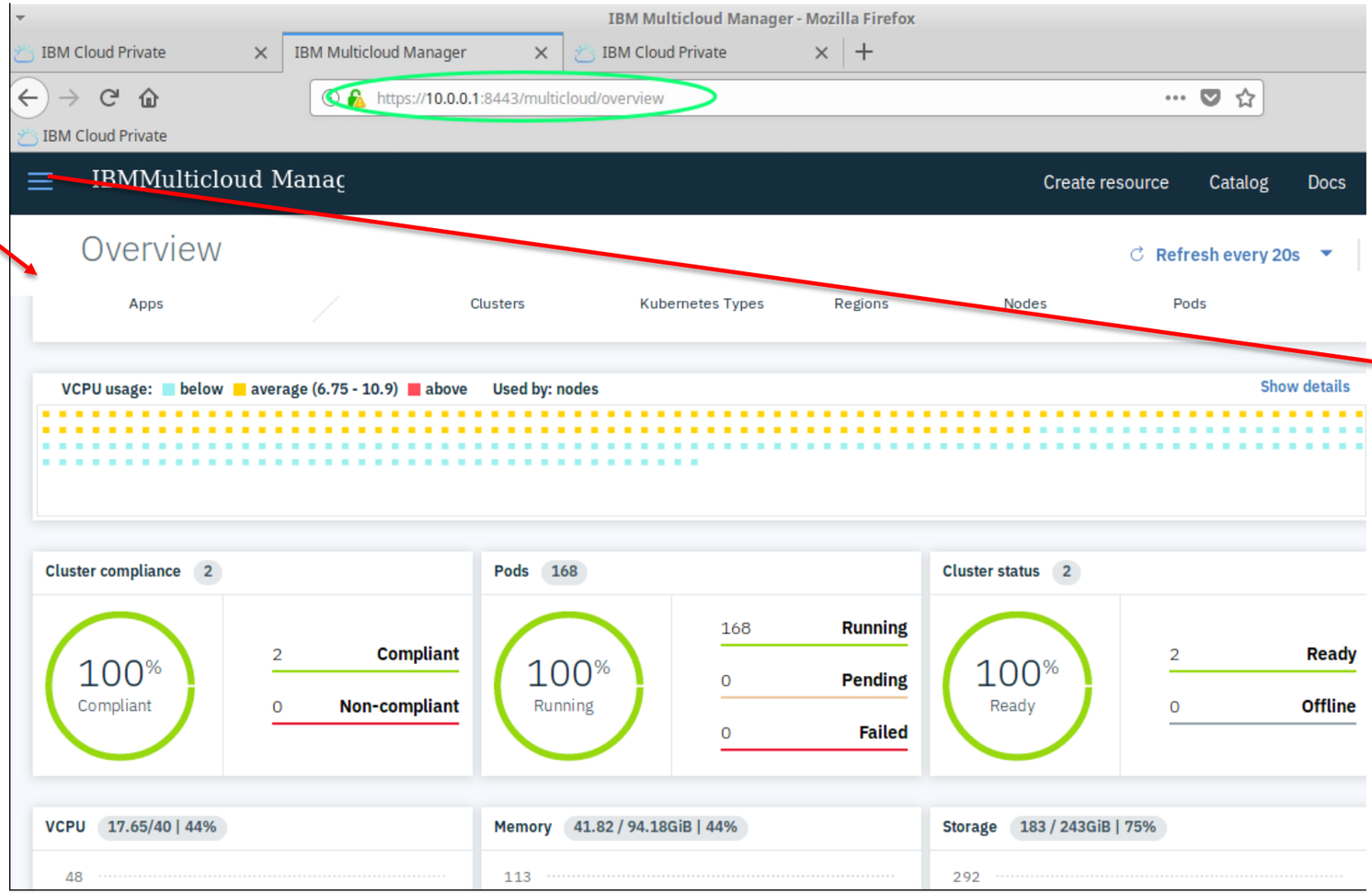
IBM Multicloud Manager Administration and CLI

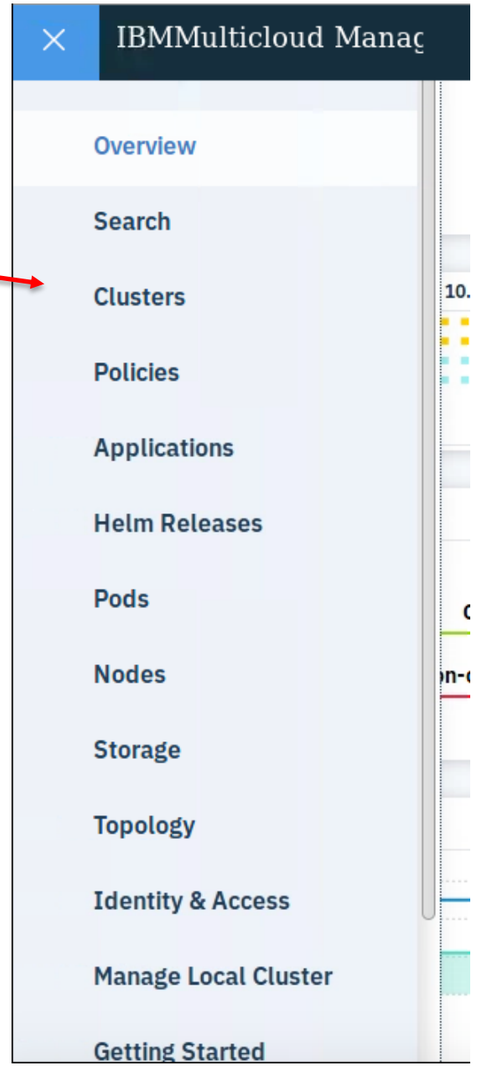
IBM Multicloud Manager roles and access control

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







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*

IBM Multicloud Manager command-line interface

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

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

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=aGNtOmNsdXN0ZXJzLHN5c3RlbTphdXRoZW50aWNhdGVk
              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
multicluster-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

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

multicluster-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.eyJhdF9oYXNoIjo
```

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

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}

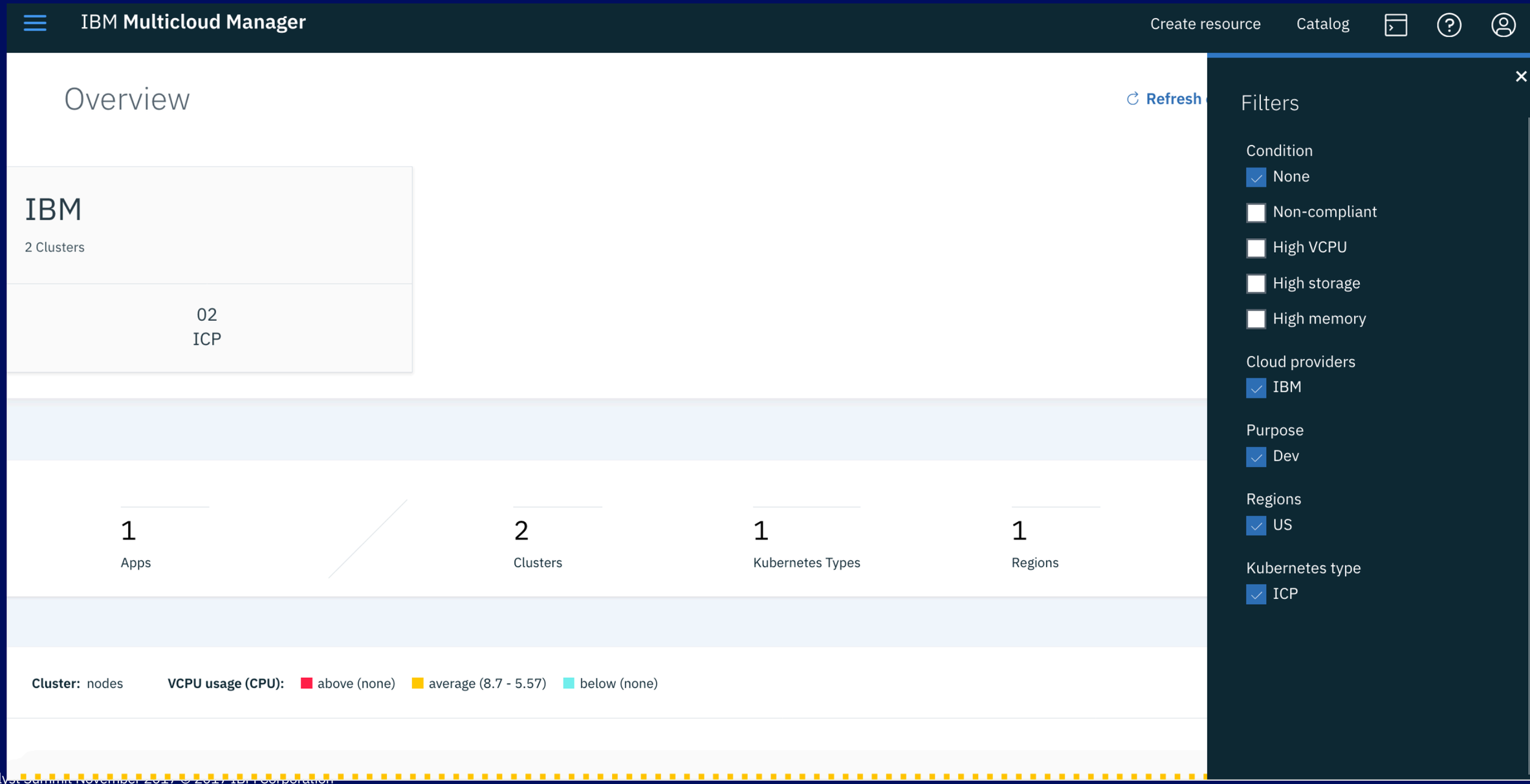
5. Run the following command to verify that the status of the cluster that you want to remove changed to *Offline*.

