# 一、UPF安装 --安装v2.1.5为例

## 1.启动虚机 --使用虚机安装upf时才需要关注

### 1.1 虚机kvm硬盘获取

|  |
| --- |
| 10.216.4.213:/vm/chris/ubuntu-18.04\_raw/ubuntu-18.04.img  root  gsta123 |

### 1.2 虚机生成

|  |
| --- |
| 创建一个6网卡, 8cpu,16G内存的虚机  第一个网卡是管理口, 桥接在br-inner  第二个网卡是n4口, 桥接在br-n4-xxx  第三个网卡是n3口, 桥接在br-n3-xxx  第四个网卡是n6口, 桥接在br-n6-xxx  第五个网卡是n9口, 桥接在br-n9-xxx  第六个网卡是ha口, 桥接在br-ha-xxx  注意: xxx要改为自己的名字  virt-install --virt-type kvm --name xxx-ubuntu-upf --ram 16384 \  --disk /vm/xxx/ubuntu-upf-xxx.img,format=raw \  --network bridge=br-inner \  --cpu host-passthrough \  --vcpus 8 \  --graphics vnc,listen=0.0.0.0 --noautoconsole \  --network bridge=br-n4-xxx \  --network bridge=br-n3-xxx \  --network bridge=br-n6-xxx \  --network bridge=br-n9-xxx \  --network bridge=br-ha-xxx \  --os-type=linux --os-variant=ubuntu18.04 --boot hd |

### 1.3 本地apt-get源

|  |
| --- |
| deb http://10.223.2.236/ubuntu/ bionic main restricted universe multiverse  deb-src http://10.223.2.236/ubuntu/ bionic main restricted universe multiverse  deb http://10.223.2.236/ubuntu/ bionic-security main restricted universe multiverse  deb-src http://10.223.2.236/ubuntu/ bionic-security main restricted universe multiverse  deb http://10.223.2.236/ubuntu/ bionic-updates main restricted universe multiverse  deb-src http://10.223.2.236/ubuntu/ bionic-updates main restricted universe multiverse  deb http://10.223.2.236/ubuntu/ bionic-proposed main restricted universe multiverse  deb-src http://10.223.2.236/ubuntu/ bionic-proposed main restricted universe multiverse  deb http://10.223.2.236/ubuntu/ bionic-backports main restricted universe multiverse  deb-src http://10.223.2.236/ubuntu/ bionic-backports main restricted universe multiverse |

## 2.UPF安装包获取

|  |
| --- |
| wget http://10.222.2.113/upf-v2/rel/pkg/release-v2.1.5/ctc-upf-v2.1.5-e6de191-20211116163333-a145d73a.tar.gz |

## 3.UPF安装

### 3.1安装UPF安装包

|  |
| --- |
| tar -xvf ctc-upf-v2.1.5-e6de191-20211116163333-a145d73a.tar.gz  cd ctc-upf-v2.1.5-e6de191-20211116163333  ./upf-mgr install |

### 3.2设置vpp startup.conf

|  |
| --- |
| unix {  nodaemon  interactive  full-coredump  cli-listen 127.0.0.1:5002  log /var/log/vpp0/vpp.log  exec /etc/ctc-upf/vpp0/init.conf  }  api-trace {  on  }  cpu {  skip-cores 2  workers 5  }  buffers {  ## Increase number of buffers allocated, needed only in scenarios with  ## large number of interfaces and worker threads. Value is per numa node.  ## Default is 16384 (8192 if running unpriviledged)  buffers-per-numa 64000  ## Size of buffer data area  ## Default is 2048  # default data-size 2048  }  dpdk {  dev default {  ## Number of receive queues, enables RSS  ## Default is 1  #num-rx-queues 2  ## Number of transmit queues, Default is equal  ## to number of worker threads or 1 if no workers treads  num-tx-queues 5  ## Number of descriptors in transmit and receive rings  ## increasing or reducing number can impact performance  ## Default is 1024 for both rx and tx  num-rx-desc 2048  num-tx-desc 2048  ## VLAN strip offload mode for interface  ## Default is off  # vlan-strip-offload on  }  dev 0000:00:05.0 {  num-rx-queues 1  workers 1  }  dev 0000:00:06.0 {  num-rx-queues 1  workers 2  }  dev 0000:00:07.0 {  num-rx-queues 1  workers 1  }  uio-driver igb\_uio  }  plugins {  plugin gtpu\_plugin.so {disable}  } |

#### 3.2.1 cpu参数

|  |
| --- |
| skip-cores 2 跳过多少个cpu开始使用, 一般vpp-upu使用numa-1上的cpu  workers 5 worker有多少个, 一般worker有11个, vpp\_wk\_0用作ipv6 rs/ra报文转送,vpp\_wk\_1和vpp\_wk\_2用作收包和rss, vpp\_wk\_3到vpp\_wk\_n用作转发处理 |

