IBM Cloud Foundry Migration Runtime Install

Setup and installation of IBM Cloud Foundry Migration Runtime is done with a Helm-based Operator. Total installation time takes about 30 to 40 minutes.

> ! **Important** Before installing IBM Cloud Foundry Migration Runtime, review the Prerequisites guide.

What is installed?

IBM Cloud Foundry Migration Runtime uses the following namespaces (OpenShift projects). By default, these namespaces start with the prefix `cfmr`.

Namespace	Purpose
cfmr	Contains Cloud Foundry system components
cfmr-cf-operator	Operator for managing the system deployed to cfmr
cfmr-eirini	Contains user applications deployed by the system installed in cfmr
cfmr-operator	Contains the operator for collectively deploying and managing the install of components for CFMR, such as Ingress, UI, CF-Operator, and Eirini, while adhering to IBM best practices for product installation (for example, use of UBI images, OLM lifecycle)
cfmr-ui	Cloud Foundry Migration Runtime UI components

Running an install

Create a directory to save cases to a local directory and export `CFMR_VERSION`

```
$ mkdir /tmp/cases
$ export CFMR_VERSION=<cfmr version, e.g. 1.0.0>
```

Download case bundle

```
$ cloudctl case save                       \
    --case ibm-cfmr-case                  \
    --version "${CFMR_VERSION}"             \
    --repo https://raw.githubusercontent.com/IBM/cloud-pak/master/repo/case \
    --outputdir /tmp/cases
```

Verify the case, dependency cases and images csv has been downloaded under the `/tmp/cases` directory.

Unpack case bundle

Unpack case bundle to access files

```
$ tar -xvzf /tmp/cases/ibm-cfmr-"${CFMR_VERSION}".tgz
$ cd /tmp/cases/ibm-cfmr/
```

Accept license agreement

Prior to installation, you must review and accept the license: http://ibm.biz/cfmr-license

Once accepted, set the license flag `license.accept` to `true` in the custom resource file `cfmr.ibm.com_<version>_ibmcfmrprod_cr.yaml`, located at `./ibm-cfmr/inventory/cfmrOperator/files`:

```
spec:
  license:
    accept: true
```

Point to entitled registry

Update the custom resource file `cfmr.ibm.com_<version>_ibmcfmrprod_cr.yaml` to use the name of the IBM Entitled Container Fulfillment Registry secret that was created as part of the

```
spec:
  global:
    image:
      # Needs to be updated
      pullSecret: "<pull_secret_name>"
```

Set default certificates for routes creation

Use the cluster's default certificates as the certificates for CFMR's routes. This can be found in namespace `openshift-ingress`.

```
$ oc get secrets -n openshift-ingress
NAME              TYPE                 DATA  AGE
...
router-certs-default      kubernetes.io/tls           2   2d21h
...
```

Update custom resource file `cfmr.ibm.com_<version>_ibmcfmrprod_cr.yaml` to use the name of the default certificate.

```
spec:
  features:
    customCertNamespace: "openshift-ingress"
    customCertSecret: "router-certs-default"
```

If there are custom certificates generated and point to that certificate's name and location.

Set SCC for operator

This chart requires adding `restricted`, `cluster-admin`, `self-provisioner` policy to service account `*-ibm-cfmr-serviceaccount` in the namespace that IBM Cloud Foundry Migration Runtime installs from. This service account is created for you, so you can ignore any `Warning: Service account not found` messages. Note that `CUSTOM_RESOURCE_NAME` is the name `metadata.name` of your custom resource file `cfmr.ibm.com_<version>_ibmcfmrprod_cr.yaml`.

```
$ export CUSTOM_RESOURCE_NAME=<custom_resource_name>
$ oc adm policy add-scc-to-user restricted system:serviceaccount:cfmr-operator:"$
$ oc adm policy add-cluster-role-to-user cluster-admin system:serviceaccount:cfmr
$ oc adm policy add-cluster-role-to-user self-provisioner system:serviceaccount:c
```

