TEL510E

Telecommunication Network Planning and Management

2015-2016 Spring

OpenFlow based Load Balancing for Fat-Tree Networks

Study of Simulation Tools

Büşra TEMEL - 504141337

Şehriban ALPAYDIN – 504151346

Emre YALÇIN - 504121341

TOOLS SELECTION

This project aims to make some improvements over load balancing in fat-tree topology networks by using SDN technology. Due to demonstrate our project, mininet network emulator will be used which allows us to create a network of virtual hosts, switches, controllers, and links, so that traffic can be generated by Iperf and packets can handle by Mininet switches with a given link speed and delay [1].

Mininet support various SDN controllers. The controllers to be used here will be floodlight which is easy and flexible to work on it. Floodlight is designed to work with the growing number of switches, routers, virtual switches, and access points that support the OpenFlow communication protocol standard [2]. Before beginning the project it is crucial to set up to Floodlight emulator and understanding the operating mechanism of the tool.

TOOLS INSTALLATION

Install java components

In this project, floodlight installed on Ubuntu OS. Floodlight is written in java thus install jdk and ant by running;

```
$ sudo apt-get install build-essential default-jdk ant python-dev eclipse
```

Download and build the floodligt

```
$ git clone git://github.com/floodlight/floodlight.git
$ cd floodlight
$ git checkout stable
$ ant;
$ sudo mkdir /var/lib/floodlight
$ sudo chmod 777 /var/lib/floodlight
```

Run the floodlight

```
$ java -jar target/floodlight.jar
```

Simulate a network

It is possible to run Mininet against the locally running Floodlight(just type “sudo mn”) but you can also run it against a remote controller [3].

BASIC CONCEPT

After the installation process, virtual machine is started where the mininet emulator and floodlight controller are installed. There are two basic process while improving the project. One of them is coding the fat-tree topology on python for mininet and the other one is coding the load balancing algorithm on java for floodlight. In this report, just basic concepts not entire coding steps are examined to understand usage of these two tools.

Obtaining a Topology on Mininet

Mininet already has its own topology samples and some of them can be created as follows:

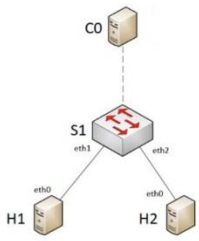

Figure 1

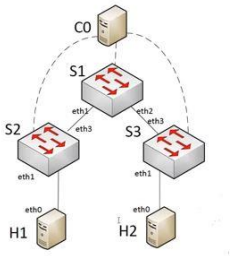

Figure 2

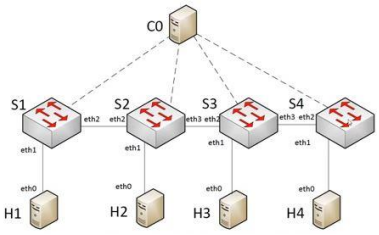

Figure 3

Command for Figure 1: $ sudo mn (default topology)

Command for Figure 2: $ sudo mn --topo=tree,2,2

Command for Figure 3: $ sudo mn --topo=linear,4

This topology codes are stored as a *.py* file under */mininet directory*. Thus, it will be replaced with the new one which will be written in python for the fat-tree topology.

Floodlight is used as an external controller and it runs in local virtual machine. It should be configured with an ip address and this address should be determined in mininet as follows:

$ sudo ifconfig eth0 192.168.1.1

$ sudo mn --controller=remote, ip=192.168.1.1, port=6633

6633 : Default port for the Open Flow Protocol

After providing this connection between network devices in the emulator and controller as floodlight, software defined networking which is the main target can be process. In this project load balancing will be aimed as a network process. The algorithm will be designed in next steps of the project is stored as a .java file under /floodlight directory.

Initial Outputs

The controller activation state is observed when Floodlight is run.

```
floodlight@floodlight:~/floodlight$ java -jar target/floodlight.jar
15:19:52.435 INFO [n.f.c.m.FloodlightModuleLoader:main] Loading modules from src/main/resources/floodlightdefault.properties
15:19:54.589 INFO [n.f.c.i.Controller:main] Controller role set to ACTIVE
```

It is learned after running floodlight which port will be used to connect devices to controller. It is observed port by 6653 is used in this version of floodlight and it is also flexible to change .

```
ClusterConfig [allNodes={32767=Node [hostname=localhost, port=6642, nodeId=3276
7, domainId=32767]}, authScheme=CHALLENGE_RESPONSE, keyStorePath=/etc/floodlight
/auth_credentials.jceks, keyStorePassword is unset]
15:12:26.153 INFO [o.s.s.i.r.RPCService:main] Listening for internal floodlight
RPC on localhost/127.0.0.1:6642
15:12:26.190 INFO [n.f.c.i.OFSwitchManager:main] Listening for switch connection
s on 0.0.0.0/0.0.0.0:6653
```

A basic tree topology is produced in mininet and connected to floodlight controller via 6653 port. Due to both of mininet and floodlight are run in the same virtual machine, it is not required to specify ip address of the controller. It is already known as localhost by default.

```
floodlight@floodlight:~$ sudo mn --mac --controller=remote,port=6653 --topo tree
,2
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1 s2 s3
*** Adding links:
(s1, s2) (s1, s3) (s2, h1) (s2, h2) (s3, h3) (s3, h4)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
c0
*** Starting 3 switches
s1 s2 s3
*** Starting CLI:
mininet>
```

Connected switches and added links are shown in the controller interface.

```
15:12:54.885 WARN [n.f.c.i.C.s.notification:main] Switch 00:00:00:00:00:00:00:01
 connected.
15:12:55.210 INFO [n.f.r.RouteManager:Scheduled-1] *** within the overrided topo
logyChanged() function in RouteManager
15:16:02.952 WARN [n.f.l.i.L.s.notification:New I/O server worker #2-5] Link added: Link [src=00:00:00:00:00:00:00:01 outPort=1,
 dst=00:00:00:00:00:00:00:02, inPort=3]
15:16:02.958 INFO [n.f.l.i.LinkDiscoveryManager:New I/O server worker #2-4] Inter-switch link detected: Link [src=00:00:00:00:00
:00:00:03 outPort=3, dst=00:00:00:00:00:00:00:01, inPort=2]
```

Connection between hosts is controlled by ping in mininet.

```
mininet> h1 ping h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=5.90 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.296 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.058 ms
```

The controller session can be killed by Control + C. There is no controller running at that moment but ping process will be still working for a while. Because controller has been sent a flow modification message to switch which results in flow entries on the switch and these entries are cashed until a timeout expires.

Based on these reviews and studies, we will keep working on our project with above-examined simulation tools.

REFERENCES

[1] Floodlight. http://www.projectfloodlight.org/

[2] Mininet. mininet.org

[3] http://www.projectfloodlight.org/getting-started/