<https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#installing-the-ric>

# Installation Guides

This document describes how to install the RIC components deployed by scripts and Helm charts under the ric-plt/dep repository, including the dependencies and required system resources.

* [Version history](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#version-history)
* [Overview](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#overview)
* [Prerequisites](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#prerequisites)
* [Installing Near Realtime RIC in RIC Cluster](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#installing-near-realtime-ric-in-ric-cluster)
  + [Getting and Preparing Deployment Scripts](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#getting-and-preparing-deployment-scripts)
  + [Deploying the Infrastructure and Platform Groups](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#deploying-the-infrastructure-and-platform-groups)
  + [Modify the deployment recipe](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#modify-the-deployment-recipe)
  + [Installing the RIC](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#installing-the-ric)
  + [Checking the Deployment Status](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#checking-the-deployment-status)
  + [Checking Container Health](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#checking-container-health)
  + [Undeploying the Infrastructure and Platform Groups](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#undeploying-the-infrastructure-and-platform-groups)
  + [Restarting the VM](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#restarting-the-vm)
  + [RIC Applications](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#ric-applications)
  + [OPTIONALLY use Redis Cluster (instead of Redis standalone)](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#optionally-use-redis-cluster-instead-of-redis-standalone)
    - [Important](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#important)
    - [Overview](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id1)
    - [Environment Set-Up](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#environment-set-up)
    - [Execution](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#execution)

## [Version history](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id2)

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Ver.** | **Author** | **Comment** |
| 2020-02-29 | 0.1.0 | Abdulwahid W |  |

## [Overview](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id3)

This section explains the installation of Near Realtime RAN Intelligent Controller Platform only.

## [Prerequisites](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id4)

The steps below assume a clean installation of Ubuntu 20.04 (no k8s, no docker, no helm)

## [Installing Near Realtime RIC in RIC Cluster](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id5)

After the Kubernetes cluster is installed, the next step is to install the (Near Realtime) RIC Platform.

### [Getting and Preparing Deployment Scripts](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id6)

Clone the ric-plt/dep git repository that has deployment scripts and support files on the target VM.

% git clone "https://gerrit.o-ran-sc.org/r/ric-plt/ric-dep"

git clone "https://gerrit.o-ran-sc.org/r/ric-plt/ric-dep"

### [Deploying the Infrastructure and Platform Groups](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id7)

Use the scripts below to install kubernetes, kubernetes-CNI, helm and docker on a fresh Ubuntu 20.04 installation. Note that since May 2022 there’s no need for anything form the repo it/dep anymore.

*# install kubernetes, kubernetes-CNI, helm and docker*

cd ric-dep/bin

./install\_k8s\_and\_helm.sh

*# install chartmuseum into helm and add ric-common templates*

./install\_common\_templates\_to\_helm.sh

After the recipes are edited and helm started, the Near Realtime RIC platform is ready to be deployed, but first update the deployment recipe as per instructions in the next section.

### [Deploying the Infrastructure and Platform Groups](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id7)

Use the scripts below to install kubernetes, kubernetes-CNI, helm and docker on a fresh Ubuntu 20.04 installation. Note that since May 2022 there’s no need for anything form the repo it/dep anymore.

near@near-VirtualBox:~/ric-dep/bin$ ll

near@near-VirtualBox:~/ric-dep$ ll

near@near-VirtualBox:~/ric-dep$ find /home/near/ -type f -name \*RECIPE\*

near@near-VirtualBox:~/ric-dep$ ll

near@near-VirtualBox:~/ric-dep$ cd RECIPE\_EXAMPLE/

near@near-VirtualBox:~/ric-dep/RECIPE\_EXAMPLE$ ll

near@near-VirtualBox:~/ric-dep/RECIPE\_EXAMPLE$ cat example\_recipe\_latest\_stable.yaml

near@near-VirtualBox:~/ric-dep/RECIPE\_EXAMPLE$ cat example\_recipe\_oran\_f\_release.yaml