Deploy an operator and custom resource:

```
# Update and deploy the Operator Custom Resource Definition and resources.
$ export PULL_SECRET=<pull_secret_name>   # entitled registry pull secret
$ oc apply -f inventory/cfmrOperatorSetup/files/op-cli/cfmr.ibm.com_ibmcfmrprods_
$ sed -e 's|REPLACE_SECRET|${PULL_SECRET}|g' inventory/cfmrOperatorSetup/files/op
$ oc apply -f inventory/cfmrOperatorSetup/files/op-cli/role.yaml
$ oc apply -f inventory/cfmrOperatorSetup/files/op-cli/role_binding.yaml
$ oc apply -f inventory/cfmrOperatorSetup/files/op-cli/operator.yaml

# Set values in, then create the CFMR Custom Resource
$ oc apply -f inventory/cfmrOperator/files/cfmr.ibm.com_<version>_ibmcfmrprod_cr.
```

Wait until the deployment process is complete. Upon completion, a message similar to `Welcome to your new deployment of KubeCF` is displayed, along with details of the deployment.

Verify Install

Check to see if you can access the CFMR UI `https://cfmr-ui.<my_domain>`

```
# Fetch CFMR UI url.
$ oc get routes -n cfmr-ui --no-headers | awk '{print $2}'
```

Check to see if you can access the CFMR API endpoint

```
$ oc get routes -n cfmr --no-headers | awk '{print $2}'
api.<my_domain>
$ cf api api.<my_domain>
```

Or if you've provided a custom domain `features.customDomain`, check to see if you can access the UI and API using that.

```
$ curl https://cfmr-ui.<custom_domain>
$ cf api api.<custom_domain>
```

Installing in an air-gapped cluster

This operator can be installed in an on-line or air-gapped cluster through either of the following install paths :

The following are required to air-gap installation.

– `helm` Helm (v3) CLI to assist with air-gapping installations

– `helm push` Helm push plugin to assist with air-gapping installations

Configure Air-Gapped OpenShift Cluster With a Bastion

Prepare Bastion Host

– Logon to the bastion machine

– Verify that the bastion machine has access

- to public internet (to download CASE and images)
- a target image registry (where the images will be mirrored)
- a target OpenShift cluster to install the operator

All the following steps should be run from the bastion machine

Set environment variables

Export the TARGET_REGISTRY, TARGET_REGISTRY_USER and TARGET_REGISTRY_SECRET environment variable with the location of the private registry and it's username/password.

```
$ export TARGET_REGISTRY_USER=<registry user>
$ export TARGET_REGISTRY_SECRET=<registry secret>
$ export TARGET_REGISTRY=<my.private-registry.org>
```

(Optional) The OpenShift image registry isn't recommended due to limitations such as lack of support for fat manifest. Quay.io enterprise is an opensource alternative. To use the image registry anyways:

1. Expose the OpenShift image registry externally

```
$ oc patch configs.imageregistry.operator.openshift.io/cluster --patch '{"spec":
```

2. Set the environment variable of the target registry.

```
$ export TARGET_REGISTRY=$(oc get route default-route -n openshift-image-regist
```

done so previously

```
$ export TARGET_NAMESPACE=cfmr-operator
$ oc new-project "${TARGET_NAMESPACE}"
```

Create auth secret for the source image registry

```
$ cloudctl case launch                                     \
    --case /tmp/cases/ibm-cfmr-"${CFMR_VERSION}".tgz             \
    --namespace "${TARGET_NAMESPACE}"                      \
    --inventory cfmrOperatorSetup                        \
    --action configure-creds-airgap                      \
    --args "--registry cp.icr.io --user iamapikey --pass <entitlement_key>"
```

Create auth secret for target image registry

```
$ cloudctl case launch                                     \
    --case /tmp/cases/ibm-cfmr-"${CFMR_VERSION}".tgz             \
    --namespace "${TARGET_NAMESPACE}"                       \
    --inventory cfmrOperatorSetup                        \
    --action configure-creds-airgap                      \
    --args "--registry "${TARGET_REGISTRY}" --user "${TARGET_REGISTRY_USER}" --pa
```

The credentials are now saved to `~/.airgap/secrets/<registry-name>.json`

Set the path of the target registry

If using OpenShift image registry, set the project to load the images to:

```
$ export TARGET_REGISTRY="${TARGET_REGISTRY}"/cfmr
```

