Barefoot 交换机使用手册

目录

1. 系统用户名和密码 2

2. 执行 P4 程序 3

1) 首先查看 SDE 路径是否存在 3

2) 加载 bf 驱动 3

3) 运行 P4 程序 3

4) 重启 P4 程序 4

3. 常用命令 5

4. 端口管理 6

1) 进入 user cli 6

2) 添加端口 6

3) 删除端口 6

4) 端口自动协商功能 6

5) Enable 端口 6

6) 操作多个端口 7

7) 查看单个端口详细信息 7

8) 查看光模块信息 7

9) 查看端口设备号 8

5. 流表管理 9

5.1 查看流表 9

5.2 下发流表 9

6. 源码解析 12

6.1 头部定义源码解析 12

6.2 数据包解封装源码解析 13

6.3 basic_test 源码解析 14

7. table 详解 17

7.1 table 创建 17

7.2 table 的 read 说明 17

8. 附录 1：P4 程序编译 18

9. 附录 2：screen 使用 19

1. 系统用户名和密码

OPEN BMC：
主板管理控制系统，相当于传统 BIOS
波特率 9600
用户名和密码：root/0penBmc

ONL：
网络操作系统，里面装了 SDE，可以运行 P4 程序。
用户名和密码：root/onl

2. 执行 P4 程序

1) 首先查看 SDE 路径是否存在

env

查看是否存在：

SDE=/root/bf-sde-8.x.x

SDE_INSTALL=/root/bf-sde-8.x.x/install

PATH=/root/bf-sde-8.x.x/install/bin:$PATH

若没有，则：

cd /root/bf-sde-8.x.x

#source set_sde_bash

或者：

export SDE=/root/bf-sde-8.x.x

export SDE_INSTALL=$SDE/install

export PATH=$SDE_INSTALL/bin:$PATH

```
root@localhost:~# env
SHELL=/bin/bash
TERM=linux
HUSHLOGIN=FALSE
SDE_INSTALL=/root/bf-sde-8.2.0/install
USER=root
SDE=/root/bf-sde-8.2.0
MAIL=/var/mail/root
PATH=/root/bf-sde-8.2.0/install/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/lib/platform-config/current/onl/bin:/lib/platform-
config/current/onl/sbin:/lib/platform-config/current/onl/lib/bin:/lib/platform-config/current/onl/lib/sbin
PWD=/root
LANG=en_US.UTF-8
SHLVL=1
HOME=/root
LOGNAME=root
_=/usr/bin/env
root@localhost:~#
```

2) 加载 bf 驱动

#./bf-sde-8.x.x/install/bin/bf_kdrv_mod_load $SDE_INSATLL

检验加载驱动是否成功：

#ls /dev/bf0

```
root@localhost:~# ./bf-sde-8.2.0/install/bin/bf_kdrv_mod_load $SDE_INSTALL
root@localhost:~#
root@localhost:~# ls /dev/bf0
/dev/bf0
```

3) 运行 P4 程序

./ bf-sde-8.x.x/run_switchd.sh -p <p4_name>

如：`./bf-sde-8.x.x/run_switchd.sh -p switch`

```
  diag:
  mavericks diag:
bf_switchd: library /root/bf-sde-8.2.0/install/lib/tofinopd/basic_switching/libpd.so loaded
bf_switchd: library /root/bf-sde-8.2.0/install/lib/tofinopd/basic_switching/libpdthrift.so loaded
bf_switchd: library /root/bf-sde-8.2.0/install/lib/libpltfm_mgr.so loaded
bf_switchd: agent[0] initialized
Tcl server started..
Tcl server: listen socket created
Tcl server: bind done on port 8008, listening...
Tcl server: waiting for incoming connections...
Health monitor started
Operational mode set to ASIC
Initialized the device types using platforms infra API
ASIC detected at PCI /sys/class/bf/bf0/device
ASIC pci device id is 16
Starting PD-API RPC server on port 9090
bf_switchd: drivers initialized
/
bf_switchd: dev_id 0 initialized

bf_switchd: initialized 1 devices
bf_switchd: libpd initialized for basic_switching
Adding Thrift service for P4 program basic_switching to server
bf_switchd: libpdthrift initialized for basic_switching
Adding Thrift service for bf-platforms to server
bf_switchd: thrift initialized for agent : 0
bf_switchd: spawning cli server thread
bf_switchd: spawning driver shell
bf_switchd: server started - listening on port 9999

        ****************************************
        *      WARNING: Authorised Access Only     *
        ****************************************

bfshell>
```

