

# 共进自研 5G 核心网环境配置

文件编号：	文件页数：第 页 共 页	版本号：				
文件名称：5G 核心网环境配置						
使用限制：需要运行 ubuntu-18.04 系统						
下载地址： <a href="http://mirrors.aliyun.com/ubuntu-releases/18.04/">http://mirrors.aliyun.com/ubuntu-releases/18.04/</a>						
更改记录 Release note						
版本号	修改描述	更改部门	更改人	审核人	批准人	生效日期
V1.0	首板发行	软件二部	刘俭	田野	田野	2021.3.18
V1.1	修改	测试部	何文亮			2023.4.13
V1.2	增加转发命令	测试部	何文亮			2023.9.18
V1.3	增加 openssh 方法和 sim 卡固定 ip	测试部	何文亮			2023.10.7
V1.4	增加 4G 核心网的配置和启动	测试部	郭锁奇			2023.10.11
V1.5	增加新版 Ubuntu 核心网启动报错处理方法	测试部	李智涌			2024.10.16

# 目录

共进自研 5G 核心网环境配置 .....	1
一、 环境配置说明.....	3
1. 环境安装.....	3
2. 5G 核心网的配置和启动 .....	4
3. 4G 核心网的配置和启动 .....	6
4. 核心网上业务转发配置.....	9
5. 安装 openssh 软件方便 ssh 设备进行操作配置 .....	10
二、 开户操作说明.....	11
1. WebUI 增加签约用户 .....	11
2. 给每个 sim 卡分配固定 ip 地址 .....	13
三、 异常情况.....	13

# 一、 环境配置说明

## 1. 环境安装

1.安装 mongodb

```
$ sudo apt update
```

```
$ sudo apt install mongodb
```

```
$ sudo systemctl start mongodb (if '/usr/bin/mongod' is not running)
```

```
$ sudo systemctl enable mongodb (ensure to automatically start it on system boot)
```

2.安装核心网依赖的软件

```
$ sudo apt install python3-pip python3-setuptools python3-wheel ninja-build build-essential flex  
bison git libsctp-dev libgnutls28-dev libgcrypt-dev libssl-dev libidn11-dev libmongoc-dev libbson-  
dev libyaml-dev libnghttp2-dev libmicrohttpd-dev libcurl4-gnutls-dev libnghttp2-dev libtins-dev  
meson
```

3.安装工具软件(缺啥安装啥)

```
$ sudo apt install vim tar
```

4. 将 5gc.tar.gz 复制到/usr/local 目录下并解压。

```
sudo tar -xvzf 5gc.tar.gz
```

5.安装 wireshark

因为 ubuntu 18.04 自带的 wireshark 版本太低，不支持 ngap 协议，所以需要升级为高版本

① <https://blog.csdn.net/valecalida/article/details/103579778> 安装 wireshark 的操作方法

② 如果安装过程库文件失败，请修改 ubuntu18.04 的更新源以后，重新进行安装。(源配置可以混用)

更新源方法:<https://www.cnblogs.com/2sheep2simple/p/10610847.html>

说明：以上安装需要将设备联网，可直接命令获取互联网库安装。

## 2. 5G 核心网的配置和启动

### 1. 配置

配置文件在 install/etc/5gc/目录下

#### (1) 修改 amf.yaml

```
# amf_name: amf1.open5gs.amf.5gc.mnc70.mcc901.3gppnetwork.org
#
amf:
  sbi:
    - addr: 127.0.0.5
      port: 7777
    ngap:
      - addr: 127.0.0.5
  guami:
    - plmn_id:
        mcc: 901
        mnc: 70
      amf_id:
        region: 2
        set: 1
  tai:
    - plmn_id:
        mcc: 901
        mnc: 70
      tac: 1
  plmn_support:
    - plmn_id:
        mcc: 901
        mnc: 70
      s_nssai:
        - sst: 1
  security:
    integrity_order : [ NIA2, NIA1, NIA0 ]
    ciphering_order : [ NEA0, NEA1, NEA2 ]
  network_name:
    full: Open5GS
  amf_name: open5gs-amf0
#
```

请将以上配置文件的 ngap 的 IP 地址修改为设备的 IP 地址，比如 192.168.106.201。

根据 gNB 的实际配置，修改核心网相关的配置，请修改上图红框部分（guami, tai 和 plmn 节点）。

主要修改 mcc 和 mnc（与 sim 卡一致），tac(与基站一致)。

#### (2)修改 upf.yaml

```
# range:
#   - cafe::a0-cafe:b0
#   - cafe::c0-cafe:d0
#
upf:
  pfcf:
    - addr: 127.0.0.7
  gtpu:
    - addr: 127.0.0.7
  pdu:
    - addr: 10.45.0.1/16
    - addr: cafe::1/64

#
# smf:
#
# <PCFP Client>
#
# o PCFP Client(127.0.0.3:8805)
#
#   pfcf:
#     addr: 127.0.0.3
#
smf:
```