#### 3.2.2 dpdk参数

|  |
| --- |
| num-rx-queues 1 收包队列, 一般为1, rss在软件做  num-tx-queues 5 发包队列, 和worker数相同  dev 0000:00:05.0 {} vpp托管网卡,一般托管N3,N6,N9口, pcie号可以通过<dpdk-devbind.py -s>查看 |

## 4.UPF卸载

|  |
| --- |
| upf-mgr uninstall |

## 5.SSH登录UPF

|  |
| --- |
| 10.216.4.80:50163  gsta  gsta123 |

# 二、UPF配置 --配置v2.1.5为例

## 1.登录UPF配置页面

|  |
| --- |
| https://10.216.4.80:8153/  admin  gsta123  第一次登录后要改密码,一般可改为通用密码Gsta@1234  如果页面登录密码重试多次被锁定了, 可以使用<upf-mgr clean>重置所有配置 |

## 2.网口配置

|  |
| --- |
| 配置N3,N6,N9口角色和IP  配置N4口IP  此处配置的IP会作用在vpp接口addr以及角色 |

## 3.运行时配置

|  |
| --- |
| 配置N3,N6,N9,N4,N4U口IP  一般N4和N4U口配置同一个IP即可  此处配置的IP大部分会作用在C面 |

# 三、landslide test server安装

## 1.启动虚机 --landslide必须安装在虚机上

### 1.1 虚机kvm硬盘获取

|  |
| --- |
| 10.216.4.213:/vm/chris/landslide\_raw/  lsGenVm\_1604\_TAS\_TS\_19.8.0.qcow2  landslide-vm.xml  root  gsta123 |

### 1.2 虚机生成

|  |
| --- |
| 创建一个5网卡, 4cpu,16G内存的虚机  第一个网卡是管理口, 桥接在br-inner  第二个网卡是n4口, 桥接在br-n4-xxx  第三个网卡是n3口, 桥接在br-n3-xxx  第四个网卡是n6口, 桥接在br-n6-xxx  第五个网卡是n9口, 桥接在br-n9-xxx  注意: xxx要改为自己的名字  landslide-vm.xml要做如下修改:  1.br-inner的mac要修改, 防止和别人冲突  2.source file= /vm/xxx/landslide\_xxx.qcow2, 改为自己复制的虚机硬盘路径  3.br-n4-xxx, br-n3-xxx, br-n6-xxx, br-n9-xxx要改为自己的名字  virsh define landslide-vm.xml  virsh start 虚机名称 |

## 2.landsilde首次启动配置

|  |
| --- |
| landslide首次安装后没有IP地址, 需要通过vnc连接, 并按下图完成静态IP配置  用户名cfguser 密码cfguser  TS IP Address即仪表管理口地址  TS IP Gateway即仪表管理口gateway  10.217.2.191是仪表的框  TS是仪表的板卡, 建议使用10.216.4.x网段    配置完成后，重启让配置生效即可。 |

# 四、landslide配置

## 1.获取landslide client

|  |
| --- |
| 浏览器打开http://10.217.2.191/，选择Landslide client后，会提示下载一个tasclient.cgi/tasclient.jnlp的文件，点击“保存文件”。    tasclient.cgi/tasclient.jnlp文件通过JAVA web start launcher打开 |

## 2.创建账号

|  |
| --- |
| 打开已保存的tasclient.cgi文件，输入默认用户名密码sms/a1b2c3d4登录。    admin--user--new user |

## 3.添加test server

|  |
| --- |
| (1)选择TSs--添加新的TS--填写TS名称和IP地址--确定  此IP地址为第三部创建的landslide test server虚机管理口地址    (2)TS状态为need license, 需选择“44-Virturl Small”, 点击apply确认，仪表会自动重启。    选择已添加的TS--configure--输入登录密码--进入配置网口界面  密码cfguser    在Ethernet DPDK-1选项里配置网口ETH1/2的IP地址和其他信息，点击apply。  【注意】若未出现如下配置网口界面，需先绑定PCI设备，然后重启Test server。  可参考下图示例： |

## 4.复制别人的测试例

### 4.1复制测试例

|  |
| --- |
| (1)从mingw账号中复制Xn-v4测试例    (2)复制该测试例到chris账号中 |

### 4.2打开复制的测试例

|  |
| --- |
|  |

### 4.3修改测试例绑定的test server

|  |
| --- |
| (1)右键呼出菜单,重新分配test server      (2)点finish |

### 4.4修改测试例绑定的网口

|  |
| --- |
|  |

### 4.5修改对应拓扑的接口和IP

|  |
| --- |
| 双击测试例, 进入测试例配置    (1)配置UPF-N4, UPF-N4U, UPF-N3    (2)配置gNB-N3    (3)配置SMF-N4    (4)配置SMF-N4U    (5)配置UE    (6)配置DN |

