RARE/FreeRouter-101 [#004] - "My name is TOFINO ... INTEL/BAREFOOT TOFINO and I speak P4 too !"

In the previous article #003 "Are you P4 compliant ?" we exposed a setup where RARE/freeRouter was controlling BMv2 P4 dataplane called ***simple_switch_grpc***. In this article we replace the open source BMv2 target by a commercial virtual target provided by INTEL/BAREFOOT. As a side note, we will show that this setup can be integrated with real networks. (with inherent software limitations)

Requirement	
• Basic Linux/Unix knowledge • Basic networking knowledge	

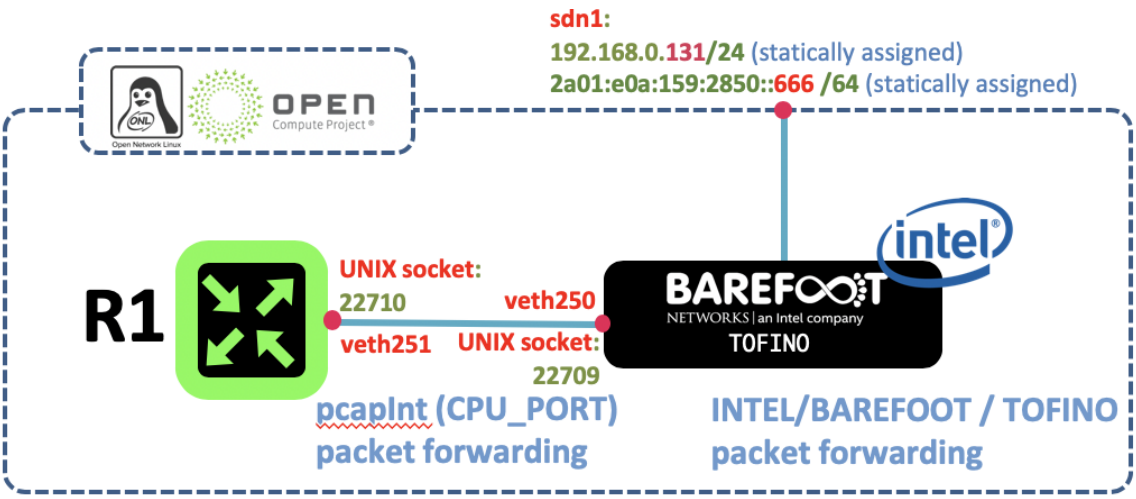
Overview

I'm repeating the core message from #003: For those who are not familiar with data plane programming and especially with P4, ***"P4 is a domain-specific programming language for specifying the behaviour of the dataplanes of network-forwarding elements."*** (from p4.org) in short it helps you to write a "***program specifying how a switch processes packets"***.

Article objective

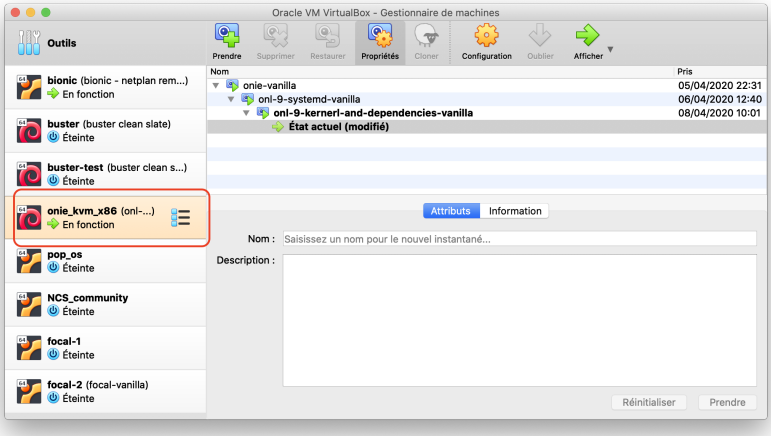
In this article we'll using freeRouter setup deployed in #003 and replace ***bmv2/simple_switch_grpc*** providing freeRouter P4Lang's dataplane by ***INTEL BAREFOOT/bf_switchd***. Actually the effective dataplane is ensured by INTEL/BAREFOOT virtual ***bf_switchd model*** running RARE P4 program called: ***bf_router.p4***.