请将以上配置文件的 gtpu 的 IP 地址修改为设备的 IP 地址，比如 192.168.106.201。

## 2. 核心网的启动和运行

进入到核心网的目录：cd /usr/local/5gc

修改 execute5gc.sh 脚本：修改 sh 为 bash

```
mcs@mcs-PowerEdge-R750xs:/usr/local/5gc$ cat execute5gc.sh
#!/bin/sh

# =====
# Prepare Block
# =====

OBJECT=$1
OPERATE=$2
TRY_COUNT=3
WEB_PATH=""
echo "**** Input Param: "$OBJECT" "$OPERATE""

# =====
# Func Block
# =====

# Func: Stop 5gc
function stop_5gc()
{
    echo "**** Stop 5gc"
    kill -9 `ps aux | grep "5gc" | grep -v grep | awk '{print $2}'` &
}

# Func: Check 5gc ready or not
function check_5gc_ready()
{
    THREAD_COUNT=`ps -ef | grep "5gc" | grep -v "grep" | grep -v "execute5gc.sh" | wc -l`
    echo "**** Check 5gc thread: "$THREAD_COUNT""
    if [ $THREAD_COUNT -lt 15 ]; then
        return 0
    else
        return 1
    fi
}
```

运行核心网：sudo bash ./execute5gc.sh c r

停止核心网：sudo bash ./execute5gc.sh c s

注意：以上命令的执行在 root 权限下执行。

脚本不能执行的情况：

ls -al /bin/sh

sudo dpkg-reconfigure dash

### 3. 4G 核心网的配置和启动

#### 1. 配置

配置文件在 install/etc/5gc/目录下

(1) 检查 amf.yaml

请将以上配置文件的 ngap 的 IP 地址还原为默认 IP 地址 127.0.0.5。

```
logger:
  file: /usr/local/5gc/var/log/5gc/amf.log
amf:
  sbi:
    - addr: 127.0.0.5
      port: 7777
  ngap:
    - addr: 127.0.0.5
  guami:
    - plmn_id:
        mcc: 901
        mnc: 70
      amf_id:
        region: 2
        set: 1
  tai:
    - plmn_id:
        mcc: 901
        mnc: 70
      tac: 1
  plmn_support:
    - plmn_id:
```

### (2) 检查 upf.yaml

请将以上配置文件的 gtpu 的 IP 地址还原为默认 IP 地址 127.0.0.7。

```
logger:
  file: /usr/local/5gc/var/log/5gc/upf.log
upf:
  pfcf:
    - addr: 127.0.0.7
  gtpu:
    - addr: 127.0.0.7
  subnet:
    - addr: 10.45.0.1/16
    - addr: 2001:230:cafe::1/48
smf:
parameter:
max:
pool:
```

### (3) 修改 mme.yaml

请将以上配置文件的 s1ap 的 IP 地址修改为设备的 IP 地址，比如 192.168.50.129。

根据 eNB 的实际配置，修改核心网相关的配置，请修改上图红框部分的值（gummei, tai 和 plmn 节点）。

主要修改 mcc 和 mnc（与 sim 卡一致）, tac(与基站一致)。

```
logger:
  file: /usr/local/5gc/var/log/5gc/mme.log
mme:
  freeDiameter: /usr/local/5gc/etc/freeDiameter/mme.conf
  s1ap:
    - addr: 192.168.50.129
  gtpc:
    - addr: 127.0.0.2
  metrics:
    - addr: 127.0.0.2
      port: 9090
  gumme1:
    plmn_id:
      mcc: 460
      mnc: 88
    mme_gid: 2
    mme_code: 1
  tai:
    plmn_id:
      mcc: 460
      mnc: 88
    tac: 1
  security:
    integrity_order : [ EIA2, EIA1, EIA0 ]
    ciphering_order : [ EEA0, EEA1, EEA2 ]
  network_name:
    full: 5GC
  mme_name: 5gc-mme0
sgwc:
  gtpc:
    - addr: 127.0.0.3
```

#### (4) 修改 sgwu.yaml

请将以上配置文件的 gtpu 的 IP 地址修改为设备的 IP 地址，比如 192.168.50.129。

```
logger :
  file: /usr/local/5gc/var/log/5gc/sgwu.log

sgwu:
  pfcf:
    - addr: 127.0.0.6
  gtpu:
    - addr: 192.168.50.129

sgwc:

parameter:

max:

pool:
```

