free5GC_Installation_Guide

I. Environment

Simulations have been tested against the following environment:

1.1. Software

- OS: Ubuntu 20.04.2.0
- gcc 7.3.0
- Go 1.17 linux/amd64
- kernel version 5.4.0-42-generic

1.2. Minimum Hardware

- CPU: Intel i5 processor
- RAM: 4GB
- Hard drive: 160GB
- NIC: Any 1Gbps Ethernet card supported in the Linux kernel

1.3. Recommended Hardware

- CPU: Intel i7 processor
- RAM: 8GB
- Hard drive: 160GB
- NIC: Any 10Gbps Ethernet card supported in the Linux kernel

This guide assumes that you will run all 5GC elements on a single machine.

II. Ubuntu Virtual Machine Installation

In this part, we:

- Install Virtualbox
- Create and install a Ubuntu virtual machine
- Install a specific Kernel version (5.4.0-42-generic).

2.1. Install Virtualbox

- Download Virtualbox from https://www.virtualbox.org/ and install (currently 6.1.26).
- Install following the instruction on the software.

2.2. Create a Ubuntu VM

- Download Ubuntu ver 20.04.2.0.
- Launch VirtualBox and create your first Ubuntu VM using the downloaded .iso image file. Some notes when creating a new VM:
  - Name the first VM using a generic name as ubuntu, or ubuntu-20.04
  - You can pick 1 or 2 (or more) CPUs, and about 2048M memory, although you can change them later
  - In addition, change the default NAT network interface to Bridged Apdater network interface (in Setting of Virtualbox interface).
- Install Ubuntu:

  It is recommended that you should:

  - Choose a short username and password for ease of typing later
  - Choose Minimal Installation and uncheck Download updates while installing ubuntu, to prevent installing newest Kernel which may cause error when running make of free5gc as well as ueransim.
- Install plugin Insert Guest Addition CD Image from Virtualbox/Devices interface after booting Ubuntu VM on the first time to enhance display solution.

2.3. Install a specific Kernel version

- Check Kernel version

```
uname -r
```

- Check a specific kernel version and install

```
# To find all kernels in version 5
sudo apt search 'linux-image-5*'

# Install specific kernel version
sudo apt install 'linux-image-5.4.0-42-generic'
```

- Update initramfs

```
sudo update-initramfs -u -k all
```

- Set default grub kernel boot

  See Set default grub kernel boot.

- Update grub then reboot

```
sudo update-grub

sudo reboot

uname -r            # check version again
```

- Install kernel header file

```
sudo apt install linux-headers-$(uname -r)
```

III. Creating and Configuring a free5GC VM

The aims of this part are:

- Cloning an existing VM to install free5GC
- Installing and testing the free5GC core network
- Configuring free5GC.

3.1. Create A free5GC VM

3.1.1. Clone a VM

First, let's clone a new VM:

- Select an existing VM (ubuntu) and Right-click and choose Clone
- Name the new VM as free5gc
- The MAC address rule: Create new MAC addresses for all network cards
- Choose the Full Clone option.

3.1.2. Change Hostname

- The cloned free5gc VM still has the hostname ubuntu (or the name you gave it in the original VM). Let's rename the VM to free5gc. You can do this by editing the file /etc/hostname (using vi or nano):

```
sudo nano /etc/hostname  # or sudo vi /etc/hostname
```

- Then, change the file /etc/hosts by replacing the ubuntu inside into free5gc:

```
sudo nano /etc/hosts
```

3.1.3. Set Static IP

- Now, we change the IP of free5gc machine to a static IP (in the IP's range of your network), e.g., 192.168.31.101/24. You also need change the DNS's mode from Automatically to Manual (8.8.8.8).

3.2. Install & Test free5GC Core Network

3.2.1. Install Basic Tools

- First make sure Golang (go) is not installed:

```
go version
```

- If go is installed, remove it first (assuming it is installed at /usr/local/go):

```
sudo rm -rf /usr/local/go
```

- To install the latest go, search golang on the web, and get the website: Golang. On the website page, choose Linux, and obtain the download URL (by right-clicking the box Download Go for Linux and chose something like Copy link address). The URL address looks like this: https://golang.org/dl/go1.17.linux-amd64.tar.gz

- With the URL address, we can install the latest go using:

```
cd ~
wget https://golang.org/dl/go1.17.linux-amd64.tar.gz
sudo tar -C /usr/local -xzf go1.17.linux-amd64.tar.gz
```

- Then, execute the following commands (copy and paste):

```
mkdir -p ~/go/{bin,pkg,src}
echo 'export GOPATH=$HOME/go' >> ~/.bashrc
echo 'export GOROOT=/usr/local/go' >> ~/.bashrc
echo 'export PATH=$PATH:$GOPATH/bin:$GOROOT/bin' >> ~/.bashrc
echo 'export GO111MODULE=auto' >> ~/.bashrc
source ~/.bashrc
```