Diagram

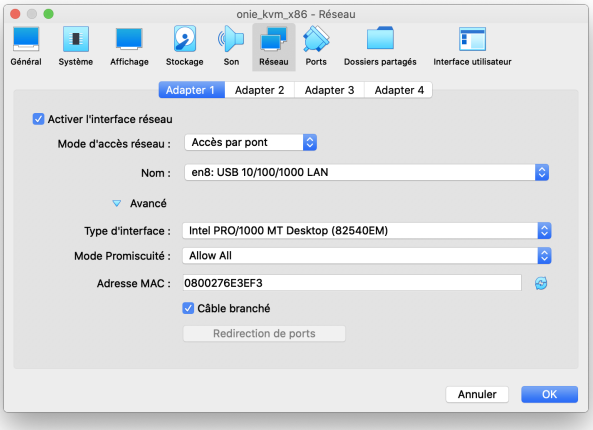


[#004] - Cookbook

In our example we will use the OpenNetworkLinux KVM image (ONL9) this is the recommended build from INTEL/BAREFOOT for SDE-9.2.0.



and we add a network interface bridged to our laptop RJ45 connection.




```
mkdir -p ~/freeRouter/bin ~/freeRouter/lib ~/freeRouter/etc ~/freeRouter/log
cd ~/freeRouter/lib
wget http://freerouter.nop.hu/rtr.jar
```

Update & Upgrade system

```
tree freeRouter
freeRouter
 bin   # binary files
 etc   # configuration files
 lib   # library files
 log   # log files
```

get freeRouter net-tools tarball

```
wget freerouter.nop.hu/rtr.tar
```

Install build tools

```
tar xvf rtr.tar -C ~/freeRouter/bin/
```

> For those you would like to rebuild these binaries you can find the compilation shell script in freeRouter cloned git repository in: ***~/freeRouter/src/native/c.sh***

In that section, you'll need to get access to INTEL/BAREFOOT Software Development Environment. For Research & Academia institution, you can apply here in order to become a FASTER member and access to INTEL/BAREFOOT resources. You can find here, a document installing INTEL/BAREFOOT SDE on ONL for a WEDGE100BF32X system. In our case, we are setting up the following environment:

- ONL9 as VM guest with kernel 8192 Mb of RAM and 2 vCPU
- SDE 9.2.0
- VirtualBox is running on MACOSX host

Just for the sake of example, SDE 9.2.0 is installed in root home directory:

SDE installation environment

```
export SDE=/root/bf-sde-9.2.0
export SDE_INSTALL=/root/bf-sde-9.2.0/install
export PATH=$PATH:$SDE_INSTALL/bin:$SDE/tools
```

TOFINO RARE bitbucket is a private repository. It is currently being reworked in order to make it public as per INTEL/BAREFOOT decision to make P4 code related to TOFINO architecture public. (It is thus inaccessible for now but will be opened to the public soon.)

Clone RARE code from repository

```
cd ~/
git clone https://bitbucket.software.geant.org/scm/rare/rare.git
```

compile RARE bf_router.p4

```
p4_build.sh -I /root/rare/p4src/ -DHAVE_MPLS /root/rare/p4src/bf_router.p4
Using SDE          /root/bf-sde-9.2.0
Using SDE_INSTALL /root/bf-sde-9.2.0/install
Using SDE version bf-sde-9.2.0

OS Name: Ubuntu 18.04.4 LTS
This system has 8GB of RAM and 1 CPU(s)
Parallelization:  Recommended: -j1   Actual: -j1

Compiling for p4_16/tna
P4 compiler path:    /root/bf-sde-9.2.0/install/bin/p4c
P4 compiler version: 9.2.0 (SHA: 639d9ec) (p4c-based)
Build Dir: /root/bf-sde-9.2.0/build/p4-build/bf_router
 Logs Dir: /root/bf-sde-9.2.0/logs/p4-build/bf_router

  Building bf_router        CLEAR CONFIGURE MAKE INSTALL ... DONE
```

FreeRouter uses 2 configuration files in order to run, let's write these configuration files for R1 in ***~/freeRouter/etc***

freeRouter hardware configuration file: tna-freerouter-hw.txt

```
int eth0 eth 0000.1111.00fb 127.0.0.1 22710 127.0.0.1 22709
tcp2vrf 2323 v1 23
tcp2vrf 9080 v1 9080
```