## 2. 核心网的启动和运行

4G 核心网的启动方法与 5GC 核心网的相同，请参照 5G 核心网启动部分章节。

## 4. 核心网上业务转发配置

```
sudo sysctl -w net.ipv4.ip_forward=1
```

可以将该命令写到 rc.local 文件中开机启动，这样在核心网侧的业务服务器的业务通过核心网转发。

```
echo test | sudo -S bash execute5gc.sh c r
exit 0
test@test-Default-string:/etc$ sudo vi rc.local
test@test-Default-string:/etc$
test@test-Default-string:/etc$
test@test-Default-string:/etc$
test@test-Default-string:/etc$
test@test-Default-string:/etc$ cat rc.local
#!/bin/bash -e
cd /usr/local/5gc
echo test | sudo -S bash execute5gc.sh c r
sudo sysctl -w net.ipv4.ip_forward=1
exit 0
test@test-Default-string:/etc$
```

```
sudo iptables -t nat -A POSTROUTING -s 10.45.0.0/16 ! -o ogstun -j MASQUERADE
```

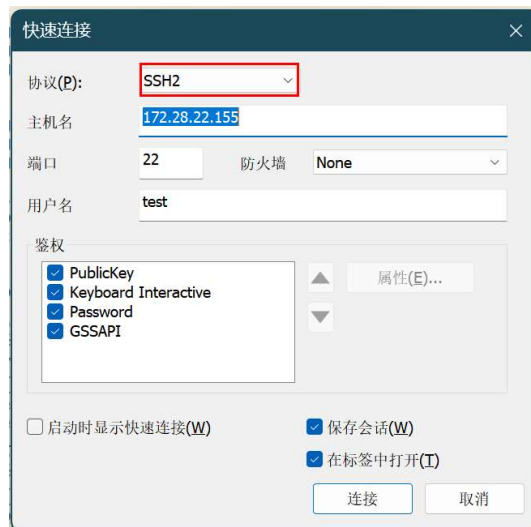
----如何接入的终端需要连接互联网，核心网需要增加这条转发命令。

## 5. 安装 openssh 软件方便 ssh 设备进行操作配置

SSH 是一种安全的远程连接方式，可以确保您的连接是加密的，并且只有授权的用户才能访问系统。在 Ubuntu 中，您可以使用 OpenSSH 服务器和客户端来建立 SSH 连接。您需要在 Ubuntu 系统上安装 OpenSSH 服务器。您可以使用以下命令安装：

```
sudo apt-get install openssh-server
```

安装后，可使用 Windows 下的支持 ssh 协议的软件登录，比如 SecureCRT，如下：



选择 ssh2 协议，输入核心网设备的 ip 地址，输入用户名，点击连接，弹出输入密码框，输入密码登录，即可 ssh 进入设备，进行命令操作。

```
test@test-EQ: /usr/local/5gc X
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.19.17-051917-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

扩展安全维护 (ESM) Infrastructure 未启用。
0 更新可以立即应用。
124 个额外的安全更新可以通过 ESM Infra 来获取安装。
可通过以下途径了解如何启用 ESM Infra, for Ubuntu 18.04 at
https://ubuntu.com/18-04

Your Hardware Enablement Stack (HWE) is supported until April 2023.
Last login: Sun Oct  8 16:29:31 2023 from 172.28.22.186
test@test-EQ:~$
test@test-EQ:~$
test@test-EQ:~$
test@test-EQ:~$ cd /usr/local/5gc/
test@test-EQ:~$ ll
总用量 32
drwxr-xr-x  7 root root 4096 10月  9 2021 ./
drwxr-xr-x 12 root root 4096  9月 12 09:47 ../
drwxr-xr-x  2 root root 4096 10月  9 2021 bin/
drwxr-xr-x  4 root root 4096 10月  9 2021 etc/
-rwxr-xr-x  1 root root 3800 10月  9 2021 execute5gc.sh*
drwxr-xr-x  3 root root 4096 10月  9 2021 lib/
drwxr-xr-x  3 root root 4096 10月  9 2021 var/
drwxr-xr-x  8 root root 4096  9月 20 16:07 webui/
test@test-EQ:~$
test@test-EQ:~$
test@test-EQ:~$
test@test-EQ:~$ ps aux|grep 5gc
test  1519  0.0  0.0 16188 1096 pts/0  S+  16:39   0:00 grep --color=auto 5gc
test@test-EQ:~$
test@test-EQ:~$
test@test-EQ:~$
test@test-EQ:~$ sudo bash execute5gc.sh c r
[sudo] test 的密码:
```

## 二、 开户操作说明

### 1. WebUI 增加签约用户

(1) 执行如下命令启动 WebUI:

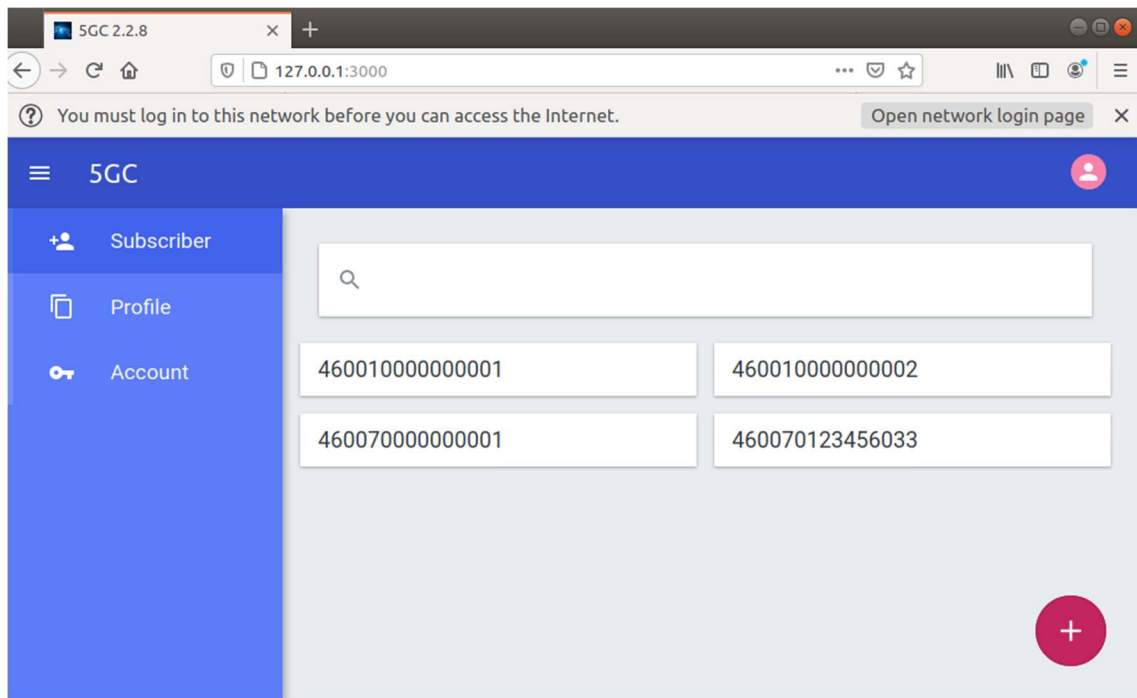
```
cd /usr/local/5gc/webui
```

```
npm run dev
```

注意：以上命令的执行在 root 权限下执行。

```
DONE Compiled successfully in 4062ms
mongoose: accounts.count({}, {})
> Ready on http://localhost:3000
mongoose: subscribers.ensureIndex({ imsi: 1 }, { unique: true, background: true })
mongoose: accounts.ensureIndex({ username: 1 }, { unique: true, background: true })
```

(2) 打开虚拟机自带火狐浏览器, 登录地址 <http://localhost:3000>, 用户名密码 admin/1423, 如下图所示。



A. 编辑操作：鼠标滑过任意 IMSI，会出现编辑和删除按钮；

B. 增加操作：点击图示右下角加号，填写如下数据即可：

IMSI：自定义，注意 Plmn 值

Subscriber Key(K)\*: 01020304050607080102030405060708（和烧录白卡一致）；

USIM Type: 选择 OP（和烧录白卡一致）；

Operator Key (OPc/OP)\*: 01020304050607080102030405060708 (和烧录白卡一致);

Access Point Name (APN)\*: internet (和烧录白卡一致);

Type: IPv4v6;

C. 点击 Save 保存即可。

### Edit Subscriber

Subscriber Configuration

IMSI\*  
26801000000690

Subscriber Key (K)\*: 01020304050607080102030405060708  
Authentication Management Field (AMF)\*: 8000