near@near-VirtualBox:~/ric-dep/bin$ ll

near@near-VirtualBox:~/ric-dep/bin$ cat install\_common\_templates\_to\_helm.sh

near@near-VirtualBox:~/ric-dep/bin$ ./install\_common\_templates\_to\_helm.sh

near@near-VirtualBox:~/ric-dep/bin$ ./install\_k8s\_and\_helm.sh

near@near-VirtualBox:~/ric-dep/bin$ sudo ./install\_k8s\_and\_helm.sh

see curl is not installed

near@near-VirtualBox:~/ric-dep/bin$ sudo apt install curl

near@near-VirtualBox:~/ric-dep/bin$ sudo ./install\_k8s\_and\_helm.sh

near@near-VirtualBox:~/ric-dep/bin$ kubectl get pods --all-namespaces

near@near-VirtualBox:~/ric-dep/bin$ sudo kubectl get pods --all-namespaces

near@near-VirtualBox:~/ric-dep/bin$ sudo ./install\_common\_templates\_to\_helm.sh

near@near-VirtualBox:~/ric-dep/bin$ cat install\_common\_templates\_to\_helm.sh

near@near-VirtualBox:~/ric-dep$ ls

near@near-VirtualBox:~/ric-dep$ cd RECIPE\_EXAMPLE/

near@near-VirtualBox:~/ric-dep/RECIPE\_EXAMPLE$ ls -al

near@near-VirtualBox:~/ric-dep/bin$ sudo ./install -f ../RECIPE\_EXAMPLE/PLATFORM/example\_recipe\_oran\_f\_release.yaml (Error because of PLATFORM path)

near@near-VirtualBox:~/ric-dep/bin$ cat ../RECIPE\_EXAMPLE/example\_recipe\_oran\_f\_release.yaml

Below the correct command

near@near-VirtualBox:~/ric-dep/bin$ sudo ./install -f ../RECIPE\_EXAMPLE/example\_recipe\_oran\_f\_release.yaml

near@near-VirtualBox:~/ric-dep/bin$ sudo helm list

near@near-VirtualBox:~/ric-dep/bin$ sudo kubectl get pods -n ricplt

***Following the demo video:***

1. create a txt file> vi <filename>.txt

ex2@ex2:~/ric-dep/bin$ vi versions.txt ( for quite and save> ::wq then enter, only quite no save> ::q! then enter)

Then write the code:

Cat versions.txt

nexus3.o-ran-sc.org:10002/o-ran-sc/ric-plt-a1:2.5.2

nexus3.o-ran-sc.org:10002/o-ran-sc/ric-plt-appmgr:0.5.6

nexus3.o-ran-sc.org:10002/o-ran-sc/ric-plt-dbaas:0.6.1

nexus3.o-ran-sc.org:10002/o-ran-sc/ric-plt-e2mgr:6.0.0

nexus3.o-ran-sc.org:10002/o-ran-sc/ric-plt-e2:6.0.0

nexus3.o-ran-sc.org:10002/o-ran-sc/ric-plt-rtmgr:0.9.1

nexus3.o-ran-sc.org:10002/o-ran-sc/ric-plt-submgr:0.9.3

nexus3.o-ran-sc.org:10002/o-ran-sc/ric-plt-vespamgr:0.7.5

nexus3.o-ran-sc.org:10002/o-ran-sc/ric-plt-o1:0.6.0

nexus3.o-ran-sc.org:10002/o-ran-sc/ric-plt-alarmmanager:0.5.13

nexus3.o-ran-sc.org:10002/o-ran-sc/it-dep-init:0.0.1

docker.io/prom/prometheus:v2.18.1

docker.io/kong/kubernetes-ingress-controller:0.7.0

docker.io/kong:1.4

docker.io/prom/alertmanager:v0.20.0

for i in `cat versions.txt`; do echo $i; docker pull $i; done

ex2@ex2:~/ric-dep$ for i in `cat versions.txt`; do echo $i; docker pull $i; done

ex2@ex2:~/ric-dep$ ip a

now we will see IP table and copy ip address.