freeRouter software configuration file: tna-freerouter-sw.txt

```
hostname tna-freerouter
buggy
!
!
vrf definition v1
 exit
!
interface ethernet0
 description freerouter@P4_CPU_PORT[veth251]
 no shutdown
 no log-link-change
 exit
!
interface sdn1
 description freerouter@sdn1[enp0s3]
 mtu 9000
 macaddr 0072.3e18.1b6f
 vrf forwarding v1
 ipv4 address 192.168.0.131 255.255.255.0
 ipv6 address 2a01:e0a:159:2850::666 ffff:ffff:ffff:ffff::
 ipv6 enable
 no shutdown
 no log-link-change
 exit
!
!
!
!
!
!
!
!
!
!
!
!
!
!
server telnet tel
 security protocol telnet
 no exec authorization
 no login authentication
 vrf v1
 exit
!
server p4lang p4
 export-vrf v1 1
 export-port sdn1 0 10
 interconnect ethernet0
 vrf v1
 exit
!
client tcp-checksum transmit
!
end
```

Setup bf_switchd dataplane communication channel via veth pair and interface adjustment (disable IPv6 at VM guest level, MTU 10240, disable TCP offload etc.)

```
echo 1 > /proc/sys/net/ipv6/conf/all/disable_ipv6
echo 1 > /proc/sys/net/ipv6/conf/default/disable_ipv6

ip link add veth251 type veth peer name veth250
ip link set veth250  up
ip link set veth251  up

ifconfig enp0s3 promisc
ifconfig veth250 promisc
ifconfig veth251 promisc

ip link set dev veth250 up mtu 10240
ip link set dev veth251 up mtu 10240
ip link set dev enp0s3 up mtu 10240
export TOE_OPTIONS="rx tx sg tso ufo gso gro lro rxvlan txvlan rxhash"

for TOE_OPTION in $TOE_OPTIONS; do
    /sbin/ethtool --offload veth250 "$TOE_OPTION" off &> /dev/null
    /sbin/ethtool --offload veth251 "$TOE_OPTION" off &> /dev/null
    /sbin/ethtool --offload enp0s3 "$TOE_OPTION" off &> /dev/null
done
```

freeRouter launch with supplied tna-freerouter-hw.txt and tna-freerouter-sw.txt with a console prompt

```
java -jar lib/rtr.jar routersc etc/p4-freerouter-hw.txt etc/p4-freerouter-sw.txt
info cfg.cfgInit.doInit:cfgInit.java:556 booting
info cfg.cfgInit.doInit:cfgInit.java:680 initializing hardware
info cfg.cfgInit.doInit:cfgInit.java:687 applying defaults
info cfg.cfgInit.doInit:cfgInit.java:695 applying configuration
info cfg.cfgInit.doInit:cfgInit.java:721 done
welcome
line ready
freerouter#
```

launch freeRouter pcapInt in order to stitch control plane and P4 bf_switchd dataplane communication

```
cd ~/freeRouter/bin
./pcapInt.bin veth251 22709 127.0.0.1 22710 127.0.0.1
binded to local port 127.0.0.1 22709.
will send to 127.0.0.1 22710.
pcap version: libpcap version 1.8.1
opening interface veth251 with pcap1.x api
serving others
>
```

Create bf_switchd RARE running environement

```
mkdir -p ~/rare-run/etc ~/rare-run/logs ~/rare-run/mibs ~/rare-run/snmp
```

create a custom ports.json file for bf_switchd model

```
cat ~/rare-run/etc/ports.json
{
    "PortToIf" : [
        { "device_port" :  0, "if" : "enp0s3" },
        { "device_port" : 64, "if" : "veth250" }
    ]
}
```

run TOFINO model in quiet mode with bf_router as program and log file (if any) should be in ~/rare-run/logs

```
cd $SDE
./run_tofino_model.sh -p bf_router -f ~/rare-run/etc/ports.json --log-dir ~/rare-run/logs/ -q
```

Run bf_switchd (logs will be in ~/rare-run/logs)

```
cd ~/rare-run/logs
 $SDE/run_switchd.sh -p bf_router
```

