

Red Hat Advanced Cluster Management Demonstration Script

Hands-on part:

Exercise 1: Environment Preparation

Following the RHPDS request for the OCP4 ACM Hub item you can connect to the ACM console. See the email for details on the URL and access details.

An example of the email looks like this, the URL for the ACM Console is at the bottom of it:

THESE ENVIRONMENTS WILL ONLY BE AVAILABLE 1 DAY AFTER PROVISIONING. THERE ARE NO EXTENSIONS.

Troubleshooting and Access issues:

Always refer to the instructions to understand how to properly access and navigate your environment. If you need help using the SSH client on your computer you can consult <http://www.opentlc.com/ssh.html>.

NOTICE: Your environment will expire and be deleted at 2021-02-18 00:00:00 -0500.

In order to conserve resources, we cannot archive or restore any data in this environment. All data will be lost upon expiration.

Here is some important information about your environment:

Openshift Master Console: <https://console-openshift-console.apps.cluster-mel-753a.mel-753a.sandbox1520.opentlc.com>
Openshift API for command line 'oc' client: <https://api.cluster-mel-753a.mel-753a.sandbox1520.opentlc.com:6443>
Download oc client from <http://d3s3zgyaz8cp2d.cloudfront.net/pub/openshift-v4/clients/ocp/4.6.12/openshift-client-linux-4.6.12.tar.gz>

HTPasswd Authentication is enabled on this cluster.

Users user1 .. user5 are created with password `openshift`

User `admin` with password `v85occoYWh0foyaEp` is cluster admin.

Your ACM console is available at:

<https://multicloud-console.apps.cluster-mel-753a.mel-753a.sandbox1520.opentlc.com>

You can access your bastion via SSH:

ssh sdelord-redhat.com@bastion.mel-753a.sandbox1520.opentlc.com

Make sure you use the username 'sdelord-redhat.com' and the password 'wcPTNpF688Ye' when prompted.

Navigate to the ACM URL and you will be prompted with the following screen



Your connection is not private

Attackers might be trying to steal your information from **multicloud-console.apps.cluster-mel-753a.mel-753a.sandbox1520.opentlc.com** (for example, passwords, messages or credit cards). [Learn more](#)

NET::ERR_CERT_AUTHORITY_INVALID

Help improve security on the web for everyone by sending [URLs of some pages that you visit, limited system information, and some page content](#) to Google. [Privacy policy](#)

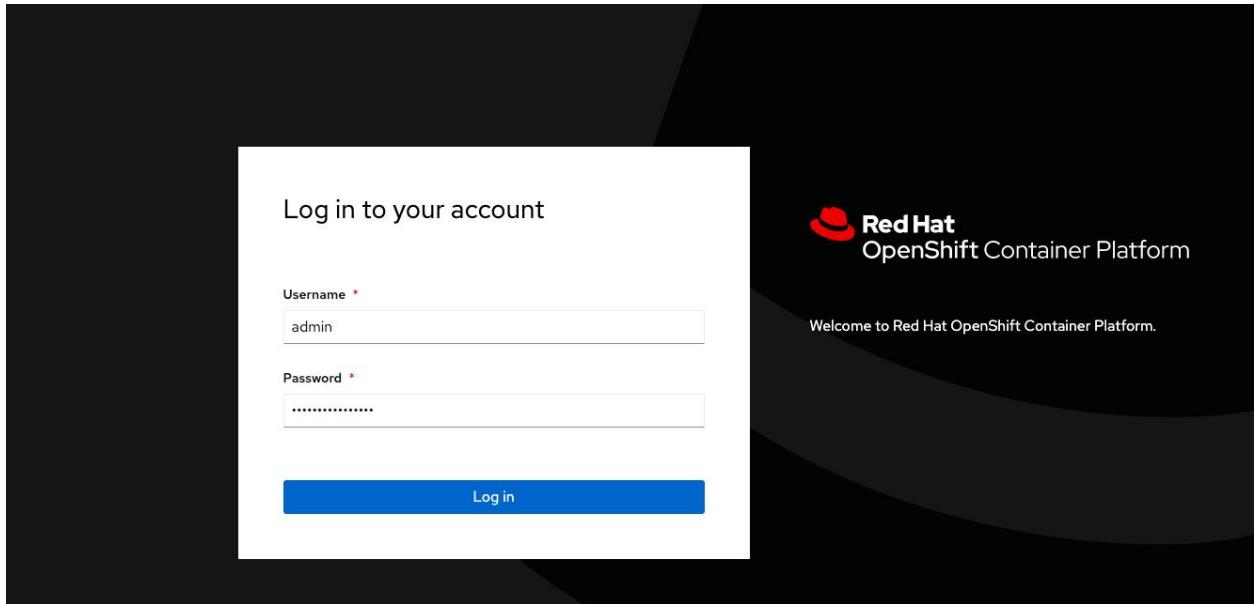
[Hide advanced](#)

[Back to safety](#)

This server could not prove that it is **multicloud-console.apps.cluster-mel-753a.mel-753a.sandbox1520.opentlc.com**; its security certificate is not trusted by your computer's operating system. This may be caused by a misconfiguration or an attacker intercepting your connection.

[Proceed to multicloud-console.apps.cluster-mel-753a.mel-753a.sandbox1520.opentlc.com \(unsafe\)](#)

Click on proceed to `multicloud-console.apps.<clustername>.opentlc.com` to enter your credentials.



Once logged in to ACM, you will be presented with the Home page of ACM.

Navigate to the **Clusters lifecycle** screen (by clicking on Go to Clusters in the Cluster lifecycle Box - highlighted in Red in the screenshot below)

Welcome! Let's get started.

Red Hat Advanced Cluster Management for Kubernetes provides the tools and capabilities to address various challenges with managing multiple clusters and consoles, distributed business applications, and inconsistent security controls across Kubernetes clusters that are deployed on-premises, or across public clouds.

⋮ Red Hat Advanced Cluster Management for Kubernetes ⋮ admin ▾

Cluster lifecycle
Create, update, scale, and remove clusters reliably, consistently using an open-source programming model that supports and encourages Infrastructure as Code best practices and design principles.
[Go to Clusters](#)

This will take you to another screen in which you will notice that the **HUB cluster** is already imported (local-cluster), we will use this later in the guide.

The screenshot shows the Red Hat Advanced Cluster Management for Kubernetes interface. At the top, there's a navigation bar with the Red Hat logo and the text "Advanced Cluster Management for Kubernetes". On the right side of the bar are icons for search, refresh, and user authentication ("admin"). Below the bar, the title "Clusters" is displayed with a help icon. Underneath, there are two tabs: "Clusters" (which is selected) and "Provider connections". A search bar labeled "Find" is positioned above a table. The table has columns for "Name", "Status", "Distribution version", "Labels", and "Nodes". One row is visible, showing "local-cluster" as the name, "Ready" as the status, "OpenShift 4.6.12(Upgrade available)" as the distribution version, a "cloud=Amazon +6" label, and 6 nodes. At the bottom of the table area, it says "Items per page: 20 | 1 of 1 pages".

This is the end of the first activity. You can do the set of extra activities below to familiarise yourself with the environment.

Extra activities:

Extra activity 1- Discover the local-cluster

In the previous screenshot, click on local-cluster. This will take you to the detailed view of the local-cluster (Hub cluster where ACM is deployed).

The screenshot shows the detailed view of the "local-cluster". At the top, there's a navigation bar with the Red Hat logo and the text "Advanced Cluster Management for Kubernetes". On the right side of the bar are icons for search, refresh, and user authentication ("admin"). Below the bar, the title "local-cluster" is displayed with a help icon. Underneath, there are three tabs: "Overview" (which is selected), "Nodes", and "Cluster settings". The main area is divided into sections. The "Details" section contains information about the cluster: Cluster ID (local-cluster), Status (Ready), Distribution version (OpenShift 4.6.12(Upgrade available)), and Labels (cloud=Amazon +6). It also shows the Cluster API address (-), Console URL (<https://console-openshift-console.apps.cluster-mel-753a.mel-753a.sandbox1520.opentlc.com>), and Username/password (-). Below this, the "Status" section shows metrics: 6 Nodes, 0 Applications, and 0 Policy violations.

The view displayed is the Overview tab, that shows the status of the cluster, how many nodes it has, how many Applications are deployed on it and how many Policy violations can be found on it.

Please also notice the Labels associated with the local-cluster. For this, just drag your mouse over the cloud=Amazon + 6 bubble. Labels play an important role for placement of applications and policies, they will be created, updated and used as part of the next exercises.

The screenshot shows the Red Hat Advanced Cluster Management for Kubernetes interface. At the top, there's a navigation bar with the Red Hat logo and the title "Advanced Cluster Management for Kubernetes". Below the navigation bar, the URL "Clusters / local-cluster" is visible, along with an "Actions" dropdown menu.

The main content area displays the cluster details under the "Overview" tab. Key information includes:

- Cluster ID:** local-cluster
- Status:** Ready (green)
- Distribution version:** OpenShift 4.6.12 (Upgrade available)
- Labels:** cloud=Amazon (+6) - A tooltip shows labels like clusterID=09b8bbaf-cf98-4bf0-a21a-2c35d5392de1, installer.name=multiclusterhub, installer.namespace=open-cluster-management, local-cluster=true, name=local-cluster, vendor=OpenShift.
- Cluster API address:** -
- Console URL:** https://console-openshift-console.apps.cluster-mel-753a.mel-753a.sandbox1520.opentlc.com
- Username/password:** -

Below the cluster details, there are three summary sections:

- Nodes:** 6 (0 nodes inactive)
- Applications:** 0 (Go to Applications)
- Policy violations:** 0 (Go to Policies)

Explore each of the other tabs (Nodes, Cluster settings).

For example when you click on the Nodes tab, you will see the various types of Nodes, their role, where they are located, their size.

From the screen below, you can see that this OpenShift cluster is sitting in AWS in the ap-southeast-1 region and that all 6 nodes (workers and masters) are of the same instance type (m5.xlarge) and split across 3 different zones (1a, 1b and 1c) to provide resiliency within the cluster.

The screenshot shows the Red Hat Advanced Cluster Management for Kubernetes interface with the "Nodes" tab selected. The cluster is named "local-cluster".

The table displays the following node information:

Name	Role	Region	Zone	Instance type	Size - Core and memory
ip-10-0-140-222.ap-southeast-1.compute.internal	worker	ap-southeast-1	ap-southeast-1a	m5.xlarge	4/15.2Gi
ip-10-0-148-199.ap-southeast-1.compute.internal	master	ap-southeast-1	ap-southeast-1a	m5.xlarge	4/15.37Gi
ip-10-0-168-126.ap-southeast-1.compute.internal	worker	ap-southeast-1	ap-southeast-1b	m5.xlarge	4/15.37Gi
ip-10-0-181-128.ap-southeast-1.compute.internal	master	ap-southeast-1	ap-southeast-1b	m5.xlarge	4/15.2Gi
ip-10-0-196-24.ap-southeast-1.compute.internal	master	ap-southeast-1	ap-southeast-1c	m5.xlarge	4/15.37Gi
ip-10-0-208-125.ap-southeast-1.compute.internal	worker	ap-southeast-1	ap-southeast-1c	m5.xlarge	4/15.37Gi

At the bottom, there are pagination controls and a message indicating 1 of 1 pages.

Extra activity 2 - Discover the Overview Screen

Go back to the ACM home page and click this time on the End-to-end visibility (Go to Overview) box.

The screenshot shows the Red Hat Advanced Cluster Management for Kubernetes home page. At the top, there's a banner with the Red Hat logo and the text: "Red Hat Advanced Cluster Management for Kubernetes". Below the banner, there's a brief description of the tool's capabilities. To the right of the description is a 3D diagram showing three interconnected nodes: a laptop, a server rack, and a cloud icon. Below the banner, there are four main sections: "End-to-end visibility" (highlighted with a red box), "Cluster lifecycle", "Application lifecycle", and "Governance, Risk, and Compliance". Each section has a small icon, a title, a brief description, and a "Go to [Section]" link.