USIM Type: OP  
Operator Key (OPc/OP)\*: 01020304050607080102030405060708

UE-AMBR Downlink (Kbps)\*: 1024000  
UE-AMBR Uplink (Kbps)\*: 1024000

APN Configurations

Access Point Name (APN)*	Type*	
		<input type="button" value="x"/>

### Edit Subscriber

UE-AMBR Downlink (Kbps)\*: 1024000  
UE-AMBR Uplink (Kbps)\*: 1024000

APN Configurations

Access Point Name (APN)*	Type*	
internet	IPv4v6	<input type="button" value="x"/>

QoS Class Identifier (QCI)\*  
 1  2  3  4  5  6  7  8  9  65  66  69  70

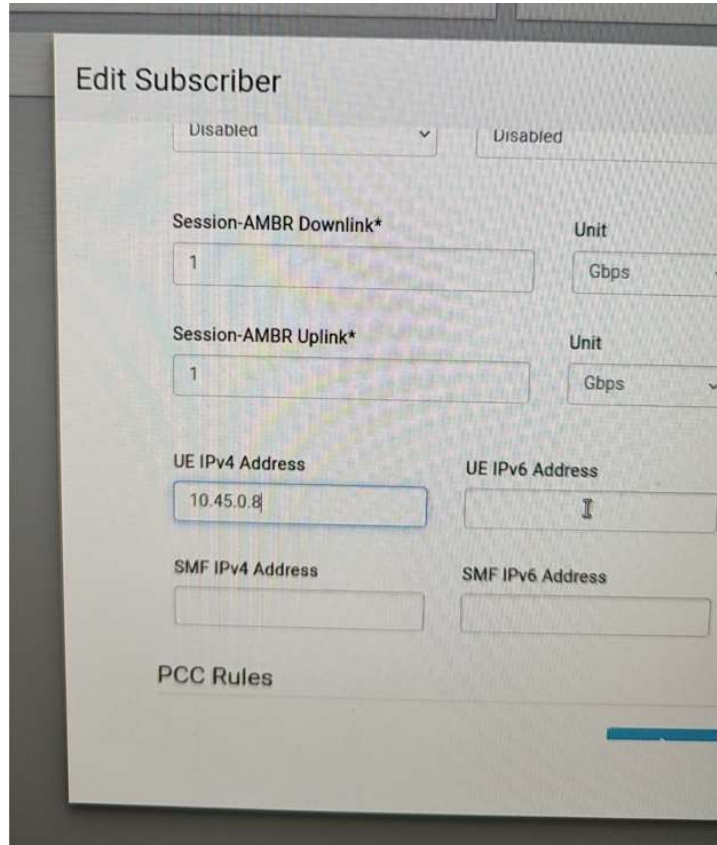
ARP Priority Level (1-15)\*: 8  
Capability\*: Disabled  
Vulnerability\*: Disabled

APN-AMBR Downlink (Kbps)\*: 1024000  
APN-AMBR Uplink (Kbps)\*: 1024000

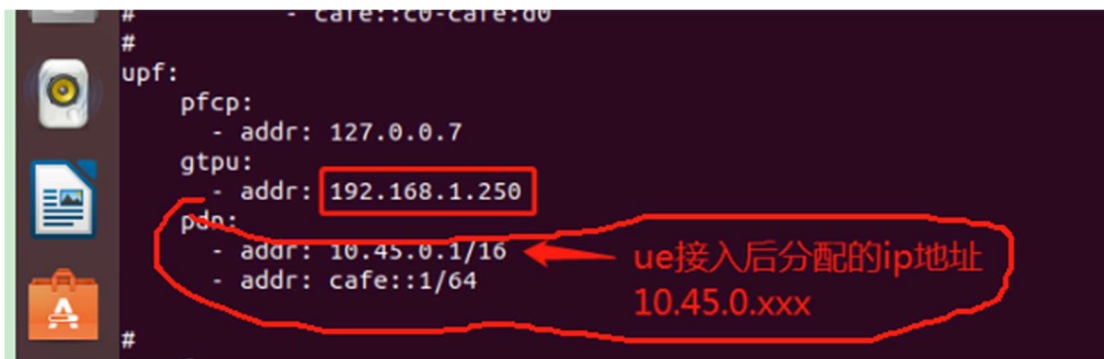
(3) 停止 WebUI, 只需要在 terminal 中 Ctrl+C 即可。

## 2. 给每个 sim 卡分配固定 ip 地址

网页上在 imsi 登记位置，在 ue ipv4 address 位置，输入要为该 sim 卡分配的 ip 地址，保存即可。



注意，分配的 ip 地址不要超过地址池范围。



```
# - care::c0-care:00
#
upf:
  pfcf:
    - addr: 127.0.0.7
  gtpu:
    - addr: 192.168.1.250
  pdp:
    - addr: 10.45.0.1/16
    - addr: cafe::1/64
#
```

## 三、 异常情况

1、安装 Ubuntu18 系统可能会出现花屏，解决办法如下：

## 安装/开机Ubuntu18.04出现花屏

在华硕主板上安装Ubuntu18.04时出现花屏，问题出现好像和显卡有关，我的是RTX 2060S，现将解决方法记录一下

- 安装时花屏：

在安装GRUB页面的时候选择`install Ubuntu`，不要点击，按`e`进入编辑页面，在`quiet splash`后面删除`---`，添加`nomodeset`以支持nvidia显卡，然后`Ctrl+X`进行安装。

- 开机时花屏：

开机后长按`Esc`键进入GRUB引导页面（不能进入换`Shift`试试），选择`advanced options for ubuntu`，按下`e`键进入编辑界面，在`ash $vt_handoff`之间加入`nomodeset`变成`ash nomodeset $vt_handoff`。然后`Ctrl+X`应该就会正常开机了。

- 开机后：

开机后记得修改grub配置文件，不然每次进入都得编辑grub选项

1. 打开grub配置文件

```
sudo gedit /etc/default/grub
```

2. 修改grub配置文件

将 `GRUB_CMDLINE_LINUX_DEFAULT="quiet splash"` 改为

```
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash nomodeset"
```

