University of Pisa
Master Degree in Computer Engineering
Computer Systems and Networks

Centralized VRRP with a Floodlight OpenFlow controller

Supervisors

Prof. Enzo Mingozzi

Ing. Antonio Virdis

Candidates

Antonio Di Tecco

Simone Pampaloni

Alexander De Roberto

20 February 2020

1 Introduction

The target of this project is developing a Floodlight Controller for SDN able to manage packets on a *SDN switch* (called *SS*) and to route them from a *Network A* to a *Network B* as in figure 1, choosing one of two routers (R1 or R2) in a transparent way for nodes of Network A, like in VRRP [2]. *Switch LS* of Network B is taken as being a legacy switch.

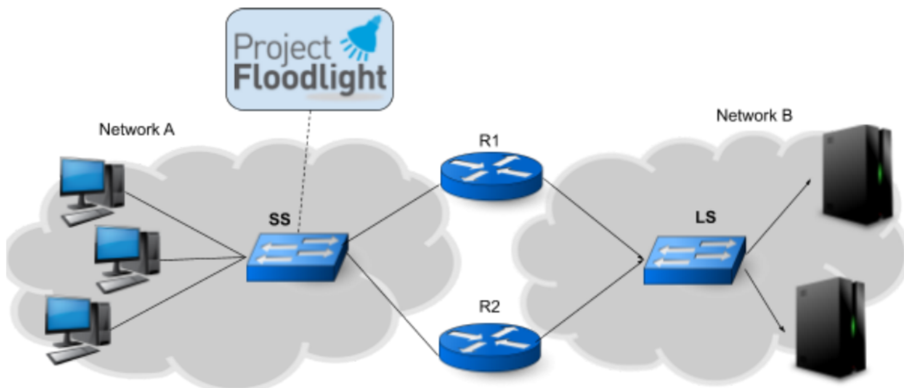

Figure 1: View of the global network

2 Building network

The network has been realized thanks to the Mininet tool that emulates a complete network of hosts, links, and switches on a single machine [1]. Mininet is useful for interactive development, testing, and demos, especially those using OpenFlow and SDN. A copy of the network shown in figure 1 is available in the *net.py* file. To execute the file is necessary to digit the following command:

```
# python ./net.py
```

Digiting the command *net* on terminal will show a linked network as follows:

```
mininet> net
r1 r1-eth1:s1-eth4 r1-eth2:s2-eth4
r2 r2-eth1:s1-eth5 r2-eth2:s2-eth5
h1 h1-eth1:s1-eth1
h2 h2-eth1:s1-eth2
h3 h3-eth1:s1-eth3
h4 h4-eth1:s2-eth1
h5 h5-eth1:s2-eth2
s1 lo:  s1-eth1:h1-eth1 s1-eth2:h2-eth1 s1-eth3:h3-eth1
        s1-eth4:r1-eth1 s1-eth5:r2-eth1
s2 lo:  s2-eth1:h4-eth1 s2-eth2:h5-eth1 s2-eth4:r1-eth2
        s2-eth5:r2-eth2
c0
```

If you are interested in looking at nodes ip can be useful for digiting the command *dump*:

```
mininet> dump
<LinuxRouter r1: r1-eth1:10.0.2.1,r1-eth2:10.0.3.1>
<LinuxRouter r2: r2-eth1:10.0.2.2,r2-eth2:10.0.3.2>
<Host h1: h1-eth1:10.0.2.3>
<Host h2: h2-eth1:10.0.2.4>
<Host h3: h3-eth1:10.0.2.5>
<Host h4: h4-eth1:10.0.3.3>
<Host h5: h5-eth1:10.0.3.4>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,
            s1-eth3:None,s1-eth4:None,s1-eth5:None>
<OVSSwitch s2: lo:127.0.0.1,s2-eth1:None,s2-eth2:None,
            s2-eth4:None,s2-eth5:None>
<RemoteController c0: 127.0.0.1:6653>
```

Where the switches *s1* and *s2* are the switches *SS* and *LS* in figure 1.

3 Centralized VRRP

To implement the centralized VRRP as in figure 2, we have realized an application to execute on network routers and some Floodlight modules.