- And check if go is installed successfully:

```
go version
```

- Next, we will install MongoDb:

```
sudo apt -y update
sudo apt -y install mongodb
sudo systemctl start mongodb
```

- Finally, install other development tools:

```
sudo apt -y install git gcc g++ cmake autoconf libtool pkg-config libmnl-dev libyaml-dev
go get -u github.com/sirupsen/logrus
```

3.2.2. Set up Networking

- To setup network rules, copy and paste:

```
sudo sysctl -w net.ipv4.ip_forward=1
sudo iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE
sudo systemctl stop ufw
```

- Note:
  - The name enp0s3 is the network interface that free5GC use to connect to Data Network (i.e., Internet).
  - These network settings will disappear after reboot. So, make sure you run the above commands after each reboot.

3.2.3. Install free5GC Core Network

- Let's install the latest free5GC directly:

```
cd ~
git clone --recursive https://github.com/free5gc/free5gc.git
```

- To build free5GC, do:

```
cd ~/free5gc
make
```

- We also need to install kernel module gtp5g:

```
cd ~
git clone https://github.com/PrinzOwO/gtp5g.git
cd gtp5g
make
sudo make install
```

- To check if gtp5g is installed successfully, see if the following command shows some information:

```
lsmod | grep gtp
```

3.2.4. Testing free5GC

- free5GC provides some testing procedures to make sure it works properly. First let's just test the basic registration procedure:

```
cd ~/free5gc
./test.sh TestRegistration
```

- If everything runs properly without RED error messages, and the word PASS appears near the end of the screen output, then free5GC is running properly.
- We can further check other free5GC procedures:

```
./test.sh TestGUTIRegistration
./test.sh TestServiceRequest
./test.sh TestXnHandover
./test.sh TestDeregistration
./test.sh TestPDUSessionReleaseRequest
./test.sh TestPaging
./test.sh TestN2Handover
./test.sh TestNon3GPP
./test.sh TestReSynchronisation
./test_ulcl.sh -om 3 TestRegistration
```

3.3. Configuring free5GC

3.3.1. Install free5GC WebConsole

- free5GC provides a simple web tool WebConsole to help to create and manage UE registrations to be used by various 5G network functions (NF). To build WebConsole we need Node.js and Yarn.
- First, remove obsolete tools that may exist:

```
sudo apt remove cmdtest
sudo apt remove yarn
```

- Then, install Node.js and Yarn:

```
curl -sS https://dl.yarnpkg.com/debian/pubkey.gpg | sudo apt-key add -
echo "deb https://dl.yarnpkg.com/debian/ stable main" | sudo tee /etc/apt/sources.list.d/yarn.list
sudo apt-get update
sudo apt-get install -y nodejs yarn
```

- Build WebConsole:

```
cd ~/free5gc
make webconsole
```

3.3.2. *Add a UE via WebConsole*

- First, start up the WebConsole server:
- The result screen shows the port number :5000 at the end. Open your web browser from your host machine, and enter the following URL (the IP address of free5gc + :5000): http://192.168.31.101:5000
  - On the login page, enter username admin and password free5gc
  - Once logged in, widen the page until you see Subscribers tab on the left-hand side column.
  - Choose Subscribers tab and create a new data
    - Note that other than the Operator Code Type field which you should choose OP for now, leave other fields unchanged. This registration data is used for ease of testing and actual use later.
  - After the data is created successfully, you can press Ctrl-C on the terminal to quit WebConsole.

3.3.3. *Setting free5GC Parameters*

In free5gc VM, we need to edit three files:

- First, change file amfcfg.yaml in path: ~/free5gc/config/amfcfg.yaml

  Replace ngapIpList IP from 127.0.0.1 to the IP address of free5gc, e.g., 192.168.31.101

```
...
   ngapIpList:  # the IP list of N2 interfaces on this AMF
   - 192.168.31.101  # 127.0.0.1
```

- Next, edit file smfcfg in path: ~/free5gc/config/smfcfg.yaml

```
  ...
     interfaces: # Interface list for this UPF
     - interfaceType: N3 # the type of the interface (N3 or N9)
        endpoints: # the IP address of this N3/N9 interface on this UPF
          - 192.168.31.101  # 127.0.0.8
```

- Finally, edit file upfcfg in path: ~/free5gc/NFs/upf/build/config/upfcfg.yaml

  Replace gtpu IP from 127.0.0.1 to the IP address of free5gc, e.g., 192.168.31.101

```
...
   gtpu:
   - addr: 192.168.31.101  # 127.0.0.8
```

IV. Creating and Configuring a UE RAN Simulator

The aims of this guide are:

- Cloning an existing VM to install ueransim
- Installing and testing the ueransim simulator
- Configuring ueransim simulator.

