

Cloud System Management Lab

Lab2

Creating SDN network topologies in Mininet

9911034 錢偉臣

Objective

使用mininet的python api, 學習建立拓樸結構及測量效能

Target

建出Fat tree topology, 並能利用python寫出腳本控制及量測

Experiment

第一目標是建Fat tree, 先看出連接順序, 慢慢熟悉python的語法。使用dict結構可以快速幫助我撰寫規則。

第二要點需要連接floodlight的controller, 預設mininet的controller會在Mininet()建構式中自行產生, 這樣一來在使用net.addController()時會因為6633port已被default controller站住而無法順利連接floodlight。兩種方法可以解決:

1. 使用myController = customConstructore()給予參數{'remote':RemoteController}及'remote,ip=127.0.0.1,port=6633', 呼叫建構式net = Mininet(topo=topo, link=TCLink, controller=myController)
2. 先給建構式無Controller設定,
net = Mininet(topo=topo, link=TCLink, controller=None)
再新增Controller,
net.addController("Owen's Controller", controller=RemoteController,
ip="127.0.0.1")

net.start()後, 需要看各節點的連接狀況:

- dumpNodeConnections(net.switches)
- dumpNodeConnections(net.hosts)

net.pingAll()或net.pingFull()讓controller知道拓樸狀況後, 再進行UDP的測試:

- 找兩個hosts分別在pod0跟pod3用popen開iperf -s當作server, 用popen是因為要避免python陷入等待狀態
- 再找一個host在pod0用cmdPrint分別對前兩個server做iperf -c

最後以net.stop()結束模擬

Result

```
floodlight@ubuntu: ~/floodlight/script
floodlight@ubuntu: ~/floodlight
*** s3002-h1 : ('iperf -c 10.0.0.1 -u -t 10 -i 1 -b 100m',)
Client connecting to 10.0.0.1, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size: 208 KByte (default)
[ 3] local 10.0.0.3 port 51181 connected with 10.0.0.1 port 5001
[ ID] Interval      Transfer      Bandwidth
[ 3] 0.0- 1.0 sec   11.3 MBytes   95.0 Mbits/sec
[ 3] 1.0- 2.0 sec   11.3 MBytes   94.9 Mbits/sec
[ 3] 2.0- 3.0 sec   11.4 MBytes   95.5 Mbits/sec
[ 3] 3.0- 4.0 sec   11.3 MBytes   95.1 Mbits/sec
[ 3] 4.0- 5.0 sec   11.4 MBytes   95.6 Mbits/sec
[ 3] 5.0- 6.0 sec   11.2 MBytes   94.1 Mbits/sec
[ 3] 6.0- 7.0 sec   11.4 MBytes   95.3 Mbits/sec
[ 3] 7.0- 8.0 sec   11.3 MBytes   94.8 Mbits/sec
[ 3] 8.0- 9.0 sec   11.3 MBytes   95.0 Mbits/sec
[ 3] 9.0-10.0 sec   11.3 MBytes   95.1 Mbits/sec
[ 3] 0.0-10.0 sec   113 MBytes   95.0 Mbits/sec
[ 3] Sent 80812 datagrams
[ 3] Server Report:
[ 3] 0.0-10.0 sec   113 MBytes   95.1 Mbits/sec   0.040 ms   20/80811 (0.025%)
[ 3] 0.0-10.0 sec   21 datagrams received out-of-order
*** s3002-h1 : ('iperf -c 10.0.0.14 -u -t 10 -i 1 -b 100m',)

floodlight@ubuntu: ~/floodlight/script
floodlight@ubuntu: ~/floodlight
*** s3002-h1 : ('iperf -c 10.0.0.14 -u -t 10 -i 1 -b 100m',)
Client connecting to 10.0.0.14, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size: 208 KByte (default)
[ 3] local 10.0.0.3 port 60232 connected with 10.0.0.14 port 5001
[ ID] Interval      Transfer      Bandwidth
[ 3] 0.0- 1.0 sec   11.3 MBytes   94.8 Mbits/sec
[ 3] 1.0- 2.0 sec   11.2 MBytes   94.0 Mbits/sec
[ 3] 2.0- 3.0 sec   11.2 MBytes   93.6 Mbits/sec
[ 3] 3.0- 4.0 sec   11.2 MBytes   94.3 Mbits/sec
[ 3] 4.0- 5.0 sec   11.1 MBytes   93.0 Mbits/sec
[ 3] 5.0- 6.0 sec   11.3 MBytes   95.0 Mbits/sec
[ 3] 6.0- 7.0 sec   11.1 MBytes   93.3 Mbits/sec
[ 3] 7.0- 8.0 sec   11.2 MBytes   94.1 Mbits/sec
[ 3] 8.0- 9.0 sec   11.4 MBytes   95.6 Mbits/sec
[ 3] 9.0-10.0 sec   11.3 MBytes   94.9 Mbits/sec
[ 3] 0.0-10.0 sec   112 MBytes   94.3 Mbits/sec
[ 3] Sent 80153 datagrams
[ 3] Server Report:
[ 3] 0.0-10.2 sec   101 MBytes   83.0 Mbits/sec   15.744 ms 7841/80152 (9.8%)
[ 3] 0.0-10.2 sec   39 datagrams received out-of-order
*** Stopping 20 switches
```

同pod的host使用UDP(loss=0.025%)

不同pod的host使用UDP(loss=9.8%)

因為設定在pod裡的link沒有loss, 而core switch跟aggregation switch間的link有5%的loss, 所以 $1-(0.95*0.95)=9.75\%$, 符合不同pod間的UDP封包loss率, 而同pod的loss率極低。

加分: 加入raw_input讓使用者決定pod數量, 練習python語法。