4) 重启 P4 程序

ps -ax | grep switchd

#sudo kill <proces id>

#./run_switchd.sh -p switch

3. 常用命令

bfshell>?　　//查看可执行命令

bfshell.pm>..　　//退出到上一级

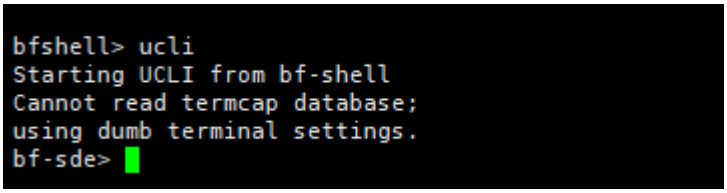
```
bfshell>
  access              Access hardware registers
  cint                C Interpreter
  exit                Exit this CLI session
  help                Display an overview of the CLI syntax
  pd-basic-switching  pd_basic_switching Related Commands
  pipemgr             Pipe manager commands
  quit                Exit this CLI session
  ucli                UCLI commands
  version             Display the SDE version

bfshell>
```

bfshell.pm>end　　//退出最初目录

4. 端口管理

1) 进入 user cli

bfshell>ucli



2) 添加端口

bf-sde>pm port-add <conn_id/chnl> <speed> <fec>

- pm：port manager，端口管理
- **conn_id/chnl**:端口号（如：1/0、1/1、1/2、1/3）

Speed	QSFP lane 0	QSFP lane 1	QSFP lane 2	QSFP lane 3
10G	Y	Y	Y	Y
25G	Y	Y	Y	Y
40G	Y			
50G	Y		Y	
100G	Y			

- **speed**: 可以设置为1G, 10G, 25G, 40G, 50G, 或 100G
- **fec**: 必须设置为**NONE**, **FC**, 或 **RS**, **NONE** 是不开启**FEC**。

3) 删除端口

bf-sde >pm port-del <conn_id/chnl>

4) 端口自动协商功能

注意：需要在 port-add 之后，port-enb 之前使用，配置才生效

bf-sde.pm> an-set -/- 0 (由 SDE 决定开启还是关闭，默认)

bf-sde.pm> an-set -/- 1 (开启 autonegotiation)

bf-sde.pm> an-set -/- 2 (关闭 autonegotiation)

5) Enable端口

bf-sde >pm `port-enb 1/0`

```
bf-sde> pm port-add 1/0 100G NONE
bf-sde> pm port-en
bf-sde> pm port-enb 1/0
bf-sde> sho
bf-sde> show
error: Module dvm does not have a log registered.
bf-sde> pm sho
bf-sde> pm show
-----+----+---+----+------+----+---+---+---+----------------+----------------+-
PORT |MAC |D_P|P/PT|SPEED |FEC |RDY|ADM|OPR|FRAMES RX       |FRAMES TX       |E
-----+----+---+----+------+----+---+---+---+----------------+----------------+-
1/0  |23/0|128|3/ 0| 100G |NONE|NO |ENB|DWN|               0|               0|
bf-sde>
```

6) 操作多个端口

用符号“-”代替数字，可以操作多个端口，如下：

添加多个端口：

port-add 2/- 10g NONE　　//添加 2 号端口所有子端口，速率 10g

port-add -/- 40g NONE　　//添加所有端口，速率 40g

```
bf-sde.pm> port-add 2/- 10g NONE
bf-sde.pm>
```

删除多个端口：

port-del 2/-　　//删除 2 号端口所有子端口

port-del -/-　　//删除所有端口

```
bf-sde.pm> port-del 2/-
bf-sde.pm>
```

enable多个端口：

port-enb 2/-　　//enable2号端口所有子端口

```
bf-sde.pm> port-enb 2/-
bf-sde.pm>
```

7) 查看单个端口详细信息

bf-sde.pm> show -p 1/0 -d

```
bf-sde.pm> show -p 1/0 -d
============================================================
            1/0 : Port Identifier
             NO : is port internal
           23/0 : MAC
            128 : Dev Port
           3/ 0 : Pipe/Port
           100G : Speed
             RS : FEC
            YES : Ready for Bring Up
            YES : Autoneg eligibility
   FORCE_ENABLE : AN policy set
            ENB : Admin State
             UP : Operational Status
              0 : FramesReceivedOK
              0 : FramesReceivedAll
              0 : FramesReceivedwithFCSError
              0 : FrameswithanyError
              0 : OctetsReceivedinGoodFrames
              0 : OctetsReceived
              0 : FramesReceivedwithUnicastAddresses
```