# 五、VPP-UPU编译 --编译v2.1.5为例

## 1.代码获取

|  |
| --- |
| git clone ssh://git@10.222.2.122:10022/5gc-dev/upu/vpp-upu.git vpp\_upu  cd vpp\_upu  git checkout r198  git submodule update --init --recursive |

## 2.release版本编译 --会生成docker image,可增量编译,转发性能较好

### 2.1 编译

|  |
| --- |
| cd vpp\_upu  make -f buildimage.mk -j local  最后会生成一个docker vpp-upu的images    提交代码前建议使用release版本验证一下 |

### 2.2 运行

|  |
| --- |
| 确认vpp-upu的image替换后, 使用<upf-mgr restart>重启所有服务 |

### 2.3 编译和打包日志

|  |
| --- |
| cd vpp\_upu  vim .log/local-build.log  vim .log/local-pack.log |

## 3.debug版本编译 --不会生成docker image,可增量编译,转发性能较差

### 3.1 创建debug版本编译和运行容器

|  |
| --- |
| docker run -d --name=vpp-upu-2 -v /var/ctc-upf/dpi:/var/ctc-upf/dpi -e UPU\_HTTPD\_PORT=4444 -e STARTUP=/etc/ctc-upf/vpp0/startup.conf -e UPU\_CFG='/etc/ctc-upf/vpp0/upu\_cfg.json' --cpuset-cpus= --privileged --ipc=host --net=host --log-opt max-size=100m --log-opt max-file=3 -v /etc/timezone:/etc/timezone:ro -v /etc/localtime:/etc/localtime:ro -v /var/crash:/var/crash -v /etc/ctc-upf:/etc/ctc-upf -v /home/xxx:/home/xxx -w /home/xxx -i 10.222.2.113:5000/upf-sdk:latest sleep 100000000  注意: xxx要改为自己目录的名字  创建一个vpp-upu-2的容器  映射/etc/timezone, /etc/localtime, /var/crash, /etc/ctc-upf, /home/xxx目录进入容器  10.222.2.113:5000是UPF项目发布用的docker镜像仓库, 如非必要, 不要往上推送docker镜像 |

### 3.2 编译

|  |
| --- |
| docker exec -it vpp-upu-2 bash  cd vpp-upu  make build |

### 3.3 运行

|  |
| --- |
| docker exec -it vpp-upu-2 bash  cd vpp-upu  make debug  run –c etc/ctc-upf/vpp0/startup.conf  注:运行前要注意UPF原来的vpp是否仍然在运行, 可通过修改如下文件禁止原vpp启动  vim /usr/sbin/upf-mgr    修改后使用<upf-mgr restart>重启所有服务, 此时只会拉起vpp-upu容器, vpp不再被拉起 |

# 六、编译vpp-upu-base镜像

## 1.docker仓库

|  |
| --- |
| http://10.222.2.113:5001/repo/tags/vpp-upu-base |

## 2.docker环境配置

|  |
| --- |
| vim /etc/docker/daemon.json  {  "insecure-registries" : ["http://10.223.2.80:5000", "http://kubernetes-master:5000","http://10.222.2.113:5000"] ,  "max-concurrent-downloads": 10  }  service docker restart  此步是把docker仓库加入进docker环境 |

## 3.代码获取

|  |
| --- |
| git clone ssh://git@10.222.2.122:10022/5gc-dev/cicd/upf-buildimage.git |

## 4.基础docker镜像获取

|  |
| --- |
| docker pull 10.222.2.113:5000/ubuntu:18.04  docker tag 10.222.2.113:5000/ubuntu:18.04 ubuntu:18.04  docker pull 10.222.2.113:5000/upf-sdk:v2.0.0  docker tag 10.222.2.113:5000/upf-sdk:v2.0.0 upf-sdk:build  注意: sdk的tag要与产品组确认,一般使用最新的tag, 不建议使用latest tag |

## 5.编译

### 5.1 修改dockerfile

|  |
| --- |
| vim tools/base-img/Dockerfile-vpp-upu  -FROM upf-sdk:v0.0.17 as sdk  +FROM upf-sdk:build as sdk  注意: build tag是第3步创建的 |

### 5.2 docker build

|  |
| --- |
| docker build -t vpp-upu-base:version -f tools/base-img/Dockerfile-vpp-upu .  10.222.2.113:5000是UPF项目发布用的docker镜像仓库, 如非必要, 不要往上推送docker镜像 |