## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>3</td>
</tr>
<tr>
<td>Scope</td>
<td>3</td>
</tr>
<tr>
<td>Terminologies</td>
<td>3</td>
</tr>
<tr>
<td>Deployment</td>
<td>3</td>
</tr>
<tr>
<td>Cloud Lab Profile</td>
<td>3</td>
</tr>
<tr>
<td>Accessing VMs/SSH to VM</td>
<td>6</td>
</tr>
<tr>
<td>Manual Build and Setup of each NGIC components</td>
<td>6</td>
</tr>
<tr>
<td>Combined GW (SAEGW mode)</td>
<td>6</td>
</tr>
<tr>
<td>SPGWC</td>
<td>6</td>
</tr>
<tr>
<td>SPGWU</td>
<td>8</td>
</tr>
<tr>
<td>ILTraffic Gen</td>
<td>10</td>
</tr>
<tr>
<td>Running OMEC Demo</td>
<td>12</td>
</tr>
<tr>
<td>Demo</td>
<td>12</td>
</tr>
<tr>
<td>O/P Stats</td>
<td>14</td>
</tr>
<tr>
<td>DPDK Binding</td>
<td>14</td>
</tr>
<tr>
<td>Recovery steps in case of reboot/shutdown</td>
<td>15</td>
</tr>
<tr>
<td>Step to perform on the Host in case of host reboot</td>
<td>15</td>
</tr>
<tr>
<td>Steps to perform on DP VM after rebooting of VM/Host.</td>
<td>16</td>
</tr>
</tbody>
</table>
Introduction
This is a quick guide to capture all the required configurations to deploy and launch Gateway tutorial working with il_trafficgen test tool.

Scope
Deployment will be considered for Combined SGW+PGW control plane (SPGWC), Combined SGW+PGW data plane (SPGWU) and traffic generator (il_trafficgen).

Terminologies
This section describes terminologies or abbreviations.

- CP - Control Plane Service/PDN Gateway node i.e. SPGWC
- DP - Data Plane Service/PDN Gateway node i.e. SPGWU
- SPGWC - Combined Serving and PDN Gateway Control plane function.
- SPGWU - Combined Serving and PDN Gateway User plane function.

Deployment
In order to run this tutorial user needs cloudlab account.

Cloud Lab Profile
Cloud Lab URL: https://www.cloudlab.us/
Sign up for an account and choose "Join Existing Project"; for "Project Name" enter cord-testdrive.
Following section will guide user to create OMEC-Demo instance on cloudbab where it has single Bare metal server with three VMS (SPGWC, SPGWU and ILTrafficGen) preinstalled and preconfigured.

1. Create Instance using OMEC-Demo profile
   Select Tab option “Project Profiles” and then search for the profile name “OMEC-Demo”
2. Instantiate the profile by clicking blue arrow against the profile.
3. Click on “Next”

4. Give some unique name e.g. “TestOMECGW1” and select cluster “Cloudlab Clemson” and then click “Next”.

5. Experiment Duration 16hr is max and default you can change it. If u want to schedule the instance then you can select the date and time and then click “Finish”. Deployment takes around 5-10mins.
6. Once deployment finishes it will display the ssh information. Use that to login to instance using your username and key setup with the account. Please note SSH is key based. User should generate SSH key and upload to user profile on cloudlab.

7. After ssh do, ‘sudo su’ and then follow the instruction to access the VMs.

Accessing VMs/SSH to VM

1. To check list of running vm on the host

   ```
   virsh list
   ```

2. Get IP address of VM

   ```
   cd /opt/deployment/scripts
   ./get_vm_ip.sh <vm_name>
   ```

3. Login to vm

   ```
   cd /opt/deployment/scripts
   ./sshvm.sh ubuntu <vm_name>
   ```

4. Login to vm using ubuntu ssh key

   ```
   ssh -i /home/ubuntu/.ssh/id_rsa ubuntu@<ip_address_of_vm>
   ```

Note: OMEC-Demo profile has prebuild and pre-configured components. If you want to manual install and configuration then follow the instruction from section Manual Build and Setup of each NGIC components. Otherwise jump directly to execution steps at to Running OMEC tutorial.