This will then take you to another screen that looks like the screenshot below

The screenshot shows the "Overview" dashboard of Red Hat Advanced Cluster Management for Kubernetes. At the top, it displays cluster information: 1 cluster, 01 OpenShift. On the right, there are buttons for "Add provider connection" and "Refresh every 1m" (with a timestamp of 09:34:01). Below this, there's a summary section with counts for Apps (0), Clusters (1), Kubernetes types (1), Regions (1), Nodes (6), and Pods (0). Further down, there are three main status indicators: "Cluster compliance" (100% Compliant), "Pods" (0 Running, 0 Pending, 0 Failed), and "Cluster status" (100% Ready). Each indicator has a circular progress bar and a breakdown of the count by status.

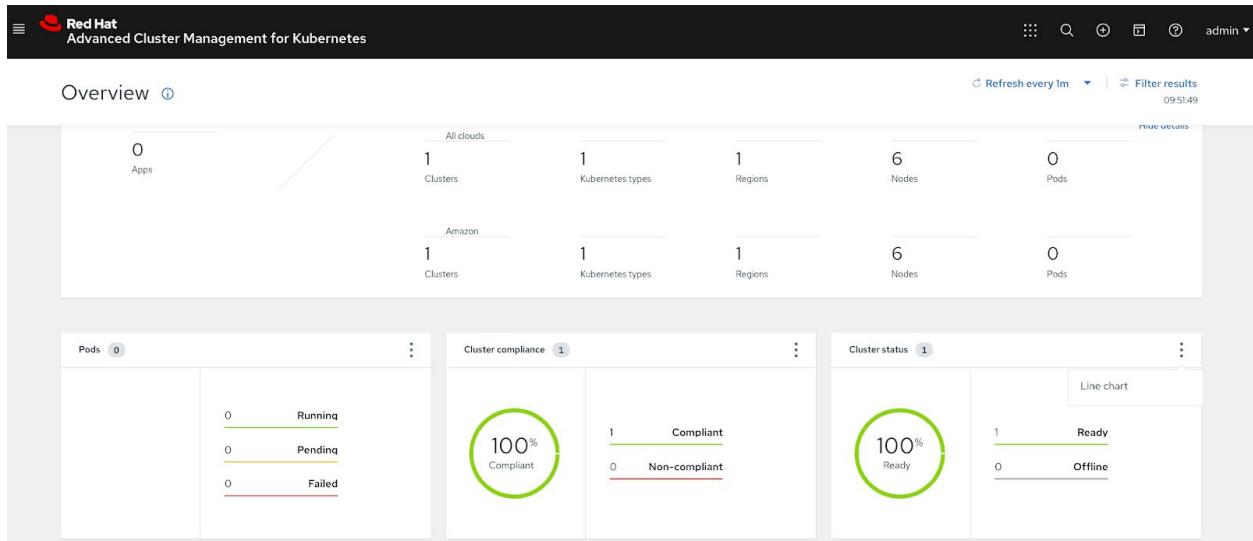
You can view the following information about your clusters on the Overview dashboard:

- Cluster, node, and pod counts across all clusters and for each provider
- Cluster status

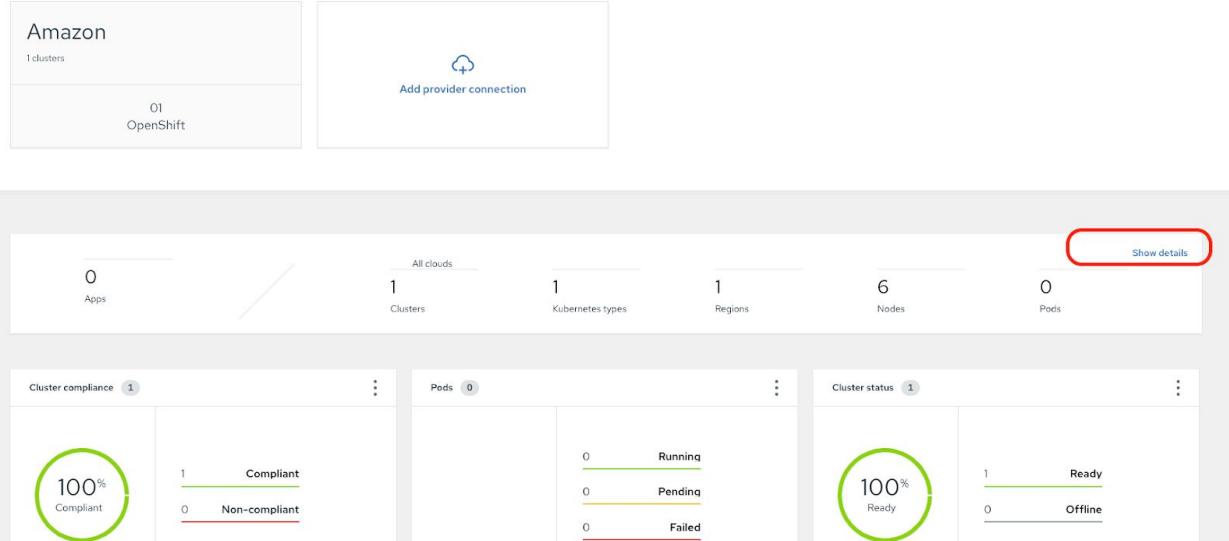
- Cluster compliance
- Pod status

You can personalize your view of the Overview dashboard by clicking and dragging to reorganize the cards.

For example in the screenshot below, I have changed the cluster status from Pie chart to Line chart (click on the 3 grey dots in the Cluster status box) and swapped around the Pods box and Cluster compliance box.



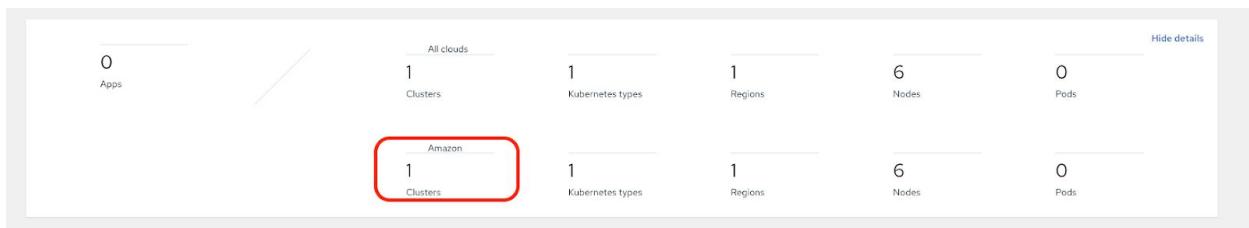
On the central box, click on Show details



This will expand the box with a summary of the various clusters.

Click on the Amazon 1 Clusters box.

It will take you to the search view (e.g clicking performed a search on cluster & local-cluster).



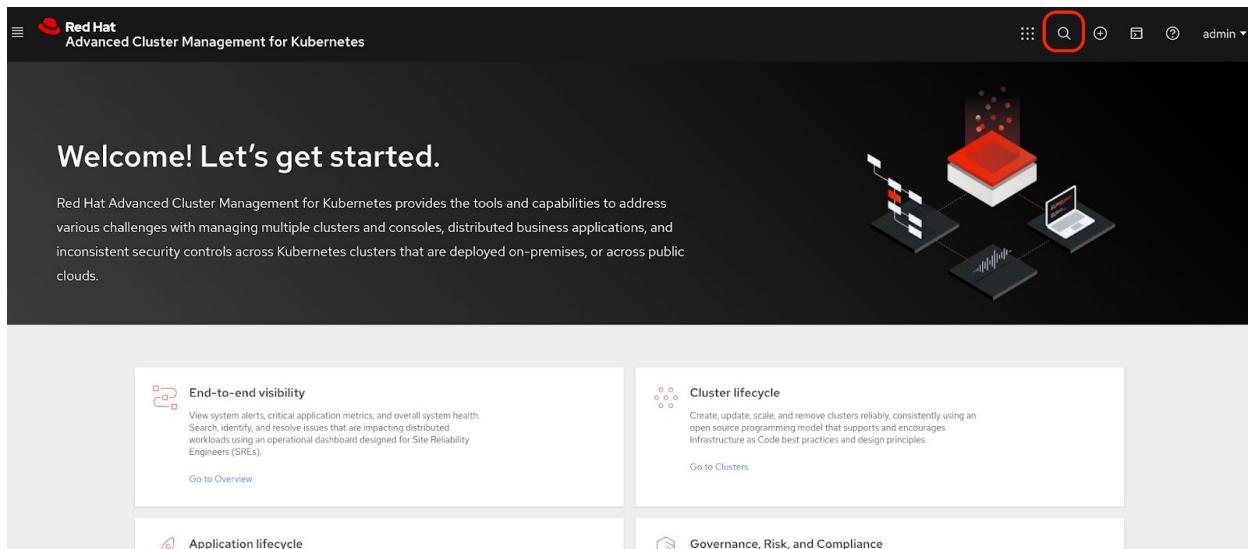
The screenshot shows the Red Hat Advanced Cluster Management for Kubernetes interface. At the top, there's a navigation bar with the Red Hat logo and the title "Advanced Cluster Management for Kubernetes". On the right side of the bar are icons for search, refresh, and user authentication ("admin"). Below the bar is a search bar with the placeholder "Search" and a magnifying glass icon. Underneath the search bar are two search filters: "Saved searches" and "Open new search tab". The main content area displays search results for "kindcluster" and "name:local-cluster". A red box highlights the search filters. Below the filters, there are several summary cards showing related resources: 337 RELATED POD, 1.3k RELATED SECRET, 6 RELATED NODE, 1 RELATED CERTPOLICYCONTROLLER, 1 RELATED APPLICATIONMANAGER, 1 RELATED IAMPOLICYCONTROLLER, 1 RELATED WORKMANAGER, 1 RELATED POLICYCONTROLLER, 1 RELATED USER, and 1 RELATED CERTIFICATESIGNINGREQUEST. Further down, a section titled "Cluster (1) ^" shows a table with one item: "local-cluster". The table columns are: Name, Available, Hub accepted, Joined, Nodes, Kubernetes version, CPU, Memory, Console URL, and Labels. The "Labels" column shows a tooltip for "cloud=Amazon" with a count of "+6". At the bottom of the table are pagination controls: "Items per page: 20" (with a dropdown arrow), "1-1 of 1 items", "1 of 1 pages", and navigation arrows.

The next extra activity will show you how to perform this search directly and how to explore some of the results for it.

Extra activity 3 - Perform a search & explore results for the search

In the previous extra activity, we performed by clicking on the overview environment a search related to kind:cluster & name:local-cluster.

We can also perform this action directly. Going back to the home screen, you can click on the search icon directly in the top right corner.



Welcome! Let's get started.

Red Hat Advanced Cluster Management for Kubernetes provides the tools and capabilities to address various challenges with managing multiple clusters and consoles, distributed business applications, and inconsistent security controls across Kubernetes clusters that are deployed on-premises, or across public clouds.