```
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

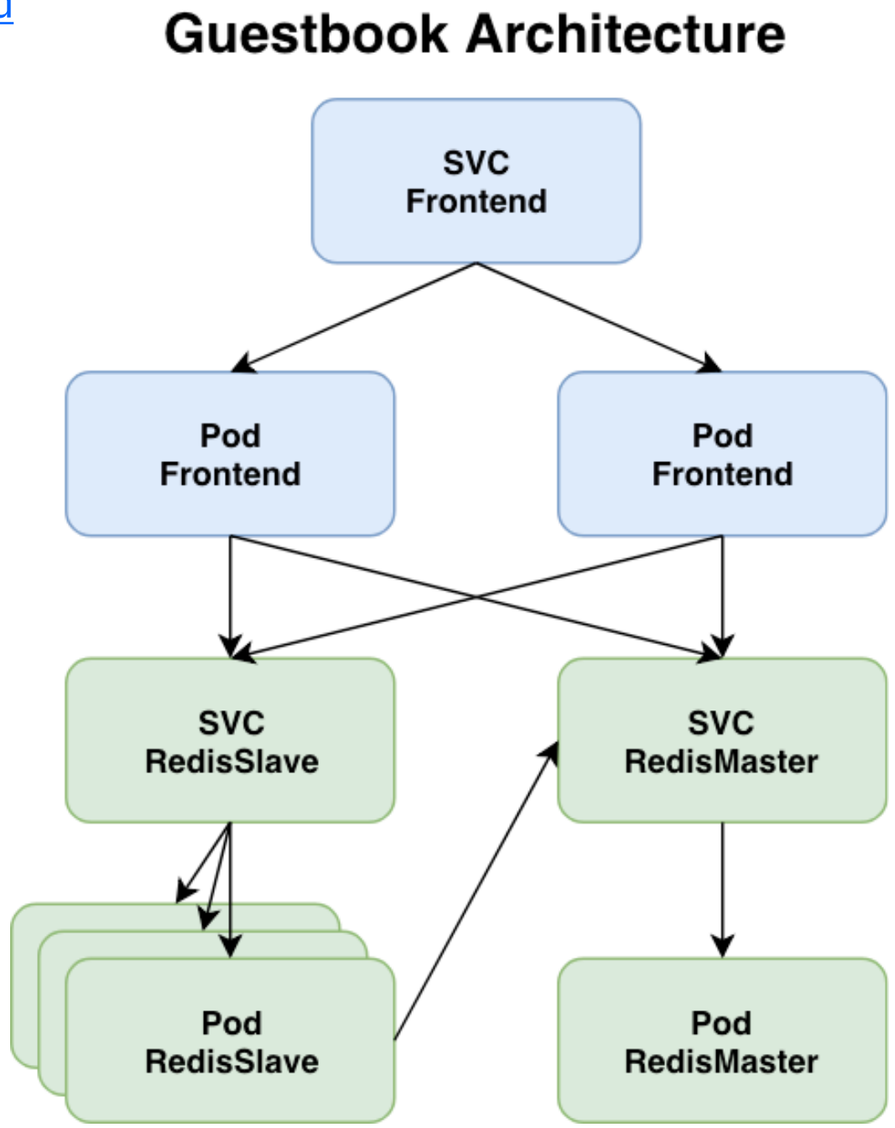

IBM Multicloud Manager
Create resource
Catalog
Overview
Refresh
IBM
2 Clusters
02
ICP
1
Apps
2
Clusters
1
Kubernetes Types
1
Regions
Cluster: nodes
VCPU usage (CPU):
above (none)
average (8.7 - 5.57)
below (none)
Filters
Condition
None
Non-compliant
High VCPU
High storage
High memory
Cloud providers
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 - MCM

This example is trying to convert k8s guest book example [https://kubernetes.io/docs/tutorials/stateless-application/guestbook] into mcm application

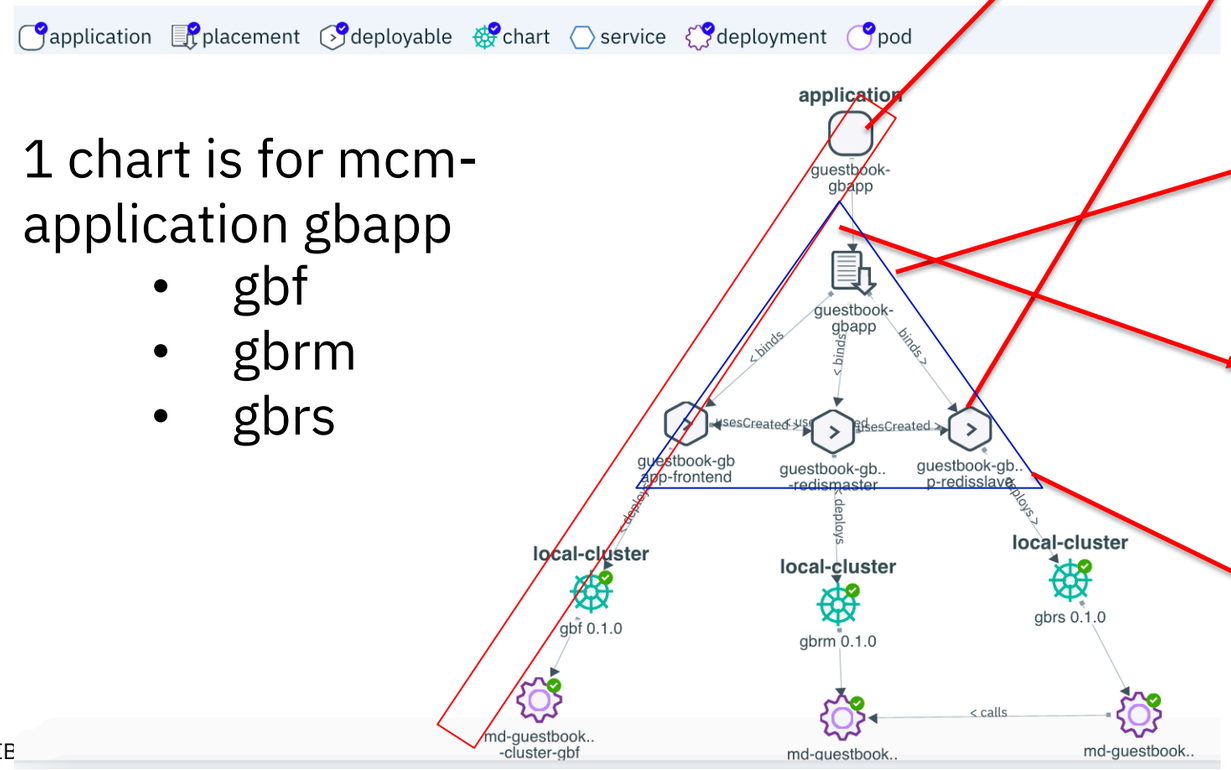
Following 3 charts are wrapper for deployment and service in k8s example, already loaded into public place
1.gbf
2.gbrm
3.gbrs
Following 1 chart is for mcm-application
1.gbapp

MCM Application

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

1 chart is for mcm-application gbapp

- gbf
- gbrm
- gbrs

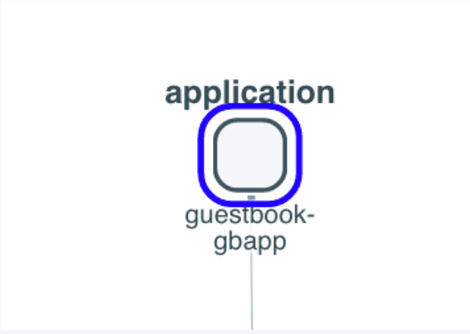



```
apiVersion: app.k8s.io/v1beta1
kind: Application
metadata:
spec:
---
apiVersion: mcm.ibm.com/v1alpha1
kind: Deployable
metadata:
spec:
---
apiVersion: mcm.ibm.com/v1alpha1
kind: PlacementPolicy
metadata:
spec:
---
apiVersion: mcm.ibm.com/v1alpha1
kind: ApplicationRelationship
metadata:
spec:
---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

```yaml
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
```



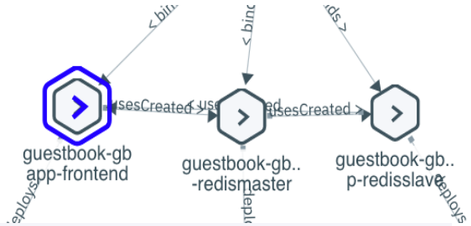

```
targetCluster:
  labelSelector:
    matchLabels:
      #vendor: "ICP"
      #region: "US"
      #environment: Dev
      #datacenter: "toronto"
      #owner: "marketing"
  resourceSelector:
    type: cpu
  compliances: ""
```

```yaml
apiVersion: app.k8s.io/v1beta1
kind: Application
metadata:
  name: guestbook-gbapp
  namespace: default
  generation: 2
  labels:
    name: guestbook-gbapp
    app: gbapp
    chart: gbapp-0.1.0
    heritage: Tiller
    release: guestbook
spec:
  componentKinds:
    - group: core
      kind: Pods
  descriptor: {}
  selector:
    matchExpressions:
      - key: app
        operator: In
        values:
          - gbapp
          - gbf
          - gbrm
          - 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:




```yaml
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 }}
```

```yaml
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
```

IE

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.

```yaml
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 }}
```

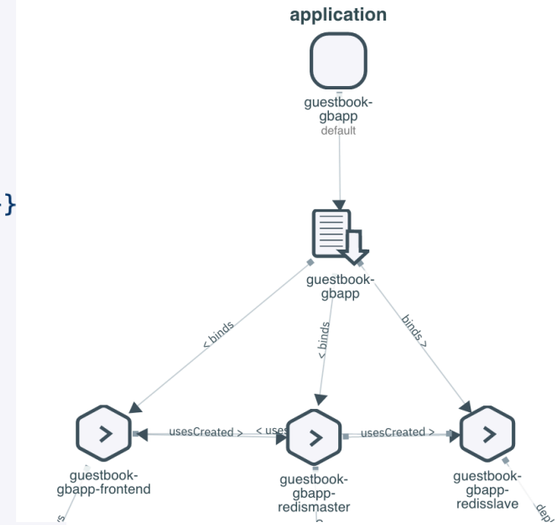

```yaml
apiVersion: mcm.ibm.com/v1alpha1
kind: PlacementPolicy
metadata:
  name: guestbook-gbapp
  namespace: default
  labels:
    name: guestbook-gbapp
    app: gbapp
    chart: gbapp-0.1.0
    heritage: Tiller
    release: guestbook
    servicekind: CacheService
spec:
  clusterLabels:
    matchLabels:
      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:

```yaml
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
```



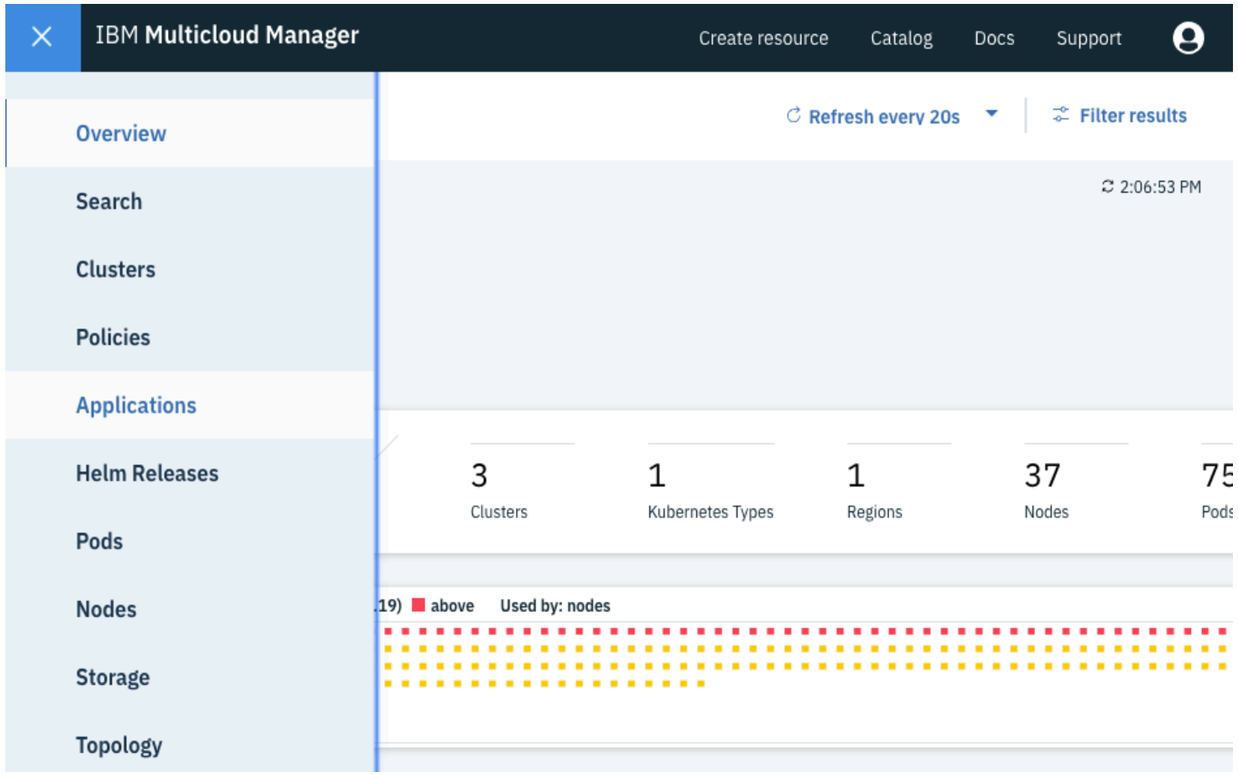

```yaml
apiVersion: mcm.ibm.com/v1alpha1
kind: PlacementBinding
metadata:
  name: guestbook-gbapp
  namespace: default
  labels:
    name: guestbook-gbapp
    app: gbapp
    chart: gbapp-0.1.0
    heritage: Tiller
    placementPolicy: guestbook-gbapp
    release: guestbook
    servicekind: CacheService
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

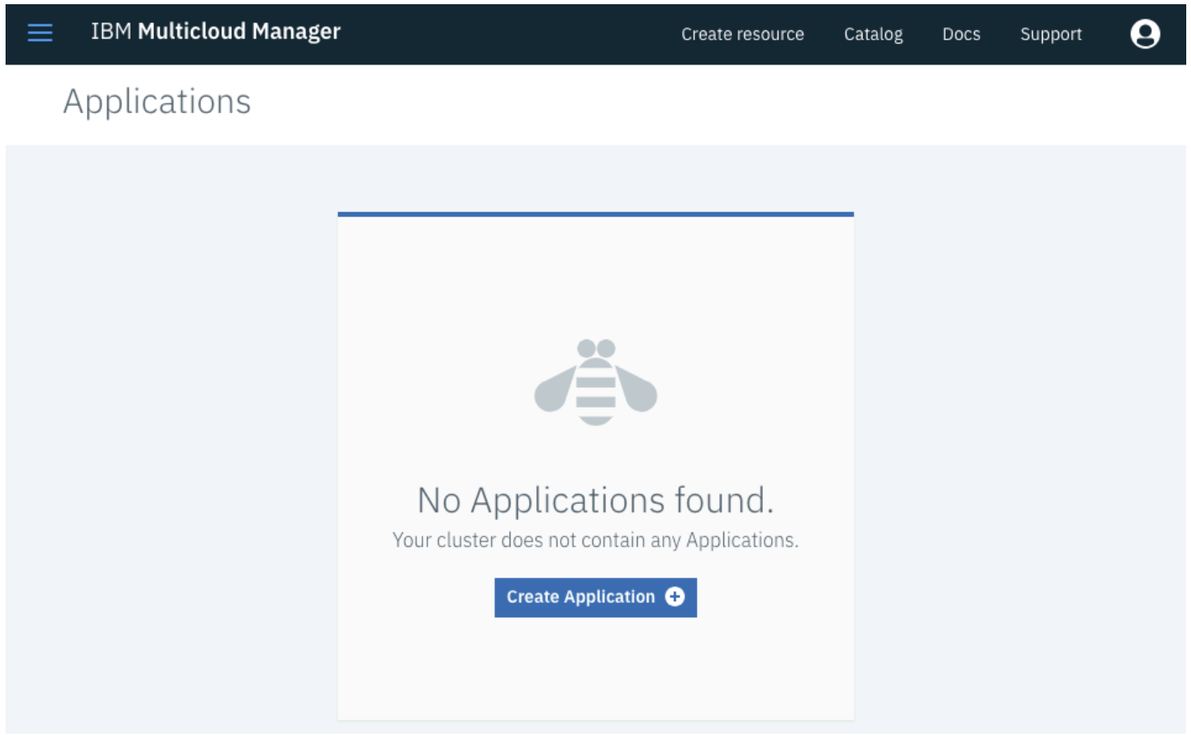
Guestbook - MCM

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:



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



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:

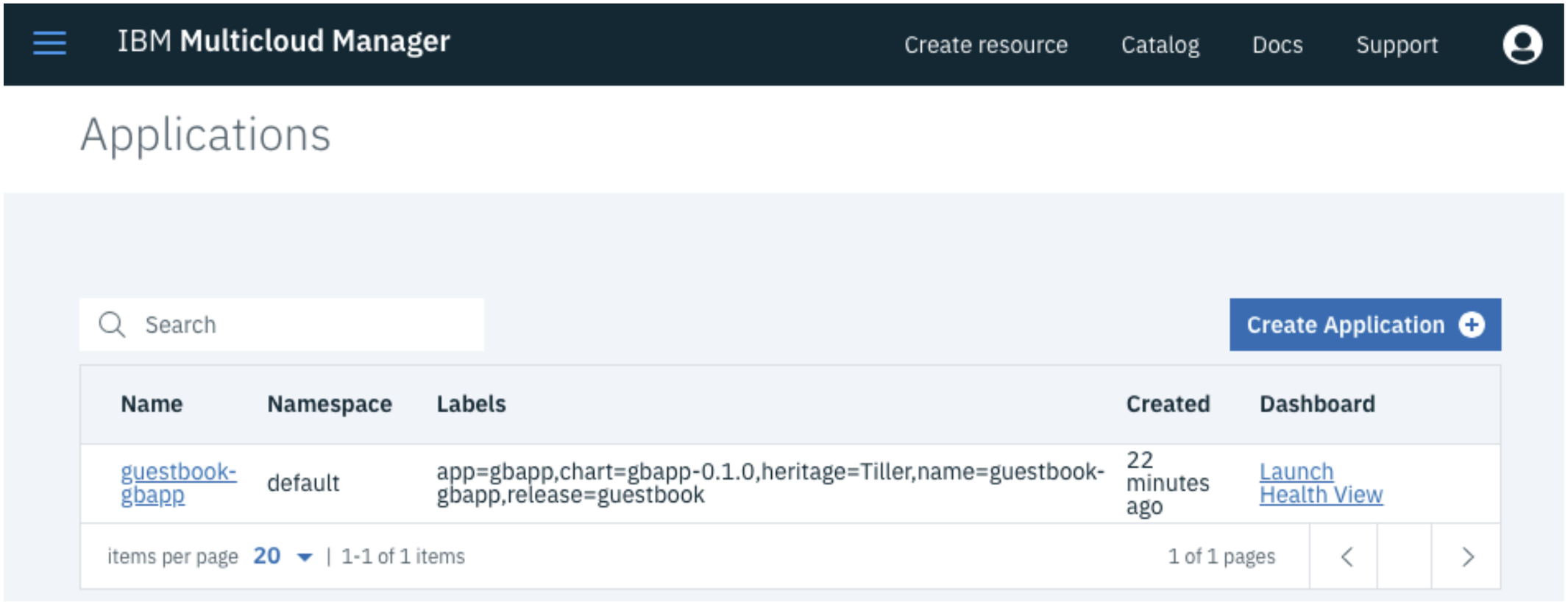
```
# Log into MCM HUB Cluster
cloudctl login -a https://HUB_CLUSTER_MASTER_IP:8443 -n default --skip-ssl-validation
 # If not already there, go to the Helm Charts directory
cd demos/guestbook
 # Deploy MCM Application Helm Chart
helm upgrade --install guestbook --set replicaCount=1 --set
targetCluster.labelSelector.matchLabels.environment=Dev ./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 should 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.

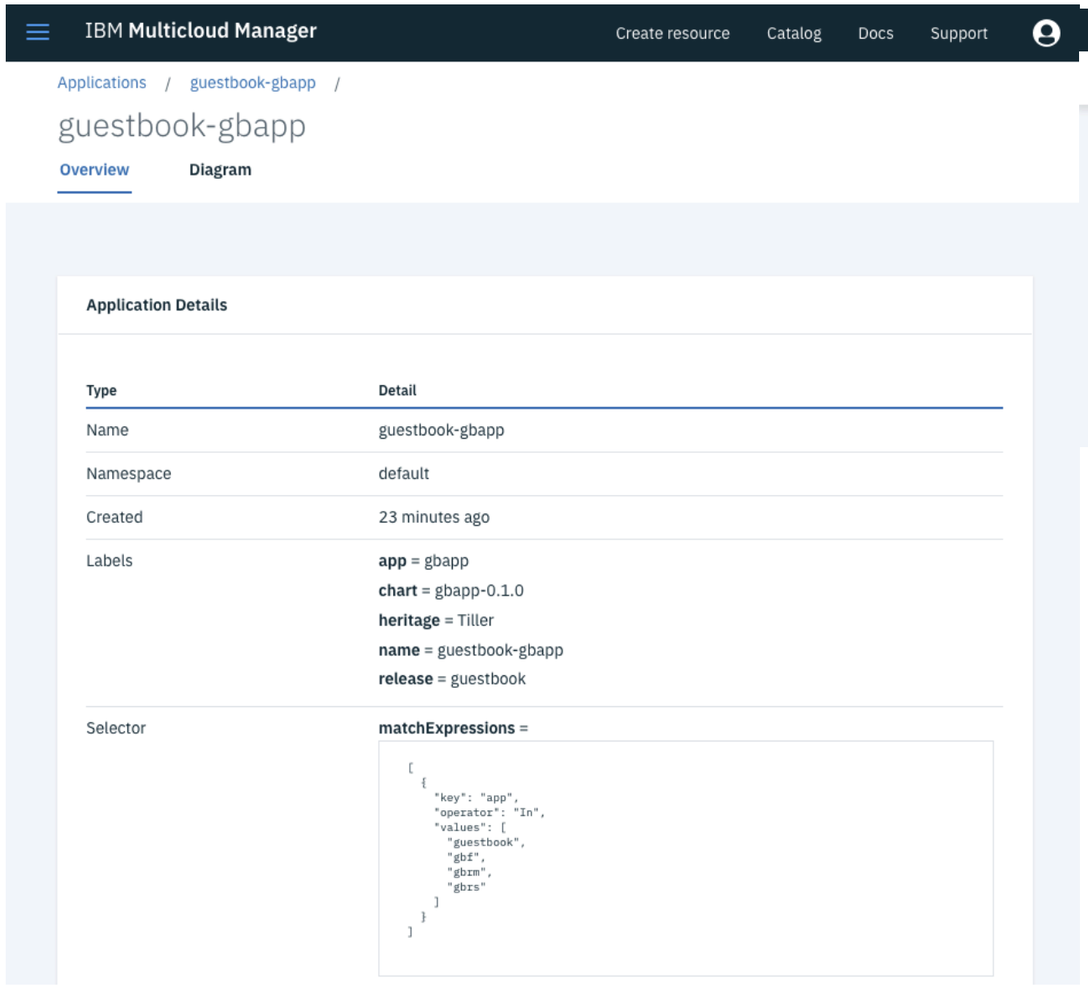
Guestbook - MCM

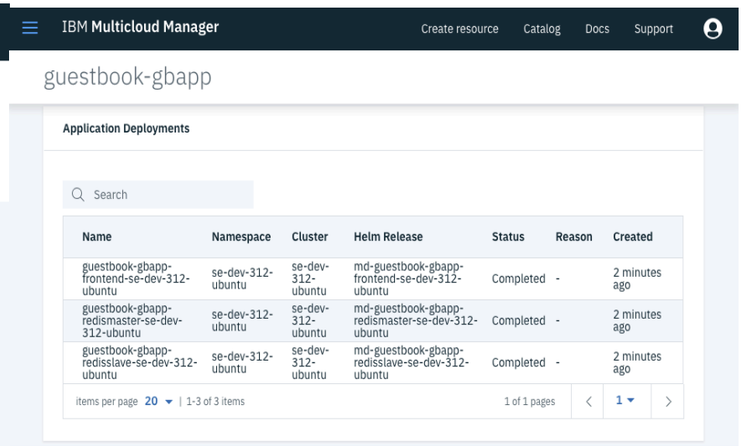
Assuming that everything went well, you should now see the guestbook-gbapp MCM Application show up in the Applications page as shown below:

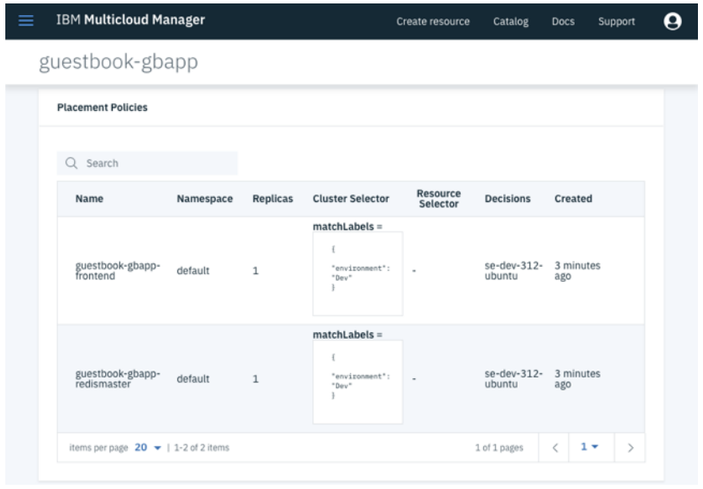


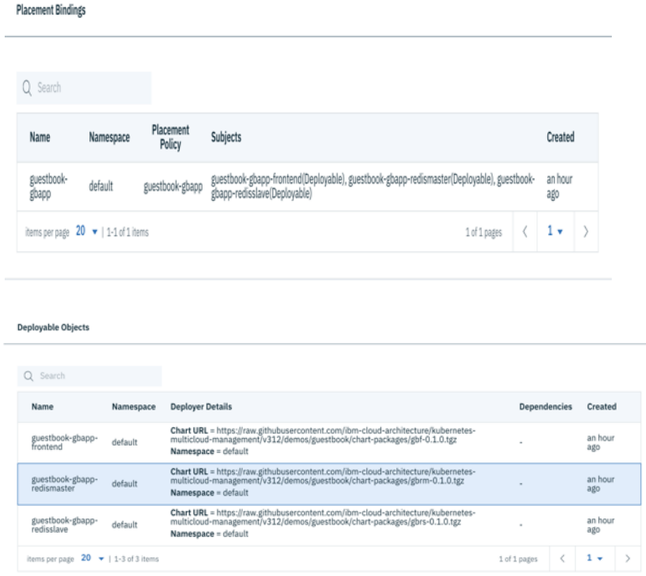
Guestbook - MCM

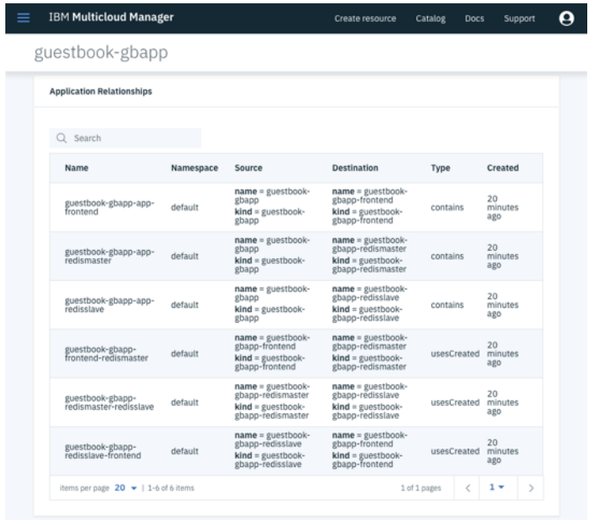
2. Exploring the Application Details Page





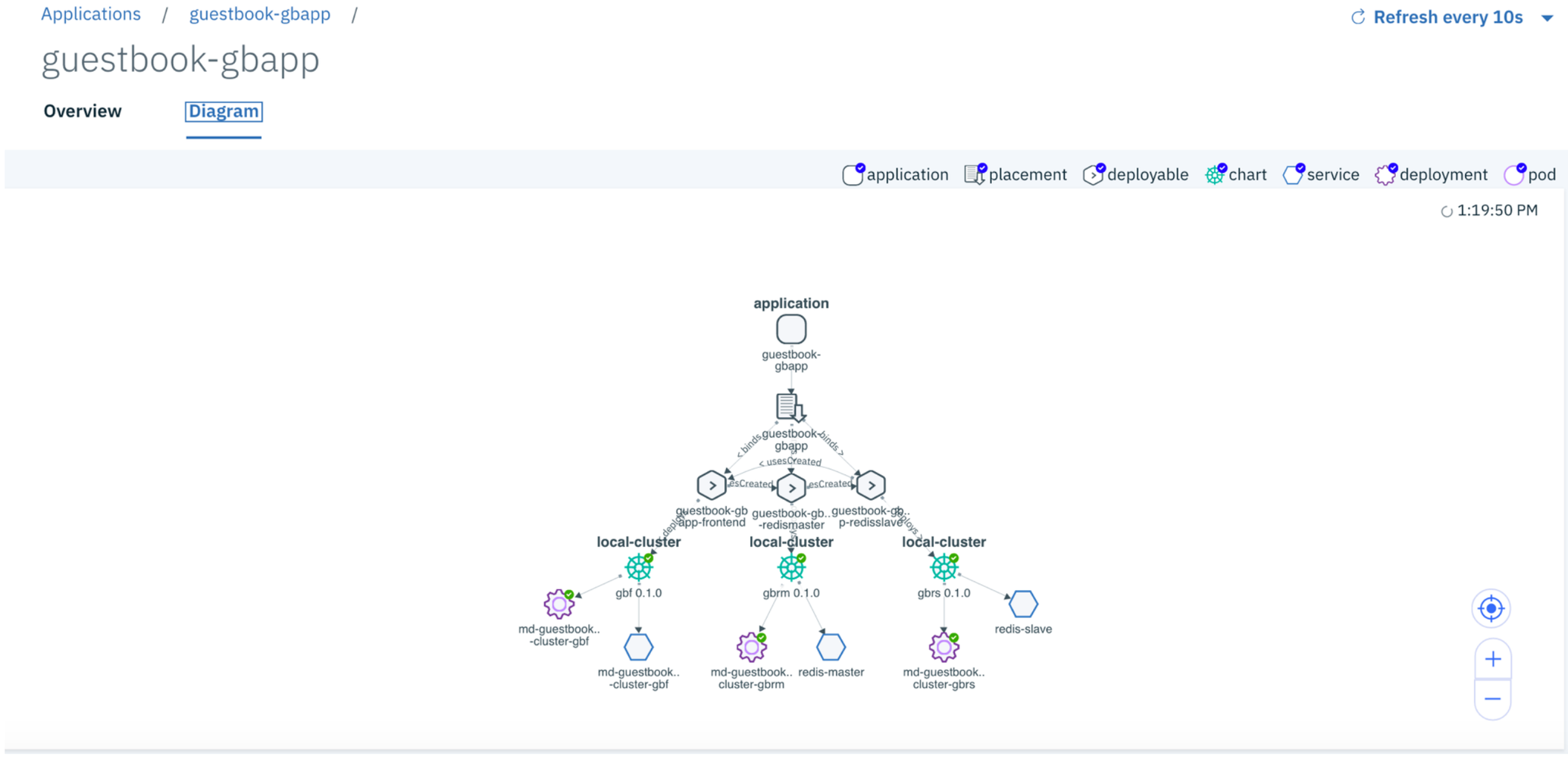






Guestbook - MCM

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.

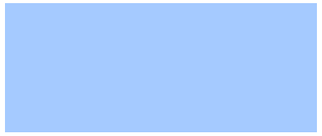


Guestbook - MCM

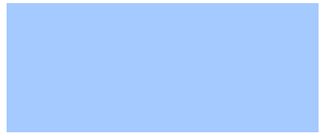
3. Redeploy to Staging Cluster

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



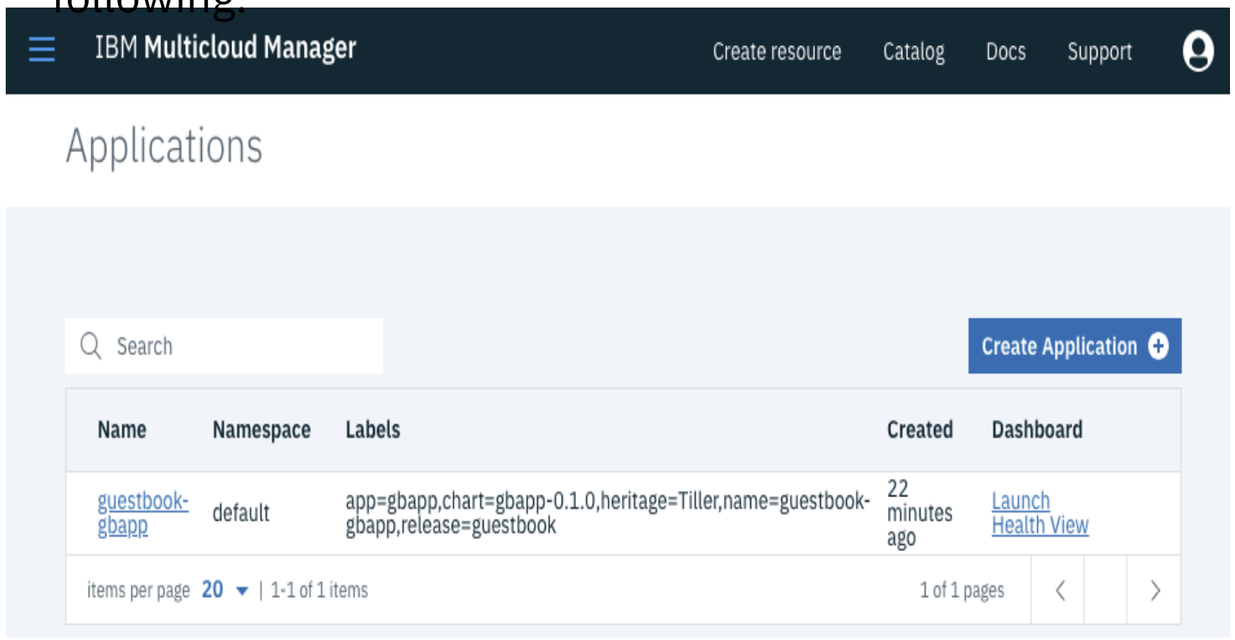
se-stg-31 cluster

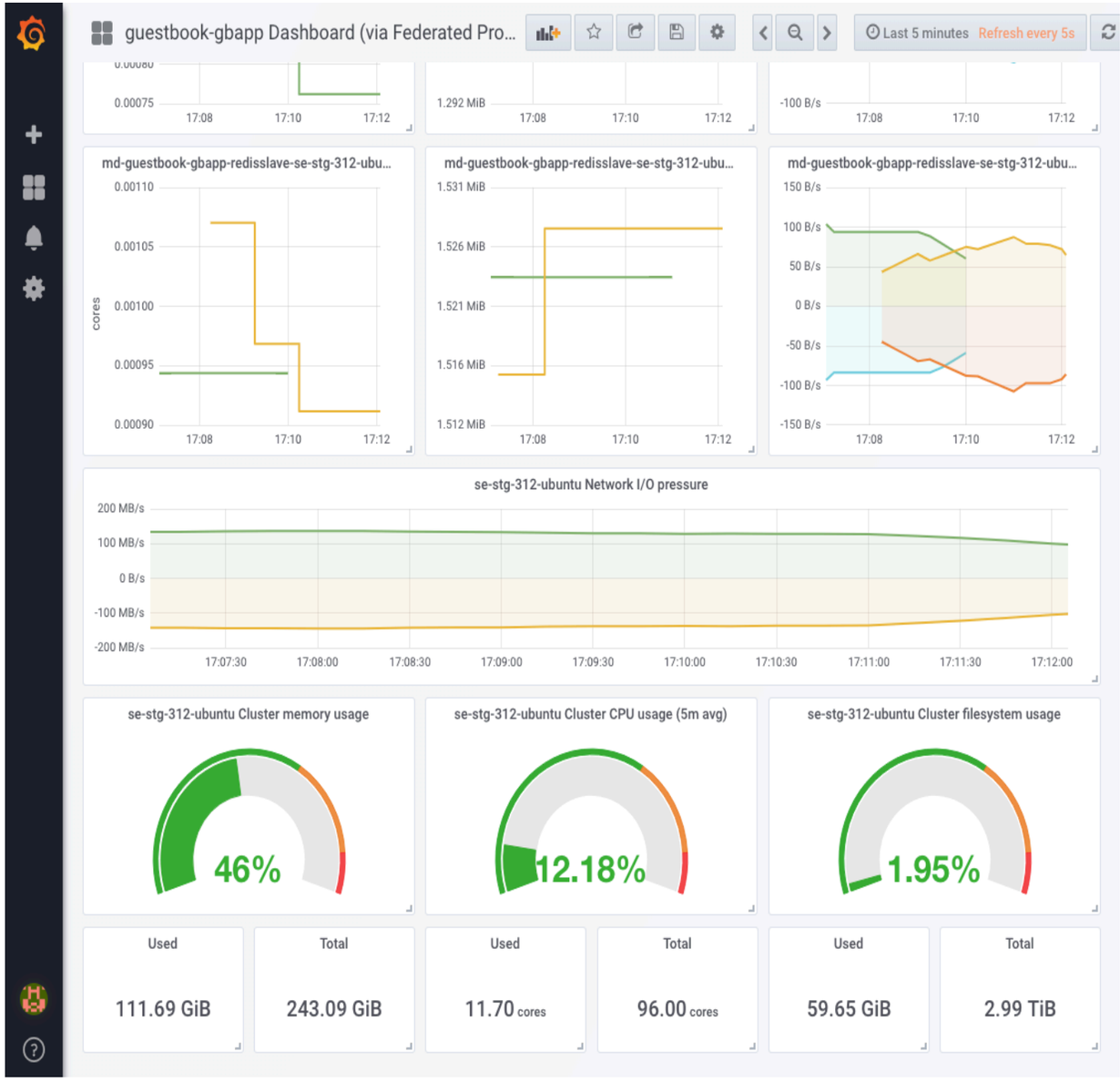
```
targetCluster:
  labelSelector:
    matchLabels:
      #vendor: "ICP"
      #region: "US"
      #environment: Dev