Manual Build and Setup of each NGIC components

This section assumes all the VMs are installed and networking is setup and validated

Combined GW (SAEGW mode)

SPGWC

   Installation
   1. Clone ngic-rtc repo under /opt/
git clone https://github.com/omc-project/ngic-rtc.git

2. **Execute the install.sh script available in ngic-rtc directory.** Install script will provide below mentioned options. Choose the options accordingly and install required items.

```
Step 1: Environment setup.
[1] Check OS and network connection
[2] Configured Service - Collocated CP and DP

Step 2: Download and Install
[3] Agree to download
[4] Download packages
[5] Download DPDK zip
[6] Install DPDK
[7] Download hyperscan

Step 3: Build NGIC
[8] Build NGIC
[9] Exit Script

Option Descriptions:
[1] This will check the OS version and network connectivity and report any anomaly. If the system is behind a proxy, it will ask for a proxy and update the required environment variables.
[2] Select services to build. You will be prompted to select following options from sub-menu, and optionally edit the memory size:
   1. CP Only
   2. DP Only
   3. Collocated CP and DP
[3] Select yes in this option to be able to download the required packages.
[4] Actual download of the dependent packages like libpcap, build essentials etc.
[5] Download dpdk as a zip file using this option.
[6] Build and set environment variables to use DPDK.
[7] Download hyperscan library. This option is displayed in menu when 'DP Only' or 'Collocated CP and DP' option is selected in [2].
[8] Build controlplane and dataplane applications. This sets the RTE_SDK environment variable and builds the applications.

Note for SGWC: use the following options in sequence

1) [3]
2) [4]
3) [5]
4) [6]
5) [2]->[1]->[no]
6) [8]
Configuration

1. Update `config/interface.cfg` file for the below parameters based on the interfaces configured on VM.

   ```
   dp_comm_ip = 10.0.7.81  \(\Leftarrow\) DP interface IP towards CP
   dp_comm_port = 20
   cp_comm_ip = 10.0.7.80  \(\Leftarrow\) CP interface IP towards DP
   cp_comm_port = 21
   ```

2. Update `config/cp_config.cfg` for the below parameters based on the configuration in this case it’s running as SGWC

   ```
   #SGPW_CFG:: SGWC=01; PGWC=02; SPGWC=03
   SPGW_CFG=03
   # Put MME IP here  MME_S11_IP
   S11_MME_IP={{ MME_S11_IP }}
   S11_SGW_IP=10.0.2.70  \(\Leftarrow\) S11 interface IP on CP
   S1U_SGW_IP=11.7.1.93  \(\Leftarrow\) S1U interface IP on DP
   MEMORY=1024
   CORELIST="0-4"  \(\Leftarrow\) core list needs 5 cores
   ```

Run

Run CP using the following:

```
ngic-rtc/cp/run.sh log
```

Note: If the any prompt complaining memory, type ‘y’ and press ‘enter’

SPGWU

Installation

1. Clone ngic-rtc repo under /opt/

   ```
   git clone https://github.com/omec-project/ngic-rtc.git
   ```

1. Execute the install.sh script available in ngic-rtc directory. Install script will provide below mentioned options. Choose the options accordingly and install required items.

   ```
   Step 1: Environment setup.
   [1] Check OS and network connection
   [2] Configured Service - Collocated CP and DP
   Step 2: Download and Install
   [3] Agree to download
   [4] Download packages
   [5] Download DPDK zip
   [6] Install DPDK
   [7] Download hyperscan
   Step 3: Build NGIC
   ```
[8] Build NGIC
[9] Exit Script

Option Descriptions:

[1] This will check the OS version and network connectivity and report any anomaly. If the system is behind a proxy, it will ask for a proxy and update the required environment variables.

[2] Select services to build. You will be prompted to select following options from sub-menu, and optionally edit the memory size:
   1. CP Only
   2. DP Only
   3. Collocated CP and DP

[3] Select yes in this option to be able to download the required packages.

[4] Actual download of the dependent packages like libpcap, build essentials etc.

[5] Download dpdk as a zip file using this option.

[6] Build and set environment variables to use DPDK.

[7] Download hyperscan library. This option is displayed in menu when 'DP Only' or 'Collocated CP and DP' option is selected in [2].

[8] Build controlplane and dataplane applications. This sets the RTE_SDK environment variable and builds the applications.

Note for SGWC: use the following options in sequence (without SGX)

1) [3]
2) [4]
3) [5]
4) [6]
5) [7]
6) [2]-->[2]-->[no]

Configuration

1. Update config/interface.cfg file for the below parameters based on the interfaces configured on VM.
2. Update config/dp_config.cfg for the below parameters based on the configuration in this case it’s running as SGWU

```
#SPGW_CFG:: SGWU=01; PGWU=02; SPGWU=03
SPGW_CFG=03
S1U_PORT=0000:00:05.0 ← PCI ID of the s1u interface
S1U_PORT=0000:00:06.0 ← PCI ID of the sgi interface
S1U_IP=11.7.1.93 ← IP address for the s1u interface
S1U_MAC=3e:bb:de:3e:28:48 ← MAC address of s1u interface
SGI_IP=13.7.1.93 ← IP address for the sgi interface
SGI_MAC=f6:ab:93:49:98:d7 ← MAC address of sgi interface
MEMORY=2048
CORE_LIST="0-3" ← Core list needs 4 cores
```

Run
Before running DP ensure the S1U port is bind to DPDK (refer DPDK Binding section)

1. Run DP using the following:

```
cd /opt/ngic-rtc/dp/
./run.sh log
```

Note: If the any prompt complaining memory, type ‘y’ and press ‘enter’

2. Open another terminal and run the KNI Scripts

```
cd /opt/ngic-rtc/kni-config
./kni-sludevcfg.sh
./kni-sgidevcfg.sh
```

ILTraffic Gen
Installation
1. Clone il_trafficgen repo under /opt/

```
git clone https://github.com/omec-project/il_trafficgen.git
```

2. Execute the install.sh script available in il_trafficgen directory. Install script will provide below mentioned options. Choose the options accordingly and install required items.

```
-------------------------------
Step 1: Environment setup.
-------------------------------
```
[1] Check OS and network connection

Step 2: Download and Install

[2] Agree to download
[4] Download packages
[5] Setup Huge Pages
[6] Download DPDK submodule
[7] Install DPDK

Step 3: Build IL_TRAFFICGEN

[8] Build IL_TRAFFICGEN

[9] Exit Script

Configuration

1. Update user_input.cfg file

```bash
$ cd /opt/il_trafficgen/pktgen/autotest/
$ vim user_input.cfg

# Il_trafficgen Generator host IP
gen_host_ip="127.0.0.1"  # IP ADDR OF IL-NPERF MGMT INTERFACE

# Il_trafficgen Generator port number
gen_host_port=5344

# Il_trafficgen Responder host IP
resp_host_ip="127.0.0.1"  # IP ADDR OF IL-NPERF MGMT INTERFACE

# Il_trafficgen Responder port number
resp_host_port=5345

# S1U port
slu_port="0000:00:04.0"

# SGI port
gsi_port="0000:00:05.0"

# il_trafficgen: generator S1U interface src mac address
p0_src_mac="46:71:87:00:ab:58"  # MAC ID OF IL-NPERF S1U INTERFACE

# System Under Test: ngic/vnf_portfwd S1U dst mac address
p0_dst_mac="4e:26:94:10:b2:52"  # MAC ID OF DP S1U INTERFACE

# il_trafficgen: responder SGI interface src mac address
p1_src_mac="56:6c:fd:3c:3d:6a"  # MAC ID OF IL-NPERF SGI INTERFACE

# System Under Test: ngic/vnf_portfwd SGI dst mac address
p1_dst_mac="aa:5d:38:f7:e2:1a"  # MAC ID OF DP SGI INTERFACE
```
2. Configure CP for simulating control sessions
   a. Edit the “/opt/ngic-rtc/config/simu_cp.cfg”

```plaintext
[0]
S1U_SGW_IP=11.7.1.93 ← S1U interface IP on DP
ENDEB_IP_START = 11.7.1.101 ← eNB IP start sync with iltraffic
UE_IP_START = 16.0.0.1
UE_IP_START RANGE = 16.0.0.0
AS_IP_START = 13.7.1.110 ← AS IP start sync with iltraffic
MAX_UE_SESS = 5000 ← Max UE sessions sync with iltraffic
TPS = 1000
BREAK_DURATION = 60
DEFAULT_BEARER = 5
ng4t_max_ue_ran = 500000
ng4t_max_enb_ran = 80
```

b. Enable flags for “DSIMU CP” and rebuild CP

```
$ cd /opt/ngic-rtc/cp/
$ vim Makefile
# Uncomment below line to read fake cp config.
CFLAGS += -DSIMU_CP
```

Re-build CP using install.sh script.

c. On DP VM Enable static ARP (default dynamic ARP) as follows:

```
$ cd /opt/ngic-rtc/dp/
$ vim Makefile
# Uncomment below line to enable STATIC ARP
#CFLAGS += -DSTATIC_ARP
```

Re-build DP using install.sh script.

d. Edit the “static_arp.cfg” file which is available under ‘config’ directory. Update the ipaddr range of SGI and S1U interface and mac addresses of SGI and S1U interfaces on il-traffic-gen(IL-NPERF) on machine

```
$ vim /opt/ngic-rtc/config/static-arp.cfg

[sgi]

[s1u]
11.8.1.101 11.8.1.180 = 9e:45:1d:9c:c1:d6
```

Running OMEC Demo

Demo

1. Complete DPDK binding on DP as mentioned in DPDK binding section.
2. Start the DP service on SPGWU VM.

