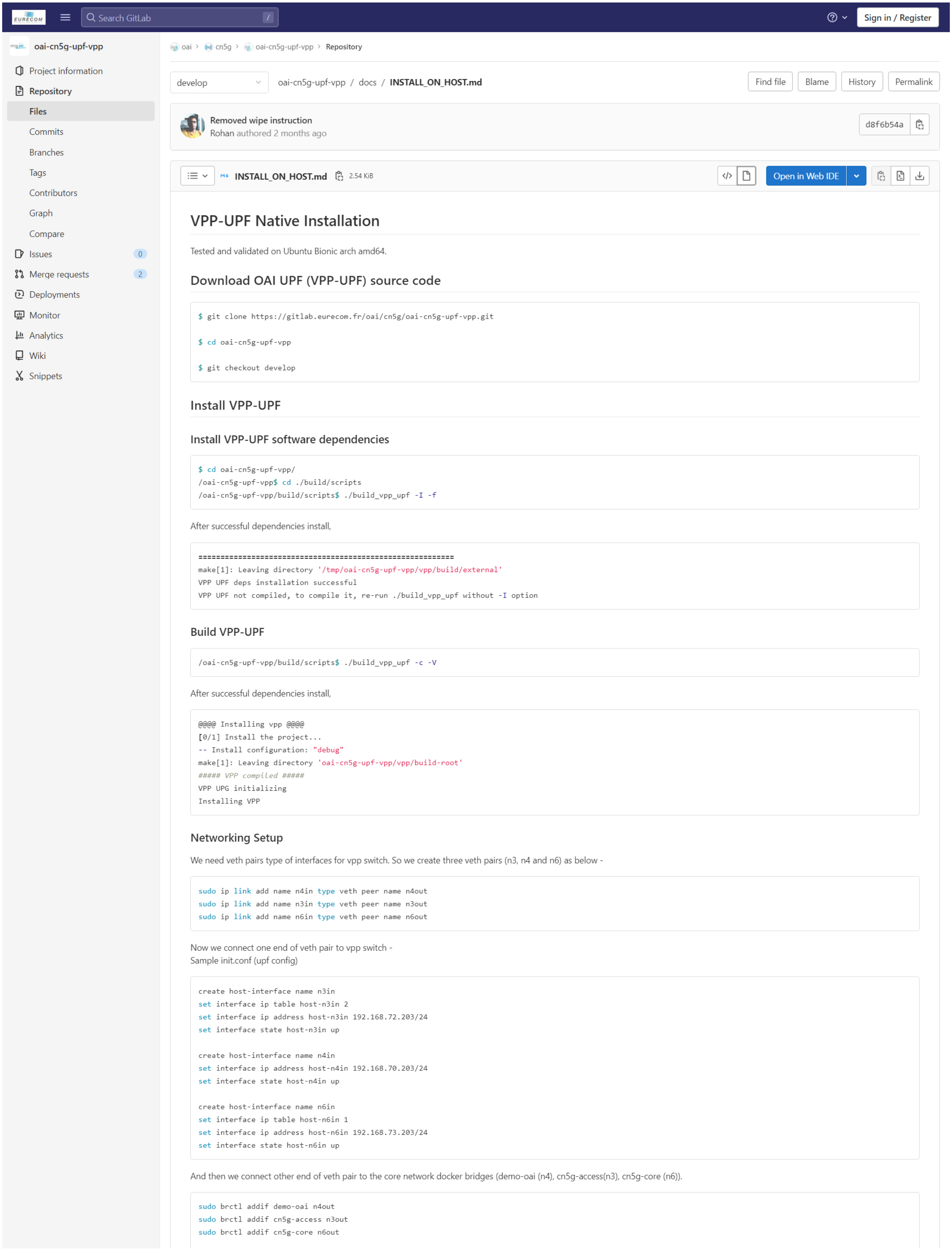

VPP-UPF Native Installation

Tested and validated on Ubuntu Bionic arch amd64.

Download OAI UPF (VPP-UPF) source code

```
$ git clone https://gitlab.eurecom.fr/oai/cn5g/oai-cn5g-upf-vpp.git

$ cd oai-cn5g-upf-vpp

$ git checkout develop
```

Install VPP-UPF

Install VPP-UPF software dependencies

```
$ cd oai-cn5g-upf-vpp/
/oai-cn5g-upf-vpp$ cd ./build/scripts
/oai-cn5g-upf-vpp/build/scripts$ ./build_vpp_upf -I -f
```

After successful dependencies install,

```
==========================================================
make[1]: Leaving directory '/tmp/oai-cn5g-upf-vpp/vpp/build/external'
VPP UPF deps installation successful
VPP UPF not compiled, to compile it, re-run ./build_vpp_upf without -I option
```

Build VPP-UPF

```
/oai-cn5g-upf-vpp/build/scripts$ ./build_vpp_upf -c -V
```

After successful dependencies install,

```
@@@@ Installing vpp @@@@
[0/1] Install the project...
-- Install configuration: "debug"
make[1]: Leaving directory 'oai-cn5g-upf-vpp/vpp/build-root'
##### VPP compiled #####
VPP UPG initializing
Installing VPP
```

Networking Setup

We need veth pairs type of interfaces for vpp switch. So we create three veth pairs (n3, n4 and n6) as below -

```
sudo ip link add name n4in type veth peer name n4out
sudo ip link add name n3in type veth peer name n3out
sudo ip link add name n6in type veth peer name n6out
```

Now we connect one end of veth pair to vpp switch -
Sample init.conf (upf config)

```
create host-interface name n3in
set interface ip table host-n3in 2
set interface ip address host-n3in 192.168.72.203/24
set interface state host-n3in up

create host-interface name n4in
set interface ip address host-n4in 192.168.70.203/24
set interface state host-n4in up

create host-interface name n6in
set interface ip table host-n6in 1
set interface ip address host-n6in 192.168.73.203/24
set interface state host-n6in up
```

And then we connect other end of veth pair to the core network docker bridges (demo-oai (n4), cn5g-access(n3), cn5g-core (n6)).

```
sudo brctl addif demo-oai n4out
sudo brctl addif cn5g-access n3out
sudo brctl addif cn5g-core n6out
sudo ip link set dev n4out up
sudo ip link set dev n6out up
sudo ip link set dev n3out up
```

The bridge used here are created by docker. For more details checkout docker-compose file used to set up core network.

You can create your own bridge or linux namespace for networking purpose, if you dont want to use docker bridges.

« Collapse sidebar