# **Near-RT RIC Deployment**

- Near-RT RIC deployment steps
   RIC Platform:
  - RIC Aux
- E2 Simulator deployment steps
  - Wireshark deployment

## Near-RT RIC deployment steps

Both VMs:

mkdir workspace cd workspace git clone "https://gerrit.o-ran-sc.org/r/ric-plt/ric-dep" cd ric-dep/bin sudo ./install\_k8s\_and\_helm.sh sudo ./install\_common\_templates\_to\_helm.sh sudo ./setup-ric-common-template

## **RIC Platform:**

Under ~/workspace/ric-dep repo:

```
nano helm/appmgr/values.yaml
# replace tiller image with these values
#
        tiller:
          registry: ghcr.io
#
         name: helm/tiller
#
#
          tag: v2.12.3
nano helm/infrastructure/values.yaml
# replace tiller image with these values
#
        tiller:
          registry: ghcr.io
#
         name: helm/tiller
#
#
          tag: v2.12.3
# edit the ../RECIPE_EXAMPLE/example_recipe_latest_stable.yaml
./install -f ../RECIPE_EXAMPLE/example_recipe_latest_stable.yaml
```

**RIC** Aux

```
cd ~/workspace
git clone "https://gerrit.o-ran-sc.org/r/it/dep"
cd dep
nano RECIPE_EXAMPLE/AUX/example_recipe.yaml
\ensuremath{\texttt{\#}} replace the ric-dashboard image with the one below
# image:
   registry: nexus3.o-ran-sc.org:10002/o-ran-sc
#
#
  name: ric-dashboard
#
   tag: 2.1.0
nano ric-aux/helm/infrastructure/subcharts/kong/values.yaml
# replace the ingressController image with the one below
#ingressController:
# enabled: true
# image:
#
   repository: docker.io/kong/kubernetes-ingress-controller
#
   tag: 0.7.0
sudo apt install dos2unix
cd dep/bin
dos2unix -o deploy-ric-aux
kubectl label --overwrite nodes $(hostname) portal-storage=enable
kubectl label --overwrite nodes $(hostname) aaf-storage=enable
./deploy-ric-aux ../RECIPE_EXAMPLE/AUX/example_recipe.yaml
```

## E2 Simulator deployment steps

```
git clone "https://gerrit.o-ran-sc.org/r/sim/e2-interface"
sudo apt-get install -y cmake build-essential libsctp-dev autoconf
automake libtool bison flex libboost-all-dev
cd e2-interface/e2sim
mkdir build
cd build
cmake .. && make package && cmake .. -DDEV_PKG=1 && make package
cp *.deb ../e2sm_examples/kpm_e2sm/
```

```
#####
# Copyright 2022 highstreet technologies GmbH
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
     http://www.apache.org/licenses/LICENSE-2.0
#
#
\ensuremath{\texttt{\#}} Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#
version: '3.8'
services:
 e2-simulator:
   image: "o-ran-sc/ntsim-e2-simulator:2.0.0"
   build:
    context: /home/ubuntu/workspace/e2-interface/e2sim/e2sm_examples
/kpm_e2sm
   container_name: e2-simulator
   entrypoint: ["kpm_sim", "10.106.23.83", "38000"]
```

```
#!/bin/bash
```

```
NODE_IP=$(kubectl get pod -n=ricplt -l app=ricplt-e2term-alpha -o
jsonpath="{.items[0].status.hostIP}")
NODE_PORT=$(kubectl get svc -n=ricplt service-ricplt-e2term-sctp-alpha -o
jsonpath="{.spec.ports[0].nodePort}")
echo "E2 term IP address: ${NODE_IP}"
echo "E2 term port: ${NODE_PORT}"
```

```
cd /workspace
mkdir e2-sim
cd e2-sim
nano e2sim-infra.sh
# copy above script for getting the IP and Port of the E2 termination point
# run the script and use the IP and port in the following docker-compose
nano docker-compose.yml
# copy above YAML into this file
# replace the entrypoint IP and port with the above IP and port
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --
dearmor -o /etc/apt/keyrings/docker.gpg
echo \
 "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings
/docker.gpg] https://download.docker.com/linux/ubuntu \
 $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.
list > /dev/null
sudo apt-get update
sudo apt-get install docker-compose-plugin
sudo docker compose build
sudo docker compose up -d
```

## Wireshark deployment

Wireshark can also be deployed to sniff the traffic inside the k8s cluster.

```
Steps:
```

krew installation (instructions here):

```
(
    set -x; cd "$(mktemp -d)" &&
    OS="$(uname | tr '[:upper:]' '[:lower:]')" &&
    ARCH="$(uname -m | sed -e 's/x86_64/amd64/' -e 's/\(arm\)\(64\)\?.*/\1\2
/' -e 's/aarch64$/arm64/')" &&
    KREW="krew-${OS}_${ARCH}" &&
    curl -fsSL0 "https://github.com/kubernetes-sigs/krew/releases/latest
/download/${KREW}.tar.gz" &&
    tar zxvf "${KREW}.tar.gz" &&
    ./"${KREW}" install krew
)
export PATH="${KREW_ROOT:-$HOME/.krew}/bin:$PATH"
```

ksniff installation (instructions here):

kubectl krew install sniff

Wireshark can be deployed inside a docker container, as described in the OAM repository (wireshark section) in O-RAN-SC.