```
$ /opt/ngic-rtc/dp
$ ./run.sh
```
3. Start CP Service no SPGWC VM

```
$ /opt/ngic rtc/cp
$.run.sh
```

Wait for CP to establish session on DP wait for the following message on the CP console

```
STATS ::

**********************************
MAX_NUM_CS    : 5000
MAX_NUM_MB    : 5000
NUM_CS_SEND   : 5000
NUM_MB_SEND   : 5000
NUM_CS_FAILED : 0
NUM_MB_FAILED : 0
**********************************

************ DP Configured successfully ************
```

1. Complete DPDK binding on IL-NPERF(il_trafficgen) machine as mentioned in DPDK binding section.
2. Open two SSH windows to IL-NPERF instance.
3. In fist SSH window of IL-NPERF start the il-trafficgen generator on IL-NPERF machine using below command. once it is successfully started then proceed next step for start il-trafficgen receiver.

```
$cd /opt/il_trafficgen/pktgen
# Start generator in one screen
$ ./il_nperf.sh -g

# Wait for generator to start and then start receiver in another screen
```

4. In second SSH window of IL-NPERF start the il-trafficgen generator on IL-NPERF machine using below command.

```
$cd /opt/il_trafficgen/pktgen
$ ./il_nperf.sh -r
```

5. To start traffic flow type start 0 on both generator and receiver prompt and press enter. Note: Do not try to start generator and receiver both simultaneously.

6. Once test completes press “quit” on both IL-NPERF consoles to see the traffic results.
Sample O/P Stats on IL-NPERF instance

DPDK Binding
Bind the S1u/Sgi port to DPDK drivers on SPGWU instance.

1. Command to get PCI address is:

```
root@spgwu:/opt/ngic-rtc/dpdk/usertools# lshw -c network -businfo
```

<table>
<thead>
<tr>
<th>Bus info</th>
<th>Device</th>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pci@0000:00:03.0</td>
<td>ens3</td>
<td>network</td>
<td>Virtio network device</td>
</tr>
<tr>
<td>pci@0000:00:04.0</td>
<td>ens4</td>
<td>network</td>
<td>Virtio network device</td>
</tr>
<tr>
<td>pci@0000:00:08.0</td>
<td>ens5</td>
<td>network</td>
<td>82599ES 10-Gigabit SFI/SFP+ Network Connection</td>
</tr>
<tr>
<td>pci@0000:00:09.0</td>
<td>ens6</td>
<td>network</td>
<td>82599ES 10-Gigabit SFI/SFP+ Network Connection</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2. Bind the port using the PCI id
cd /opt/ngic-rtc/dpdk/usertools/
./dpdk-devbind.py -b igb_uio 00:08.0

3. Lists ports

root@spgwu:/opt/ngic-rtc/dpdk/usertools# ./dpdk-devbind.py --status

Network devices using DPDK-compatible driver
============================================
0000:00:08.0 '82599ES 10-Gigabit SFI/SFP+ Network Connection 10fb' 
drv=igb_uio unused=ixgbe

Network devices using kernel driver
===================================
0000:00:03.0 'Virtio network device 1000' if=ens3 drv=virtio-pci 
unused=igb_uio *Active*
0000:00:04.0 'Virtio network device 1000' if=ens4 drv=virtio-pci 
unused=igb_uio *Active*
0000:00:09.0 '82599ES 10-Gigabit SFI/SFP+ Network Connection 10fb' 
if=ens6 drv=ixgbe unused=igb_uio

Recovery steps in case of reboot/shutdown

Step to perform on the Host in case of host reboot

1) Insert following command on host server.

   modprobe vfio-pci

2) Identify the physical interface on which the VFs were configured and the number of VFs

   Following command will give you interface name.
   e.g.

   $ cat /opt/deployment/terraform/c3povm_defs.cfg | grep CTRL_PFDEV
   CTRL_PFDEV=ens786f1

   Following command will give you count of the VFs

   $ cat /opt/deployment/terraform/c3povm_defs.cfg | grep NUM_CTRL_VF
   NUM_CTRL_VF=17

3) Create virtual network function using following command.

   echo 17 > /sys/class/net/ens786f1/device/sriov_numvfs

4) Up the NIC

   ifconfig ens786f1 up

5) Start VM on host using virsh command.

   virsh start <VM name>
Steps to perform on DP VM after rebooting of VM/Host.

Following step need to perform on the DP (SGWU/PGWU/SPGWU) VM after reboot.

1) Login to particular VM from host using following command.

```
    cd opt/deployment/script
    ./sshvm.sh ubuntu <VM name>
```

2) Load DPDK drivers

```
    cd /opt/ngic-rtc/dpdk/usertools/
    ./dpdk-setup.sh

    Select the following options and then exit

    Options : 14  #set environment
               17  #insert igb_uio module
               19  #insert kni module
```

3) Bind the S1u/Sgi port to DPDK drivers (refer DPDK Binding Section)