情報ネットワーク学演習２
レポート課題３

学籍番号 33E16010 提出者 佐竹 幸大

課題内容（パッチパネルの機能拡張）

パッチパネルに機能を追加しよう。

授業で説明したパッチの追加と削除以外に、以下の機能をパッチパネルに追加してください。

1.ポートのミラーリング
2.パッチとポートミラーリングの一覧
それぞれ patch_panel のサブコマンドとして実装してください。

なお 1 と 2 以外にも機能を追加した人には、ボーナス点を加点します。

解答内容

ポートのミラーリング

プログラム説明

ポートのミラーリングは、スイッチやルータの機能の一つである。
あるポートの送受信データを同時に別ポートから送受信する機能である。
ミラーリングを作成するためのサブコマンドとして、create_mirror を定義した。
create_mirror コマンドは以下のように用いる

./bin/patch_panel create_mirror (datapath ID) (モニターポート番号) (ミラーポート番号)

create_mirror は、./bin/patch_panel に追記する形で行った。定義箇所は以下である。

```
  desc 'Creates a mirror patch'
  arg_name 'dpid monitor_port mirror_port'
  command :create_mirror do |c|
    c.desc 'Location to find socket files'
    c.flag [:S, :socket_dir], default_value: Trema::DEFAULT_SOCKET_DIR

    c.action do |_global_options, options, args|
      dpid = args[0].hex
      monitor_port = args[1].to_i
      mirror_port = args[2].to_i
      Trema.trema_process('PatchPanel', options[:socket_dir]).controller.
        create_mirror_patch(dpid, monitor_port, mirror_port)
    end
  end
```

上述のcreate_mirror_patchメソッドは、 ./lib/patch_panel.rb内で定義している。定義箇所を以下に示す。

```
def create_mirror_patch(dpid, monitor_port, mirror_port)
    add_mirror_flow_entries dpid, monitor_port, mirror_port
    @mirror_patch[dpid] << [monitor_port, mirror_port]
  end
```

ミラーパッチ設定にミラーパッチ情報を追加する。

また、上述の add_mirror_flow_entries は以下のように定義した。

ここでは、あるミラーパッチを書き込む。

```
def add_mirror_flow_entries(dpid, monitor_port, mirror_port)
    source_port = nil
    @patch[dpid].each do |port_a, port_b|
      if port_a == monitor_port then source_port = port_b
      elsif port_b == monitor_port then source_port = port_a
      end
    end
    if source_port == nil then return false
    end
    send_flow_mod_delete(dpid, match: Match.new(in_port:
source_port))
    send_flow_mod_delete(dpid, match: Match.new(in_port:
monitor_port))
    send_flow_mod_add(dpid,
                      match: Match.new(in_port: source_port),
                      actions: [
                        SendOutPort.new(monitor_port),
                        SendOutPort.new(mirror_port)
                      ])
    send_flow_mod_add(dpid,
                      match: Match.new(in_port: monitor_port),
                      actions: [
                        SendOutPort.new(source_port),
                        SendOutPort.new(mirror_port)
                      ])
    return true
  end
```