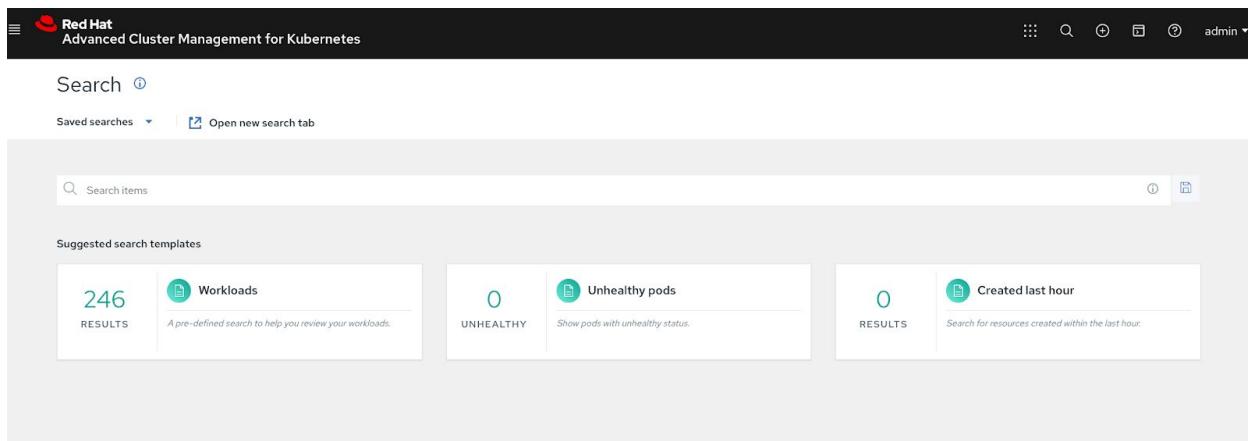
End-to-end visibility
View system alerts, critical application metrics, and overall system health. Search, identify, and resolve issues that are impacting distributed workloads using an operational dashboard designed for Site Reliability Engineers (SREs).

Cluster lifecycle
Create, update, scale, and remove clusters reliably, consistently using an open source programming model that supports and encourages Infrastructure as Code best practices and design principles.

Application lifecycle

Governance, Risk, and Compliance

This takes you to the search page.



Search ①

Saved searches ▼ | Open new search tab

Search items

Suggested search templates

RESULTS	Workloads	UNHEALTHY	RESULTS	Created last hour
246	A pre-defined search to help you review your workloads.	0	Unhealthy pods	0
RESULTS	Show pods with unhealthy status.	UNHEALTHY	RESULTS	Search for resources created within the last hour.

By clicking on the Search items box, you will see a filter tab appearing.

You can choose between many FILTERS and associated FILTER VALUES. Once you've selected the FILTERS, click on it and then click next to the filter and all associated values will appear.

For example, in the screenshot below I have performed the kind (FILTERS) cluster (KIND VALUE) search category and I am now performing the name (FILTER) local-cluster (NAME VALUE)

Red Hat Advanced Cluster Management for Kubernetes

Search ①

Saved searches ▼ Open new search tab

kindcluster X name: X

NAME VALUES
local-cluster

337 RELATED POD	1 RELATED IAMPOLICYCONTROLLER	RELATED SECRET	6 RELATED NODE	1 RELATED CERTPOLICYCONTROLLER	1 RELATED APPLICATIONMANAGER
WORKMANAGER		1 RELATED POLICYCONTROLLER	1 RELATED USER	1 RELATED CERTIFICATESIGNINGREQUEST	

Show all (119) ▾

Cluster (1) ▾

Name	Available	Hub accepted	Joined	Nodes	Kubernetes version	CPU	Memory	Console URL	Labels
local-cluster	True	True	True	6	v1.19.0+9c69bdc	24	94094Mi	Launch	cloud=Amazon +6

Items per page: 20 ▾ | 1-1 of 1 items

1 of 1 pages < >

Click on the RELATED POD box, we can see all the Pods deployed on the local-cluster.

Red Hat Advanced Cluster Management for Kubernetes

Search ①

Saved searches ▼ Open new search tab

kindcluster X name:local-cluster X

337 RELATED POD	1.3k RELATED SECRET	6 RELATED NODE	1 RELATED CERTPOLICYCONTROLLER	1 RELATED APPLICATIONMANAGER
1 RELATED IAMPOLICYCONTROLLER	1 RELATED WORKMANAGER	1 RELATED POLICYCONTROLLER	1 RELATED USER	1 RELATED KLUSTERLETADDONCONFIG

Show all (118) ▾

Cluster (1) ▾

Related Pod (337)

Name	Namespace	Cluster	Status	Restarts	Host IP	Pod IP	Created	Labels
cert-manager-fb256-5b4f78b889-nt4cw	open-cluster-management	local-cluster	Running	0	10.0.168.126	10.131.0.24	2 hours ago	app=cert-manager +5
cert-manager-fb256-5b4f78b889-65lzx	open-cluster-management	local-cluster	Running	0	10.0.140.222	10.128.2.16	2 hours ago	app=cert-manager +5
multus-wiftv	openshift-multus	local-cluster	Running	0	10.0.181.128	10.0.181.128	3 hours ago	app=multus +5
cert-manager-webhook-85dfbb8c4-4h7kt	open-cluster-management	local-cluster	Running	2	10.0.140.222	10.128.2.17	2 hours ago	app=webhook +5

You can then drill down on any of the Pods there.

Clicking on the cert-manager-webhook-85dfbb8c4-4h7kt Pod takes me to another screen with the YAML view of the Pod and the associated Logs.

Red Hat Advanced Cluster Management for Kubernetes

Search / cert-manager-webhook-85dfbbd8c4-4h7kt

[YAML](#) [Logs](#)

Cluster: local-cluster Namespace: open-cluster-management

```

1: kind: Pod
2: apiVersion: v1
3: metadata:
4:   name: cert-manager-webhook-85dfbbd8c4-4h7kt
5:   generateName: cert-manager-webhook-85dfbbd8c4-
6:   namespace: open-cluster-management
7:   selfLink: ...
8:   ... /api/v1/namespaces/open-cluster-management/pods/cert-manager-webhook-85dfbbd8c4-4h7kt
9:   uid: f5b73a3-319e-4359-b141-dab875203eb7
10:  resourceVersion: '36316'
11:  creationTimestamp: '2021-02-11T21:29:81Z'
12:  labels:
13:    app: webhook
14:    chart: webhook-2.1.0
15:    heritage: Helm
16:    k8s-app: cert-manager-webhook
17:    pod-template-hash: 85dfbbd8c4
18:    releases: cert-manager-webhook-0844e
19:  annotations:
20:    k8s.v1.cni.cncf.io/network-status: |-
21:      {
22:        "name": "",
23:        "interface": "eth0",
24:        "ips": [
25:          "10.128.2.17"
26:        ],
27:        "default": true,

```

[Edit](#)

Red Hat Advanced Cluster Management for Kubernetes

Search / cert-manager-webhook-85dfbbd8c4-4h7kt

[YAML](#) [Logs](#)

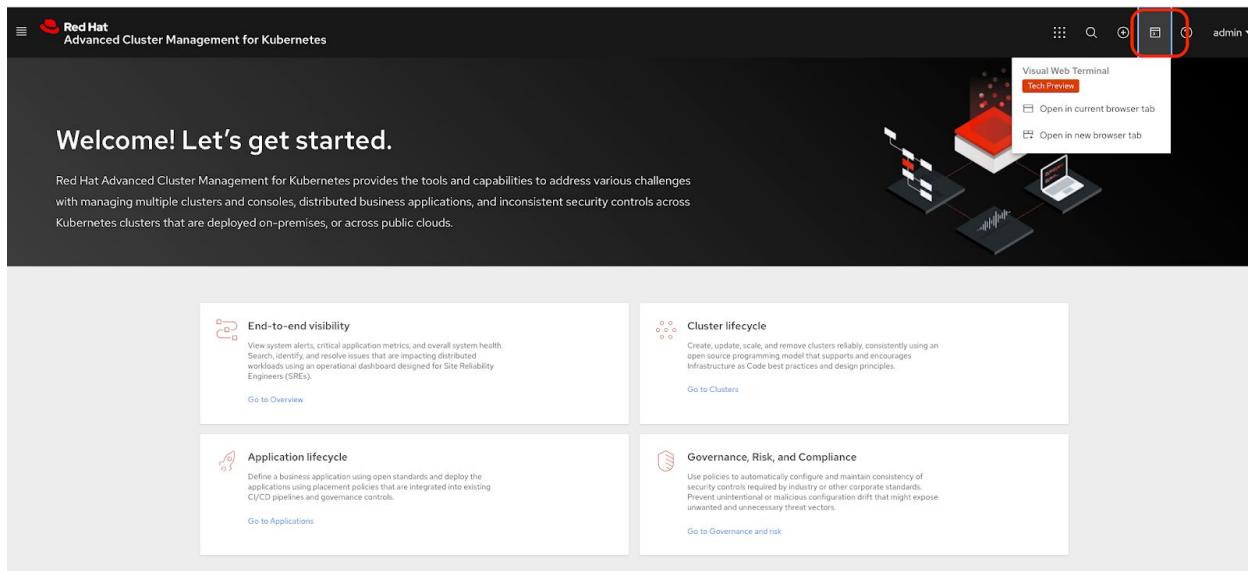
Cluster: local-cluster	Namespace: open-cluster-management	Container: cert-manager-webhook
I021 21:30:33.490975	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:30:33.528991	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:30:58.932159	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[{"op": "\\\\"remove\"", "path": "\\\"spec\\\"\\\"dnsNames\\\""}, {"op": "\\\\"add\"", "path": "\\\"status\\\"\\\"value\\\""}, {"value: \"{}"}]"	
I021 21:31:00.129187	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:31:00.184854	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:31:00.184815	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:31:49.517328	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:31:49.715829	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:31:50.416565	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:31:50.615733	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:31:50.812349	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:31:51.536613	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:31:52.523613	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:31:52.540095	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:31:52.540096	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:32:26.708590	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:32:26.708591	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:32:26.709486	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:32:26.710432	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:32:26.729921	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:32:26.873430	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:32:27.073615	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:32:27.273634	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:33:26.073838	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:33:28.274192	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	
I021 21:33:28.874926	I mutation.go:120] cert-manager "msg" = "generated patch" "patch": "[]"	

Extra activity 4 - Use the Visual Web Terminal to run commands from a central location

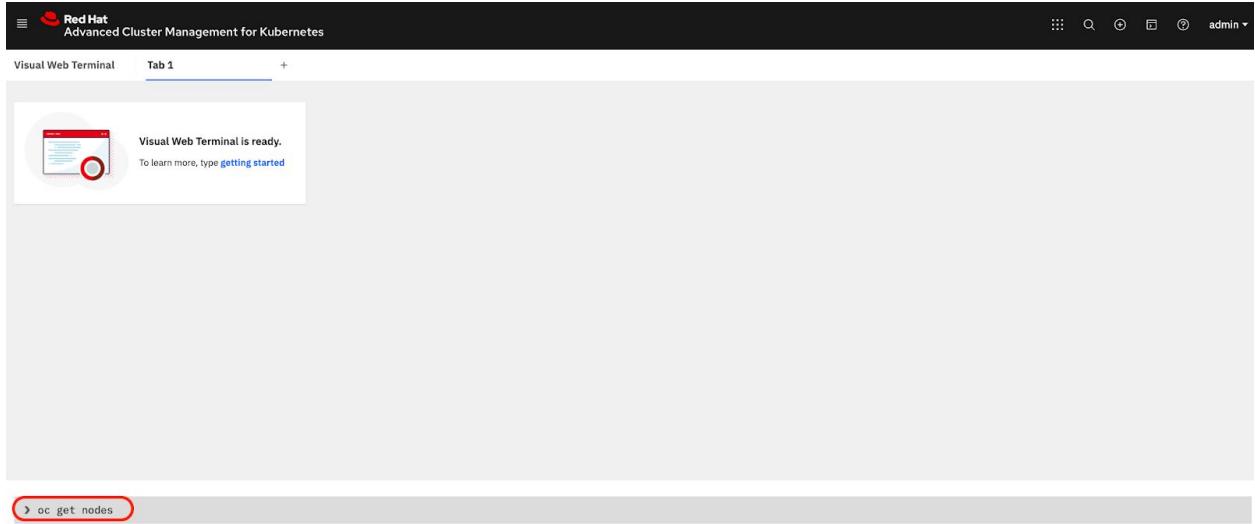
The Visual Web Terminal is a tech preview feature. You can use the Visual Web Terminal to run many commands across your cluster(s).