3. 更新grub

```
sudo update-grub
```

<https://www.cnblogs.com/real010/p/14678546.html>

2、由于某些原因而不得不使用其他更高版本 Ubuntu 系统时，在启动核心网时会出现一条名为：`./bin/5gc-upfd: error while loading shared libraries: libtins.so.3.4: cannot open shared object file: No such file or directory` 的报错。

```
See 'snap info <snapname>' for additional versions.
dlgj@dlgj-Default-string:/usr/local/sgc$ sudo apt-get update
命中:1 http://security.ubuntu.com/ubuntu focal-security InRelease
命中:2 http://mirrors.tuna.tsinghua.edu.cn/ubuntu focal InRelease
命中:3 http://mirrors.tuna.tsinghua.edu.cn/ubuntu focal-updates InRelease
命中:4 http://mirrors.tuna.tsinghua.edu.cn/ubuntu focal-backports InRelease
正在读取软件包列表... 完成
dlgj@dlgj-Default-string:/usr/local/sgc$ sudo bash ./executeSgc.sh c r
**** Input Param: c r
**** run Sgc
**** Run Sgc
./bin/sgc-upfd: error while loading shared libraries: libtins.so.3.4: cannot open shared object file: No such file or directory
SGC daemon v2.3.3+
10/15 18:21:53.245: [app] INFO: Configuration: '/usr/local/sgc/etc/sgc/sqwc.yaml' (./lib/app/ogs-init.c:129)
10/15 18:21:53.245: [app] INFO: File Logging: '/usr/local/sgc/var/log/sgc/sqwc.log' (./lib/app/ogs-init.c:132)
SGC daemon v2.3.3+
10/15 18:21:53.248: [app] INFO: Configuration: '/usr/local/sgc/etc/sgc/mme.yaml' (./lib/app/ogs-init.c:129)
10/15 18:21:53.248: [app] INFO: File Logging: '/usr/local/sgc/var/log/sgc/mme.log' (./lib/app/ogs-init.c:132)
SGC daemon v2.3.3+
10/15 18:21:53.250: [app] INFO: Configuration: '/usr/local/sgc/etc/sgc/sqwu.yaml' (./lib/app/ogs-init.c:129)
10/15 18:21:53.251: [app] INFO: File Logging: '/usr/local/sgc/var/log/sgc/sqwu.log' (./lib/app/ogs-init.c:132)
SGC daemon v2.3.3+
10/15 18:21:53.260: [app] INFO: Configuration: '/usr/local/sgc/etc/sgc/udm.yaml' (./lib/app/ogs-init.c:129)
10/15 18:21:53.267: [app] INFO: File Logging: '/usr/local/sgc/var/log/sgc/udm.log' (./lib/app/ogs-init.c:132)
10/15 18:21:53.268: [sbi] INFO: nhttp2_server() [127.0.0.12]:7777 (./lib/sbi/nhttp2-server.c:145)
SGC daemon v2.3.3+
10/15 18:21:53.269: [app] INFO: Configuration: '/usr/local/sgc/etc/sgc/nssf.yaml' (./lib/app/ogs-init.c:129)
10/15 18:21:53.269: [app] INFO: File Logging: '/usr/local/sgc/var/log/sgc/nssf.log' (./lib/app/ogs-init.c:132)
SGC daemon v2.3.3+
10/15 18:21:53.269: [app] INFO: Configuration: '/usr/local/sgc/etc/sgc/smf.yaml' (./lib/app/ogs-init.c:129)
10/15 18:21:53.269: [sbi] INFO: nhttp2_server() [127.0.0.14]:7777 (./lib/sbi/nhttp2-server.c:145)
10/15 18:21:53.273: [pfcp] INFO: pfcp_server() [127.0.0.6]:8805 (./lib/pfcp/path.c:31)
10/15 18:21:53.273: [gtp] INFO: gtp_server() [127.0.0.6]:2152 (./lib/gtp/path.c:31)
SGC daemon v2.3.3+
```

究其原因，新版本的 Ubuntu 的 libtins 库默认是 4.0 版本，而核心网要求的是 3.4 版本。导致在执行脚本时找不到名为 libtins.so.3.4 的库文件。解决思路也很简单，只需要替换下库文件即可。文件地址：<https://github.com/mfontanini/libtins/releases/tag/v3.4>

- (1) 将压缩包放在任意目录下解压：

```
dlgj@dlgj-Default-string:~/Download$ sudo tar -xvzf libtins-3.4.tar.gz
[sudo] dlgj 的密码:
libtins-3.4/
libtins-3.4/.gitignore
libtins-3.4/.gitmodules
libtins-3.4/.travis.yml
libtins-3.4/AUTHORS
libtins-3.4/CHANGES
libtins-3.4/CMakeLists.txt
libtins-3.4/CONTRIBUTING.md
libtins-3.4/LICENSE
libtins-3.4/README.md
libtins-3.4/THANKS
libtins-3.4/cmake/
libtins-3.4/cmake/Modules/
libtins-3.4/cmake/Modules/CheckCXXFeatures.cmake
libtins-3.4/cmake/Modules/CheckCXXFeatures/
```

- (2) 进入解压完成后的 libtins-3.4 的目录后，执行以下 5 条命令

Mkdir bulid

Cd buile

Sudo cmake ..