Mirror Images

In this step, images from saved CASE (ibm-cfmr-"${CFMR_VERSION}"-images.csv) are copied to target registry in the air-gapped environment.

```
$ cloudctl case launch                                   \
    --case /tmp/cases/ibm-cfmr-"${CFMR_VERSION}".tgz           \
    --namespace "${TARGET_NAMESPACE}"                     \
    --inventory cfmrOperatorSetup                      \
    --action mirror-images                             \
    --args "--registry $TARGET_REGISTRY --inputDir /tmp/cases"
```

Configure Cluster for Air-gapping

– creates a global image pull secret for the target registry (skipped if target registry is unauthenticated)

– creates a imagesourcecontentpolicy

WARNING:

– Cluster resources must adjust to the new pull secret, which can temporarily limit the usability of the cluster. Authorization credentials are stored in $HOME/.airgap/secrets and /tmp/airgap* to support this action

– Applying imagesourcecontentpolicy causes cluster nodes to recycle.

```
$ cloudctl case launch                                   \
    --case /tmp/cases/ibm-cfmr-"${CFMR_VERSION}".tgz           \
    --namespace "${TARGET_NAMESPACE}"                     \
    --inventory cfmrOperatorSetup                      \
    --action configure-cluster-airgap                  \
    --args "--registry "${TARGET_REGISTRY}" --inputDir /tmp/cases"
```

(Optional) If you are using an insecure registry, you must add the local registry to the cluster insecureRegistries list.

```
$ oc patch image.config.openshift.io/cluster --type=merge -p '{"spec":{"registryS
```

Configure Helm Repository

Prepare a private helm chart repository on the OpenShift cluster that can be used during installation.

Locate chartmuseum helm chart in `/tmp/cases/charts` folder. Should be named `chartmuseum-3.1.0.tgz`.

Initialize helm chart repository on the cluster

```
$ cloudctl case launch                                   \
    --case /tmp/cases/ibm-cfmr-"${CFMR_VERSION}".tgz     \
    --namespace "${TARGET_NAMESPACE}"                    \
    --inventory cfmrOperatorSetup                        \
    --action init-helm-repository                        \
    --args "-chartmuseum chartmuseum-3.1.0.tgz"
```

After helm repo is initialized, helm repository URL and username/password are created

```
[INFO] username = admin
[INFO] password = feb92d0ebc038522f407c4642a4acf14
```

Load Helm Repository

Loads helm charts for `quarks`, `kubecf`, and `console` in defaults charts `/tmp/cases/charts` into helm repository.

Export helm repo URL and credentials.

```
$ export HELM_REPO_URL=<private-helm-repo URL e.g. http://private-helm-repo-chart
$ export HELM_REPO_USERNAME=<e.g. admin>
$ export HELM_REPO_PASSWORD=<e.g. feb92d0ebc038522f407c4642a4acf14>
```

Load helm charts into helm repository

```
$ cloudctl case launch                                     \
    --case /tmp/cases/ibm-cfmr-"${CFMR_VERSION}".tgz     \
    --namespace "${TARGET_NAMESPACE}"                      \
    --inventory cfmrOperatorSetup                          \
    --action load-helm-repository                          \
    --args "-u "${HELM_REPO_USERNAME}" -p "${HELM_REPO_PASSWORD}" --url "${HELM_R
```

Once complete, this should list loaded charts. This will be used in the custom resource during installation.

Update your Custom Resource file `cfmr.ibm.com_<version>_ibmcfmrprod_cr.yaml` to use the helm repository.

```
spec:
  features:
    chartRepository: "http://private-helm-repo-chartmuseum-private-helm-repo.mycl
    chartRepositoryName: "private-helm-repo"
```

In Air-Gapped OpenShift Cluster Without a Bastion

Prepare a portable device

Prepare a portable device (such as laptop) that can be used to download the case and images and can be carried into the air-gapped environment

- Verify that the portable device has access
  - to public internet (to download CASE and images)
  - a target image registry (where the images will be mirrored)

All the following steps should be run from the portable device

Configure Registry Auth

See instructions from previous Configure Registry Auth section

Set environment variables

See instructions from previous Set environment variables section

Mirror Images

See instructions from previous Mirror Images section

Configure Cluster for Air-gapping

See instructions from previous Configure Cluster for Air-gapping section

Configure Helm Repository

See instructions from previous Configure Helm Repository section

Load Helm Repository

See instructions from previous Load Helm Repository section

Tell us what you think

Was this topic helpful?

Yes No

English

Contact IBM

Privacy

Accessibility