Run commands and get outputs without leaving the entire console. You can run multiple commands, similar to what you can do in a standard terminal.

With Visual Web Terminal, commands that return cluster resource data are displayed visually in interactive tabular format, rather than plain text that is displayed in a standard terminal. You can see the data, click on a row, and then see more details about that resource.

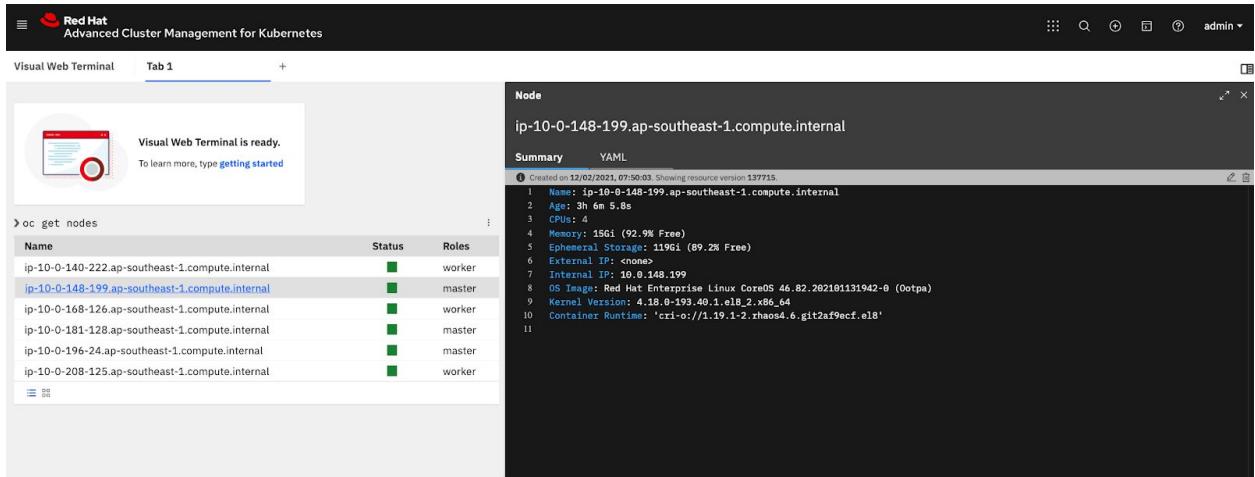


You will then be able to run commands on the various clusters. For example, type `oc get nodes` (as per the screenshot).



Results will be displayed and you can then interact directly with them by clicking on them.

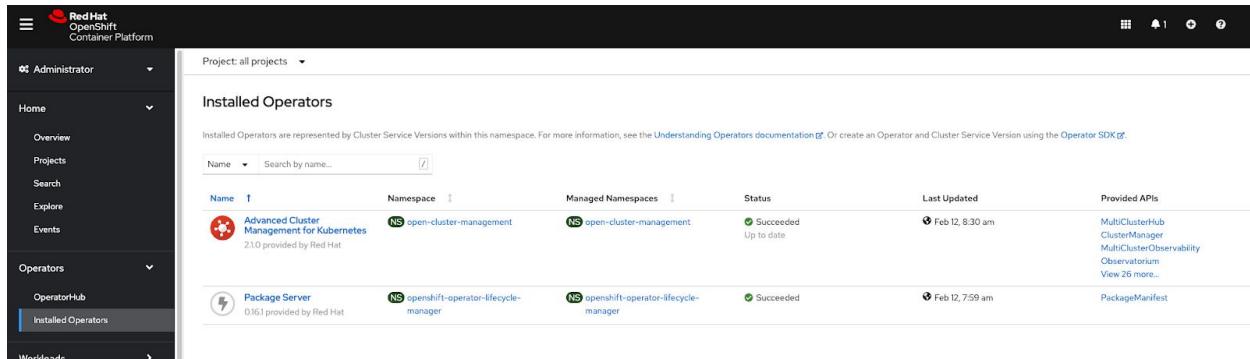
For example by clicking on one of the nodes, you get more information about this node. In this screenshot, I have clicked on ip-10-0-148-199.ap-southeast-1-compute.internal node. And on the right side, I can see all the information related to it.

A screenshot of the Red Hat Advanced Cluster Management for Kubernetes interface. On the left, a terminal window shows the command 'oc get nodes' and a list of nodes. One node, 'ip-10-0-148-199.ap-southeast-1-compute.internal', is highlighted with a red box. On the right, a detailed view of this node is shown in a separate window titled 'Node ip-10-0-148-199.ap-southeast-1.compute.internal'. The window has tabs for 'Summary' and 'YAML'. The 'Summary' tab displays node details: Name: ip-10-0-148-199.ap-southeast-1.compute.internal, Status: master, Roles: master. The 'YAML' tab shows the node configuration in YAML format. The configuration includes fields like Name, Age, CPU, Memory, and OS Image. The entire right-hand panel is also highlighted with a red box.

Extra activity 5 - Check ACM deployment on the Hub cluster (local-cluster)

You can also check how ACM has been deployed on the OpenShift Hub cluster. For this, go back to the email from RHPDS and login to the OpenShift Console (you may have to reenter the username and password).

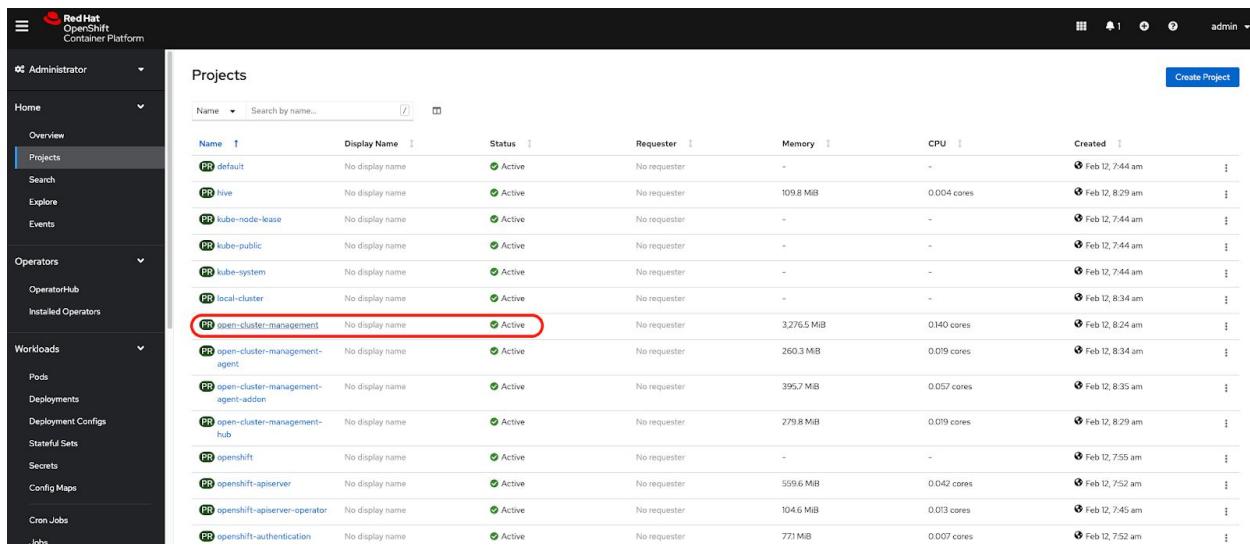
Once you are on the OpenShift console, go to Operators-> Installed Operators.



Name	Namespace	Managed Namespaces	Status	Last Updated	Provided APIs
Advanced Cluster Management for Kubernetes 2.10 provided by Red Hat	NS open-cluster-management	NS open-cluster-management	Succeeded Up to date	Feb 12, 8:30 am	MultiClusterHub ClusterManager MultiClusterObservability Observatorium View 26 more...
Package Server 0.16.1 provided by Red Hat	NS openshift-operator-lifecycle-manager	NS openshift-operator-lifecycle-manager	Succeeded	Feb 12, 7:59 am	PackageManifest

You can see that ACM is deployed in the open-cluster-management Namespace.

Click on the Projects (Home -> Projects Tab) and click on the open-cluster-management Project



Name	Display Name	Status	Requester	Memory	CPU	Created
PR default	No display name	Active	No requester	-	-	Feb 12, 7:44 am
PR hive	No display name	Active	No requester	109.8 MiB	0.004 cores	Feb 12, 8:29 am
PR kube-node-lease	No display name	Active	No requester	-	-	Feb 12, 7:44 am
PR kube-public	No display name	Active	No requester	-	-	Feb 12, 7:44 am
PR kube-system	No display name	Active	No requester	-	-	Feb 12, 7:44 am
PR local-cluster	No display name	Active	No requester	-	-	Feb 12, 8:34 am
PR open-cluster-management	No display name	Active	No requester	3,276.5 MiB	0.140 cores	Feb 12, 8:24 am
PR open-cluster-management-agent	No display name	Active	No requester	260.3 MiB	0.019 cores	Feb 12, 8:34 am
PR open-cluster-management-agent-addon	No display name	Active	No requester	395.7 MiB	0.057 cores	Feb 12, 8:35 am
PR open-cluster-management-hub	No display name	Active	No requester	279.8 MiB	0.019 cores	Feb 12, 8:29 am
PR openshift	No display name	Active	No requester	-	-	Feb 12, 7:55 am
PR openshift-apiserver	No display name	Active	No requester	559.6 MiB	0.042 cores	Feb 12, 7:52 am
PR openshift-apiserver-operator	No display name	Active	No requester	104.6 MiB	0.013 cores	Feb 12, 7:45 am
PR openshift-authentication	No display name	Active	No requester	771 MiB	0.007 cores	Feb 12, 7:52 am

This is where ACM has been deployed. Click on the 1 Route item and you will recognise the ACM Web console URL that was sent by email.

The screenshot shows two main sections of the ACM Web Console:

- Project Overview:** This section displays details about the project "open-cluster-management". It includes tabs for Overview, Details, YAML, Workloads, and Role Bindings. The Details tab shows the project's name, requester, labels, and description. The Status tab shows utilization metrics for CPU, Memory, Filesystem, Network Transfer, and Pod count over a 1-hour period. The Activity tab shows recent events, including updates to API services and custom resource definitions, along with a log of ingress requests.
- Routes:** This section lists routes for the "open-cluster-management" project. A single route named "multicloud-console" is shown, with its status as "Accepted" and its location as "https://multicloud-console.apps.cluster-mel-753a.mel-753a.sandbox1520.openshift.com:80".

This is the end of the extra activities for the first exercise.

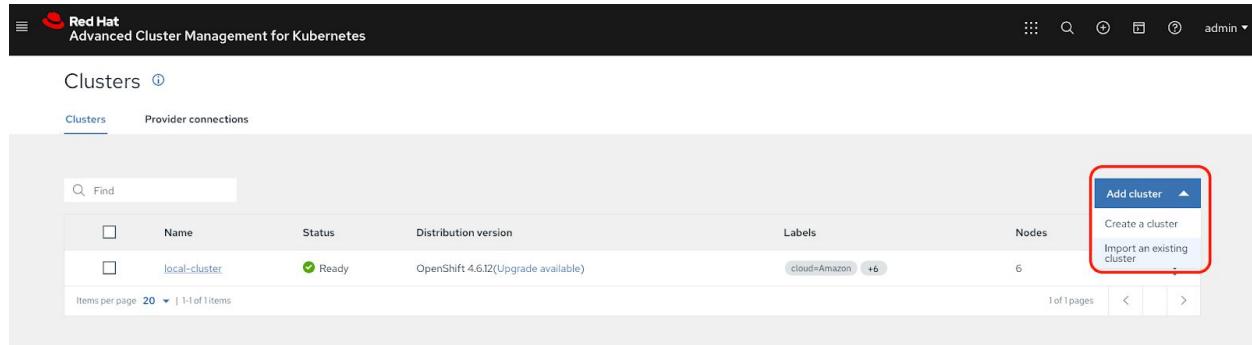
Feel free to go through the various components of the ACM Web console. More information is available under [ACM Web Console](#)

Second Exercise - Import a Cluster

Back to the Home Page in ACM.