8) 查看光模块信息

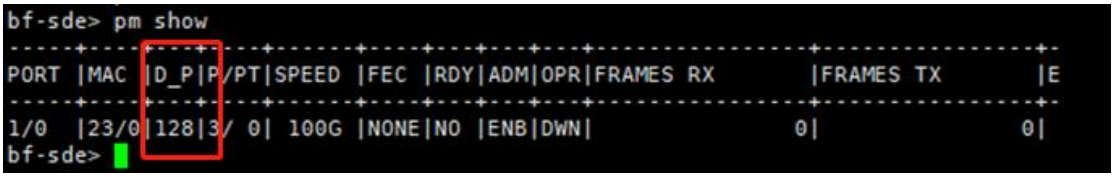
bf-sde> bf_pltfm

bf-sde.bf_pltfm> dump-info 1/0

9) 查看端口设备号

bf-sde>pm show

其中“D_P”为device port，设备端口号，用于p4程序使用，如流表的操作



5. 流表管理

进入 pd（program dependent）界面，根据 p4 程序生成，可以查看各个流表项和下发流表，指导交换机怎么处理数据包。具体功能根据 P4 程序来实现。
SDE 自带了部分 P4 程序，目录在 bf-sde-8.x.x\pkgsrc\p4-examples\programs

以下 p4 程序介绍：
basic_switching.p4　　//根据目的 mac 地址进行指定端口转发
switch-test.p4　　//根据源端口进行指定端口转发

可根据需求对程序进行修改，再重新编译，即可使交换机拥有相应的功能。
下面的案例以 basic_switching.p4 为例：

5.1 查看流表

命令如下：
bfshell> pd-switch-test　　//进入 pd 界面
pd-switch-test:0>dump_table forward　　//查看名称为 forward 的表格里的表项
//有哪些表格根据 P4 程序来决定

```
bfshell> pd-switch-test
pd-switch-test:0> dum
dump_profile dump_table
pd-switch-test:0> dump_table
acl     forward
pd-switch-test:0> dump_table forward
-------------
entry_hdl: 1
match_spec_ig_intr_md_ingress_port: 133 (0x85)
action_spec_name: p4_pd_switch_test_set_egr
action_egress_spec: 155 (0x9b)

-------------
pd-switch-test:0>
```

5.2 下发流表

bfshell> pd-switch-test　　//进入 pd 界面
pd-switch-test:0>pd forward add_entry set_egr ig_intr_md_ingress_port 133 action_egress_spec 155
添加一条流表项到名为 forward 的 table，执行的操作为：设备端口号 133 进来的数据包从设

备端口号 155 转发出去。设备端口号查询见上一章节《端口管理》

命令详解：

- pd-switch-test:0>pd　　//按“?”健，可查看 table 名称

```
pd-switch-test:0> pd
  acl
  forward
  init
```

- pd-switch-test:0> pd forward　　//对 forward 的 table 进行操作，按“?”健,可查看可进行的操作。

```
pd-switch-test:0> pd forward
  add_entry                       Synonym for table_add_with
  del_entry                       Synonym for table_delete
  get_entry
  get_entry_count
  get_first_entry_handle
  get_next_entry_handles
  match_spec_to_entry_hdl
  mod_entry                       Synonym for table_modify_with
  set_default_action
  table_add_with
  table_delete
  table_get_default_entry_handle
  table_modify_with
  table_reset_default_entry
```

- pd-switch-test:0> pd forward add_entry　//进行添加流表的操作，按“?”健,可查看行为

nop：进行数据包丢弃

set_egr：进行数据包转发

```
pd-switch-test:0> pd forward add_entry
  nop
  set_egr
```

- pd-switch-test:0> pd forward add_entry set_egr　　//按“?”健,可匹配相应数据包

  ig_intr_md_ingress_port：匹配数据包进来的端口号

  注：可匹配其他参数，如源或目的 mac 地址等，根据 P4 程序来生成

```
pd-switch-test:0> pd forward add_entry set_egr
  sess_hdl                : Integer type: i32 (optional, default=1)
  device_id               dev_tgt.: Integer type: i32 (optional, default = your selection when opening pdcli, 0 if you didnt do this)
  dev_pipe_id             dev_tgt.: Integer type: i16 (optional, default=65535)
  ig_intr_md_ingress_port  match_spec.: Integer type: i16 (required)
```

- pd forward add_entry set_egr ig_intr_md_ingress_port 133 action_egress_spec 155