      #datacenter: "toronto"
      #owner: "marketing"
```

Guestbook - MCM

4. Health View Dashboard

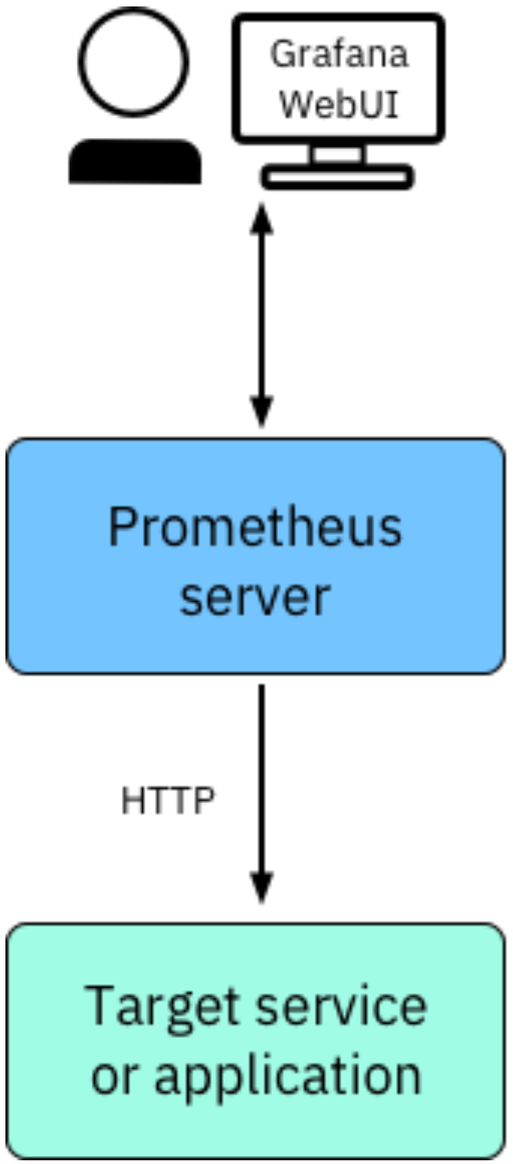
Now click the Launch Health View link under the Dashboard column and it should open a Grafana dashboard in a new page that looks similar to the following:





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

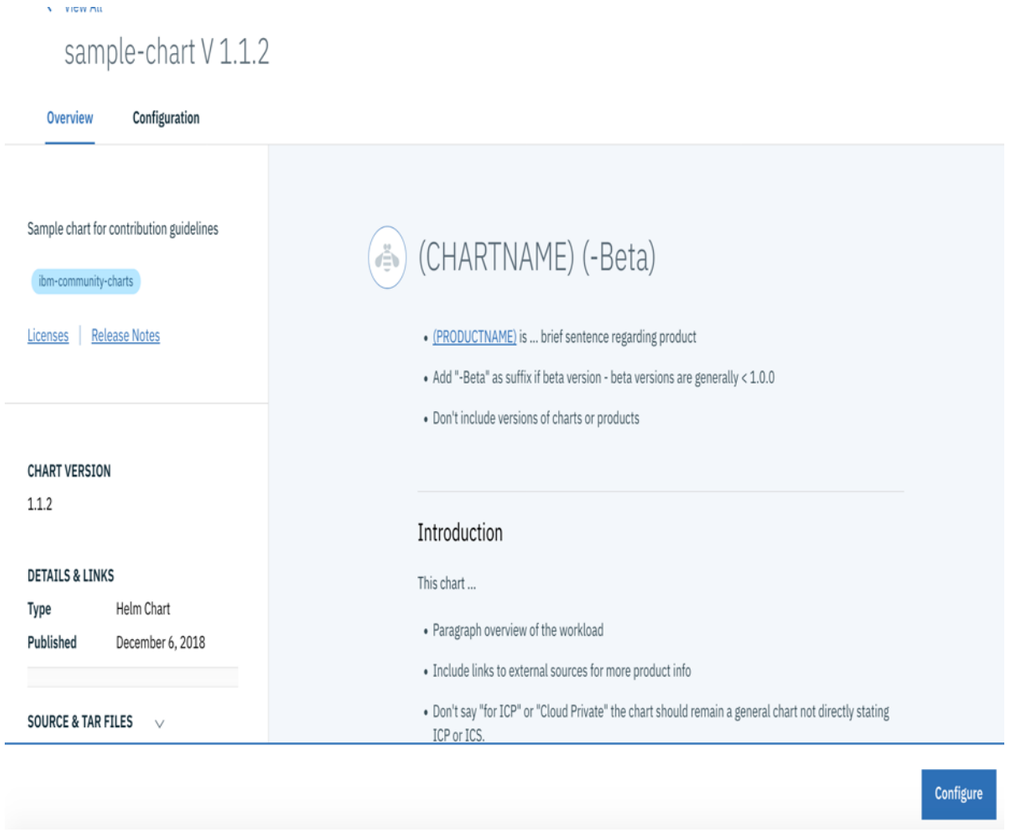


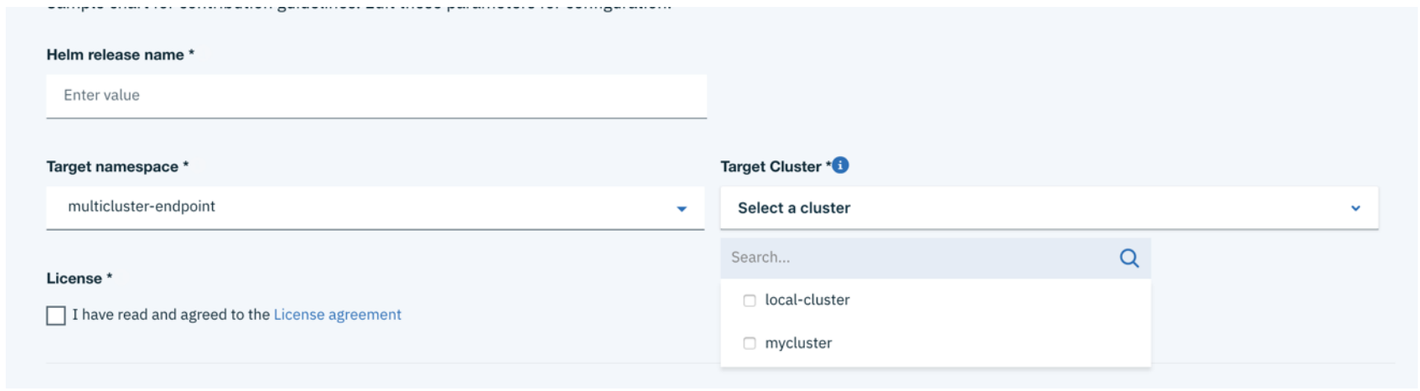
Grafana dashboards are used to visialize the information collected by Prometheous server

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

Helm Chart from Catalog - MCM

4. Deploy sample chart services to remote cluster from catalog



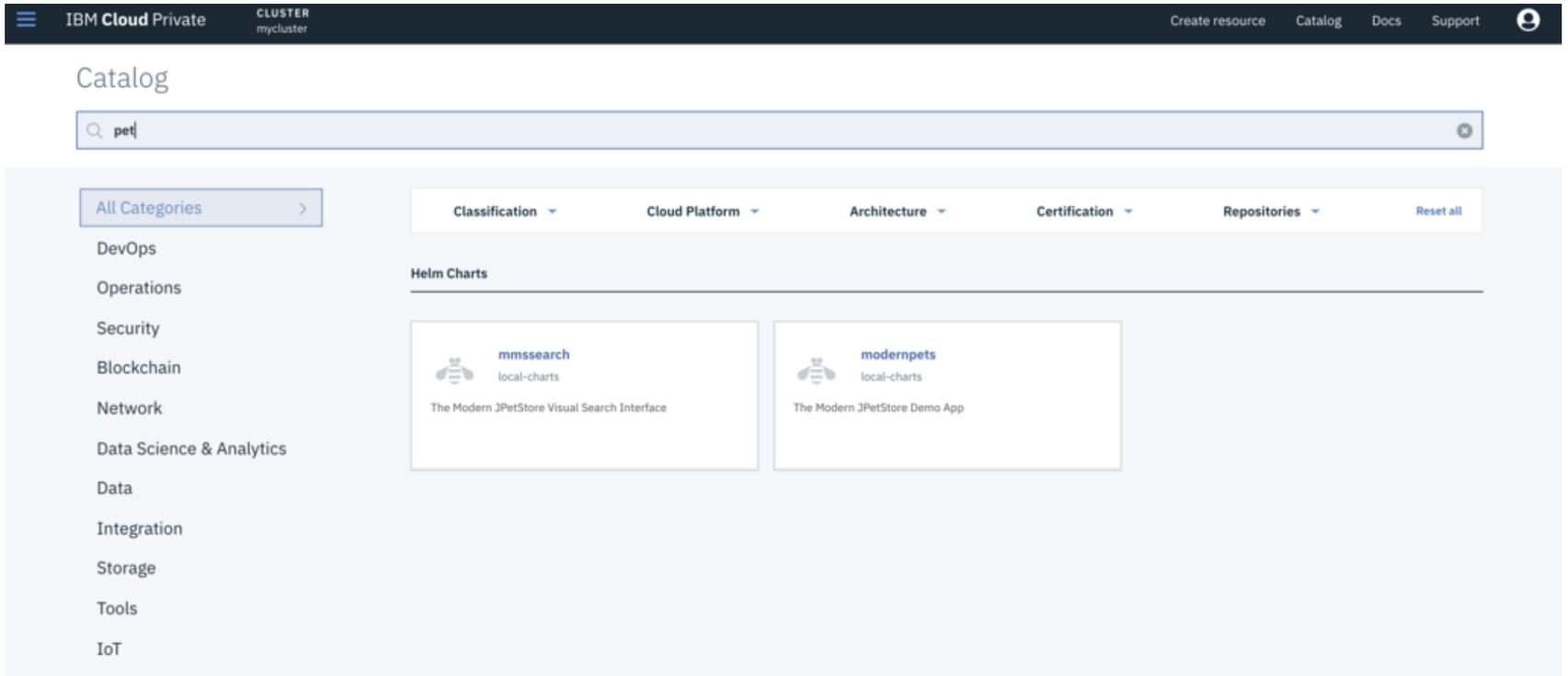



