

# 情報ネットワーク学演習 2

## 課題: マルチフローテーブルを読む

33E16022 村上 遼

平成 28 年 11 月 8 日

### 1 課題内容

ルータのコマンドラインインタフェース (CLI) を作ろう。次の操作ができるコマンドを作ろう。

- ルーティングテーブルの表示
- ルーティングテーブルエントリの追加と削除
- ルータのインタフェース一覧の表示
- そのほか、あると便利な機能

コントローラを操作するコマンドの作りかたは、第 3 回パッチパネルで作った `patch_panel` コマンドを参考にしてください。

### 2 課題に対する回答

今回の課題は成元君の回答を元に考えた。

#### 2.1 ルーティングテーブルの表示

まず、`RoutingTable` のクラスにルーティングテーブル内の内容を表示するメソッド `getDB` を追加した。リスト 1 の `getDB` メソッドでは、変数 `@db` を変換して新たなルーティングテーブルを `ret` を生成している。これは、変数 `@db` では `Hash` の `key` が `int` 型で表現されており、表示の際に不都合が生じるためである。

リスト 1: `getDB` メソッド

```
1 def getDB()  
2   ret = Array.new()  
3   @db.each do |each|  
4     tmp = Hash.new()  
5     each.each do |key, value|  
6       tmp[IPAddress.new(key).to_s] = value.to_s  
7     end  
8     ret << tmp  
9   end  
10  return ret  
11 end
```

次に、`SimpleRouter` のクラスにルーティングテーブルの内容を表示するメソッド `print_routing_table()` を追加した。リスト 2 の `print_routing_table` メソッドでは、`routing_table` クラスに追加した `getDB` メソッドを用いてルーティングテーブルの内容を表示している。

## リスト 2: print\_routing\_table メソッド

```
1 def print_routing_table()
2   return @routing_table.getDB
3 end
```

最後に SimpleRouter クラスを操作するコマンドを作成するために、bin ディレクトリ下に router\_control ファイルを追加した。router\_control ファイルでは、ルーティングテーブルの表示はリスト 3 のように実装されている。ルーティングテーブルを表示する場合はコンソールに ./bin/router\_control printRT と入力する。

## リスト 3: printRT コマンドの実装

```
1 desc 'Print routing table'
2 arg_name '@routing_table'
3 command :printRT do |c|
4   c.desc 'Location to find socket files'
5   c.flag [:S, :socket_dir], default_value: Trema::DEFAULT_SOCKET_DIR
6
7   c.action do |_global_options, options, args|
8     @routing_table = Trema.trema_process('SimpleRouter', options[:socket_dir]).
9       controller.print_routing_table()
10    print "\"destination/netmask\" \"next hop\"\n"
11    @routing_table.each do |each|
12      mask = @routing_table.find_index each
13      each.each do |key, value|
14        print sprintf("%-21s %s\n", key+"/"+mask.to_s, value)
15      end
16    end
17  end
18 end
```

## 2.2 ルーティングテーブルエントリの追加

RoutingTable クラスの add メソッドを用いて、ルーティングテーブルエントリの追加を行う。そのために、SimpleRouter クラスに add\_entry\_for\_routing\_table メソッドを追加する。リスト 4 の add\_entry\_for\_routing\_table メソッドでは、add メソッドに必要な引数 option を作成して変数 @routing\_table の add メソッドを呼び出している。

## リスト 4: add\_entry\_for\_routing\_table メソッド

```
1 def add_entry_for_routing_table(destination_ip, netmask_length, next_hop)
2   options = {:destination => destination_ip, :netmask_length => netmask_length, :next_hop => next_hop}
3   @routing_table.add(options)
4 end
```

router\_control ファイルでは、ルーティングテーブルエントリの追加はリスト 5 のように実装されている。ルーティングテーブルエントリを追加する場合はコンソールに ./bin/router\_control addRT [宛先 IP] [ネットマスク長] [送信先 IP アドレス] と入力する。

## リスト 5: addRT コマンドの実装

```
1 desc 'Add routing table entry'
2 arg_name 'destination_ip netmask_length next_hop'
3 command :addRT do |c|
4   c.desc 'Location to find socket files'
5   c.flag [:S, :socket_dir], default_value: Trema::DEFAULT_SOCKET_DIR
6
7   c.action do |_global_options, options, args|
8     destination_ip = args[0]
9     netmask_length = args[1].to_i
10    next_hop = args[2]
11    Trema.trema_process('SimpleRouter', options[:socket_dir]).controller.
12      add_entry_for_routing_table(destination_ip, netmask_length, next_hop)
13  end
14 end
```