Sudo make

Sudo make install

用来编译安装我们解压出来的库文件

```

dlgj@dlgj-Default-string:~/Download/libtins-3.4/build$ sudo make install
[100%] Built target tins
Install the project...
-- Install configuration: "RelWithDebInfo"
-- Installing: /usr/local/lib/pkgconfig/libtins.pc
-- Installing: /usr/local/CMake/libtinsConfig.cmake
-- Installing: /usr/local/CMake/libtinsConfigVersion.cmake
-- Installing: /usr/local/CMake/libtinsTargets.cmake
-- Installing: /usr/local/CMake/libtinsTargets-relwithdebinfo.cmake
-- Installing: /usr/local/include/tins/address_range.h
-- Installing: /usr/local/include/tins/arp.h
-- Installing: /usr/local/include/tins/bootp.h

```

(3) 编译完成后使用命令 `sudo find / -name "libtins.so*"` 来查看原始库文件（libtins.so.4.0）和我们刚编译完的库文件（libtins.so.3.4）分别在哪个路径下。

```

root@dlgj-Default-string:/usr/local/5gc# sudo find / -name "libtins.so*"
/usr/lib/libtins.so.4.0
/usr/lib/libtins.so
/usr/local/lib/libtins.so.3.4
/usr/local/lib/libtins.so
/usr/local/lib/libtins.so.4.5
find: '/run/user/1000/doc': 权限不够
find: '/run/user/1000/gvfs': 权限不够
/home/dlgj/.local/share/Trash/files/libtins-4.5/build/lib/libtins.so
/home/dlgj/.local/share/Trash/files/libtins-4.5/build/lib/libtins.so.4.5
/home/dlgj/Download/libtins-3.4/build/lib/libtins.so.3.4
/home/dlgj/Download/libtins-3.4/build/lib/libtins.so

```

(4) 用命令：`sudo rm -rf /usr/lib/libtins.so*` 删掉原始库文件（4.0），再用命令：`cp /usr/local/lib/libtins* /usr/lib/` 把我们编译好的库文件（3.4）放到库目录下。

```

root@dlgj-Default-string:/usr/lib# sudo rm -rf /usr/lib/libtins.so*
root@dlgj-Default-string:/usr/lib# ls
accountsservice  dpkg          gnupg          linux          os-probes      sasl2          tmpfiles.d
app              eject         gnupg2        linux-boot-probes  os-release     skin           ubiquity
apparmor        emacsen-common  gold-ld       linux-sound-base  packagekit    shotwell       ubuntu-advantage
apt             environment-d  groff         locale         pcnclautls    snapp         ubuntu-release-upgrader
aspell          evolution-data-server  grub         lp_solve      pkgconf       software-properties  udev
bfd-plugins     file          grub-legacy  lsb           pkg-config,multitarch  speech-dispatcher-modules  ufw
binfmt.d        firefox      gst-install-plugins-helper  man-db        policykit-1   sudo          unity-settings-daemon-1.0
bluetooth       firefox-addons  gvfs          mntest86+     pppd          sysctl.d      update-notifier
brlty           firmware     hdparm        mime          pulse-13.99.1  syslinux     upower
cnf-update-db   gcc          libus        modprobe.d    python2.7     SLSLINUX     valgrind
command-not-found  gdm3        init         modules        python3       system        vino
compst-ld       ghostscript   initt        modules-load.d  python3.8     systemd       X11
console-setup   git-core     kernel       netplan       python3.9     sysusers.d   x86_64-linux-gnu
cups            gnome-initial-setup  klibc        networkd-dispatcher  python3       tc            xorg
cups            gnome-session  klibc-zqGENFMIYUJZUBUCGjBo3LnHoMg.so  NetworkManager  recovery-mode  ternInfo     xserver-xorg-video-intel
dbus-1.0        gnome-settings-daemon-3.0  language-selector  openssh       rhythmbox     thunderbird
debug           gnome-shell   libreoffice  os-prober     rygel         thunderbird-addons
root@dlgj-Default-string:/usr/lib# sudo ldconfig
root@dlgj-Default-string:/usr/lib# cp /usr/local/lib/libtins.so* /usr/lib/
root@dlgj-Default-string:/usr/lib# ls
accountsservice  dpkg          gold-ld        linux          os-release     shotwell       ubuntu-release-upgrader
app              emacsen-common  groff         linux-boot-probes  packagekit    snapp         udev
apparmor        environment-d  grub-legacy  linux-sound-base  pcnclautls    software-properties  ubuntu-release-upgrader
apt             evolution-data-server  grub-legacy  locale         pkg-config,multitarch  speech-dispatcher-modules  ufw
aspell          evolution-data-server  gst-install-plugins-helper  lp_solve      pkg-config,multitarch  ssl          unity-settings-daemon-1.0
bfd-plugins     file          hdparm        libus          pm-utils     sudo          update-notifier
binfmt.d        firefox      initt         libus          pppd         sysctl.d      upower
bluetooth       firefox-addons  init          mntest86+     pulse-13.99.1  syslinux     valgrind
brlty           firmware     initt        modules        python2.7     SLSLINUX     vino
cnf-update-db   gcc          initt        modprobe.d    python3       system        X11
command-not-found  gdm3        kernel       modules-load.d  python3.8     sysusers.d   x86_64-linux-gnu
compst-ld       ghostscript   klibc        networkd-dispatcher  python3.9     tc            xorg
console-setup   git-core     klibc        NetworkManager  recovery-mode  ternInfo     xserver-xorg-video-intel
cups            gnome-initial-setup  language-selector  openssh       rhythmbox     thunderbird
cups            gnome-session  libreoffice  os-prober     rygel         thunderbird-addons
dbus-1.0        gnome-settings-daemon-3.0  libreoffice  os-prober     rygel         tmpfiles.d
debug           gnome-shell   libtins.so   os-prober     sasl2        ubiquity