```
nginx-7cdbd8cdc9-w2pgr   1/1     Running   0          2d1h
mikes-mbp:sample mikechang$ kubectl get pods -n default
NAME                     READY   STATUS    RESTARTS   AGE
nginx-7cdbd8cdc9-f5f5t   1/1     Running   0          2d1h
nginx-7cdbd8cdc9-w2pgr   1/1     Running   0          2d1h
mikes-mbp:sample mikechang$ kubectl describe pod nginx-7cdbd8cdc9-f5f5t
Name:               nginx-7cdbd8cdc9-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=7cdbd8cdc9
                    run=nginx
Annotations:        kubernetes.io/psp: ibm-privileged-psp
Status:             Running
IP:                 192.168.110.157
Controlled By:      ReplicaSet/nginx-7cdbd8cdc9
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



MCM GRC dashboard IBM Multicloud Manager policy framework

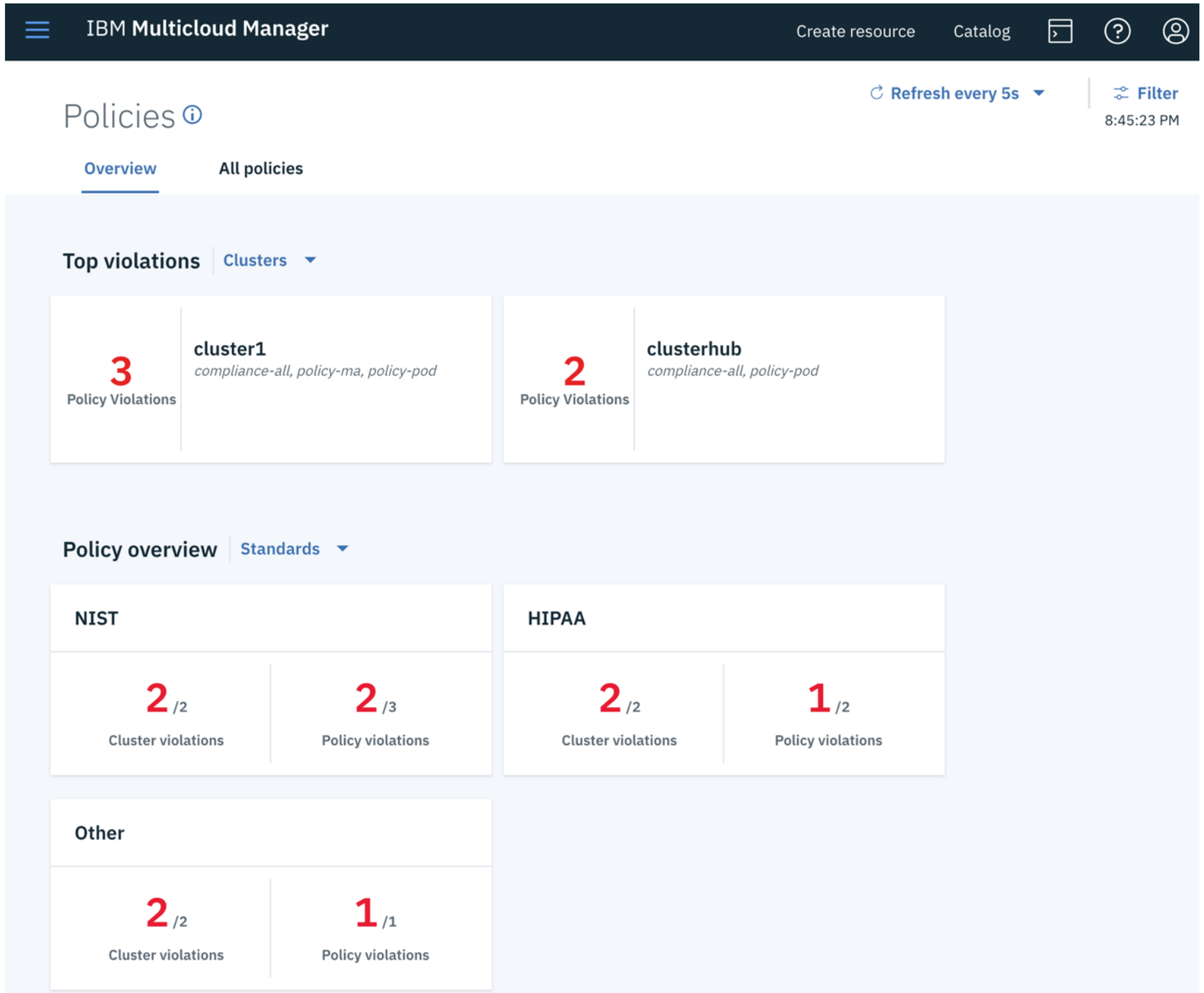
GRC dashboard overview

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



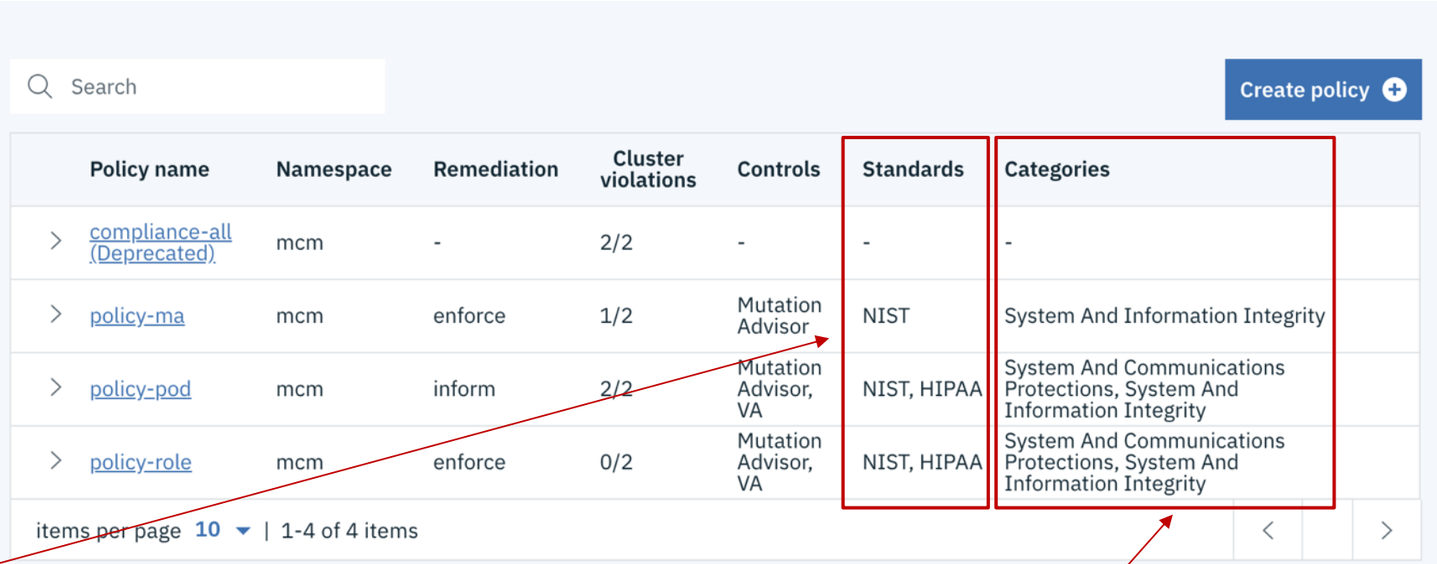
Customizing dashboard cards

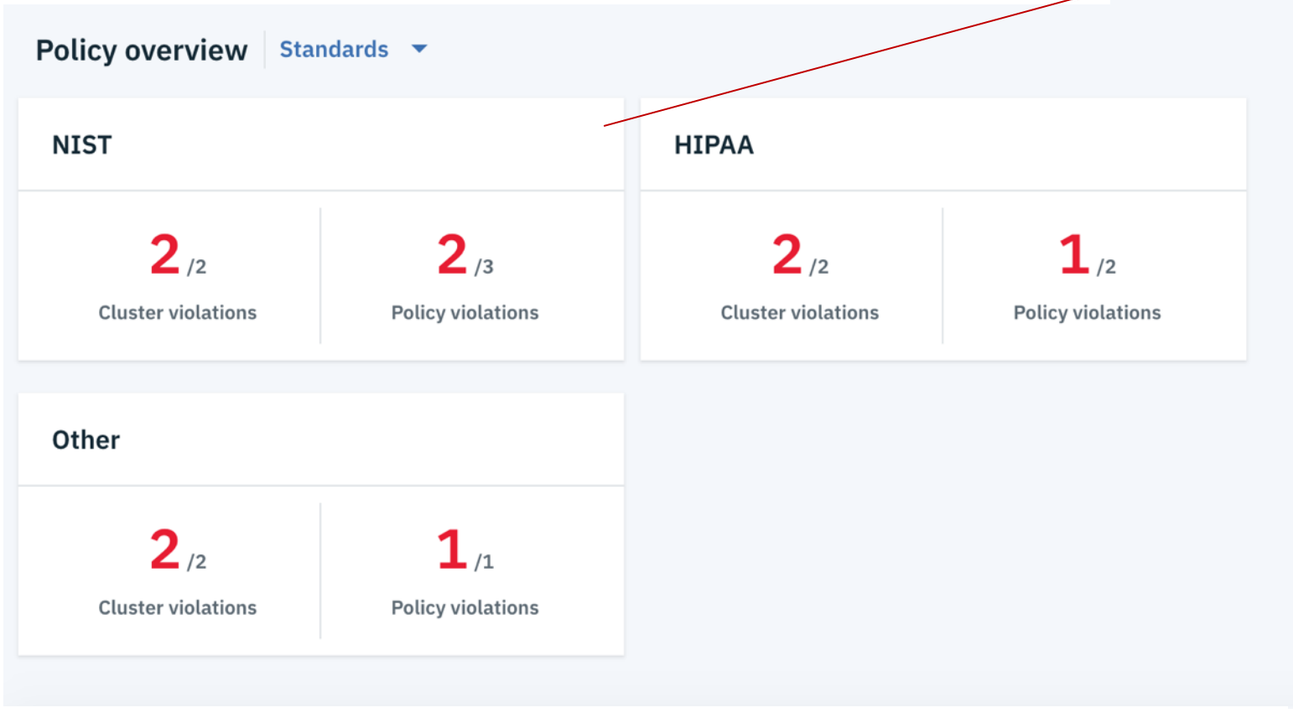
Annotations metadata

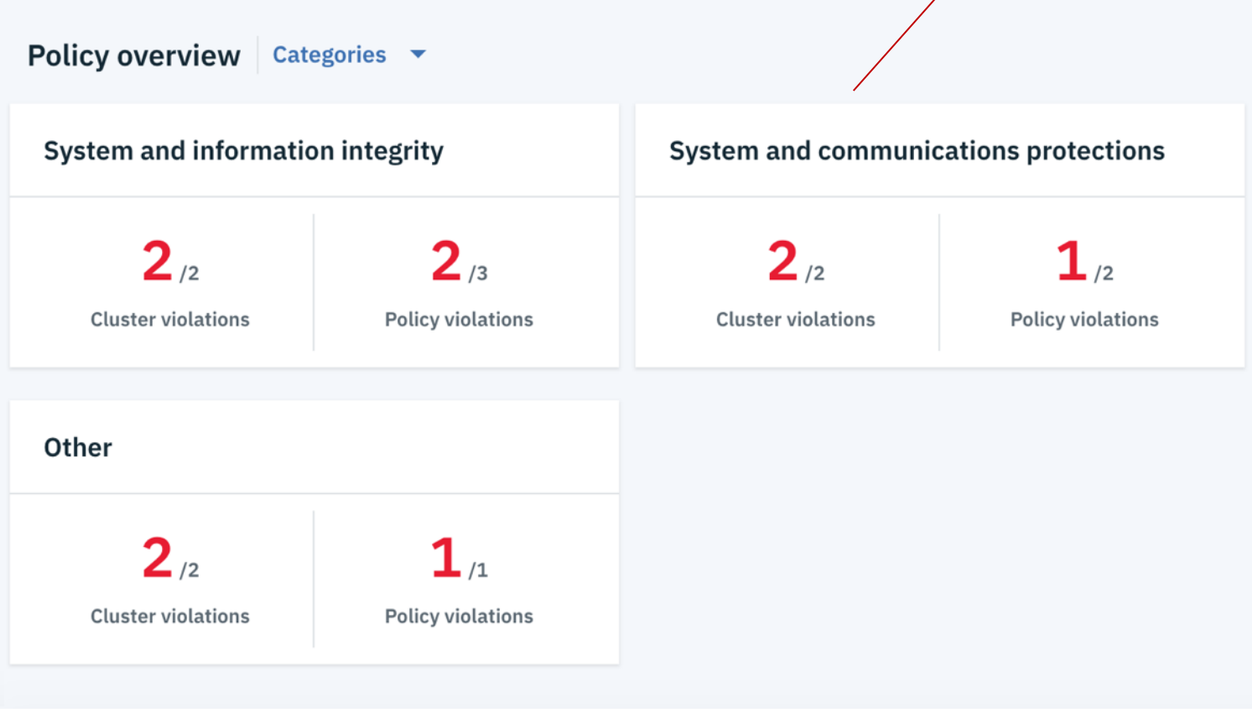
– policy.mcm.ibm.com/categories

– policy.mcm.ibm.com/standards

– policy.mcm.ibm.com/controls



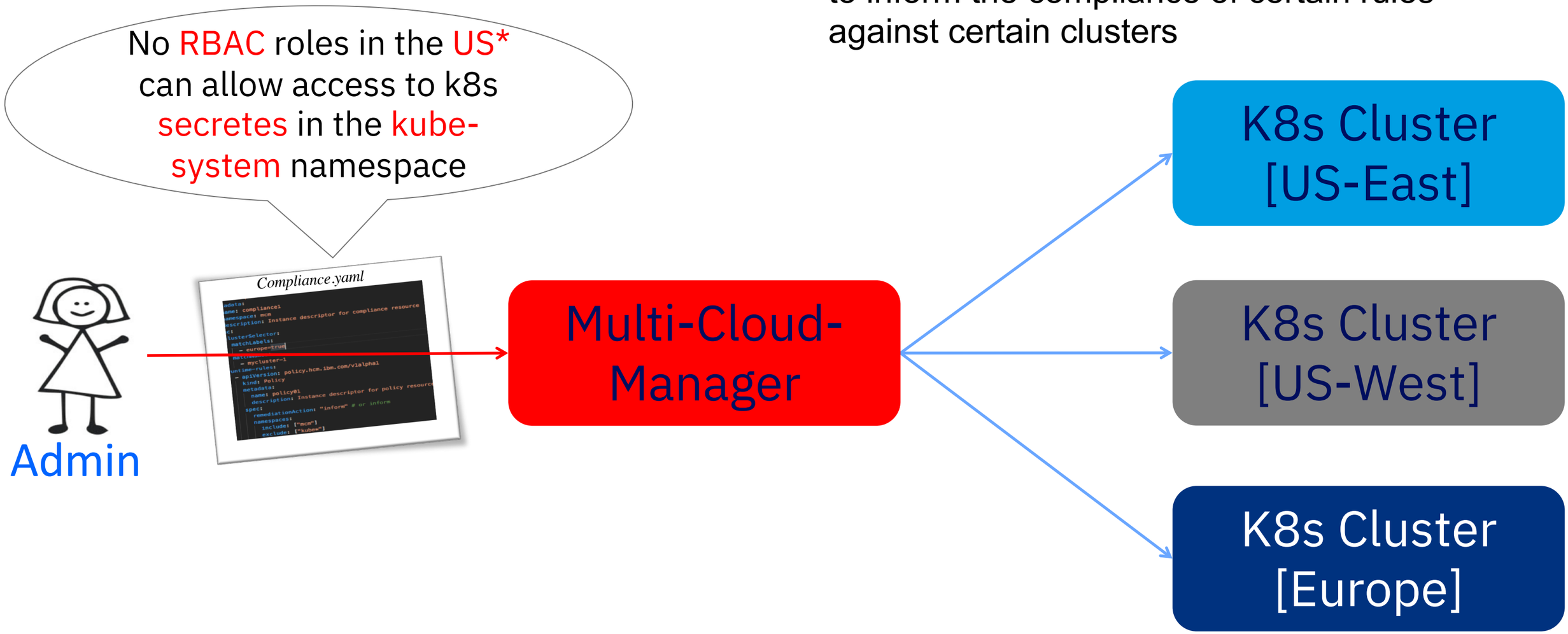




IBM Multicloud Manager Policy overview

IBM Multicloud Manager Policy is a template that you can define

- to enforce certain rules to be applied on certain clusters
- to inform the compliance of certain rules against certain clusters