In order to see the packets captured with `kubectl sniff` in Wireshark, a pipe needs to be created (instructions here):

mkfifo /tmp/capture.pcap

Please make sure that the above pipe (/tmp/capture.pcap) is mounted as a volume in the Wireshark docker container!

Configure Wireshark to capture packets from that pipe:

### Capture Options

← → C ▲ Not secure   10.31.1.100:3000														
	4													
	Eile	Edi	it <u>v</u>	iew	<u>3</u> 0		ure	<u>A</u> nalyze	<u>S</u> tatistics	Telephony	<u>W</u> ireless <u>T</u> ool	s <u>H</u> elp		
	1		A		-	0		ns				θΘ		
Ļ			(ACA)	•		1	Start	~		Ctrl+E			-•	
l	to	p.po:	rt ==	4560										
1	No.		Time		_		- ·			C++1+P	ort Destina	tion		
1	4196 2022-06-08 09				1442	iesta.				4560 10.244.0.54 57210 10.96.83.131				
		4283	83 2022-06-08 09 Capture <u>F</u> ilter				re <u>F</u> 11te	rs						
		4284 2022-06-08 09 Refresh In			sh Inter	faces	F5	4560 10.244.	10.244.0.54					
		4285 2022-06-08 09.34.37.390372				0372	10.244.0.34		5/210 10.96.83.131					
		4286	2022	-00-08	5 09	:34:	57.59	6622	10.96.83.131		4560 10.244.	0.54		
		4387	2022	-06-08	5 09	:36:	02.60	5330	10.244.0.54		57210 10.96.8	3.131		
		4388	2022	-06-08	8 09	:36:	02.60	5436	10.96.83.131		4560 10.244.	0.54		
		4389	2022	-06-08	09	:36:	02.60	5934	10.244.0.54		57210 10.96.8	3.131		
		4390	2022	-06-08	09	:36:	02.60	6008	10.96.83.131		4560 10.244.	0.54		
		4489	2022	-06-08	09	:37:	07.58	7989	10.244.0.54		57210 10.96.8	3.131		
		4490	2022	-06-08	09	:37:	07.58	8041	10.96.83.131		4560 10.244.	0.54		
		4491	2022	-06-08	09	:37:	07.58	8254	10.244.0.54		57210 10.96.8	3.131		
	L	4492	2022	-06-08	09	:37:	07.58	8280	10.96.83.131		4560 10.244.	0.54		

### Manage Interfaces:

interface	Traffic	Link-layer Header	Promisc	Snapler
▶ ens3		Ethernet	<b>v</b>	default
⊧ docker0		Ethernet	<b>v</b>	default
flannel.1		Ethernet	<b>v</b>	defaul
> cni0		Ethernet	<b>v</b>	defaul
vethbb276c1d		Ethernet	<b>v</b>	defaul
veth4d2784ae		Ethernet	V	defaul
vethScc53a57		Ethernet	<b>v</b>	defau]
vethffdc412b		Ethernet	V	defau]
vethd9605355		Ethernet	<b>v</b>	defaul
veth52fdbf64		Ethernet	<b>v</b>	defau]
vethfa34ddaa		Ethernet	<b>v</b>	defau]
vetha3fe9498		Ethernet	×	defau]
veth491c1f74		Ethernet	<b>v</b>	defau]
veth7be7e4b1		Ethernet	<b>v</b>	defau]
Enable promiscuous mode on all interf	faces			nterfac
apture filter for selected interfaces:	Enter a capture filter -		* Co	noile A

Add a new pipe pointing to the pipe inside the mapped volume of the Wireshark docker container.

<b>4</b>		
Local Interfaces	ripes	
Local Pipe Path		
/config/pcap/capture	: pcap	
+ -		
This version of Wireshark does not sa	ve pipe settings.	
		🖉 OK 🛛 🗶 Cancel 👯 Help

### Select that pipe as the interface to run the capture on:

nput	Output	Options					
Interf.	ace			Traffic	Link-layer Header	Promisc	Snaplen
veti	hbc261f2				Ethernet	✓	default
> veti	hd73f93b3				Ethernet	✓	default
vetł	hb8ec3387				Ethernet	✓	default
br-i	b159492d4a	8b			Ethernet	✓	default
veti	h331fbað				Ethernet	✓	default
any					Linux cooked v1	✓	default
Loop	pback: lo				Ethernet	<b>v</b>	default
kub	e-ipvs0				Unknown	✓	default
nfl	og				Linux netfilter log messages	✓	default
nfq	ueue				Raw IPv4	✓	default
Dis	plavPort A	UX channel moni	tor capture: dpauxmon		DisplayPort AUX channel monitor		
Ran	dom packet	generator: ran	dpkt		Generator dependent DLT		
UDP	Listener	remote capture:	udpdump		Exported PDUs		
/co	nfig/pcap/	capture.pcap			Unknown	✓	default
							•
Enab	le promise	uous mode on al	1 interfaces			Manage II	nterfaces
opture	e filter f	or selected inte	erfaces: 📕 Enter a ca	apture filter _	•	Cor	npile BPF

### Start the k8s sniffer on the Pod of interest:

<pre>kubectl sniff ricxapp-rc-86f7dfd8ff-5klz8 -n ricxapp -o /tmp/capture.pcap</pre>														
#	this	will	sniff	traffic	from	the	RC	xApp	k8s	pod	to	the	/tmp/capture.	
p	pcap pipe													

Important: the order matters! The Wireshark capture needs to be started first, and only then the sniffing of the traffic towards that pipe!