ex2@ex2:~/ric-dep/RECIPE\_EXAMPLE$ vim ../RECIPE\_EXAMPLE/example\_recipe\_oran\_f\_release.yaml

ex2@ex2:~/ric-dep/bin$ sudo ./install -f ../RECIPE\_EXAMPLE/example\_recipe\_oran\_f\_release.yaml

ex2@ex2:~/ric-dep/bin$ sudo kubectl get pods -n ricplt

**To check IP table>>** ex2@ex2:~/ric-dep/bin$ ss -ltpn

### [Modify the deployment recipe](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id8)

Edit the recipe files ./RECIPE\_EXAMPLE/example\_recipe\_latest\_stable.yaml (which is a softlink that points to the latest release version). “example\_recipe\_latest\_unstable.yaml points to the latest example file that is under current development.

extsvcplt:

ricip: ""

auxip: ""

* Deployment scripts support both helm v2 and v3. The deployment script will determine the helm version installed in cluster during the deployment.
* To specify which version of the RIC platform components will be deployed, update the RIC platform component container tags in their corresponding section.
* You can specify which docker registry will be used for each component. If the docker registry requires login credential, you can add the credential in the following section. Please note that the installation suite has already included credentials for O-RAN Linux Foundation docker registries. Please do not create duplicate entries.

docker-credential:

enabled: true

credential:

SOME\_KEY\_NAME:

registry: ""

credential:

user: ""

password: ""

email: ""

For more advanced recipe configuration options, please refer to the recipe configuration guideline.

### [Installing the RIC](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id9)

After updating the recipe you can deploy the RIC with the command below. Note that generally use the latest recipe marked stable or one from a specific release.

cd ric-dep/bin

./install -f ../RECIPE\_EXAMPLE/PLATFORM/example\_recipe\_latest\_stable.yaml

### [Checking the Deployment Status](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id10)

Now check the deployment status after a short wait. Results similar to the output shown below indicate a complete and successful deployment. Check the STATUS column from both kubectl outputs to ensure that all are either “Completed” or “Running”, and that none are “Error” or “ImagePullBackOff”.

*# helm list -A*

NAME REVISION UPDATED STATUS CHART APP VERSION NAMESPACE

r3-a1mediator 1 Thu Jan 23 14:29:12 2020 DEPLOYED a1mediator-3.0.0 1.0 ricplt

r3-appmgr 1 Thu Jan 23 14:28:14 2020 DEPLOYED appmgr-3.0.0 1.0 ricplt

r3-dbaas1 1 Thu Jan 23 14:28:40 2020 DEPLOYED dbaas1-3.0.0 1.0 ricplt

r3-e2mgr 1 Thu Jan 23 14:28:52 2020 DEPLOYED e2mgr-3.0.0 1.0 ricplt

r3-e2term 1 Thu Jan 23 14:29:04 2020 DEPLOYED e2term-3.0.0 1.0 ricplt

r3-infrastructure 1 Thu Jan 23 14:28:02 2020 DEPLOYED infrastructure-3.0.0 1.0 ricplt

r3-jaegeradapter 1 Thu Jan 23 14:29:47 2020 DEPLOYED jaegeradapter-3.0.0 1.0 ricplt

