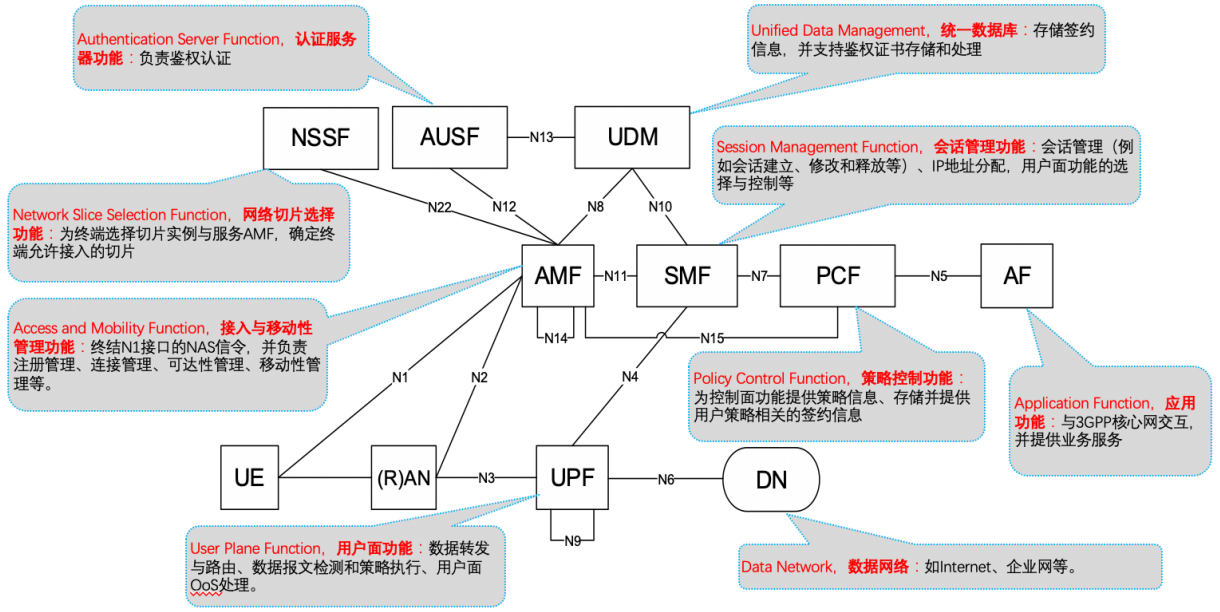
Free5GC的搭建说明

文档对5GC环境配置进行说明，配置方式为多机虚拟机为主（与官方的单机配置不同），是对官方的补充和详细说明，推荐和官方说明一块使用。

一、5GC的架构简介

2.6 核心网架构图-2



其中free5GC项目中包含的网元包括

NF_LIST= "nrf amf smf udr pcf udm nssf ausf " upf n3iwf

Nrf为网络存储库功能单元，负责各网元之间的服务发现和存储。N3iwf为非标准接入的方式，暂时无需部署。

二、部署方法

free5GC可采用虚拟机部署和容器化部署两种方式。推荐使用多个虚拟机的方式部署（即2.1和2.2.1的内容。）

2.1虚拟机环境的搭建

虚拟机部署的首要问题是虚拟机的配置和安装。需要在实验室的ubuntu服务器上安装kvm。虚拟机之间的网络通信方式有两种：NAT模式、网桥模式。我们使用网桥模式，这样可以为每个虚拟机分配独立的IP，并且自己搭建网桥，通过学校的DHCP动态分配给每个虚拟机一个校园局域网的IP，这样可以不通过宿主机而是直接ssh登录虚拟机，方便登录和管理（坏处是每当学校的网络出现故障或者停电时，机器的IP便会改变）。

具体步骤：（推荐使用MobaXterm终端进行登录，在虚拟机安装时可以使用图形化界面）

- 1.安装kvm及其相关组件。
- 2.创建网桥，我们选择自己创建网桥，并将宿主机和虚拟机全都绑到网桥上。

```
创建网桥br0: sudo brctl addbr br0
```

- 3.修改配置文件/etc/netplan/00-installer，其中红色圈内改为宿主机对应的网卡名。然后重启网络配置。这一步将网卡绑定到网桥上，并动态分配ip，后面安装的虚拟机也将绑定到br0上。

