Software Defined Networking

Dr. Nick Feamster
Professor

In this course, you will learn about software defined networking and how it is changing the way communications networks are managed, maintained, and secured.

Testing a Simple Mininet Setup

- Try setting up a simple topology with three hosts connected to a single switch:
  - sudo mn --test pingall --topo single,3
- This setup uses a default switch controller and switch
  - Mininet also allows you to use custom remote controllers (and custom switches)

Basic Mininet Command Line

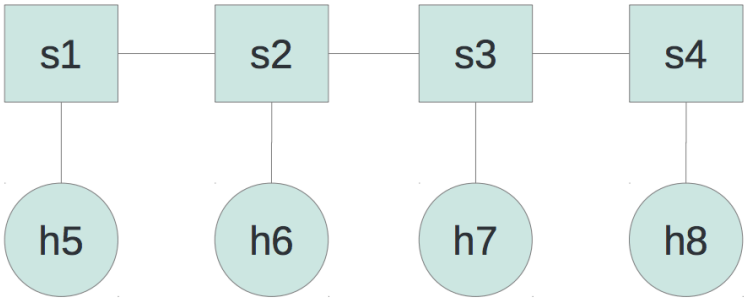
- **--topo** – defines a topology via command line upon mininet start-up.
- **--switch** – defines the switch to be used. By default the OVSK software switch is used.
- **--controller** – defines the controller to be used. If unspecified default controller is used with a default hub behavior.

Trying Out Different Mininet Topologies

- Minimal network with two hosts, one (1) switch
  - sudo mn –topo minimal
- Example with 4 hosts and 4 switches
  - sudo mn --topo linear,4
- Example with 3 hosts all connected to one switch.
  - sudo mn --topo single,3
- Tree topology with defined depth and fan-out.
  - sudo mn --topo tree,depth=2,fanout=2

How mn Works: mn executes Python

- "mn" is a launch script that executes Python
- Consider: "—topo linear, 4"




```
from mininet.net import Mininet

from mininet.topo import LinearTopo

Linear = LinearTopo(k=4)

net = Mininet(topo=Linear)

net.start()
net.pingAll()
net.stop()
```

Writing Your Own Mininet Topologies

- Example: Two hosts, one switch
- **mininet.cli.CLI(net)** before net.stop() will escape to interactive CLI before script terminates
- **addLink** allows you to specify: Bandwidth (bw) in Mbps, Delay (delay), Maximum Queue Size (max_queue_size), Loss (loss) in percentage

```
from mininet.net import Mininet
net = Mininet()

# Creating nodes in the network.
c0 = net.addController()
h0 = net.addHost('h0')
s0 = net.addSwitch('s0')
h1 = net.addHost('h1')

# Creating links between nodes in network (2-ways)
net.addLink(h0, s0)
net.addLink(h1, s0)

# Configuration of IP addresses in interfaces
h0.setIP('192.168.1.1', 24)
h1.setIP('192.168.1.2', 24)

net.start()
net.pingAll()
net.stop()
```

https://github.com/mininet/mininet/wiki/Introduction-to-Mininet

More Complicated Topology Generation

```
#!/usr/bin/python

from mininet.topo import Topo
from mininet.net import Mininet
from mininet.util import dumpNodeConnections
from mininet.log import setLogLevel

class SingleSwitchTopo(Topo):
    "Single switch connected to n hosts."
    def build(self, n=2):

        # Initialize topology and default options
        switch = self.addSwitch('s1')

        # Python's range(N) generates 0..N-1
        for h in range(n):
            host = self.addHost('h%s' % (h + 1))
            self.addLink(host, switch)
```

```
def simpleTest():
    "Create and test a simple network"
    topo = SingleSwitchTopo(n=4)
    net = Mininet(topo)
    net.start()
    print "Dumping host connections"
    dumpNodeConnections(net.hosts)
    print "Testing network connectivity"
    net.pingAll()
    net.stop()

if __name__ == '__main__':
    # Tell mininet to print useful information
    setLogLevel('info')
    simpleTest()
```

Things Not (Yet) Covered

- Access to (common) filesystem
- Setting link speeds and properties
- Using custom controllers and switches
- Host configuration
- Performance measurement

Summary

- Mininet is a network emulator that runs in a Virtual Machine
  - Lightweight OS virtualization to achieve scale
  - Fast, easy, sharable
- Next Part of Lesson: Topology examples
  - mn wrapper, Python
  - Topologies and controllers