動作確認

動作確認を以下のように行った。

```
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ bundle exec ./bin/trema
show_stats host1
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ bundle exec ./bin/trema
show_stats host2
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ bundle exec ./bin/trema
show_stats host3
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ bundle
exec ./bin/patch_panel create 0xabc 1 2
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ bundle exec ./bin/trema
send_packet --source host1 --dest host2
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ bundle exec ./bin/trema
show_stats host1
Packets sent:
   192.168.0.1 -> 192.168.0.2 = 1 packet
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ trema show_stats host2
Packets received:
   192.168.0.1 -> 192.168.0.2 = 1 packet
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ trema show_stats host3
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ bundle
exec ./bin/patch_panel create_mirror 0xabc 1 3
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ trema send_packets --
source host1 --dest host2
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ trema show_stats host1
Packets sent:
   192.168.0.1 -> 192.168.0.2 = 2 packets
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ trema show_stats host2
Packets received:
   192.168.0.1 -> 192.168.0.2 = 2 packet
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ trema show_stats host3
Packets received:
   192.168.0.1 -> 192.168.0.2 = 1 packet
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ trema send_packets --
source host2 --dest host1
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ trema show_stats host1
Packets sent:
   192.168.0.1 -> 192.168.0.2 = 2 packets
Packets received:
   192.168.0.2 -> 192.168.0.1 = 1 packet
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ trema show_stats host2
```

```
Packets sent:
   192.168.0.2 -> 192.168.0.1 = 1 packet
Packets received:
   192.168.0.1 -> 192.168.0.2 = 2 packets
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ trema show_stats host3
Packets received:
   192.168.0.1 -> 192.168.0.2 = 1 packet
   192.168.0.2 -> 192.168.0.1 = 1 packet
```

上記の作業により、所望の動作をしていることが確認できた。まず、パッチ操作の後、パケット送信を行うことによって、host1及びhost2が接続できたことが読み取れる。その後、ミラーリング設定を行い、再びパケットの送信によって、host1に関するパケットがhost3にミラーリングされていることが読み取れた。

パッチとポートミラーリングの一覧

パッチ及びポートミラーリングの状態を示すコマンドとして、 ./bin/patch_panel に list を定義した。定義箇所は以下である。

```
desc 'Prints patch and mirror patch'
  arg_name 'dpid patchlist patch mirror_patch'
  command :list do |c|
    c.desc 'Location to find socket files'
    c.flag [:S, :socket_dir], default_value:
Trema::DEFAULT_SOCKET_DIR

    c.action do |_global_options, options, args|
      dpid = args[0].hex
      patchlist = Trema.trema_process('PatchPanel',
options[:socket_dir]).controller.
        list_patch(dpid)
      @patch = patchlist[0]
```

```
      @mirror_patch = patchlist[1]
      @patch[dpid].each do |port_a, port_b|
        print("Port ", port_a, " is connected to port ",
port_b, "¥n")
      end
      @mirror_patch[dpid].each do |monitor_port, mirror_port|
        print("Monitor port:", monitor_port, ", Mirror port:",
mirror_port, "¥n")
      end
    end
  end
```

上述のlist_patchメソッドは、 ./lib/patch_panel.rb で定義されている。定義箇所は以下である。

```
def list_patch(dpid)
    list = Array.new()
    list << @patch
    list << @mirror_patch
    return list
  end
```

---

lib/patch_panel.rbのstartハンドラを以下のように変更した。

```
def start(_args)
    @patch = Hash.new { |hash,key| hash[key]=[] }
    @mirror_patch = Hash.new { |hash,key| hash[key]=[] }
    logger.info 'PatchPanel started.'
  end
```

動作確認

---

ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ bundle exec ./bin/patch_panel create 0xabc 1 2ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ bundle exec ./bin/patch_panel create_mirror 0xabc 1 3

```
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$ bundle
exec ./bin/patch_panel list 0xabc
Port 1 is connected to port 2
Monitor port:1, Mirror port:3
ensyuu2@ensyuu2-VirtualBox:~/patch-panel-ksatake$
```

上記の動作確認により、以下のことが確認できる。具体的には、最後の list コマンドの出力結果から、以下のことが確認できるので、正しく動作していると言える。

・パッチ操作で port1 及び port2 が接続されたこと
・port1 のミラーが port3 に生成されている。

参考文献

デビッド・トーマス+アンドリュー・ハント(2001)「プログラミング Ruby」ピアソン・エデュケーション. テキスト :6 章“インテリジェントなパッチパネル“

謝辞

プログラム及び本レポートの作成にあたり、田中達也氏のものを大いに参考にさせて頂いたため、ここに感謝の意を示す。