r3-rsm 1 Thu Jan 23 14:29:39 2020 DEPLOYED rsm-3.0.0 1.0 ricplt

r3-rtmgr 1 Thu Jan 23 14:28:27 2020 DEPLOYED rtmgr-3.0.0 1.0 ricplt

r3-submgr 1 Thu Jan 23 14:29:23 2020 DEPLOYED submgr-3.0.0 1.0 ricplt

r3-vespamgr 1 Thu Jan 23 14:29:31 2020 DEPLOYED vespamgr-3.0.0 1.0 ricplt

*# kubectl get pods -n ricplt*

NAME READY STATUS RESTARTS AGE

deployment-ricplt-a1mediator-69f6d68fb4-7trcl 1/1 Running 0 159m

deployment-ricplt-appmgr-845d85c989-qxd98 2/2 Running 0 160m

deployment-ricplt-dbaas-7c44fb4697-flplq 1/1 Running 0 159m

deployment-ricplt-e2mgr-569fb7588b-wrxrd 1/1 Running 0 159m

deployment-ricplt-e2term-alpha-db949d978-rnd2r 1/1 Running 0 159m

deployment-ricplt-jaegeradapter-585b4f8d69-tmx7c 1/1 Running 0 158m

deployment-ricplt-rsm-755f7c5c85-j7fgf 1/1 Running 0 158m

deployment-ricplt-rtmgr-c7cdb5b58-2tk4z 1/1 Running 0 160m

deployment-ricplt-submgr-5b4864dcd7-zwknw 1/1 Running 0 159m

deployment-ricplt-vespamgr-864f95c9c9-5wth4 1/1 Running 0 158m

r3-infrastructure-kong-68f5fd46dd-lpwvd 2/2 Running 3 160m

*# kubectl get pods -n ricinfra*

NAME READY STATUS RESTARTS AGE

deployment-tiller-ricxapp-d4f98ff65-9q6nb 1/1 Running 0 163m

tiller-secret-generator-plpbf 0/1 Completed 0 163m

### [Checking Container Health](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id11)

Check the health of the application manager platform component by querying it via the ingress controller using the following command.

% curl -v http://localhost:32080/appmgr/ric/v1/health/ready

The output should look as follows.

\* Trying 10.0.2.100...

\* TCP\_NODELAY set

\* Connected to 10.0.2.100 (10.0.2.100) port 32080 (*#0)*

> GET /appmgr/ric/v1/health/ready HTTP/1.1

> Host: 10.0.2.100:32080

> User-Agent: curl/7.58.0

> Accept: \*/\*

>

< HTTP/1.1 200 OK

< Content-Type: application/json

< Content-Length: 0

< Connection: keep-alive

< Date: Wed, 22 Jan 2020 20:55:39 GMT

< X-Kong-Upstream-Latency: 0

< X-Kong-Proxy-Latency: 2

< Via: kong/1.3.1

<

\* Connection *#0 to host 10.0.2.100 left intact*

### [Undeploying the Infrastructure and Platform Groups](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id12)

To undeploy all the containers, perform the following steps in a root shell within the it-dep repository.

*# cd bin*

*# ./uninstall*

Results similar to below indicate a complete and successful cleanup.

*# ./undeploy-ric-platform*

Undeploying RIC platform components [appmgr rtmgr dbaas1 e2mgr e2term a1mediator submgr vespamgr rsm jaegeradapter infrastructure]

release "r3-appmgr" deleted

release "r3-rtmgr" deleted

release "r3-dbaas1" deleted

release "r3-e2mgr" deleted

release "r3-e2term" deleted

release "r3-a1mediator" deleted

release "r3-submgr" deleted

release "r3-vespamgr" deleted

release "r3-rsm" deleted

release "r3-jaegeradapter" deleted

release "r3-infrastructure" deleted

configmap "ricplt-recipe" deleted

namespace "ricxapp" deleted

namespace "ricinfra" deleted

namespace "ricplt" deleted

### [Restarting the VM](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id13)

After a reboot of the VM, and a suitable delay for initialization, all the containers should be running again as shown above.

### [RIC Applications](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id14)

**xApp Onboarding using CLI tool called dms\_cli**

xApp onboarder provides a cli tool called dms\_cli to fecilitate xApp onboarding service to operators. It consumes the xApp descriptor and optionally additional schema file, and produces xApp helm charts.

Below are the sequence of steps to onboard, install and uninstall the xApp.

Step 1: (OPTIONAL ) Install python3 and its dependent libraries, if not installed.

Step 2: Prepare the xApp descriptor and an optional schema file. xApp descriptor file is a config file that defines the behavior of the xApp. An optional schema file is a JSON schema file that validates the self-defined parameters.

Step 3: Before any xApp can be deployed, its Helm chart must be loaded into this private Helm repository.

