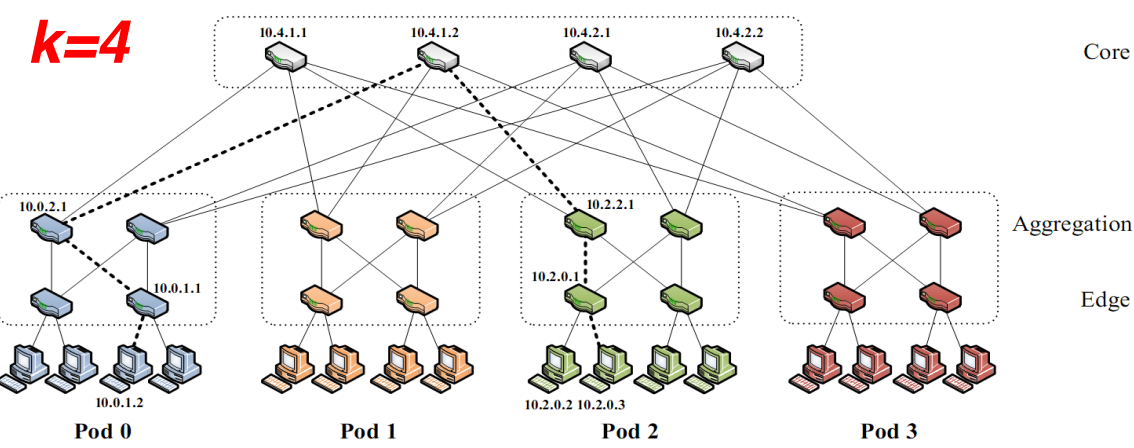
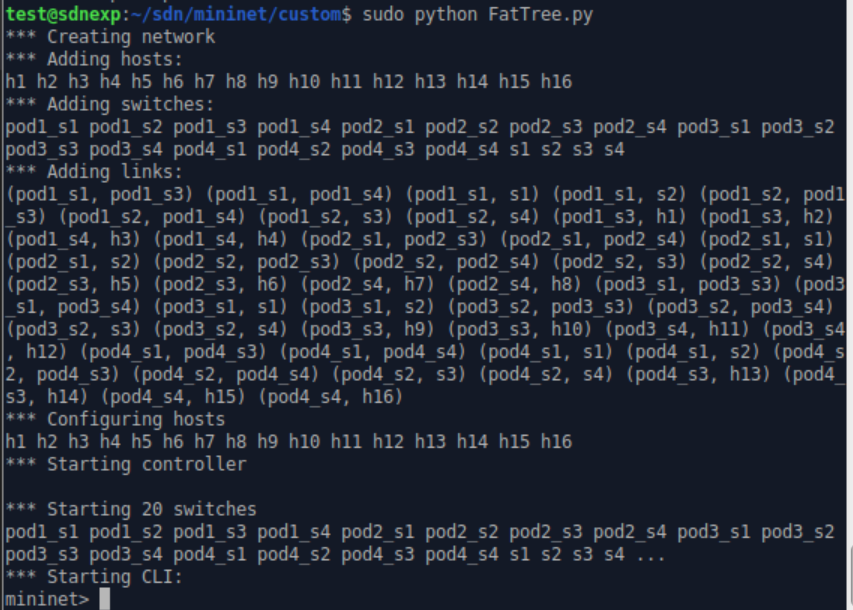
**Lab1——Mininet**

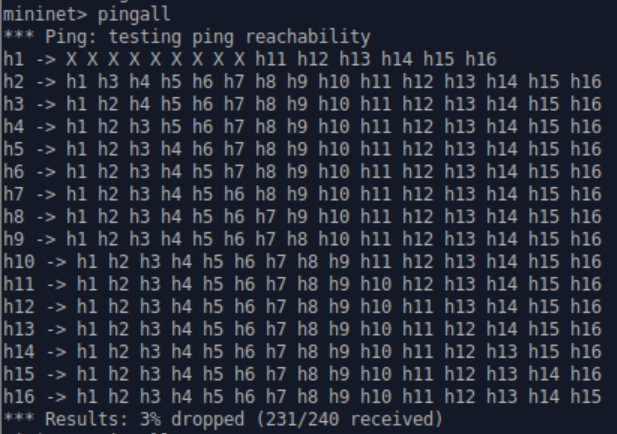
搭建 Fat Tree 网络拓扑（k=4）



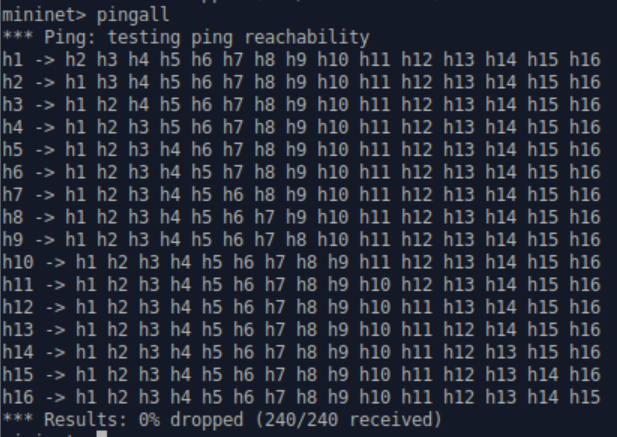
在创建的拓扑中，s1-s4表示的是最上面的四个Core交换机，在其他交换机的命名中均有pod表示其所属的pod，其中s1和s2表示其属于Aggregation类型的交换机，s3和s4表示其属于Edge类型的交换机，例如pod1\_s1表示属于pod1的Aggregation类型交换机