Click on Cluster lifecycle (Go to Cluster).

Click on **Add Cluster --> Import an existing cluster**



The screenshot shows the ACM web interface. At the top, there's a navigation bar with the Red Hat logo and the text "Advanced Cluster Management for Kubernetes". On the far right of the bar, there are icons for search, add, and help, along with a dropdown menu for "admin". Below the bar, the main content area has a title "Clusters" with a tooltip. Underneath, there are two tabs: "Clusters" (which is selected) and "Provider connections". A search bar labeled "Find" is positioned above a table. The table has columns for "Name", "Status", "Distribution version", "Labels", and "Nodes". One row in the table is for a cluster named "local-cluster" which is "Ready", running "OpenShift 4.6.12(Upgrade available)", has labels "cloud=Amazon" and "+6", and 6 nodes. At the bottom of the table, there are buttons for "Items per page" (set to 20), a page number indicator "1 of 1 pages", and navigation arrows. In the top right corner of the main content area, there is a blue button labeled "Add cluster" with a dropdown menu. The dropdown menu contains two options: "Create a cluster" and "Import an existing cluster", with "Import an existing cluster" being the one highlighted by a red box.

This will take you to the Import an existing cluster page.

Fill the various fields on the page:

- Configuration -> Cluster name: Just specify any lower-case name. In the rest of this tutorial, we have used **newcluster** as the name (please note that the name you use for the cluster is not relevant).
- Labels:
 - Cloud: select auto-detect
 - Environment: select the **dev** label.
 - Additional labels: leave this field empty

We will be using those labels later on to provision applications.

Your screen should look like this

Clusters / Import an existing cluster ⓘ YAML: off

Cancel Generate command

^ Configuration

Cluster name* ⓘ

newcluster

^ Labels

Cloud ⓘ

auto-detect X ▼

Environment ⓘ

dev X ▼

Additional labels ⓘ

Enter key=value then press enter or comma

Once finished click **Generate command** and copy the kubectl command.

The screenshot shows the Red Hat Advanced Cluster Management for Kubernetes interface. At the top, there's a navigation bar with the Red Hat logo and the title "Advanced Cluster Management for Kubernetes". Below the title, the path "Clusters / Import an existing cluster" is shown. On the right side of the header, there are several icons: three dots, a magnifying glass, a plus sign, a question mark, and a user icon labeled "admin". The main content area has a heading "Import an existing cluster" with a help icon. To the right are two buttons: "Import another" and "View cluster". Below the heading, a message says "Command generated" with a green checkmark. A note below it says "Run the command with kubectl configured for your targeted cluster to start the cluster import." A copy button icon (a clipboard with a pencil) is highlighted with a red circle. At the bottom, there's a progress bar.

Using the credentials for the managed cluster, SSH to the Bastion host **on the managed cluster**. Make sure you use the Bastion host from the RHPDS OCP4 ACM Managed email (as per the screenshot below).

Thank you for ordering OCP4 ACM Managed
Your environment for RHPDS-RH-sdelord-redhat.com-PROD_OCP4_ACM_MANAGED_RHPDS-mel-2bda_COMPLETED has been provisioned.
Access to the environment will be granted as soon as the environment deployment is complete.

Please refer to the instructions for next steps and how to access your environment.

Troubleshooting and Access issues:

Always refer to the instructions to understand how to properly access and navigate your environment. If you need help using the SSH client on your computer you can consult <http://www.opentlc.com/ssh.html>.

NOTICE: Your environment will expire and be deleted in **7 day(s)** at **2021-02-18 00:00:00 -0500**.

In order to conserve resources, we cannot archive or restore any data in this environment. All data will be lost upon expiration.

Here is some important information about your environment:

Openshift Master Console: <https://console-openshift-console.apps.cluster-mel-2bda.mel-2bda.sandbox944.opentlc.com>
Openshift API for command line 'oc' client: <https://api.cluster-mel-2bda.mel-2bda.sandbox944.opentlc.com:6443>
Download oc client from <http://d3s3zgyaz8cp2d.cloudfront.net/pub/openshift-v4/clients/ocp/4.5.16/openshift-client-linux-4.5.16.tar.gz>

HTPasswd Authentication is enabled on this cluster.

Users user1 .. user5 are created with password 'openshift'

User 'admin' with password 'w9v1W0fufSuFI0WM' is cluster admin.

You can access your bastion via SSH:

ssh sdelord-redhat.com@bastion.mel-2bda.sandbox944.opentlc.com

Make sure you use the username 'sdelord-redhat.com' and the password 'nV3p8ASa9Pmb' when prompted.

Once you are logged in, double check that you are connected to the right cluster by running an oc whoami --show-server command and check that the name of the cluster matches the one in the email.

```
[sdelord-redhat.com@bastion ~]$ oc whoami --show-server
https://api.cluster-mel-2bda.mel-2bda.sandbox944.opentlc.com:6443
[sdelord-redhat.com@bastion ~]$ ]
```

Paste the **Kubectl** command that you copied from ACM earlier in the bastion host, if all worked well you should see this on your terminal window

```
GVsYXlT2WWnvbmRz01AyC1AgICAgIHNlcnZpY2VBV2NvdWS01mFtz1oga2x1c3Rlcmx1dAoKLS0tCmFwaVz1cnNpb2461g9wZXJhdG9yLm9wZW4tY2x1c3R1c1tYW5h22Vt2W5
0Lm1vL3YxCmt:pbmQ61Et5dXN0ZXjsZXQkbWV0YWRhdGE6CiAgbmfzTzOga2x1c3Rlcmx1dApzcGVj0gogIGNsdXN0ZXJOYW1l0iBuZXdjbHVzdG9yCiAgbmFtZXNwYWNl0iBvc
GVU1WNsdxN0ZXItbWFuYWdlbWVudC1hZ2VuAogIHJ1Z21zdiJhdGlvbkltYWdlUHVsbfNwZWM6IHJ1Z21zdHJ5LnJlZGhhC5pb9yaGFjbT1vcmVnaXN0cmF0d9uLXJoZlw
4QHNoYTI1NjkpZDNhNDBjYjf1nZl0NTE0YzhmOTA0NGE3NjQ4YTQzYTVmYTY4MWJ1YmVjZGNkMNUyZDB1ZWRhY2ZiZDZhYTyxCiAgd29ya0ltYWdlUHVsbfNwZWM6IHJ1Z21zd
HJ5LnJlZGhhC5pb9yaGFjbT1vZay1yaGVs0EBzaGEyNTY6ZDg2Y2YzMjFizjgwYjk1MmNmNDUZZGFjMGY5ZWU0NDEwNdczMzc4jE3NTAzNjVjNzY10MMwNjMxDQ1ZTA
2NAo= | base64 --decode | kubectl apply -f -
customresourcedefinition.apirextensions.k8s.io/klusterlets.operator.open-cluster-management.io created
namespace/open-cluster-management-agent created
serviceaccount/klusterlet created
secret/bootstrap-hub-kubeconfig created
clusterrole.rbac.authorization.k8s.io/klusterlet created
clusterrole.rbac.authorization.k8s.io/open-cluster-management:klusterlet-admin-aggregate-clusterrole created
clusterrolebinding.rbac.authorization.k8s.io/klusterlet created
deployment.apps/klusterlet created
klusterlet.operator.open-cluster-management.io/klusterlet created
[sdelord-redhat.com@bastion ~]$ ]
```

On the ACM console, navigate back to the Clusters view. And wait for the cluster to become available (should be no more than 5 min).

You will first see the cluster newcluster becoming Ready as a Status and then after a short wait, all the rest of the information will be displayed (Distribution version, Nodes).

Clusters

Name	Status	Distribution version	Labels	Nodes
local-cluster	Ready	OpenShift 4.6.12(Upgrade available)	cloud=Amazon +6	6
newcluster	Ready	-	cloud=auto-detect +3	5

Clusters

Name	Status	Distribution version	Labels	Nodes
local-cluster	Ready	OpenShift 4.6.12(Upgrade available)	cloud=Amazon +6	6
newcluster	Ready	OpenShift 4.5.16(Upgrade available)	cloud=Amazon +4	5

Extra activities:

Extra activity 1- Discover the local-cluster

Similarly as Extra activity 1 in Exercise 1, explore the various tabs under newcluster (Overview, Nodes, Cluster settings).

newcluster

Cluster ID	newcluster	Cluster API address	-
Status	Ready	Console URL	https://console-openshift-console.apps.cluster-mel-2bda.mel-2bda.sandbox944.opentlc.com
Distribution version	OpenShift 4.5.16(Upgrade available)	Username/password	-
Labels	cloud=Amazon +4		

Nodes: 5

Applications: 0

Policy violations: 0

Extra activity 2 - Perform a search

Go to the Search Icon on the top right of ACM and issue a search for kind:cluster & name:newcluster.

The screenshot shows the ACM interface with a search bar at the top containing 'kind:cluster' and 'name:newcluster'. Below the search bar, there are several summary cards:

Count	Related Type
263	RELATED POD
1.1k	RELATED SECRET
5	RELATED NODE
1	RELATED KUBECONTROLLERMANAGER
1	RELATED CLOUDCREDENTIAL
1	RELATED OAUTH
1	RELATED ETCD
2	RELATED PRIORITYCLASS
2	RELATED CONSOLELINK
1	RELATED RANGEALLOCATION

Below these cards is a table titled 'Cluster (1) ▾' showing one entry:

Name	Available	Hub accepted	Joined	Nodes	Kubernetes version	CPU	Memory	Console URL	Labels
newcluster	True	True	True	5	v1.18.3+2fbd7c7	20	78356Mi	Launch	cloud=Amazon +4

At the bottom left, it says 'Items per page: 20 ▾ | 1-1 of 1 items'. At the bottom right, it says '1 of 1 pages' with navigation icons.

Extra activity 3 - Check the Klusterlet Agent on the managed cluster (newcluster)

You can also check how the various parts for the Klusterlet and associated functions have been deployed on the managed cluster. For this, go back to the email from RHPDS for the OCP4 ACM Managed request and login to the OpenShift Console.

Click on the Projects (Home -> Projects Tab) and click on the open-cluster-management-agent Project

The screenshot shows the OpenShift Container Platform interface with the 'Administrator' user logged in. The left sidebar has 'Home' selected under 'Projects'. The main area displays a table of projects:

Name	Display Name	Status	Requester	Memory	CPU	Created
PR default	No display name	Active	No requester	-	-	Feb 12, 7:46 am
PR kube-node-lease	No display name	Active	No requester	-	-	Feb 12, 7:46 am
PR kube-public	No display name	Active	No requester	-	-	Feb 12, 7:46 am
PR kube-system	No display name	Active	No requester	-	-	Feb 12, 7:46 am
PR newcluster	No display name	Active	No requester	-	-	Feb 12, 11:43 am
PR open-cluster-management-agent	No display name	Active	No requester	284.0 MiB	0.017 cores	Feb 12, 11:40 am
PR open-cluster-management-agent-addon	No display name	Active	No requester	399.9 MiB	0.075 cores	Feb 12, 11:42 am
PR openshift	No display name	Active	No requester	-	-	Feb 12, 7:57 am
PR openshift-apiserver	No display name	Active	No requester	345.1 MiB	0.029 cores	Feb 12, 7:54 am
PR openshift-apiserver-operator	No display name	Active	No requester	59.6 MiB	0.012 cores	Feb 12, 7:47 am

A red box highlights the two entries for 'open-cluster-management-agent' and 'open-cluster-management-agent-addon'.