*#Create a local helm repository with a port other than 8080 on host*

docker run --rm -u 0 -it -d -p 8090:8080 -e DEBUG=1 -e STORAGE=local -e STORAGE\_LOCAL\_ROOTDIR=/charts -v **$(**pwd**)**/charts:/charts chartmuseum/chartmuseum:latest

Step 4: Set up the environment variables for CLI connection using the same port as used above.

*#Set CHART\_REPO\_URL env variable*

export CHART\_REPO\_URL=http://0.0.0.0:8090

Step 5: Install dms\_cli tool

*#Git clone appmgr*

git clone "https://gerrit.o-ran-sc.org/r/ric-plt/appmgr"

*#Change dir to xapp\_onboarder*

cd appmgr/xapp\_orchestrater/dev/xapp\_onboarder

*#If pip3 is not installed, install using the following command*

yum install python3-pip

*#In case dms\_cli binary is already installed, it can be uninstalled using following command*

pip3 uninstall xapp\_onboarder

*#Install xapp\_onboarder using following command*

pip3 install ./

Step 6: (OPTIONAL ) If the host user is non-root user, after installing the packages, please assign the permissions to the below filesystems

*#Assign relevant permission for non-root user*

sudo chmod 755 /usr/local/bin/dms\_cli

sudo chmod -R 755 /usr/local/lib/python3.6

sudo chmod -R 755 /usr/local/lib/python3.6

Step 7: Onboard your xApp

*# Make sure that you have the xapp descriptor config file and the schema file at your local file system*

dms\_cli onboard CONFIG\_FILE\_PATH SCHEMA\_FILE\_PATH

OR

dms\_cli onboard --config\_file\_path=CONFIG\_FILE\_PATH --shcema\_file\_path=SCHEMA\_FILE\_PATH

*#Example:*

dms\_cli onboard /files/config-file.json /files/schema.json

OR

dms\_cli onboard --config\_file\_path=/files/config-file.json --shcema\_file\_path=/files/schema.json

Step 8: (OPTIONAL ) List the helm charts from help repository.

*#List all the helm charts from help repository*

curl -X GET http://localhost:8080/api/charts | jq .

*#List details of specific helm chart from helm repository*

curl -X GET http://localhost:8080/api/charts/<XAPP\_CHART\_NAME>/<VERSION>

Step 9: (OPTIONAL ) Delete a specific Chart Version from helm repository.

*#Delete a specific Chart Version from helm repository*

curl -X DELETE http://localhost:8080/api/charts/<XAPP\_CHART\_NAME>/<VERSION>

Step 10: (OPTIONAL ) Download the xApp helm charts.

dms\_cli download\_helm\_chart XAPP\_CHART\_NAME VERSION --output\_path=OUTPUT\_PATH

OR

dms\_cli download\_helm\_chart --xapp\_chart\_name=XAPP\_CHART\_NAME --version=VERSION --output\_path=OUTPUT\_PATH

Example:

dms\_cli download\_helm\_chart ueec 1.0.0 --output\_path=/files/helm\_xapp

OR

dms\_cli download\_helm\_chart --xapp\_chart\_name=ueec --version=1.0.0 --output\_path=/files/helm\_xapp

Step 11: Install the xApp.

dms\_cli install XAPP\_CHART\_NAME VERSION NAMESPACE

OR

dms\_cli install --xapp\_chart\_name=XAPP\_CHART\_NAME --version=VERSION --namespace=NAMESPACE

Example:

dms\_cli install ueec 1.0.0 ricxapp

OR

dms\_cli install --xapp\_chart\_name=ueec --version=1.0.0 --namespace=ricxapp

Step 12: (OPTIONAL ) Install xApp using helm charts by providing the override values.yaml.