## 2.3 ルーティングテーブルエントリの削除

RoutingTable クラスの add メソッドを元に delete メソッドを作成した (リスト 6)。また, SimpleRouter クラスの add\_entry\_for\_routing\_table メソッドを元にした delete\_entry\_from\_routing\_table クラスを追加した (リスト 7)。

リスト 6: delete メソッド

```
1 def delete(options)
2   netmask_length = options.fetch(:netmask_length)
3   prefix = IPv4Address.new(options.fetch(:destination)).mask(netmask_length)
4   @db[netmask_length].delete(prefix.to_i)
5 end
```

リスト 7: delete\_entry\_from\_routing\_table メソッド

```
1 def delete_entry_from_routing_table(destination_ip, netmask_length)
2   options = {:destination => destination_ip, :netmask_length => netmask_length}
3   @routing_table.delete(options)
4 end
```

router\_control ファイルでは, ルーティングテーブルエントリの削除はリスト 8 のように実装されている。ルーティングテーブルエントリを削除する場合はコンソールに ./bin/router\_control delRT [宛先 IP] [ネットマスク長] と入力する。

リスト 8: delRT コマンドの実装

```
1 desc 'Delete routing table entry'
2 arg_name 'destination_ip netmask_length'
3 command :delRT do |c|
4   c.desc 'Location to find socket files'
5   c.flag [:S, :socket_dir], default_value: Trema::DEFAULT_SOCKET_DIR
6
7   c.action do |_global_options, options, args|
8     destination_ip = args[0]
9     netmask_length = args[1].to_i
10    Trema.trema_process('SimpleRouter', options[:socket_dir]).controller.
11      delete_entry_from_routing_table(destination_ip, netmask_length)
12  end
13 end
```

## 2.4 インターフェース一覧の表示

SimpleRouter クラスに接続しているインターフェースを表示する print\_interface メソッドを追加した。リスト 9 の print\_interface メソッドでは, 接続している各インターフェースについてポート番号, IP アドレス, MAC アドレスを変数 ret に記録している。

リスト 9: print\_interface メソッド

```
1 def print_interface()
2   ret = Array.new()
3   Interface.all.each do |each|
4     ret << {:port_number => each.port_number, :mac_address => each.mac_address.to_s,
5           :ip_address => each.ip_address.value.to_s, :netmask_length => each.netmask_length}
6   end
7   return ret
8 end
```

router\_control ファイルでは, インターフェース一覧の表示はリスト 10 のように実装されている。ルーティングテーブルエントリを削除する場合はコンソールに ./bin/router\_control printInterface と入力する。

リスト 10: printInterface コマンドの実装

```
1 desc 'Print interface'
2 arg_name 'interfaces'
3 command :printInterface do |c|
4   c.desc 'Location to find socket files'
5   c.flag [:S, :socket_dir], default_value: Trema::DEFAULT_SOCKET_DIR
```

```

6
7   c.action do |_global_options, options, args|
8     interfaces = Trema.trema_process('SimpleRouter', options[:socket_dir]).controller.print_interface()
9     print sprintf("%s %-17s %s", "\"port number\"", "\"mac address\"", "\"ip address/netmask\"")
10    interfaces.each do |each|
11      print sprintf("%-13s %-17s %s\n",
12        each[:port_number].to_s, each[:mac_address], each[:ip_address]+"/"+each[:netmask_length].to_s)
13    end
14  end
15 end

```

## 2.5 あると便利な機能の実装

あると便利な機能の実装として、宛先 IP アドレスを入力した場合、次の送信先アドレスを示すコマンドを作成した。これには、RoutingTable クラスの lookup メソッドを用いる。そのために SimpleRouter クラスに find\_next\_hop メソッドを追加する。リスト 11 の fund.next\_hop メソッドでは、string 型の引数 destination を IPv4address のクラスに変換して変数 @routing\_table の lookup メソッドを呼び出している。

リスト 11: find\_next\_hop メソッド

```

1   def find_next_hop(destination)
2     destination_ip = IPv4Address.new(destination)
3     tmp = @routing_table.lookup(destination_ip).to_s
4     return tmp if tmp
5   end

```

router\_control ファイルでは、インターフェース一覧の表示はリスト 12 のように実装されている。ルーティングテーブルエントリを削除する場合はコンソールに ./bin/router\_control nextHop [宛先 IP] と入力する。

リスト 12: nextHop コマンドの実装

```

1   desc 'Return next hop IP address'
2   arg_name 'destination_ip'
3   command :nextHop do |c|
4     c.desc 'Location to find socket files'
5     c.flag [:S, :socket_dir], default_value: Trema::DEFAULT_SOCKET_DIR
6
7     c.action do |_global_options, options, args|
8       destination_ip = args[0]
9       nextHop = Trema.trema_process('SimpleRouter', options[:socket_dir]).controller.
10         find_next_hop(destination_ip)
11
12       if nextHop
13         print sprintf("Next Hop is %s\n", nextHop)
14       else
15         print sprintf("No Entry in Routing Table")
16       end
17     end
18   end