pingall测试连接情况

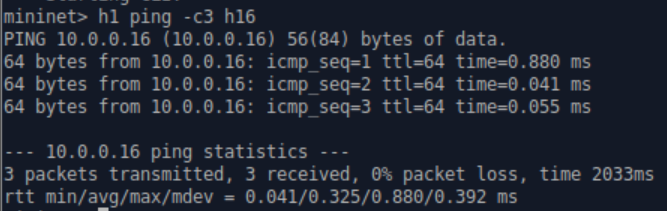


在开启Mininet的STP协议功能后，第一次pingall时，h1前几次均未ping通，存在一定的丢包，思考后发现由于交换机有一个学习链路的过程，所以会导致一开始ping失败



再pingall后发现所有主机之间都能互相Ping通

以h1 ping h16为例，分析数据包的传输路径



由于h1和pod1\_s3直连，所以我们先来看pod1\_s3的mac表，发现目的Mac对应的是端口2



然后查表得pod1\_s3的2端口对应的是pod1\_s2,再查其Mac表，得到目的Mac对应的是端口2



然后查表得pod1\_s2的2端口对应的是s4,再查其Mac表，得到目的Mac对应的是端口4

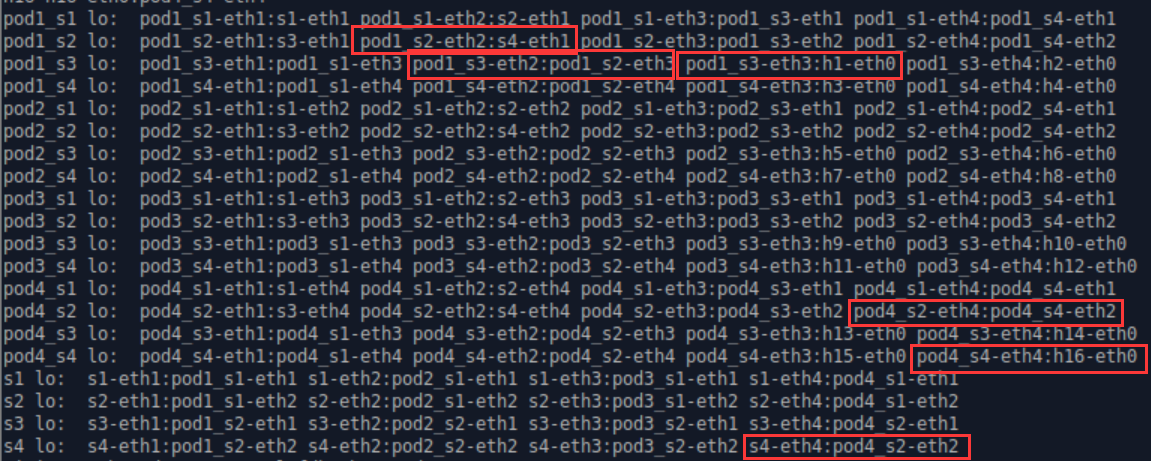


然后查表得s4的4端口对应的是pod4\_s2,再查其Mac表，得到目的Mac对应的是端口4



然后查表得pod4\_s2的4端口对应的是pod4\_s4,再查其Mac表，得到目的Mac对应的是端口4





然后查表得pod4\_s4的4端口对应的是目标主机h16。所以h1 ping h16的路径是h1--pod1\_s3--pod1\_s2—s4--pod4\_s2--pod4\_s4—h16

源代码

from mininet.topo import Topo

from mininet.net import Mininet

from mininet.cli import CLI

from mininet.log import setLogLevel

import os

class FatTree( Topo ):

def build(self):

for i in range(4):

self.addSwitch("s{}".format(i+1))

for pod in range(4):

for i in range(4):

self.addSwitch("pod{}\_s{}".format(pod+1,i+1))

for i in range(16):

self.addHost("h{}".format(i+1))

# Add links

for pod in range(4):

for i in range(2):

self.addLink("pod{}\_s{}".format(pod+1,i+1),"s{}".format(2\*i+1))

self.addLink("pod{}\_s{}".format(pod+1,i+1),"s{}".format(2\*i+2))

self.addLink("pod{}\_s{}".format(pod+1,i+1),"pod{}\_s{}".format(pod+1,3))

self.addLink("pod{}\_s{}".format(pod+1,i+1),"pod{}\_s{}".format(pod+1,4))

for pod in range(4):

for i in range(2,4):

self.addLink("pod{}\_s{}".format(pod+1,i+1),"h{}".format(pod\*4+2\*i-3))

self.addLink("pod{}\_s{}".format(pod+1,i+1),"h{}".format(pod\*4+2\*i-2))

def run():

topo=FatTree()

net=Mininet(topo=topo,controller=Non)

net.start()

for i in range(4):

os.system("sudo ovs-vsctl set bridge s{} stp\_enable=true".format(i+1))

os.system("sudo ovs-vsctl del-fail-mode s{}".format(i+1))

for pod in range(4):

for i in range(4):

os.system("sudo ovs-vsctl set bridge pod{}\_s{} stp\_enable=true".format(pod+1,i+1))

os.system("sudo ovs-vsctl del-fail-mode pod{}\_s{}".format(pod+1,i+1))

CLI(net)

net.stop()

if \_\_name\_\_=='\_\_main\_\_':

setLogLevel('info')

run()