The open-cluster-management-agent project is where the klusterlet agent gets deployed to connect back to the Hub cluster. Click on the Pods and see the various Pods deployed in there.

Red Hat OpenShift Container Platform

Administrator

Home Overview Projects Search Explore Events Operators Workloads Pods Deployments Deployment Configs Stateful Sets Secrets Config Maps Cron Jobs

Projects > Project Details

PR open-cluster-management-agent Active

Overview Details YAML Workloads Role Bindings

Details		Status		Activity	
Name	open-cluster-management-agent	Requester	No requester	View events	
Labels	No labels	Utilization		Ongoing	
		Resource	Usage	12:15 12:30 12:45 13:00	Recent Events
		CPU	16.37m	20m 10m	13:01 NS Updated CustomResour...
		Memory	284 MiB	400 MiB 300 MiB 200 MiB 100 MiB	12:57 D Updated open-cluster-m...
		Filesystem	348 KiB	600 KiB 400 KiB 200 KiB	12:57 D Updated CustomResour...
				30 KBps	12:57 D Updated /certpolicycontr...
					12:57 D Updated /policycontroller...
					12:57 D Updated /applicationman...
					12:57 D Updated /searchcollector...
					12:57 D Updated /iampolicycontr...

Inventory

- 3 Deployments
- 0 Deployment Configs
- 0 Stateful Sets
- 7 Pods

Check the open-cluster-management-addon project and see the various Pods used for applications, adds-on for policy, applications)

Red Hat OpenShift Container Platform

Administrator

Home Overview Projects Search Explore Events Operators Workloads Pods Deployments Deployment Configs Stateful Sets Secrets Config Maps Cron Jobs

Project: open-cluster-management-agent-addon

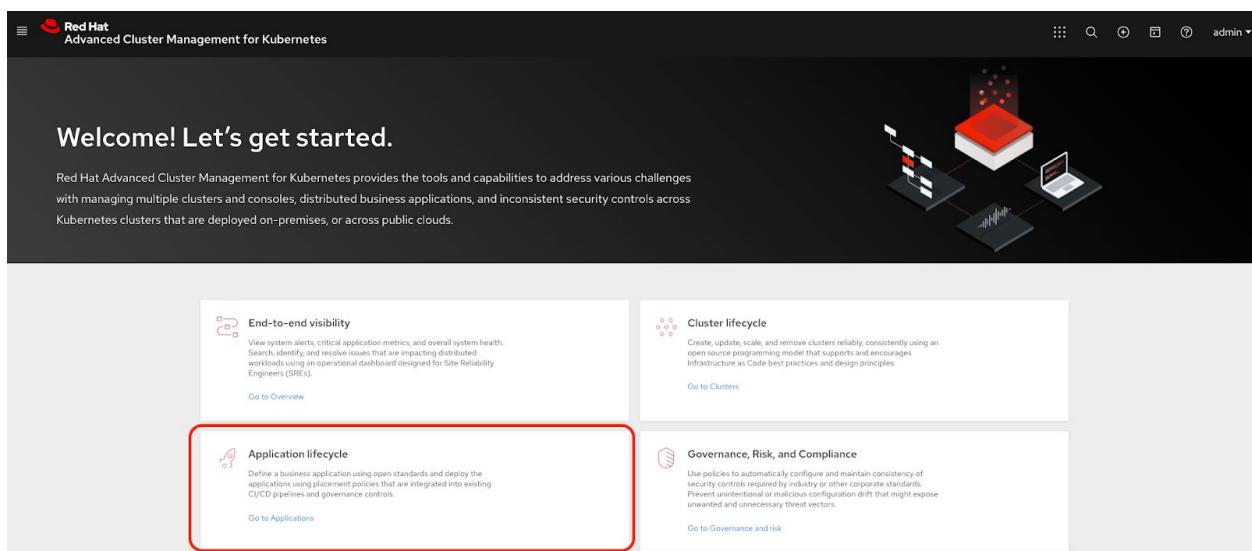
Pod Name	Namespace	Status	Replicas	Available	Ready	Image	Memory	Cores	Last Seen	Action
klusterlet-addon-appmgr-779dbc7bd6-5l9w8	NS	Running	2/2	0	RS	klusterlet-addon-appmgr-779dbc7bd6	60.0 MiB	0.003 cores	Feb 12, 11:43 am	⋮
klusterlet-addon-certpolicyctr-56b94d75bd-g54f2	NS	Running	2/2	0	RS	klusterlet-addon-certpolicyctr-56b94d75bd	42.6 MiB	0.003 cores	Feb 12, 11:43 am	⋮
klusterlet-addon-iampolicyctr-746ccdd46c-8lvhs	NS	Running	2/2	0	RS	klusterlet-addon-iampolicyctr-746ccdd46c	43.1 MiB	0.003 cores	Feb 12, 11:43 am	⋮
klusterlet-addon-operator-ffd67497-dwfmv	NS	Running	1/1	0	RS	klusterlet-addon-operator-ffd67497	53.2 MiB	0.042 cores	Feb 12, 11:42 am	⋮
klusterlet-addon-policyctrl-config-policy-6775b8dd7-jq7vn	NS	Running	1/1	0	RS	klusterlet-addon-policyctrl-config-policy-6775b8dd7	20.8 MiB	0.009 cores	Feb 12, 11:43 am	⋮
klusterlet-addon-policyctrl-framework-cc7db656c-sbp27	NS	Running	4/4	0	RS	klusterlet-addon-policyctrl-framework-cc7db656c	76.2 MiB	0.002 cores	Feb 12, 11:43 am	⋮
klusterlet-addon-search-	NS	Running	2/2	0	RS	klusterlet-addon-search-	50.9 MiB	0.020 cores	Feb 12, 11:43 am	⋮

Exercise 2: Application Lifecycle

In the previous exercise, you explored the Cluster Lifecycle functionality in RHACM. This allowed you to import an OpenShift® cluster to RHACM, which you can now use to deploy applications.

Application Lifecycle functionality in RHACM provides the processes that are used to manage application resources on your managed clusters.

Go back to the Home screen of ACM and click on the Application Lifecycle (Go to Applications) box.



Creating an Application

Within the Applications window click on **Create application**.

The screenshot shows the Red Hat Advanced Cluster Management for Kubernetes interface. At the top, there's a navigation bar with the Red Hat logo and the text "Advanced Cluster Management for Kubernetes". Below it, the main title "Applications" is displayed, with "Overview" and "Advanced configuration" as sub-options. In the center, there's a large blue sphere icon with the text "No applications found" below it. A message says, "You don't have any applications, yet. Click the Create application button to create the resource and view the documentation." In the top right corner of the main content area, there's a small box containing "Refresh every 15s", "Last update 13:35:07", and a prominent red-outlined "Create application" button.

Enter the following information:

Name: book-import

Namespace: book-import

Under repository types, select the GIT repository

URL: <https://github.com/jnpacker/book-import-fest2020.git>

Branch: hugo-nginx

The screenshot shows the "bookimport" application creation form. At the top, there's a header with the Red Hat logo and "Advanced Cluster Management for Kubernetes". Below it, the title "bookimport" is shown with a "YAML: On" switch. The "Editor" tab is selected. On the left, there are input fields for "Name*" (bookimport) and "Namespace*" (book-import). On the right, the "Application YAML" section displays the generated YAML code:

```

1  apiVersion: app.k8s.io/v1beta1
2  kind: Application
3  metadata:
4    namespace: book-import
5    name: bookimport
6    annotations: {}
7    selfLink: /apis/app.k8s.io/v1beta1/namespaces/book-import/applications/bookimport
8  spec:
9    componentKind:
10      - group: apps.open-cluster-management.io
11        kind: Subscription
12        descriptor: {}
13        selector:
14          matchExpressions:
15            - key: app
16              operator: In
17              values:
18                - bookimport
19
20  apiVersion: apps.open-cluster-management.io/v1
21  kind: Subscription
22  metadata:
23    namespace: book-import
24    name: bookimport-subscription-0
25    annotations:
26      apps.open-cluster-management.io/git-branch: hugo-nginx
27      apps.open-cluster-management.io/git-path: ''
28    labels:
29      app: bookimport
30      selfLink: /apis/apps.open-cluster-management.io/v1/namespaces/book-import/subscriptions/bookimport-subscription-0
31    spec:
32      channel: githubcom-jnpacker-book-import-fest2020-ns/githubcom-jnpacker-book-import-fest2020
33      placement:
34        placementRef:
35          name: bookimport-placement-0
36          kind: PlacementRule
37
38  apiVersion: apps.open-cluster-management.io/v1
39  kind: PlacementRule
40  metadata:
41    namespace: book-import
42    name: bookimport-placement-0
43    annotations: {}

```

Select Deploy application resources only on clusters matching specified labels

Label: environment

Value: dev

```
12   matchExpressions:
13     - key: app
14       operator: In
15       values:
16         - book-import
17
18   apiVersion: apps.open-cluster-management.io/v1
19   kind: Subscription
20   metadata:
21     annotations:
22       apps.open-cluster-management.io/git-branch: hugo-nginx
23       apps.open-cluster-management.io/git-path:
24       labels:
25         app: book-import
26       name: book-import-subscription-0
27       namespace: book-import
28   spec:
29     channel:
30       gitHubcom/jnpacker-book-import-fest2020-ms/githubcom/jnpacker-book
31     placement:
32       placementRef:
33         kind: PlacementRule
34         name: book-import-placement-0
35
36   apiVersion: apps.open-cluster-management.io/v1
37   kind: PlacementRule
38   metadata:
39     labels:
40       app: book-import
41       name: book-import-placement-0
42       namespace: book-import
43   spec:
44     clusterSelector:
45       matchLabels:
46         environment: 'dev'
```

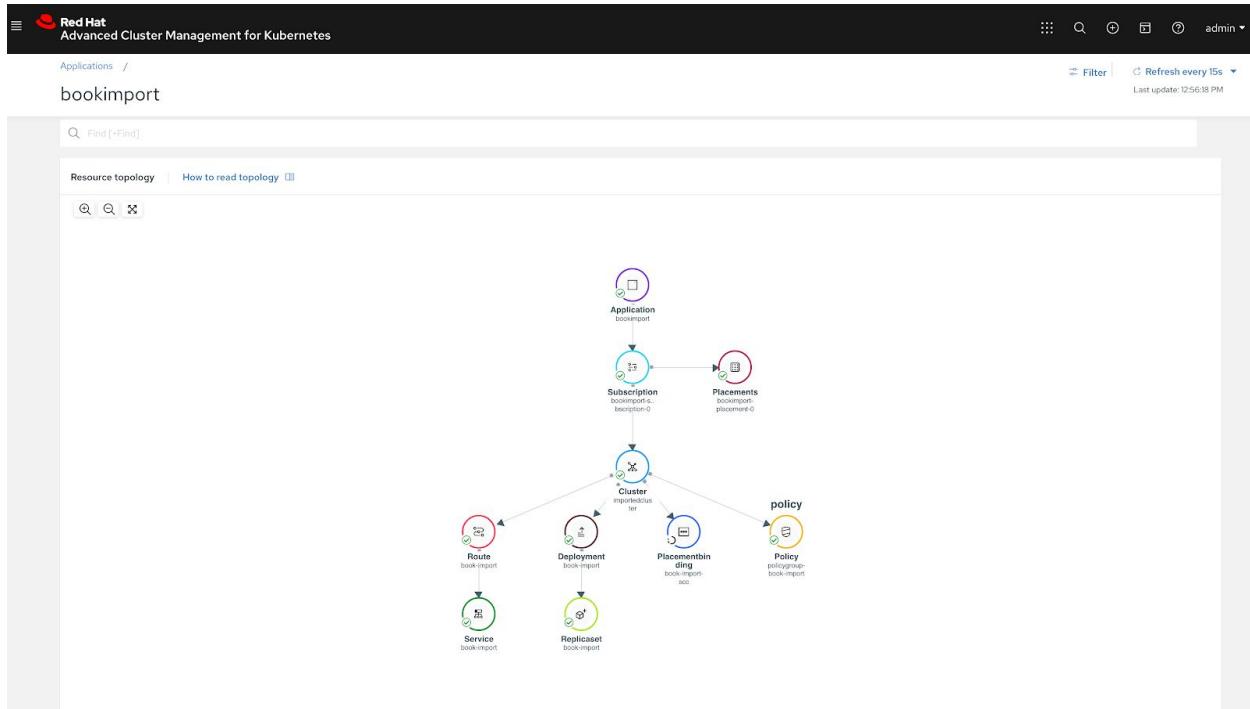
Click **Save**

After a few minutes you will see the application available in ACM Applications Window.