*#Download the default values.yaml*

dms\_cli download\_values\_yaml XAPP\_CHART\_NAME VERSION --output\_path=OUTPUT\_PATH

OR

dms\_cli download\_values\_yaml --xapp\_chart\_name=XAPP\_CHART\_NAME --version=VERSION --output\_path=OUTPUT\_PATH

Example:

dms\_cli download\_values\_yaml traffic-steering 0.6.0 --output-path=/tmp

OR

dms\_cli download\_values\_yaml --xapp\_chart\_name=traffic-steering --version=0.6.0 --output-path=/tmp

*#Modify values.yaml and provide it as override file*

dms\_cli install XAPP\_CHART\_NAME VERSION NAMESPACE OVERRIDEFILE

OR

dms\_cli install --xapp\_chart\_name=XAPP\_CHART\_NAME --version=VERSION --namespace=NAMESPACE --overridefile=OVERRIDEFILE

Example:

dms\_cli install ueec 1.0.0 ricxapp /tmp/values.yaml

OR

dms\_cli install --xapp\_chart\_name=ueec --version=1.0.0 --namespace=ricxapp --overridefile=/tmp/values.yaml

Step 13: (OPTIONAL ) Uninstall the xApp.

dms\_cli uninstall XAPP\_CHART\_NAME NAMESPACE

OR

dms\_cli uninstall --xapp\_chart\_name=XAPP\_CHART\_NAME --namespace=NAMESPACE

Example:

dms\_cli uninstall ueec ricxapp

OR

dms\_cli uninstall --xapp\_chart\_name=ueec --namespace=ricxapp

Step 14: (OPTIONAL) Upgrade the xApp to a new version.

dms\_cli upgrade XAPP\_CHART\_NAME OLD\_VERSION NEW\_VERSION NAMESPACE

OR

dms\_cli upgrade --xapp\_chart\_name=XAPP\_CHART\_NAME --old\_version=OLD\_VERSION --new\_version=NEW\_VERSION --namespace=NAMESPACE

Example:

dms\_cli upgrade ueec 1.0.0 2.0.0 ricxapp

OR

dms\_cli upgrade --xapp\_chart\_name=ueec --old\_version=1.0.0 --new\_version=2.0.0 --namespace=ricxapp

Step 15: (OPTIONAL) Rollback the xApp to old version.

dms\_cli rollback XAPP\_CHART\_NAME NEW\_VERSION OLD\_VERSION NAMESPACE

OR

dms\_cli rollback --xapp\_chart\_name=XAPP\_CHART\_NAME --new\_version=NEW\_VERSION --old\_version=OLD\_VERSION --namespace=NAMESPACE

Example:

dms\_cli rollback ueec 2.0.0 1.0.0 ricxapp

OR

dms\_cli rollback --xapp\_chart\_name=ueec --new\_version=2.0.0 --old\_version=1.0.0 --namespace=ricxapp

Step 16: (OPTIONAL) Check the health of xApp.

dms\_cli health\_check XAPP\_CHART\_NAME NAMESPACE

OR

dms\_cli health\_check --xapp\_chart\_name=XAPP\_CHART\_NAME --namespace=NAMESPACE

Example:

dms\_cli health\_check ueec ricxapp

OR

dms\_cli health\_check --xapp\_chart\_name=ueec --namespace=ricxapp

### [OPTIONALLY use Redis Cluster (instead of Redis standalone)](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id15)

#### [Important](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id16)

The redis-cluster currently is NOT part of RIC platform & hence is completely optional. This piece of document has been created as part of delivery item for below jira ticket <https://jira.o-ran-sc.org/browse/RIC-109> This ticket is about assessing the feasibility of redis-cluster (with data sharding) supporting desired pod anti-affinity for high availability as per the ticket.

#### [Overview](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id17)

This document describes the environment/conditions used to test the feasibility of Redis cluster set-up as detailed in the above ticket. Redis Cluster is a distributed implementation of Redis with high performance goals. More details at <https://redis.io/topics/cluster-spec>

#### [Environment Set-Up](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id18)

**The set up was tested with kubernetes v1.19 cluster with**

1. Pod topology spread constraint enabled Reference: <https://kubernetes.io/docs/concepts/workloads/pods/pod-topology-spread-constraints>
2. CEPH as the Cluster Storage Solution.
3. Three worker nodes in the kubernet cluster

#### [Execution](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/installation-guides.html#id19)

Once environment is set-up, a redis-cluster can be set up using the helm-chart (also provided with this commit). Once cluster is running, any master/slave of the redis instance pods can be deleted which will be compensated automatically by new instances

At this stage the perl utility program (included with helm-chart) can be run. The helm chart installation output generates the requirement commands to invoke.

