H.264 SVC performance evaluation over SDN (2)

[Goal]
Based on "Using Bellman-Ford to find a shortest path (version 2)" and myEvalSVC-Mininet, I re-write some tools so that users can create any topology and do the H.264/SVC performance evaluations. Moreover, metrics, such as packet loss rate, packet end-to-end delay, and throughput, can be obtained.

[Tools]
1. l2_bellmardford.py (put this file under /pox/ext)
2. mystg_svc.c (H.264 sender: This version can generate the sending packet trace, i.e. sender_trace.txt)
3. myrtg_svc.c (H.264 receiver: This version can generate the receiving packet trace, i.e received_trace.txt)
4. measure-throughput.pl (for measuring the throughput)
   (above files can be downloaded from here)
5. Other related can refer to myEvalSVC-Mininet.

[Steps]
1. Go to http://www.ramonfontes.com/vnd/# to create your own SDN topology. (More detail operations can refer to https://www.youtube.com/playlist?list=PLccoFREVAT_4nEtrkl59mjJf5ZzRX8DZA)
2. You can create a topology like the following figure.
3. click the link and set the parameters, such as bandwidth, delay, loss, and etc.
4. click File -> Export -> Export to Mininet
(mininetScript0224.sh)

```python
#!/usr/bin/python

from mininet.net import Mininet
from mininet.node import Controller, RemoteController, OVSKernelSwitch,
OVSLegacyKernelSwitch, UserSwitch
from mininet.cli import CLI
from mininet.log import setLogLevel
from mininet.link import Link, TCLink

def topology():
    """Create a network.""
    net = Mininet( controller=RemoteController, link=TCLink,
    switch=OVSKernelSwitch )

    print "*** Creating nodes"
```

s1 = net.addSwitch( 's1', listenPort=6673, mac='00:00:00:00:00:01' )
s2 = net.addSwitch( 's2', listenPort=6674, mac='00:00:00:00:00:02' )
s3 = net.addSwitch( 's3', listenPort=6675, mac='00:00:00:00:00:03' )
s4 = net.addSwitch( 's4', listenPort=6676, mac='00:00:00:00:00:04' )
s5 = net.addSwitch( 's5', listenPort=6677, mac='00:00:00:00:00:05' )
h6 = net.addHost( 'h6', mac='00:00:00:00:00:06', ip='10.0.0.6/8' )
h7 = net.addHost( 'h7', mac='00:00:00:00:00:07', ip='10.0.0.7/8' )
h8 = net.addHost( 'h8', mac='00:00:00:00:00:08', ip='10.0.0.8/8' )
h9 = net.addHost( 'h9', mac='00:00:00:00:00:09', ip='10.0.0.9/8' )
c10 = net.addController( 'c10', controller=RemoteController, ip='127.0.0.1', port=6633 )

print "*** Creating links"
net.addLink(s3, h9, 4, 0, bw=10, delay='1ms', max_queue_size=20, loss=0)
net.addLink(s3, h8, 3, 0, bw=10, delay='1ms', max_queue_size=20, loss=0)
net.addLink(s5, s3, 2, 2, bw=10, delay='1ms', max_queue_size=20, loss=0)
net.addLink(s4, s5, 2, 1, bw=10, delay='1ms', max_queue_size=20, loss=0)
net.addLink(s2, s3, 3, 1, bw=10, delay='1ms', max_queue_size=20, loss=0)
net.addLink(h7, s2, 0, 2, bw=10, delay='1ms', max_queue_size=20, loss=0)
net.addLink(s1, s4, 2, 1, bw=10, delay='1ms', max_queue_size=20, loss=0)
net.addLink(s1, s2, 2, 1, bw=10, delay='1ms', max_queue_size=20, loss=0)
net.addLink(h6, s1, 0, 1, bw=10, delay='1ms', max_queue_size=20, loss=0)

print "*** Starting network"
net.build()
c10.start()
s3.start( [c10] )
s5.start( [c10] )
s4.start( [c10] )
s2.start( [c10] )
s1.start( [c10] )

print "*** Running CLI"
CLI( net )

print "*** Stopping network"
et.stop()

if __name__ == '__main__':
    setLogLevel( 'info' )
topology()

5. Open a terminal and run pox controller.
6. compile the H.264 sender and receiver program.

Video encoding and some related operations can refer to myEvalSVC-Mininet.

7. Open another terminal to run the mininet script.
8. type `xterm h6 h7 h8 h9` to open four xterm windows. (h6->h8: H.264/SVC video transmission, h7->h9: iperf for background traffic)

9. use `iperf` in h7 and h9 to generate the background traffic
10. use mystg_svc in h6 and myrtg_svc in h8 to transmit the video packets.
11. After evaluation, run the following commands to do performance evaluation.
(throughput)
(packet loss rate)
count the number of record in sender_trace.txt
count the number of record in received_trace.txt.
So the packet loss rate $= \frac{894 - 865}{894} \times 100 = 3.24\%$
Check the output of pox controller and we can find out that video traffic will go from s1-s2-s3 and background traffic will go s2-s3. If the video traffic can choose another path, i.e. s1-s4-s5-s3, the video can get better video delivered quality.

Dr. Chih-Heng Ke
Department of Computer Science and Information Engineering, National Quemoy University, Kinmen, Taiwan
Email: smallko@gmail.com / smallko@nqu.edu.tw