Launch RARE bf_forwarder.p4 (BfRuntime GRPC based interface)

```
cd ~/rare/bfrt_python/
./bf_forwarder.py --ifmibs-dir ~/rare-run/mibs/ --ifindex ~/rare-run/snmp/ifindex
bf_forwarder.py running on: MODEL
GRPC_ADDRESS: 127.0.0.1:50052
P4_NAME: bf_router
CLIENT_ID: 0
Subscribe attempt #1
Subscribe response received 0
Received bf_router on GetForwarding
Binding with p4_name bf_router
Binding with p4_name bf_router successful!!
BfForwarder - loop
  Clearing Table pipe.ig_ctl.ig_ctl_mpls.tbl_mpls_fib
  Clearing Table pipe.ig_ctl.ig_ctl_acl_in.tbl_ipv6_acl
BfIfSnmpClient - main
BfIfSnmpClient - No active ports
  Clearing Table pipe.ig_ctl.ig_ctl_ipv4.tbl_ipv4_fib_host
  Clearing Table pipe.ig_ctl.ig_ctl_copp.tbl_ipv6_copp
  Clearing Table pipe.ig_ctl.ig_ctl_acl_in.tbl_ipv4_acl
  Clearing Table pipe.ig_ctl.ig_ctl_ipv6.tbl_ipv6_fib_host
  Clearing Table pipe.ig_ctl.ig_ctl_mpls.tbl_mpls_fib_decap
  Clearing Table pipe.ig_ctl.ig_ctl_nexthop.tbl_nexthop
  Clearing Table pipe.ig_ctl.ig_ctl_vlan_out.tbl_vlan_out
  Clearing Table pipe.ig_ctl.ig_ctl_vlan_in.tbl_vlan_in
  Clearing Table pipe.ig_ctl.ig_ctl_acl_out.tbl_ipv6_acl
  Clearing Table pipe.ig_ctl.ig_ctl_ipv4.tbl_ipv4_fib_lpm
  Clearing Table pipe.ig_ctl.ig_ctl_acl_out.tbl_ipv4_acl
  Clearing Table pipe.ig_ctl.ig_ctl_vrf.tbl_vrf
  Clearing Table pipe.ig_ctl.ig_ctl_copp.tbl_ipv4_copp
  Clearing Table pipe.ig_ctl.ig_ctl_ipv6.tbl_ipv6_fib_lpm
  Clearing Table pipe.ig_ctl.ig_ctl_bridge.tbl_bridge_target
  Clearing Table pipe.ig_ctl.ig_ctl_bridge.tbl_bridge_learn
Bundle specific clearing: (Order matters)
  Clearing Bundle Table pipe.ig_ctl.ig_ctl_bundle.tbl_nexthop_bundle
  Clearing Bundle Table pipe.ig_ctl.ig_ctl_bundle.ase_bundle
  Clearing Bundle Table pipe.ig_ctl.ig_ctl_bundle.apr_bundle
BfForwarder - Main
BfForwarder - Entering message loop
rx: ['myaddr4_add', '224.0.0.0/4', '0', '1', '\n']
BfIfStatus - main
BfIfStatus - No active ports
```

```
rx: ['myaddr4_add', '255.255.255.255/32', '0', '1', '\n']
BfSubIfCounter - main
BfSubIfCounter - No active ports
rx: ['myaddr6_add', 'ff00::/8', '0', '1', '\n']
rx: ['myaddr4_add', '192.168.0.0/24', '-1', '1', '\n']
rx: ['myaddr4_add', '192.168.0.131/32', '-1', '1', '\n']
rx: ['myaddr6_add', '2a01:e0a:159:2850::/64', '-1', '1', '\n']
rx: ['myaddr6_add', '2a01:e0a:159:2850::666/128', '-1', '1', '\n']
rx: ['myaddr6_add', 'fe80::/64', '-1', '1', '\n']
rx: ['mylabel4_add', '186286', '1', '\n']
rx: ['mylabel6_add', '842368', '1', '\n']
rx: ['state', '0', '1', '10', '\n']
rx: ['mtu', '0', '9000', '\n']
rx: ['portvrf_add', '0', '1', '\n']
rx: ['neigh6_add', '20989', 'fe80::224:d4ff:fea0:cd3', '00:24:d4:a0:0c:d3', '1', '00:72:3e:18:1b:6f', '0', '\n']
BfIfSnmpClient - added stats for port 0
rx: ['keepalive', '\n']
rx: ['neigh4_add', '29777', '192.168.0.254', '00:24:d4:a0:0c:d3', '1', '00:72:3e:18:1b:6f', '0', '\n']
rx: ['keepalive', '\n']
rx: ['neigh6_add', '25745', 'fe80::bc6a:83ad:7897:8461', '00:13:46:3c:a9:4f', '1', '00:72:3e:18:1b:6f', '0',
'\n']
rx: ['keepalive', '\n']
rx: ['neigh6_add', '41106', 'fe80::e23f:49ff:fe6d:1899', 'e0:3f:49:6d:18:99', '1', '00:72:3e:18:1b:6f', '0',
'\n']
rx: ['keepalive', '\n']
rx: ['neigh6_add', '35111', '2a01:e0a:159:2850:e23f:49ff:fe6d:1899', 'e0:3f:49:6d:18:99', '1', '00:72:3e:18:1b:
6f', '0', '\n']
rx: ['keepalive', '\n']
rx: ['neigh6_del', '25745', 'fe80::bc6a:83ad:7897:8461', '00:13:46:3c:a9:4f', '1', '00:72:3e:18:1b:6f', '0',
'\n']
rx: ['keepalive', '\n']
rx: ['neigh6_add', '20371', 'fe80::bc6a:83ad:7897:8461', '00:13:46:3c:a9:4f', '1', '00:72:3e:18:1b:6f', '0',
'\n']
rx: ['keepalive', '\n']
...
rx: ['keepalive', '\n']
rx: ['neigh4_add', '34182', '192.168.0.62', 'e0:3f:49:6d:18:99', '1', '00:72:3e:18:1b:6f', '0', '\n']
...
```