  //action_egress_spec：数据包转发出去的端口号

```
bfshell> pd-switch-test
pd-switch-test:0> pd forward add_entry
nop     set_egr
pd-switch-test:0> pd forward add_entry set_egr
sess_hdl               device_id              dev_pipe_id
ig_intr_md_ingress_port
pd-switch-test:0> pd forward add_entry set_egr ig_intr_md_ingress_port 133 action_egress_spec 155
entry_hdl: 1 (0x1)
pd-switch-test:0>
```

6. 源码解析

6.1 头部定义源码解析

headers.p4：

```
/*
Copyright 2013-present Barefoot Networks, Inc.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
*/

// Template headers.p4 file for basic_switching
// Edit this file as needed for your P4 program

// Here's an ethernet and ipv4 header to get started.

header_type ethernet_t {        //以太网头部定义
    fields {
        dstAddr : 48;
        srcAddr : 48;
        etherType : 16;
    }
}

header ethernet_t ethernet;

header_type vlan_tag_t {
    fields {
        pcp : 3;
        cfi : 1;
        vid : 12;
        etherType : 16;
```

```
    }
}

header ipv4_t ipv4;
header_type ipv4_t {
    fields {
        version : 4;
        ihl : 4;
        diffserv : 8;
        totalLen : 16;
        identification : 16;
        flags : 3;
        fragOffset : 13;
        ttl : 8;
        protocol : 8;
        hdrChecksum : 16;
        srcAddr : 32;
        dstAddr : 32;
        options : *; // Variable length options
    }
    length : ihl * 4;
    max_length : 60;
}

header vlan_tag_t vlan;
```

6.2 数据包解封装源码解析

```
/*
Copyright 2013-present Barefoot Networks, Inc.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
*/
```

```
16.
17. // Template parser.p4 file for basic_switching
18. // Edit this file as needed for your P4 program
19.
20. // This parses an ethernet and ipv4 header
21.
22. parser start {
23.     return parse_ethernet;   //一开始先解析数据包以太网头部
24. }
25.
26. #define ETHERTYPE_VLAN 0x8100
27. #define ETHERTYPE_IPV4 0x0800
28.
29. parser parse_ethernet {
30.     extract(ethernet);
31.     return select(latest.etherType) {
32.         ETHERTYPE_VLAN : parse_vlan;     //解析 vlan
33.         ETHERTYPE_IPV4 : parse_ipv4;      //解析 ipv4
34.         default: ingress;
35.     }
36. }
37.
38. parser parse_vlan {
39.     extract(vlan);
40.     return select(latest.etherType) {
41. //        ETHERTYPE_VLAN : parse_vlan;
42. //        ETHERTYPE_IPV4 : parse_ipv4;
43.         default: ingress;
44.     }
45. }
46.
47. parser ipv4 {
48.     extract(ipv4);
49.     return ingress; // All done with parsing; start matching
50. }
```

6.3 basic_test 源码解析

```
1.  /*
2.  Copyright 2013-present Barefoot Networks, Inc.
3.
4.  Licensed under the Apache License, Version 2.0 (the "License");
```

```
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
*/

// This is P4 sample source for switch_test

#include "includes/headers.p4"
#include "includes/parser.p4"
#include <tofino/intrinsic_metadata.p4>
#include <tofino/constants.p4>

action set_egr(egress_) {
    modify_field(ig_intr_md_for_tm.ucast_egress_port, egress_spec);
}
action nop() {
}

action _drop() {
    drop();
}

table forward {           //添加 table，名称为 forward，用于数据包转发
    reads {
        ig_intr_md.ingress_port: exact;       //匹配哪个端口进来的数据包,“exact”表示精确匹配。
    }
    actions {
        set_egr; nop;       //执行转发操作
    }
}

```

```
43. table acl {                              //添加 table，名称为 acl，用作数据包丢弃
44.     reads {
45.         ethernet.dstAddr : ternary;     //匹配数据包的目的 mac 地址
46.         ethernet.srcAddr : ternary;     //匹配数据包的源 mac 地址
47.     }
48.     actions {
49.         nop;
50.         _drop;              //执行丢弃的动作
51.     }
52. }
53.
54. control ingress {       //数据包进去时，匹配 forward 的 table
55.     apply(forward);
56. }
57.
58. control egress {       //数据包出去时，匹配 acl 的 table
59.     apply(acl);
60. }
```

7. table 详解

7.1 table 创建

```
table 名称 {
    read {
        匹配字段：exact/ternary/lpm
        }
     action{
        动作
     }
}
```