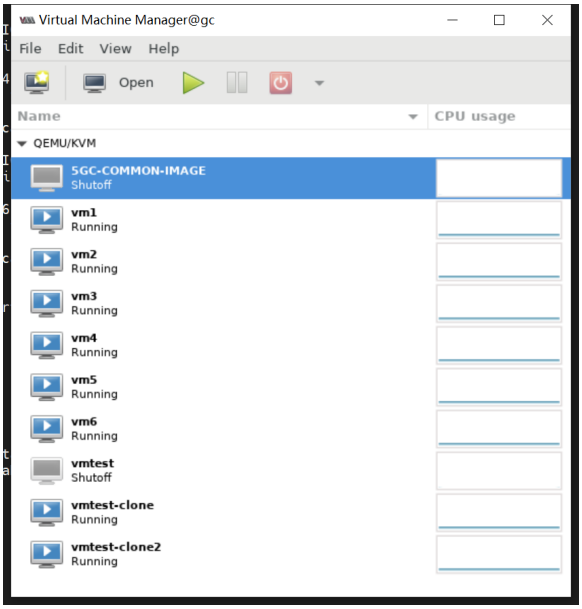
```
ebupt@gc:/etc/netplan$ ls
00-installer-config.yaml  00-installer-config.yaml.back
ebupt@gc:/etc/netplan$ cat 00-installer-config.yaml
# This is the network config written by 's
#
network:
  ethernets:
    enp4s0f3:
      dhcp4: no
      dhcp6: no
  version: 2
  bridges:
        br0:
           interfaces: [enp4s0f3]
           dhcp4: yes
            #dhcp6: no
            #addresses: [10.108.124.212/22]
           # gateway4: 10.108.124.1
           #nameservers:
                   # addresses: [10.3.9.4,10.3.9.5]

ebupt@gc:/etc/netplan$
```

- 4.开启网桥相关的内核配置。（这一步非常关键，是一个大坑，一定要开启以下iptable配置！）

```
net.bridge.bridge-nf-call-ip6tables=0
net.bridge.bridge-nf-call-iptables=0
net.bridge.bridge-nf-call-arptables=0
```

- 5.安装虚拟机。在终端输入 sudo virt-manager（如果使用MobaXterm会弹出如下图形化界面）。可以选择本地的ubuntu镜像安装虚拟机，在网络配置时，选择brigde模式并绑定到新建的网桥br0上。这样可以为虚拟机分配独立的校园网IP。



2.2 free5GC的网元启动和配置。

2.2.1 虚拟机方式部署

在2.1中已经可以安装多台虚拟机，我们将会把不同的网元放置在不同的虚拟机中启动（有一些网元可以放在同一台虚拟机中比如udr,udm,nrf）， 这一步之前，要先按照free5GC项目的要求配置好本地虚拟机所依赖的环境和组件。

建议多安装一些虚拟机，不同的组件放在不同的虚拟机中。

网元启动的顺序推荐为：mongodb nrf amf udr udm nssf ausf pcf upf smf

- 首先需要安装数据库mongodb。（参考free5GC官方的即可）
- NRF配置文件及启动。（以下是原始配置文件，IP须改为具体配置对应的地址，只截取了部分关键的配置信息，所有网元同理）

```
configuration:
  MongoDBName: "free5gc"
  MongoDBUrl: "mongodb://127.0.0.1:27017" (此处的ip为mongodb所在虚拟机的ip)
  DefaultServiceIP: "127.0.0.1" (此处的ip为nrf网元所在虚拟机的ip)
  sbi:
    scheme: http
    ipv4Addr: 127.0.0.1 (此处的ip为nrf所在虚拟机的ip)
    port: 29510
  DefaultPlmnId:
    mcc: "208"
    mnc: "93"
  serviceNameList:
    - nnrf-nfm
    - nnrf-disc
```

配置完成后可启动nrf网元，然后在项目的free5gc/log目录下可以看到相关的日志，没有error的报错信息即可。

- amf配置

```
configuration:
  amfName: AMF
  ngapIpList:
    - 10.108.124.114 #amf所在的虚拟机IP
  sbi:
    scheme: http
    registerIPv4: 10.108.124.114 # IP used to register to NRF（amf所在的虚拟机IP）
    bindingIPv4: 10.108.124.114  # IP used to bind the service（amf所在的虚拟机IP）
    port: 29518

   ....
  supportDnnList:
    - internet
  nrfUri: http://10.108.126.198:29510 #（NRF的IP）
```

- udr配置

```
configuration:
  sbi:
    scheme: http
    registerIPv4: 127.0.0.1# IP used to register to NRF （udr所在主机ip）
    bindingIPv4: 127.0.0.1# IP used to bind the service（udr所在主机ip）
    port: 29504
  mongodb:
    name: free5gc
    url: mongodb://localhost:27017   （mongodb所在ip）
    nrfUri: http://localhost:29510   （nrf所在ip）
```

- udm配置

```
sbi:
  scheme: http
  registerIPv4: 127.0.0.1# IP used to register to NRF (udm所在的Ip)
  bindingIPv4: 127.0.0.1# IP used to bind the service (udm所在的Ip)
  port: 29503
  tls:
    log: free5gc/udmsslkey.log
    pem: free5gc/support/TLS/udm.pem
    key: free5gc/support/TLS/udm.key

udrclient:
  scheme: http
  ipv4Addr: 127.0.0.1 (udr所在的Ip)
  port: 29504

nrfclient:
  scheme: http
  ipv4Addr: 127.0.0.1 (nrf所在的Ip)
  port: 29510
nrfUri: http://localhost:29510 (nrf所在的Ip)
```