This utility program identifies the missing anti-affinity(as per above ticket) of redis instances required in a redis-cluster. When executed it communicates to redis nodes to switch roles (e.g. master/slave) such that the end-state meets the desired anti-affinity.

[Previous](https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/release-notes.html)

**E2 Simulator**

1. Clone e2 interfece

ex@ex-VirtualBox:~$ git clone <https://gerrit.o-ran-sc.org/r/sim/e2-interface>

2. install cmake g++ libsctp-dev

ex@ex-VirtualBox:~$ sudo apt-get install cmake g++ libsctp-dev

3. Go to e2 interfece

ex@ex-VirtualBox:~$ cd e2-interface/e2sim

4. enter docker file

ex@ex-VirtualBox:~/e2-interface/e2sim$ cd docker/

5. Check files

ex@ex-VirtualBox:~/e2-interface/e2sim/docker$ ll

6. edit docker file

ex@ex-VirtualBox:~/e2-interface/e2sim/docker$ vi Dockerfile

7. edit docker file>> sleep 100000000

8. create a path (build) in e2sim

ex@ex-VirtualBox:~/e2-interface/e2sim$ mkdir build

9. enter build path

ex@ex-VirtualBox:~/e2-interface/e2sim$ cd build/

10. install compiler

ex@ex-VirtualBox:~/e2-interface/e2sim/build$ cmake .. && make package && cmake .. -DDEV\_PKG=1 && make package

11. use this command

ex@ex-VirtualBox:~/e2-interface/e2sim/build$ cp \*.deb ../e2sm\_examples/kpm\_e2sm/

12. use this command

ex@ex-VirtualBox:~/e2-interface/e2sim/build$ cd ../e2sm\_examples/kpm\_e2sm/

13. docker build

ex@ex-VirtualBox:~/e2-interface/e2sim/e2sm\_examples/kpm\_e2sm$ sudo docker build -t oransim:0.0.999 .

14. check docker file running or not

ex@ex-VirtualBox:~/e2-interface/e2sim/e2sm\_examples/kpm\_e2sm$ sudo docker run -d --name oransim -it oransim:0.0.999

15. to go root@ee3c390d1bdc

ex@ex-VirtualBox:~/e2-interface/e2sim/e2sm\_examples/kpm\_e2sm$ sudo docker exec -ti oransim /bin/bash

16. To check docker images

ex@ex-VirtualBox:~/e2-interface/e2sim/e2sm\_examples/kpm\_e2sm$ sudo docker image ls

17. To check docker images in details

ex@ex-VirtualBox:~/e2-interface/e2sim/e2sm\_examples/kpm\_e2sm$ sudo docker ps

18. To stop docker container

ex@ex-VirtualBox:~/e2-interface/e2sim/e2sm\_examples/kpm\_e2sm$ sudo docker container stop oransim

19. To remove docker container

ex@ex-VirtualBox:~/e2-interface/e2sim/e2sm\_examples/kpm\_e2sm$ sudo docker container rm oransim

20. To run docker container

ex@ex-VirtualBox:~/e2-interface/e2sim/e2sm\_examples/kpm\_e2sm$ sudo docker container start oransim

21. To go to the root………./playpen

ex@ex-VirtualBox:~/e2-interface/e2sim/e2sm\_examples/kpm\_e2sm$ sudo docker exec -ti oransim /bin/bash

22. to check RIC

Kpm\_sim 10.96.235.53 36422

23. check ric running or not

ex2@ex2:~/ric-dep/bin$ curl -X GET http://10.110.69.145:3800/v1/nodeb/states 2>/dev/null|jq