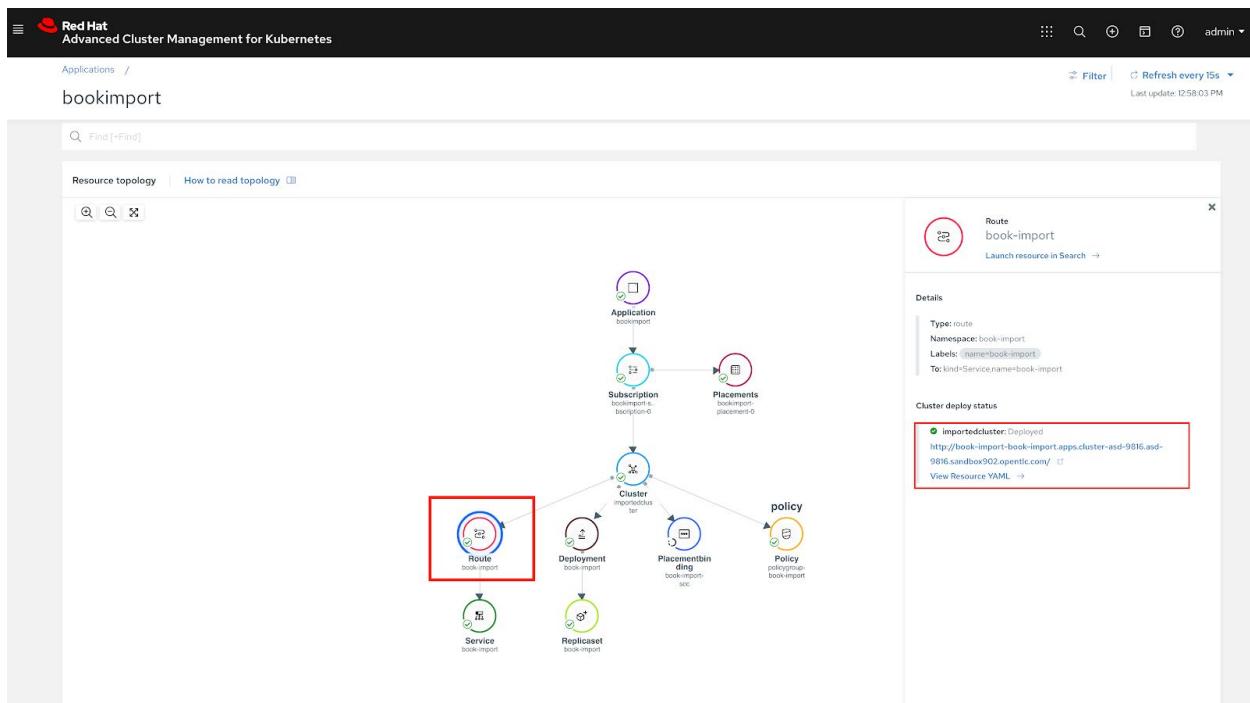
Name	Namespace	Clusters	Resource	Time window	Created
bookimport	book-import	1 Remote	Git	2 hours ago	2 hours ago

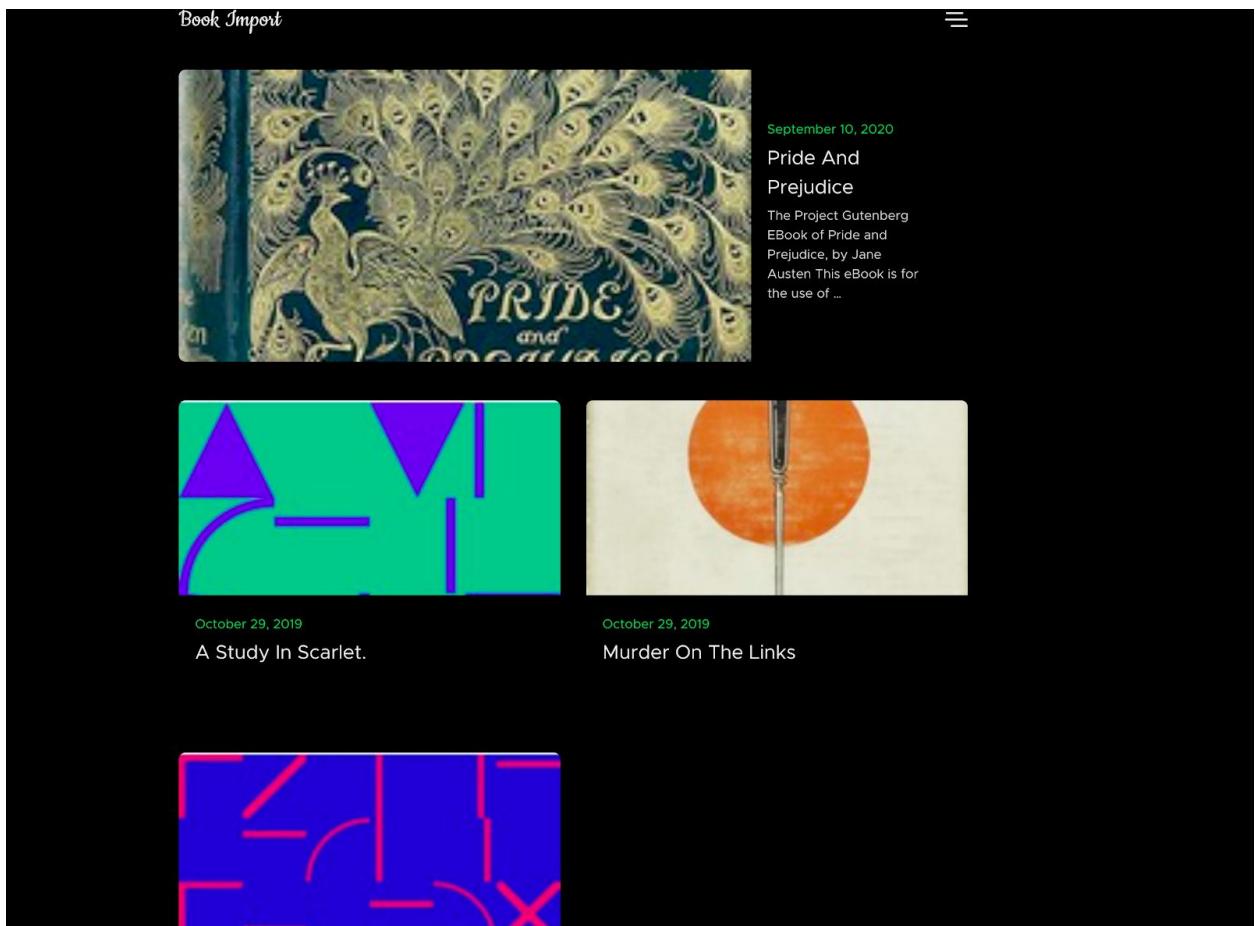
If everything was done correctly you should be able to see the application deployed to 1 Remote cluster.

Click the **book-import** application and have a look at the topology view



Select the Route and click on the URL provided, you should see the Book Import application





You have now completed the overview of the **Application Lifecycle functionality in RHACM**.

You successfully deployed an application to a target cluster using RHACM.

This approach leveraged a Git repository which housed all of the manifests that defined your application. RHACM was able to take those manifests and use them as deployables, which were then deployed to the target cluster.

You also leverage the power of labels and deploy the application to your imported cluster.

Extra activities:

Extra activity 1 - Compare the Git Repository and the ACM Display

Open a new tab in your browser and go to

<https://github.com/jnpacker/book-import-fest2020/tree/master/book-import>

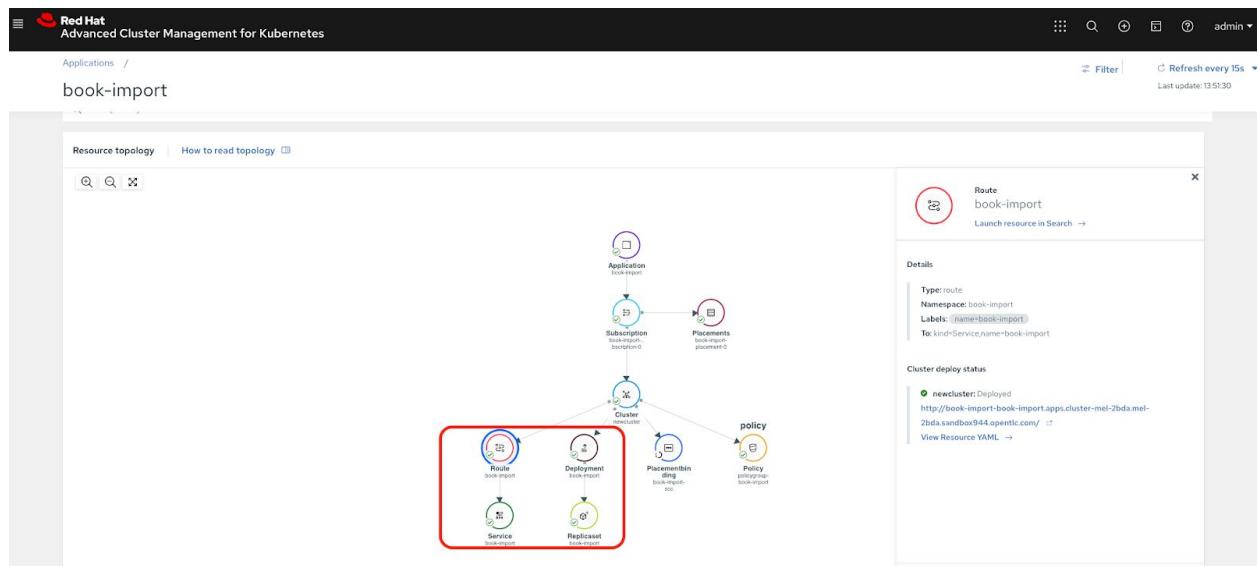
master ➔ book-import-fest2020 / book-import /

jnpacker Fix replica-set

..

deployment.yaml	Fix replica-set
route.yaml	Initial commit, book-import
service.yaml	Initial commit, book-import

You should be able to see that the Git Repo resources have been represented as part of the book-info deployed App (the Replicaset being part of the deployment.yaml).



Extra activity 2 - Compare the OpenShift newcluster and the ACM Display

Log into the OCP managed cluster (newcluster) using the details from the RHPDS email. Under the Home->Projects tab check the book-import project.

The screenshot shows the 'Projects' section of the OpenShift web interface. On the left is a sidebar with navigation links like Home, Projects, Search, Explore, Events, Operators, and Workloads. The main area displays a table of projects. One project, 'book-import', is highlighted with a red box around its name in the first column. The table columns include Name, Display Name, Status, Requester, Memory, CPU, and Created.