Verification

FreeRouter telnet access from Virtualbox VM guest via port 2323

```
telnet localhost 2323
Trying ::1...
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
welcome
line ready
tna-freerouter#
```

freerouter running configuration

```
tna-freerouter#term len 0
tna-freerouter#sh run
hostname tna-freerouter
buggy
!
!
vrf definition v1
 exit
!
interface ethernet0
 description freerouter@P4_CPU_PORT[veth251]
 no shutdown
 no log-link-change
 exit
!
interface sdn1
 description freerouter@sdn1[enp0s9]
 mtu 9000
 macaddr 0072.3e18.1b6f
 vrf forwarding v1
 ipv4 address 192.168.0.131 255.255.255.0
 ipv6 address 2a01:e0a:159:2850::666 ffff:ffff:ffff:ffff::
 ipv6 enable
 no shutdown
 no log-link-change
 exit
!
!
!
!
!
!
!
!
!
!
!
!
!
server telnet tel
 security protocol telnet
 no exec authorization
 no login authentication
 vrf v1
 exit
!
server p4lang p4
 export-vrf v1 1
 export-port sdn1 0 10
 interconnect ethernet0
 vrf v1
 exit
!
client tcp-checksum transmit
!
end
```

Check control plane is communicating with bf_switchd p4 dataplane

```
tna-freerouter#sh int sum
interface  state  tx     rx         drop
ethernet0  up     89955  128007451  0
sdn1       up     87291  127572417  0
```

Ping IPv4 from freerouter -> LAN router gateway

```
tna-freerouter#ping 192.168.0.254 /vrf v1 /repeat 11111
pinging 192.168.0.254, src=null, cnt=11111, len=64, tim=1000, ttl=255, tos=0, sweep=false
..!........!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!*
result=95%, recv/sent/lost=197/207/10, rtt min/avg/max/total=27/54/645/20764
```

The output above indicates that there are packet losses. This is due to the fact fact as soon as bf_switch port 0 is bridged to enp0s3 on the local area netwrok it receives a lot of packet. These packet have to be process by bf_switchd increasing the processing delay as all the packet received via enp0s3 has to be queued and processed by bf_switchd model.

IPv4 arp check

```
tna-freerouter#sh ipv4 arp sdn1
mac             address        time      static
e03f.496d.1899  192.168.0.62   00:05:27  false    <----- Host server
9ceb.e8d5.2c51  192.168.0.77   00:05:27  false    <----- VM guest bridged IP
0024.d4a0.0cd3  192.168.0.254  00:01:27  false    <----- LAN gateway
```

Ping IPv6 from freerouter -> Host server and SSH connection test

```
tna-freerouter#..1:e0a:159:2850:e23f:49ff:fe6d:1899 /vrf v1 /repeat 111111
pinging 2a01:e0a:159:2850:e23f:49ff:fe6d:1899, src=null, cnt=111111, len=64, tim=1000, ttl=255, tos=0,
sweep=false
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!*
result=100%, recv/sent/lost=89/89/0, rtt min/avg/max/total=30/50/600/4467
```

IPv6 neighbor discovery check

```
tna-freerouter#show ipv6 neighbors sdn1
mac             address                                time      static  router
0024.d4a0.0cd3  2a01:e0a:159:2850::1                   00:00:26  false   false    <----- LAN gateway
e03f.496d.1899  2a01:e0a:159:2850:e23f:49ff:fe6d:1899  00:03:26  false   false    <----- Host server
0024.d4a0.0cd3  fe80::224:d4ff:fea0:cd3                00:01:26  false   false
e03f.496d.1899  fe80::e23f:49ff:fe6d:1899              00:02:26  false   false
```