7.2 table 的 read 说明

匹配类型如下：

匹配类型	描述
excat	精确匹配。
ternary	三重匹配，动作-匹配表的每个表项都有一个掩码，将掩码和字段值进行逻辑与运算，再执行匹配。为了避免导致多条表项匹配成功，每条表项都需要设定一个优先级。
lpm	这是三重匹配的一种特殊情况，当多个表项匹配成功时，选择掩码最长的最为最高优先级进行匹配。
index	字段值作为表项索引。
range	表项中确定一个范围，字段值在此范围内皆能成功匹配。
valid	仅用于包头字段匹配，表项值只能为 true/false。

8. 附录 1：P4 程序编译

若对需要执行的 P4 程序进行修改，需要进行重新编译，编译的步骤如下：

1） 将 P4-name.p4 程序文件拷贝到 bf-sde-8.x.x/pkgsrc/p4-examples/programs/“P4-name”/文件夹中，若无和 P4 程序相同名称的文件则创建一个。

cd ./bf-sde-8.x.x/pkgsrc/p4-examples/programs/

mkdir XXX //创建名为“XXX”的文件夹，与 P4 程序相同名称

使用 cp 命令进行拷贝

注：若在原 P4 程序文件进行修改，则跳过此步骤。

2） 进入到 p4-build 文件夹里

```
root@localhost:~# cd bf-sde-8.2.0/pkgsrc/p4-build/
root@localhost:~/bf-sde-8.2.0/pkgsrc/p4-build#
```

3） 进行编译

对 switch_test.p4 程序进行编译：

#./autogen.sh

#./configure --prefix=$SDE_INSTALL --with-tofino P4_NAME=switch_test P4_PATH=/root/bf-sde-4.1.1.15/pkgsrc/p4-examples/programs/switch_test/switch_test.p4 --enable-thrift

#make

#make install

完成编译后，生成相应的 pd 文件。

注：configure 详细参数如下：

./configure --prefix=$SDE_INSTALL \

[--with-bmv2/ tofino/ tofinobm] \选择 bmv2、tofino、tofinobm 三种模式选择一个，由于运行在 tofino 物理芯片上，故选择 tofino 即可

P4_NAME=<name of P4 program> \ P4 程序的名称

P4_PATH=<absolute path to P4 program> \ p4 程序绝对路径

4） 为 P4 程序生成.conf 文件

注：若在原 P4 程序文件进行修改，则跳过此步骤。

cd ./bf-sde-8.x.x/pkgsrc/p4-examples/

./configure --prefix=$SDE_INSTALL

#make

#make install

执行完后，可在./bf-sde-8.x.x/install/share/p4/targets/tofino/文件里看到与 P4 程序相同名称的.conf 文件。

```
conf localhost:~# ls ./bf-sde-8.2.0/install/share/p4/targets/tofino/switch_test.c
./bf-sde-8.2.0/install/share/p4/targets/tofino/switch_test.conf
```

9. 附录 2：screen 使用

screen –S switch　　//创建名为“switch”的新会话

按 CTRL a+d　　//将会话挂起，会话里运行的程序会在后台继续执行，退出当前会话
如：

```
  non_default_port_ppgs: 0
  Agent[0]: /root/bf-sde-8.2.0/install/lib/libpltfm_mgr.so
  diag:
  mavericks diag:
bf_switchd: library /root/bf-sde-8.2.0/install/lib/tofinopd/switch_test/libpd.so loaded
bf_switchd: library /root/bf-sde-8.2.0/install/lib/tofinopd/switch_test/libpdthrift.so loaded
bf_switchd: library /root/bf-sde-8.2.0/install/lib/libpltfm_mgr.so loaded
bf_switchd: agent[0] initialized
Tcl server started..
Tcl server: listen socket created
Tcl server: bind done on port 8008, listening...
Tcl server: waiting for incoming connections...
Health monitor started
Operational mode set to ASIC
Initialized the device types using platforms infra API
ASIC detected at PCI /sys/class/bf/bf0/device
ASIC pci device id is 16
Starting PD-API RPC server on port 9090
bf_switBits 55-60 of /proc/PID/pagemap entries are about to stop being page-shift some time soon
ails.
chd: drivers initialized

[detached from 1336.switch]
root@localhost:~#
```

其中 1336 为该会话的进程号，switch 为该会话的名称

screen –ls　　//查看所有的会话

```
root@localhost:~# screen -ls
There is a screen on:
        1336.switch     (08/15/2018 07:23:25 AM)        (Detached)
1 Socket in /var/run/screen/S-root.
```

kill 会话进程号　　//关闭该会话

screen –r 会话名称/进程号　　//进入会话，进入到相应的会话名称，或者进程号的会话
如：

```
root@localhost:~# screen -r switch
```