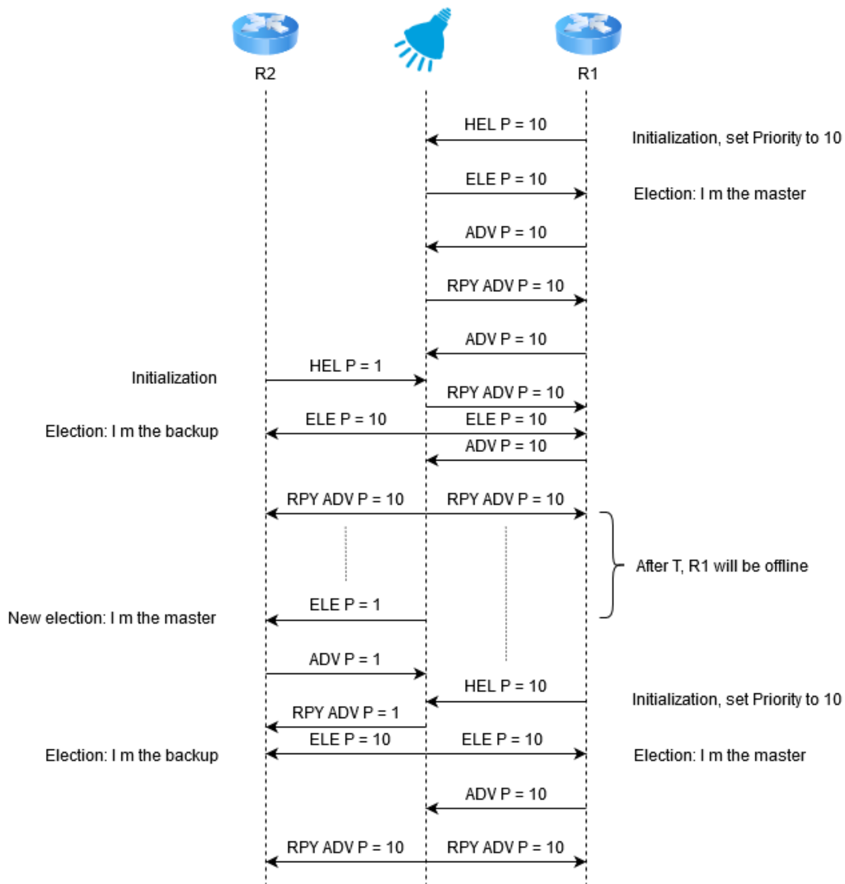

Figure 2: Sequence diagram of Centralized VRRP

3.1 Set up and configure routers

The application installed on routers has been written in Python and it has been thought for executing on each border router, with an SDN switch and different *priority* values. The file name is *router.py*. The parameters you probably need to configure are the following:

```
ADVERTISEMENT_INTERVAL = 1
CTR_DOWN_INTERVAL = 3 * ADVERTISEMENT_INTERVAL
```

The parameter ADVERTISEMENT_INTERVAL is the time between two advertisements sent by the router; of course, the less the value, the better the system will be able to manage faults. This parameter has been used also to set up the CTR_DOWN_INTERVAL parameter which is the time interval required for the router to declare the controller down and to stop the protocol.

To start the application on the router it is necessary to specify the *physical port*, which will communicate with the controller, and the *priority value*:

```
# python ./router.py [Port] [Priority]
```

After the parameters have been configured, the application starts. There will be a short initialization phase and an election phase at following, where the router is going to send a UDP *HEL* (Hello) packet in broadcast with its priority value. The router will wait for an *ELE* (Election) message from the controller with the priority chosen as the priority of the master router. If the reply does not arrive at the router, the algorithm stops because the controller will be down, otherwise the algorithm continues to bring the router in the backup state or the master state and changes the router state as needed when a new *ELE* packet arrives. The application workflow is shown in the diagram of figure 3.

3.2 Set up and configure the controller

The Floodlight controller has been realized to implement the Centralized VRR protocol. It is compound of 3 modules (three controllers) each one with dedicated tasks:

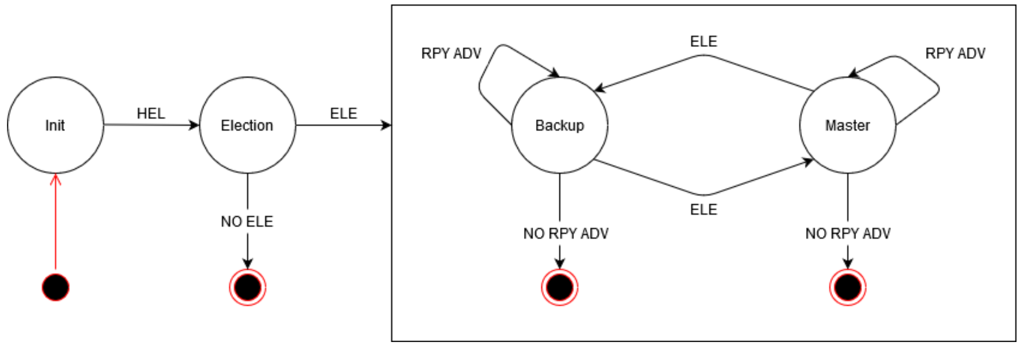

Figure 3: Description of the software executed on the router

- ARPController
- VAController
- VREController

The module *ARPController* deals with managing ARP requests sent into Network A and sent to the virtual router. Broadcast ARP requests sent between two nodes into the network will be retransmitted in all switch links, while ARP requests sent to the virtual router will be processed from the controller which retransmits an ARP reply to the sender with the *virtual MAC* of the virtual router.

The module *VAController* (Virtual Address) deals with managing the translation mechanism of MAC address, looking for all packets to retransmit to the external network and to route them to the master router. The module deals also with installing flow rules on the *switch s1* to avoid packets being processed each time from the controller.

In the end, the module *VREController* (Virtual Router Election) deals also with master and backup router's election based on priority values, managing connection and disconnection of routers.

Also here, in order to configure the protocol it would be necessary to edit some parameters in the file *Parameters.java*:

```
MASTER_ADVERTISEMENT_INTERVAL = 1000;
MASTER_DOWN_INTERVAL = 3*MASTER_ADVERTISEMENT_INTERVAL;
```

The parameter MASTER_ADVERTISEMENT_INTERVAL must be similar to the parameter ADVERTISEMENT_INTERVAL of paragraph 3.1, at most to ADVERTISEMENT_INTERVAL + Δ (here 1 second is equal to 1000!), because this value is used by the controller to set up the parameter MASTER_DOWN_INTERVAL where from [2] is the time interval for the controller to declare master down (seconds).

References

[1] Mininet, Dec 2019.

[2] S. Nadas and Ed. Virtual router redundancy protocol (vrrp) version 3 for ipv4 and ipv6, Jan 1970.