Name	Display Name	Status	Requester	Memory	CPU	Created
PR book-import	No display name	Active	No requester	26.2 MiB	-	Feb 12, 1:40 pm
PR default	No display name	Active	No requester	-	-	Feb 12, 7:46 am
PR kube-node-lease	No display name	Active	No requester	-	-	Feb 12, 7:46 am
PR kube-public	No display name	Active	No requester	-	-	Feb 12, 7:46 am

Check the 1 Route and see if it matches the ACM Route display in the Applications book-import window.

This screenshot shows the detailed view for the 'book-import' project. The left sidebar includes options for Home, Overview, Projects, Search, Explore, Events, Operators, Workloads (Pods, Deployments, ConfigMaps, StatefulSets, Secrets), CronJobs, Jobs, and DaemonSets. The main content area has tabs for Overview, Details, YAML, Workloads, and Role Bindings. Under the Overview tab, there are sections for Details, Status, Utilization (CPU, Memory, Filesystem, Network Transfer, Pod count), and Activity. The 'Routes' tab is selected, showing a single route entry for 'book-import' in the namespace 'book-import'. The location for this route is highlighted with a red box.

This screenshot shows the 'Routes' list for the 'book-import' project. The left sidebar is identical to the previous screenshot. The main area lists routes, with one route named 'book-import' highlighted by a red box. The route table includes columns for Name, Namespace, Status, Location, and Service.

Name	Namespace	Status	Location	Service
RT book-import	NS book-import	Accepted	http://book-import-book-import.apps.cluster-mel-2bda.mel-2bda.sandbox944.opentlc.com/	book-import

Extra activity 3 - Deploy the book-import application to the local-cluster

Go back to the Clusters view. By browsing over the Labels column, check that both the local-cluster and newcluster have the following label **vendor=OpenShift**

Name	Status	Distribution version	Labels	Nodes
local-cluster	Ready	OpenShift 4.6.12(Upgrade available)	cloud=Amazon, clusterID=D9fb8bbaf-cf98-4bf0-a21a-2c35d592de1, installer.name=multiclusterhub, installer.namespace=open-cluster-management, local-cluster, vendor=OpenShift	6
newcluster	Ready	OpenShift 4.5.16(Upgrade available)	cloud=Amazon, clusterID=D9fb8bbaf-cf98-4bf0-a21a-2c35d592de1, installer.name=multiclusterhub, installer.namespace=open-cluster-management, local-cluster, vendor=OpenShift	6

We will now go back to the Applications view. On the book-import line, click on the 3 grey dots and select Delete application.

Name	Namespace	Clusters	Resource	Time window	Created
book-import	book-import	1 Remote	Git		3 minutes ago

We will now Create a new application (reusing the same Git Repo) with a different label.

Click on the Create application button.

Use the following values for the different fields:

- Name: book-import2
- Namespace: book-import2
- Repository Type: Git
 - URL: <https://github.com/jnpacker/book-import-fest2020.git>
 - Branch: hugo-nginx
- Label: vendor
- Value: OpenShift (be careful this is case sensitive)

Then click Save.

```

12   matchExpressions:
13     - key: app
14       operator: In
15       values:
16         - book-import2
17
18   apiVersion: apps.open-cluster-management.io/v1
19   kind: Subscription
20   metadata:
21     annotations:
22       apps.open-cluster-management.io/git-branch: hugo-nginx
23       apps.open-cluster-management.io/git-path:
24       label:
25         app: book-import2
26       name: book-import2-subscription-0
27       namespace: book-import2
28   spec:
29     channel: githubcom-jnpacker-book-import-fest2020-ns/githubcom-jnpacker-book
30     placement:
31       placementRef:
32         kind: PlacementRule
33         name: book-import2-placement-0
34
35   apiVersion: apps.open-cluster-management.io/v1
36   kind: PlacementRule
37   metadata:
38     labels:
39       app: book-import2
40       namespace: book-import2-placement-0
41   spec:
42     clusterSelector:
43       matchLabels:
44         vendor: 'OpenShift'
45

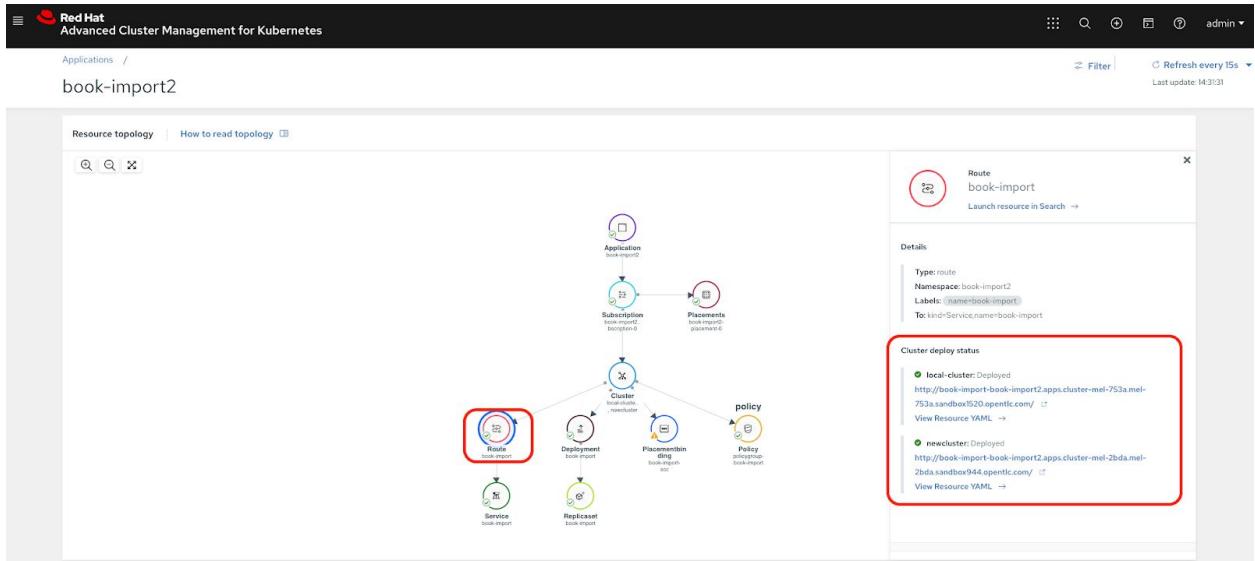
```

Going back to the Applications window, you will see after a bit of time that this new application (book-import2) is deployed on both the local-cluster and the newcluster.

Name	Namespace	Clusters	Resource	Time window	Created
book-import	book-import	1Remote	Git	9 minutes ago	...
book-import2	book-import2	1Remote, 1Local	Git	a few seconds ago	...

Click on book-import2, and you will see that there are now 2 clusters where this application is deployed.

Click the Route icon, and you will see that there 2 entries to which you can now navigate to access the Application.



Exercise 3: Creating Policies in ACM

At this point, you have completed the overview labs for Cluster Lifecycle and Application Lifecycle capabilities in RHACM.

Now that you have a cluster and a deployed application, you need to make sure that they do not drift from their original configurations. This kind of drift is a serious problem, because it can happen from benign and benevolent fixes and changes, as well as malicious activities that you might not notice but can cause significant problems.

The solution that RHACM provides for this is the Governance, Risk, and Compliance, or GRC, functionality.

Review GRC Functionality

The RHACM console provides an easy way to start creating basic policies, as shown below. In this example, you can see that when you change values for elements in the RHACM console, the YAML content is updated.

The screenshot shows the RHACM console's Governance and risk interface. At the top, there are summary statistics for NIST CSF and NIST standards. Below this, a detailed list of policies is provided, each with its name, namespace, remediation level, cluster violations, standards, controls, and categories. The policies listed include `policy-erc-hg`, `policy-namespace-i`, `policy-iamepolicy-j`, `policy-erc-cluster`, `policy-subscription-controlplane`, `policy-auth-provider`, `policy-certificatepolicy`, `policy-container-security`, `policy-iampolicy`, and `policy-namespace`. The categories for these policies include PR DS Data Security, DE CM Security Continuous Monitoring, PR AC 4 Access Control, PR IP 1 Baseline Configuration, PR AC 5 Network Integrity, PR DS 2 Data In Transit, PR PT Protective Technology, PR DS Data Security, PR IP Information Protection Processes And Procedures, PR DS Data Security, PR SP 800 190 3 11 Image Vulnerabilities, PR IP Information Protection Processes And Procedures, PR AC 4 Access Control, PR AC Identity Management Authentication And Access Control, and PR IP 1 Baseline Configuration.

This is a complex topic, and this course is only providing an overview. Please consult the [GRC product documentation](#) for more details on any of these policy controllers.

From the Home page, navigate to the **Governance, Risk and Compliance** screen by clicking on Go to Governance and risk.

Welcome! Let's get started.

Red Hat Advanced Cluster Management for Kubernetes provides the tools and capabilities to address various challenges with managing multiple clusters and consoles, distributed business applications, and inconsistent security controls across Kubernetes clusters that are deployed on-premises, or across public clouds.

End-to-end visibility
View system alerts, critical application metrics, and overall system health. Search, identify, and resolve issues that are impacting distributed workloads using an operational dashboard designed for Site Reliability Engineers (SREs).

[Go to Overview](#)

Cluster lifecycle
Create, update, scale, and remove clusters reliably, consistently using an open source programming model that supports and encourages Infrastructure as Code best practices and design principles.

[Go to Clusters](#)

Application lifecycle
Define a business application using open standards and deploy the application using placement policies that are integrated into existing CI/CD pipelines and governance controls.

[Go to Applications](#)

Governance, Risk, and Compliance
Use policies to automatically configure and maintain consistency of security protocols required by industry or other corporate standards. Prevent unintentional or malicious configuration drift that might expose unwanted and unnecessary threat vectors.

[Go to Governance and risk](#)

No policies found

You don't have any policies, yet. Use the [create policy](#) button to create the resource or view the documentation to find samples for creating new policies.

[Create policy](#)

[View documentation](#)

Build a policy by clicking on Create policy and provide the following information:

- **Name:** policy-grc-namespace
- **Namespace:** default
- **Specifications:** CertificatePolicy -cert management expiration

Leave everything else as default and click **Create**. Please note that initially it will complain that there is an issue with the policy but shortly after should go green and get a checkmark

```

apiVersion: policy.open-cluster-management.io/v1
kind: Policy
metadata:
  name: policy-grc-namespace
  namespace: default
  annotations:
    policy.open-cluster-management.io/standards: NIST-CSF
    policy.open-cluster-management.io/categories: PR-DS Data Security
    policy.open-cluster-management.io/controls: PR-05-2 Data-in-transit
spec:
  remediationAction: inform
  disabled: false
  policyTemplates:
  - objectDefinitions:
    - type: policy.open-cluster-management.io/v1
      kind: CertificatePolicy # cert management expiration
      metadata:
        name: policy-grc-namespace-example
      spec:
        namespaceSelectors:
          include: ["default"]
          exclude: ["kube-*"]
        remediationAction: inform # will be overridden by remediationAction in parent policy
        severity: low
        minimumDuration: 300h
  apiversion: policy.open-cluster-management.io/v1
  kind: PlacementBinding
  metadata:
    name: binding-policy-grc-namespace
    namespace: default
    placementRefs:
      name: placement-policy-grc-namespace

```

Policy name	Namespace	Remediation	Cluster violations	Standards	Categories	Controls
policy-grc-namespace	default	inform	0/2	NIST CSF	PR DS Data Security	PR DS 2 Data In Transit

You can also play around by creating more policies if needed, we have some examples in the [GitPolicy repo](#) .

There is also a smaller subject of these policies I normally use for demos all located [here](#)