```

(5) 最后启用核心网成功

```

0/16 10:36:10.388: [sctp] INFO: MME initialize...done (./src/mme/app-init.c:33)
0/16 10:36:10.399: [app] INFO: HSS initialize...done (./src/hss/app-init.c:31)
0/16 10:36:10.400: [dian] INFO: CONNECTED TO 'hss.localdomain' (SCTP,soc#7): (./lib/diameter/common/logger.c:108)
0/16 10:36:10.400: [dian] INFO: CONNECTED TO 'mme.localdomain' (SCTP,soc#15): (./lib/diameter/common/logger.c:108)
0/16 10:36:10.451: [gtp] INFO: gtp_server() [127.0.0.4]:2123 (./lib/gtp/path.c:31)
0/16 10:36:10.452: [gtp] INFO: gtp_server() [::1]:2123 (./lib/gtp/path.c:31)
0/16 10:36:10.452: [gtp] INFO: gtp_server() [127.0.0.4]:2152 (./lib/gtp/path.c:31)
0/16 10:36:10.452: [gtp] INFO: gtp_server() [::1]:2152 (./lib/gtp/path.c:31)
0/16 10:36:10.452: [pfcsp] INFO: pfcsp_server() [127.0.0.4]:8805 (./lib/pfcsp/path.c:31)
0/16 10:36:10.452: [pfcsp] INFO: pfcsp_server() [::1]:8805 (./lib/pfcsp/path.c:31)
0/16 10:36:10.452: [pfcsp] INFO: ogs_pfcsp_connect() [127.0.0.7]:8805 (./lib/pfcsp/path.c:60)
0/16 10:36:10.452: [pfcsp] INFO: ogs_pfcsp_connect() [127.0.0.4]:8805 (./lib/pfcsp/path.c:60)
0/16 10:36:10.452: [upf] INFO: PFCSP associated (./src/upf/pfcsp-sm.c:173)
0/16 10:36:10.452: [sbi] INFO: nhttp2_server() [127.0.0.4]:7777 (./lib/sbi/nhttp2-server.c:145)
0/16 10:36:10.452: [app] INFO: SMF initialize...done (./src/smf/app.c:31)
0/16 10:36:10.452: [dian] INFO: CONNECTED TO 'smf.localdomain' (SCTP,soc#11): (./lib/diameter/common/logger.c:108)
0/16 10:36:10.452: [smf] INFO: PFCSP associated (./src/smf/pfcsp-sm.c:174)
0/16 10:36:10.453: [dian] INFO: CONNECTED TO 'pcrf.localdomain' (SCTP,soc#17): (./lib/diameter/common/logger.c:108)
0/16 10:36:10.453: [nrf] INFO: [675a4ae2-8b67-41ef-907b-47069bbe8c29] NF registred [Heartbeat:10s] (./src/nrf/nf-sm.c:193)
0/16 10:36:10.453: [smf] INFO: [675a4ae2-8b67-41ef-907b-47069bbe8c29] NF registred [Heartbeat:10s] (./src/smf/nf-sm.c:200)
0/16 10:36:12.796: [pfcsp] INFO: ogs_pfcsp_connect() [127.0.0.3]:8805 (./lib/pfcsp/path.c:60)
0/16 10:36:12.796: [sgwu] INFO: PFCSP associated (./src/sgwu/pfcsp-sm.c:168)
0/16 10:36:12.796: [sgwc] INFO: PFCSP associated (./src/sgwc/pfcsp-sm.c:172)
*** Check ogstun: ogstun:
*** Check 5gc thread: 15
*** Success to run 5gc

```