1

Ingegneria del traffico - MPLS

mpls.py

```
from ryu.base import app_manager
from ryu.controller import ofp_event
from ryu.controller.handler import CONFIG_DISPATCHER,
↪  MAIN_DISPATCHER
from ryu.controller.handler import set_ev_cls
from ryu.ofproto import ofproto_v1_3

# Topologia di riferimento
# sudo mn --mac --arp --topo linear,3 --switck ovsk,datapath=user
↪  --controller remote
# --mac e --arp perche' non vogliamo gestire il broadcast
# --switck ovsk,datapath=user perche' OpenVSwitch supporta MPLS
↪  solo in userspace

#
#
#       s1 -- s2 -- s3
#     /        |      \
#    h1       h2       h3
# I pacchetti da h1 a h3 sono etichettati in s1 con etichetta 1000
# I pacchetti da h3 a h1 sono etichettati in s3 con etichetta 1001
# In s1 inserire regola:
```

```
# ** destinazione 10.0.0.3/32 => push 1000, next-hop=s2
# In s2 insirire la regola:
# ** 1000 => 1000, next-hop=s3
# In s3 inserire regola:
# ** destinazine 10.0.0.1/32 => push 1001, next-hop=s2

class Mpls(app_manager.RyuApp):
    OFP_VERSIONS = [ofproto_v1_3.OFP_VERSION]

    # execute at switch registration
    @set_ev_cls(ofp_event.EventOFPSwitchFeatures,
    ↪  CONFIG_DISPATCHER)
    def switch_features_handler(self, ev):
        datapath = ev.msg.datapath
        ofproto = datapath.ofproto
        parser = datapath.ofproto_parser

        # match all packets
        match = parser.OFPMatch()
        # send to controller
        actions = [parser.OFPActionOutput(ofproto.OFPP_CONTROLLER,

                                          ↪  ofproto.OFPCML_NO_BUFFER)]
        inst =
        ↪  [parser.OFPInstructionActions(ofproto.OFPIT_APPLY_ACTIONS,
                                         actions)]
        mod = parser.OFPFlowMod(datapath=datapath, priority=0,
                                match=match, instructions=inst)
        datapath.send_msg(mod)

        if datapath.id == 1:

            match = parser.OFPMatch(eth_type=0x0800,
            ↪  ipv4_dst="10.0.0.3")
            actions = [
                parser.OFPActionPushMpls(),
                parser.OFPActionSetField(mpls_label=1000),
                parser.OFPActionOutput(2)
            ]
            inst =
            ↪  [parser.OFPInstructionActions(ofproto.OFPIT_APPLY_ACTIONS,
                                             actions)]
```

```
            mod = parser.OFPFlowMod(datapath=datapath, priority=1,
                                    match=match, instructions=inst)
            datapath.send_msg(mod)

            match = parser.OFPMatch(eth_type=0x8847,
            ↪  mpls_label=1001)
            actions = [
                parser.OFPActionPopMpls(),
                parser.OFPActionOutput(1)
            ]
            inst =
            ↪  [parser.OFPInstructionActions(ofproto.OFPIT_APPLY_ACTIONS,
                                             actions)]
            mod = parser.OFPFlowMod(datapath=datapath, priority=1,
                                    match=match, instructions=inst)
            datapath.send_msg(mod)


        if datapath.id == 2:

            match = parser.OFPMatch(eth_type=0x8847,
            ↪  mpls_label=1000)
            actions = [
                parser.OFPActionOutput(3)
            ]
            inst =
            ↪  [parser.OFPInstructionActions(ofproto.OFPIT_APPLY_ACTIONS,
                                             actions)]
            mod = parser.OFPFlowMod(datapath=datapath, priority=1,
                                    match=match, instructions=inst)
            datapath.send_msg(mod)

            match = parser.OFPMatch(eth_type=0x8847,
            ↪  mpls_label=1001)
            actions = [
                parser.OFPActionOutput(2)
            ]
            inst =
            ↪  [parser.OFPInstructionActions(ofproto.OFPIT_APPLY_ACTIONS,
                                             actions)]
            mod = parser.OFPFlowMod(datapath=datapath, priority=1,
                                    match=match, instructions=inst)
```

```
            datapath.send_msg(mod)

        if datapath.id == 3:

            match = parser.OFPMatch(eth_type=0x8847,
            ↪  mpls_label=1000)
            actions = [
                parser.OFPActionPopMpls(),
                parser.OFPActionOutput(1)
            ]
            inst =
            ↪  [parser.OFPInstructionActions(ofproto.OFPIT_APPLY_ACTIONS,
                                             actions)]
            mod = parser.OFPFlowMod(datapath=datapath, priority=1,
                                    match=match, instructions=inst)
            datapath.send_msg(mod)

            match = parser.OFPMatch(eth_type=0x0800,
            ↪  ipv4_dst="10.0.0.1")
            actions = [
                parser.OFPActionPushMpls(),
                parser.OFPActionSetField(mpls_label=1001),
                parser.OFPActionOutput(2)
            ]
            inst =
            ↪  [parser.OFPInstructionActions(ofproto.OFPIT_APPLY_ACTIONS,
                                             actions)]
            mod = parser.OFPFlowMod(datapath=datapath, priority=1,
                                    match=match, instructions=inst)
            datapath.send_msg(mod)
```