4.1. Create A ueransim VM

Refer to the instruction in Creating and Configuring a free5GC VM with a name of ueransim and an IP address, e.g., 192.168.31.102.

4.2. Install & Test ueransim Simulator

4.2.1. Install ueransim

- Search "ueransim" on the web and get the github.
- In the github page, find the Installation page.
- Download UERANSIM:

```
cd ~
git clone https://github.com/aligungr/UERANSIM
cd UERANSIM
git checkout v3.1.0
```

- Install required tools:

```
sudo apt install make
sudo apt install g++
sudo apt install libsctp-dev lksctp-tools
sudo apt install iproute2
sudo snap install cmake --classic
```

- Build UERANSIM:

```
cd ~/UERANSIM
make
```

4.2.2. Setting parameters for ueransim simulator

In the ueransim VM, there are two files related to free5GC

- Edit the file free5gc-gnb.yaml in the path ~/UERANSIM/config/free5gc-gnb.yaml, and change the ngapIp IP and gtpIp IP, from 127.0.0.1 to the IP address of the ueransim VM, e.g., 192.168.31.102, and also change the IP in amfConfigs into the IP address of the free5gc VM, e.g., 192.168.31.101:

```
  ...
      ngapIp: 192.168.31.102   # 127.0.0.1    # gNB's local IP address
for N2 Interface (Usually same with local IP)
      gtpIp: 192.168.31.102   # 127.0.0.1     # gNB's local IP address
for N3 Interface (Usually same with local IP)


      # List of AMF address information
      amfConfigs:
        - address: 192.168.31.101   # 127.0.0.1
```

- Next, we examine the file free5gc-ue.yaml in the path ~/UERANSIM/config/free5gc-ue.yaml and see if the settings are consistent with those of the Subscriber in the free5GC (via WebConsole).

V. Testing UERANSIM against free5GC

5.1. Run on free5gc VM

- Run network setting (must do every time rebooting):

```
sudo sysctl -w net.ipv4.ip_forward=1
sudo iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE
sudo systemctl stop ufw
sudo iptables -I FORWARD 1 -j ACCEPT
```

- Run free5gc:

```
cd ~/free5gc
./run.sh
```

  At this time free5GC has been started.

5.2. Run on ueransim VM

- Open a Terminal 1, execute nr-gnb and leave it running:

```
cd ~/UERANSIM
sudo build/nr-gnb -c config/free5gc-gnb.yaml
```

- Open a Terminal 2, execute nr-ue and leave it running:

```
cd ~/UERANSIM
sudo build/nr-ue -c config/free5gc-ue.yaml
```

- Open a Terminal 3, ping 192.168.31.101 to see free5gc is alive. Then, use ifconfig to see if the tunnel uesimtun0 has been created (by nr-ue).
- Ping google.com via the tunnel uesimtun0 to test the connection to the internet via 5G core network.

```
ping -I uesimtun0 google.com
```

  If ping gets replies, then free5GC is running properly. Congratulations!

APPENDIX

A. Set default grub kernel boot

The following text shows how to set Default GRUB kernel boot.

1. **Get the $menuentry_id_option for the submenu named "Advanced options for Ubuntu":**

```
grep submenu /boot/grub/grub.cfg
```

- The result should be:

```
submenu 'Advanced options for Ubuntu' $menuentry_id_option 'gnulinux-advanced-43fea22c-500a-4846-ab6d-ecd32e472183' {
```

- Copy the YELLOW text in the result:

2. **Get the $menuentry_id_option for the 5.4.0-42-generic kernel boot:**

```
grep gnulinux /boot/grub/grub.cfg
```

- The result should contain the text like this:

```
menuentry 'Ubuntu, with Linux 5.4.0-42-generic' --class ubuntu --class gnu-linux --class gnu --class os $menuentry_id_option 'gnulinux-5.4.0-42-generic-advanced-43fea22c-500a-4846-ab6d-ecd32e472183' {
```

- Copy the YELLOW text in the result.

3. **Edit Default GRUB:**

- Create a backup and edit the GRUB file:

```
# Make a backup of the file, to our home directory:
sudo cp /etc/default/grub ~

# Edit GRUB file:
sudo vi /etc/default/grub
```

- Press i to insert # to comment the following commands:

```
#GRUB_DEFAULT=0
#GRUB_HIDDEN_TIMEOUT=0
```

- Press **ESC** to escape insert mode, press Ctrl + V to paste the copied texts in the above:

  ```
  GRUB_DEFAULT='gnulinux-advanced-43fea22c-500a-4846-ab6d-
  ecd32e472183>gnulinux-5.4.0-42-generic-advanced-43fea22c-500a-4846-
  ab6d-ecd32e472183'
  ```

  Remember that there is > between these texts.

- Press **:wq** to write and close this file.

Follow this guide for more detail.