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 policyRule**s 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*

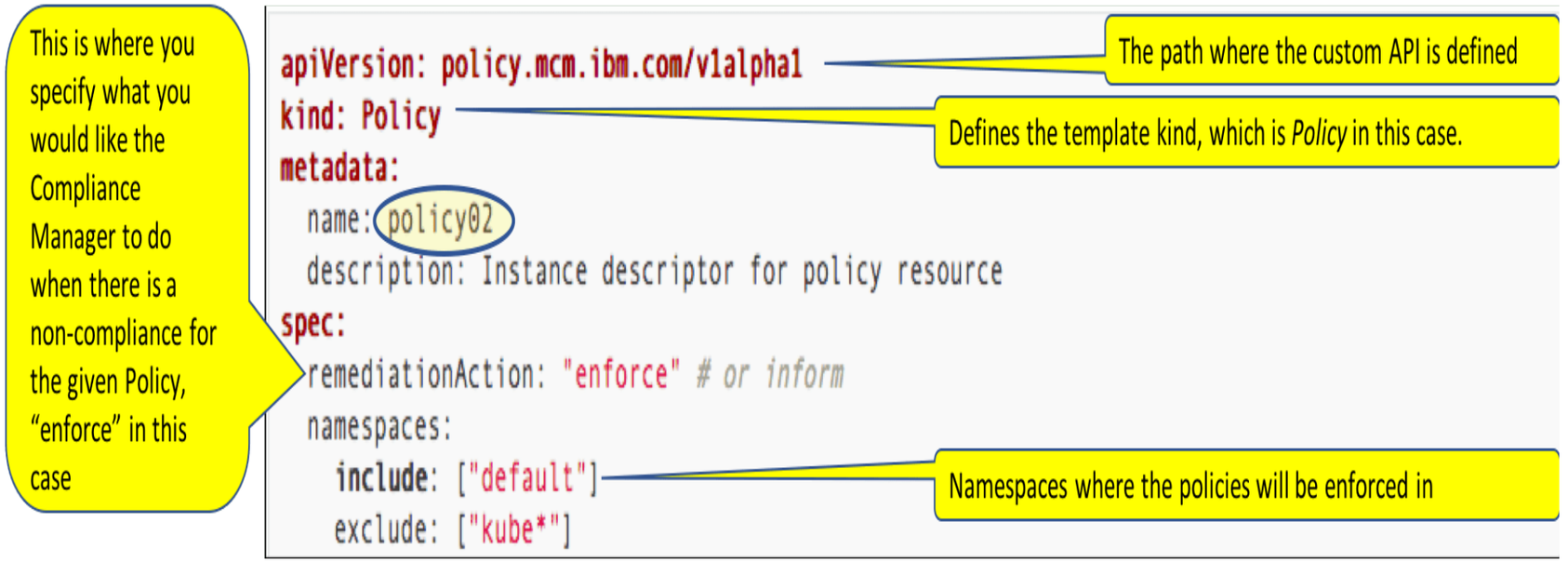
Multicloud Manager - **Policy**

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



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

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.

```
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"
#      hippa: "true"
#    matchExpressions:
#    - key: key1
#      operator: "NotIn"
#      values:
#      - "cl3"
#      - "cl4"
#    matchConditions:
#    - type: "OK"
#      status: "True"
  runtime-rules:
    - apiVersion: policy.mcm.ibm.com/v1alpha1
      kind: Policy
      metadata:
        name: policy01
        description: Instance descriptor for policy resource
```

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

API-group/version

action

where

template

who

how

what

if the name resolves to a set of RBAC roles, or if the RBAC role already exists, then the controller will perfrom 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/IBMPrivateCloud/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/IBMPrivateCloud/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/IBMPrivateCloud/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/IBMPrivateCloud/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

```yaml
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"]
```

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

```yaml
---
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

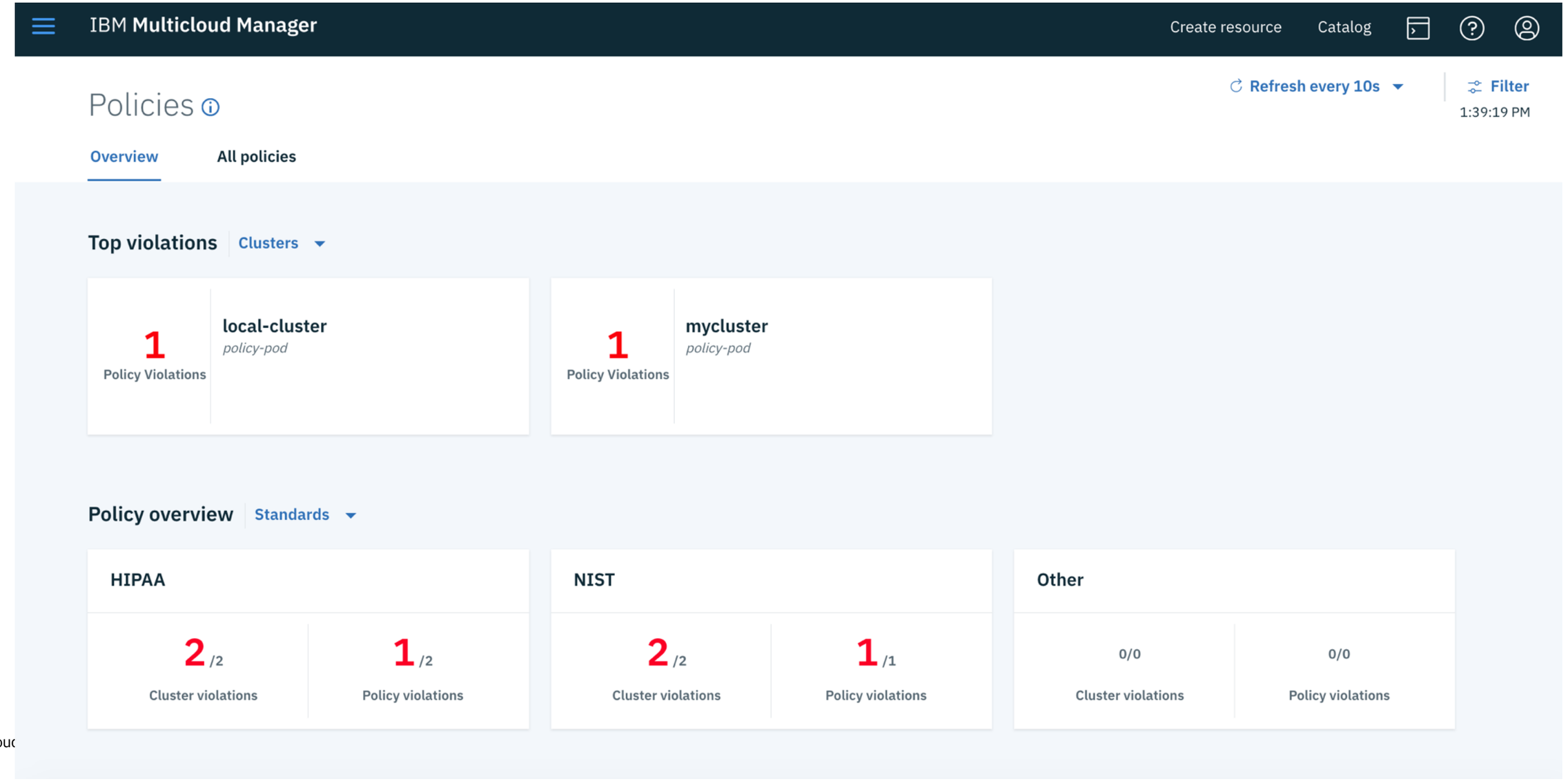
```yaml
apiVersion: policy.mcm.ibm.com/v1alpha1
kind: Policy
metadata:
  name: policy-pod
  namespace: kube-system
  annotations:
    policy.mcm.ibm.com/standards: NIST,HIPAA
    policy.mcm.ibm.com/categories: SystemAndCommunicationsProtections,SystemAndInformationIntegrity
    policy.mcm.ibm.com/controls: 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
```

```yaml
---
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**, where ${MCM_HUB_CLUSTER_MASTER_IP} is the IP Address of the master node in the MCM Hub Cluster.



Policies

Refresh every 10s | Filter

1:24:23 PM

Overview | All policies

Search

Create policy

Policy name	Namespace	Remediation	Cluster violations	Controls	Standards	Categories
policy-role	kube-system	enforce	0/2	Mutation Advisor, VA	HIPAA	System And Communications Protections, System And Information Integrity
policy2	kube-system	enforce	0/0	-	-	-
policy3	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

```yaml
apiVersion: policy.mcm.ibm.com/v1alpha1
kind: Policy
metadata:
  name: policy-role
  namespace: kube-system
  annotations:
    policy.mcm.ibm.com/categories: 'SystemAndCommunicationsProtections,Syste
    policy.mcm.ibm.com/controls: 'MutationAdvisor,VA'
    policy.mcm.ibm.com/standards: HIPAA
    seed-generation: '1'
  finalizers:
    - propagator.finalizer.mcm.ibm.com
  generation: 7
  resourceVersion: '1390799'
spec:
  namespaces:
    exclude:
      - kube*
    include:
      - default
  remediationAction: enforce
```

Policy-pod (Policy Compliant)

Policy status

Search

Cluster name	Policy compliant	Policy valid
local-cluster	1/1	1/1
mycluster	1/1	1/1

items per page 10 | 1-2 of 2 items — 1 of 1 pages — 1

Policies

Search

Compliant	Name	Cluster compliant	Cluster not compliant
Compliant	policy-role	local-cluster mycluster	-

items per page 10 | 1-1 of 1 items — 1 of 1 pages — 1

Policy-pod (Placement policy and bindings)

Placement policies

Search

Name	Namespace	Replicas	Cluster selector	Resource selector	Decisions	Created
placement-role	mcm	-	-	-	local-cluster, mycluster	18 minutes ago

items per page 10 | 1-1 of 1 items 1 of 1 pages 1

Placement bindings

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/