- ausf配置

```
configuration:
  sbi:
    scheme: http
    registerIPv4: 127.0.0.1# IP used to register to NRF （ausf所在的ip）
    bindingIPv4: 127.0.0.1# IP used to bind the service   （ausf所在的ip）
    port: 29509
  serviceNameList:
```

```
    - nausf-auth
  nrfUri: http://localhost:29510   (nrf所在的ip)
  plmnSupportList:
```

nssf配置

```
configuration:
  nssfName: NSSF
  sbi:
    scheme: http
    registerIPv4: 127.0.0.1# IP used to register to NRF (nssf所在ip)
    bindingIPv4: 127.0.0.1# IP used to bind the service (nssf所在ip)

...

nsiInformationList:
    - nrfId: http://localhost:29510/nnrf-nfm/v1/nf-instances #(与nrfID相关的全都改为nrf所在的ip)
       nsiId: 10
```

- pcf配置

```
configuration:
  pcfName: PCF
  sbi:
    scheme: http
    registerIPv4: 127.0.0.1# IP used to register to NRF (pcf所在主机ip)
    bindingIPv4: 127.0.0.1# IP used to bind the service (pcf所在主机ip)
    port: 29507
  timeFormat: 2019-01-02 15:04:05
  defaultBdtRefId: BdtPolicyId-
  nrfUri: http://localhost:29510   (nrf所在主机ip)
```

- smf配置

```
configuration:
  smfName: SMF
  sbi:
    scheme: http
    registerIPv4: 127.0.0.1# IP used to register to NRF (smf所在主机地址)
    bindingIPv4: 127.0.0.1# IP used to bind the service (smf所在主机地址)
    port: 29502

...
```

```
pfcp:
    addr: 127.0.0.1     (smf所在ip)
  userplane_information:
    up_nodes:
      gNB1:
        type: AN
        an_ip: 127.0.0.100
      UPF:
        type: UPF
        node_id: 127.0.0.8   (upf所在ip)
```

💡 所有网元均可通过日志查看是否启动正常，可以通过日志来进行问题排查。

2.2.2 容器化部署的方法。

容器化部署方式推荐使用docker-compose方式，这是官方的项目。

项目github地址为：https://github.com/free5gc/free5gc-compose

💡 需要熟练使用docker和docker编排工具docker-compose，以及了解docker网桥等组件。在2.2.1中虚拟机配置的时候，可以参考free-compose项目中的许多配置信息，会对free5GC项目的配置有更深刻的理解。

容器化部署方式按照官方的方法配置即可，**推荐使用2.2.1中的虚拟机部署方式，**原因如下：

- 容器化方式会共用宿主机内核，也需要配置宿主机环境（尤其是upf）,并没有方便很多。
- 容器化方式在进行quic改造的时候并不方便，并且会调用共用的宿主机内核的协议栈，没有真正跨主机通信。

2.3 5GC接入互联网与终端设备接入5GC

这一块张龙继续做了一些工作，并且实用性和效果都很好，可以请教张龙，包括系统的可靠性测试，张龙搞了一套新工具和方法。

3.现有环境

- 目前实验室有一台服务器环境配置完好，装有多台虚拟机并有完整的free5gc系统。

```
服务器地址：10.112.57.140
user: ebupt
passwd: 82325588
ssh端口： 246（学校封了22端口，这台机器和机器上所有的虚拟机都是通过246端口登录）
```

- 虚拟机安装情况如下：

VM1中有free5gc-compose 容器化部署的镜像和环境（服务未启动）。

VM2部署 mongodb 、nrf 、ausf、nssf、pcf、udr、udm。

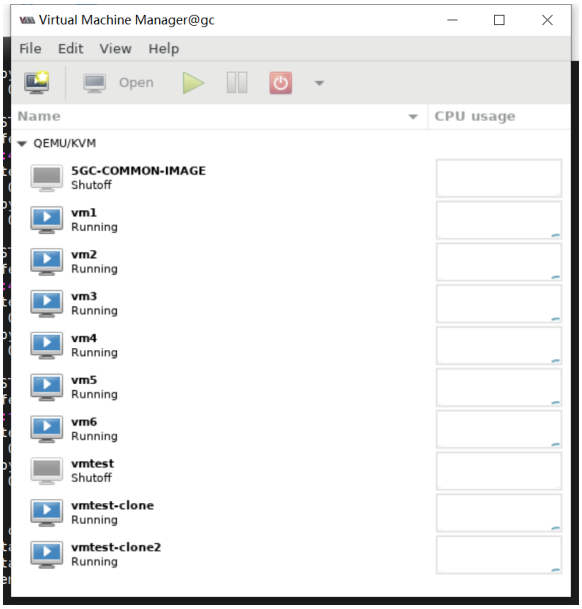
VM3中部署amf

VM4中部署smf

VM5中部署upf

虚拟机user: cjn　passwd: 123456　ssh端口：246

其他虚拟机作为测试和平时教学使用。



以上系统完成了部分quic改造，可以参考这台机器进行5GC的环境配置，（若要进行修改或改进需与张龙进行商议）。从头学习并熟悉这一套系统，建议再从机柜上找一台闲置的机器，从零进行一次5GC的搭建，这样更能锻炼自己并且会对系统更加熟悉。

新建虚拟机时推荐使用这台机器上的5GC-COMMON-IMAGE镜像，我已经配置了该镜像运行5gc所需的环境，直接使用此镜像会少踩很多坑。