Initiate IPv4 ssh from freerouter -> LAN router gateway

```
tna-freerouter#ssh 192.168.0.62 /vrf v1 /user my-nas
 - connecting to 192.168.0.62 22
password: *******

 - securing connection

Last login: Fri Jul  3 10:57:02 2020 from 192.168.0.66
FreeBSD 11.3-RELEASE-p9 (FreeNAS.amd64) #0 r325575+588899735f7(HEAD): Mon Jun  1 15:04:31 EDT 2020

        FreeNAS (c) 2009-2020, The FreeNAS Development Team
        All rights reserved.
        FreeNAS is released under the modified BSD license.

        For more information, documentation, help or support, go here:
        http://freenas.org
Welcome to FreeNAS
MY-NAS%
```

Initiate IPv6 ssh from freerouter -> LAN router gateway

```
tna-freerouter#..:e0a:159:2850:e23f:49ff:fe6d:1899 /vrf v1 /user my-nas
 - connecting to 2a01:e0a:159:2850:e23f:49ff:fe6d:1899 22
password: *******

 - securing connection

Last login: Mon Jul  6 11:05:31 2020 from 192.168.0.131
FreeBSD 11.3-RELEASE-p9 (FreeNAS.amd64) #0 r325575+588899735f7(HEAD): Mon Jun  1 15:04:31 EDT 2020

        FreeNAS (c) 2009-2020, The FreeNAS Development Team
        All rights reserved.
        FreeNAS is released under the modified BSD license.

        For more information, documentation, help or support, go here:
        http://freenas.org
Welcome to FreeNAS
MY-NAS%
```

Conclusion

In this article you:

- had a demonstration of how to integrate freeRouter into a local area network (Similar to article #002)
- However instead of using ***bmv2*** we used a INTEL/BAREFOOT P4 dataplane called: ***TOFINO*** (bf_switchd)
- TOFINO ***bf_switchd*** target is running RARE ***bf_router.p4***
- communication between freeRouter control plane and TOFINO is ensured by ***pcapInt*** via veth pair [veth250 - veth251]
- This communication is possible via RARE ***bf_forwarder.py*** based on GRPC P4Lang BfRuntime python binding
- In this example the TOFINO bf_switchd P4 virtual switch model has only 1 dataplane interface that is bound to enp0s3 VM interface exposed to the local network as a bridged interface

[#004] RARE/FreeRouter-101 - key take-away

- FreeRouter is using UNIX socket in order to forward packet dedicated to control plane + dataplane communication.

This essential paradigm is used to ensure communication between freeRouter and TOFINO bf_switchd P4 dataplane. It is ensured by ***pcapInt*** binary from freeRouter net-tools that will bind freeRouter socket (veth251@locathost:22710) to a virtual network interface (veth250@localhost:22709) connected to CPU_PORT 64.

- freeRouter control plane and dataplane communication is enabled by RARE ***bf_forwarder.py***

bf_forwarder.py is a simple python script based on GRPC client BfRuntime python library.

- freeRouter is the control plane for TOFINO bf_switchd P4 dataplane

freeRouter is doing all the control plane route computation and write/modify/remove message entry via BfRuntime so that P4 entries are created /modified/removed accordingly from P4 tables

- TOFINO bf_switchd virtual model target

While TOFINO bf_switchd **virtual model** target is a very good choice for packet processing algorithm validation on TOFINO platform, the virtual model is not a target for production use. We will see in next articles how we can reach **TREMENDOUS** traffic throughput required by Internet Service Provider's use cases. Indeed, while with the model we can validate algorithm accuracy, traffic transfers achieved have a very low throughput. (I could barely make my setup described above working)

- TOFINO bf_switchd hardware target

In a subsequent article we will demonstrate how we can create with RARE/freeRouter/TOFINO TNA architecture, a service provider/carrier grade router that technically is able to switch **3.3 Tbps** of traffic (line rate) using EdgeCore WEDGE100BF32X hardware switch.

TOFINO family most powerful Programmable Switching ASIC has the ability to switch 6.5 Tbps traffic throughput, our **WEDGE100BF 32X** switches are powered by the ASIC's little brother that is able to handle **3.3 Tbps** line rate traffic throughput.