```

## 3 動作確認

今回の動作確認に用いた trema.conf ファイルと simple\_router.conf ファイルをそれぞれリスト 13, 14 に示す。

リスト 13: trema.conf

```

1   vswitch('0x1') { dpid 0x1 }
2   netns('host1') {
3     ip '192.168.1.2'
4     netmask '255.255.255.0'
5     route net: '0.0.0.0', gateway: '192.168.1.1'
6   }
7   netns('host2') {
8     ip '192.168.2.2'
9     netmask '255.255.255.0'
10    route net: '0.0.0.0', gateway: '192.168.2.1'
11  }
12  link '0x1', 'host1'
13  link '0x1', 'host2'

```

#### リスト 14: simple\_router.conf

```

1  # Simple router configuration
2  module Configuration
3      INTERFACES = [
4          {
5              port: 1,
6              mac_address: '01:01:01:01:01:01',
7              ip_address: '192.168.1.1',
8              netmask_length: 24
9          },
10         {
11             port: 2,
12             mac_address: '02:02:02:02:02:02',
13             ip_address: '192.168.2.1',
14             netmask_length: 24
15         }
16     ]
17
18     ROUTES = [
19         {
20             destination: '0.0.0.0',
21             netmask_length: 0,
22             next_hop: '192.168.1.2'
23         }
24     ]
25 end

```

今回は各コマンドを用いて状態を確認する。リスト 15 のようにコマンドを入力，出力結果を得た。リスト 15 のような結果が得られたため，各コマンドは正常に動作していると考えられる。

#### リスト 15: 実行結果

```

1  ensyuu2@ensyuu2-VirtualBox:~/week5/simple-router-Ryo-Murakami$ ./bin/router_control printRT
2  "destination/netmask" "next hop"
3  0.0.0.0/0             192.168.1.2
4  ensyuu2@ensyuu2-VirtualBox:~/week5/simple-router-Ryo-Murakami$ ./bin/router_control printInterface
5  "port number" "mac address"      "ip address/netmask"
6  1             01:01:01:01:01:01  192.168.1.1/24
7  2             02:02:02:02:02:02  192.168.2.1/24
8  ensyuu2@ensyuu2-VirtualBox:~/week5/simple-router-Ryo-Murakami$ ./bin/router_control nextHop 192.168.1.1
9  Next Hop is 192.168.1.2
10 ensyuu2@ensyuu2-VirtualBox:~/week5/simple-router-Ryo-Murakami$
11 ./bin/router_control addRT 192.168.1.1 16 192.168.1.3
12 ensyuu2@ensyuu2-VirtualBox:~/week5/simple-router-Ryo-Murakami$ ./bin/router_control printRT
13 "destination/netmask" "next hop"
14 0.0.0.0/0             192.168.1.2
15 192.168.0.0/16        192.168.1.3
16 ensyuu2@ensyuu2-VirtualBox:~/week5/simple-router-Ryo-Murakami$ ./bin/router_control nextHop 192.168.1.1
17 Next Hop is 192.168.1.3
18 ensyuu2@ensyuu2-VirtualBox:~/week5/simple-router-Ryo-Murakami$
19 ./bin/router_control addRT 192.168.1.1 24 192.168.1.4
20 ensyuu2@ensyuu2-VirtualBox:~/week5/simple-router-Ryo-Murakami$ ./bin/router_control printRT
21 "destination/netmask" "next hop"
22 0.0.0.0/0             192.168.1.2
23 192.168.0.0/16        192.168.1.3
24 192.168.1.0/24        192.168.1.4
25 ensyuu2@ensyuu2-VirtualBox:~/week5/simple-router-Ryo-Murakami$ ./bin/router_control nextHop 192.168.1.1
26 Next Hop is 192.168.1.4
27 ensyuu2@ensyuu2-VirtualBox:~/week5/simple-router-Ryo-Murakami$ ./bin/router_control delRT 192.168.1.1 24
28 ensyuu2@ensyuu2-VirtualBox:~/week5/simple-router-Ryo-Murakami$ ./bin/router_control printRT
29 "destination/netmask" "next hop"
30 0.0.0.0/0             192.168.1.2
31 192.168.0.0/16        192.168.1.3
32 ensyuu2@ensyuu2-VirtualBox:~/week5/simple-router-Ryo-Murakami$ ./bin/router_control nextHop 192.168.1.1
33 Next Hop is 192.168.1.3

```