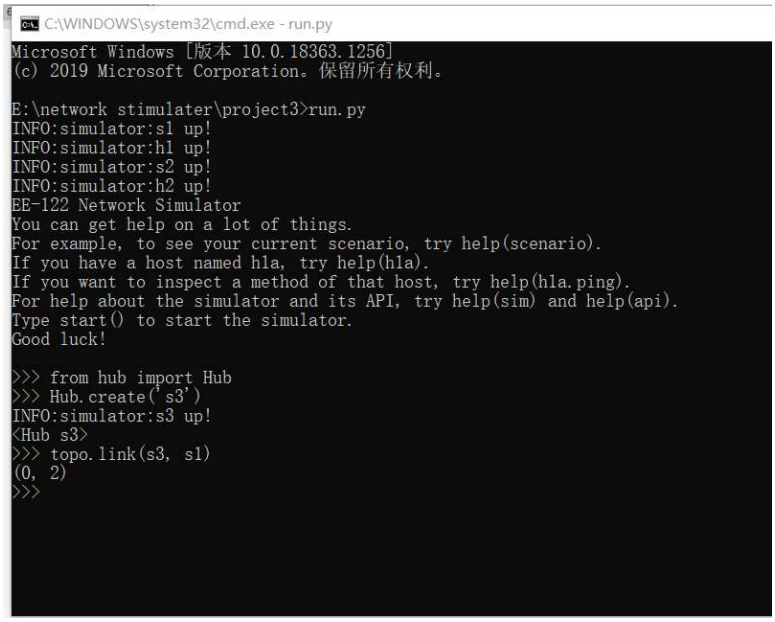
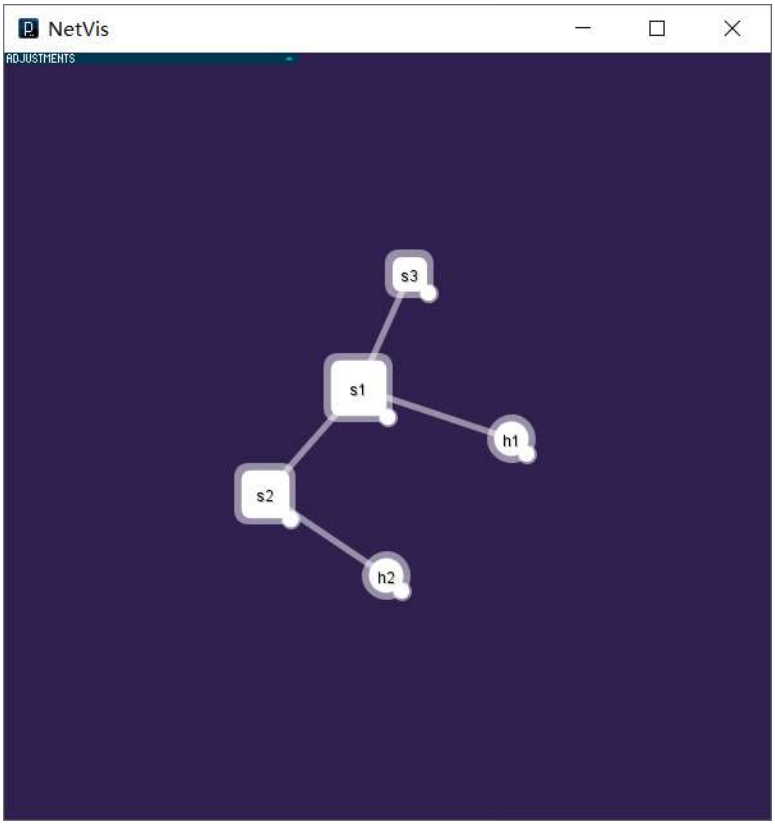
姓名：滕德淋　　班级：111192

第一步：根据文档操作

根据《先读我.docx》文档的前四步教程，了解到整体框架是什么样的，并且运行了一下：





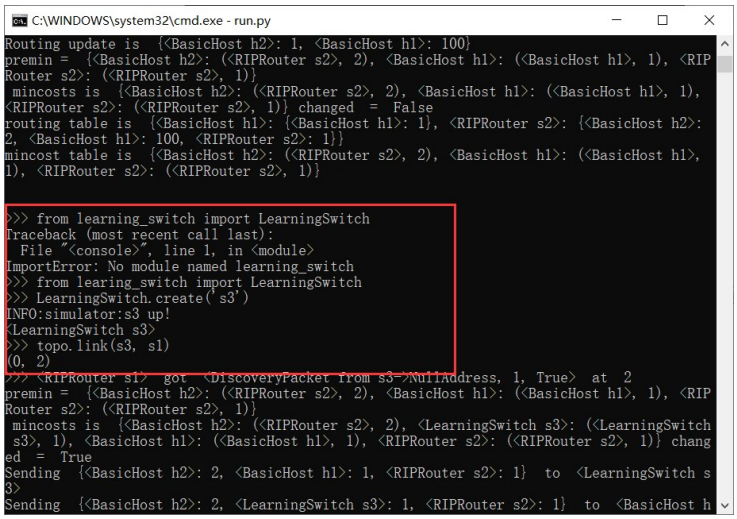
进行了创建点、连接线、删除线、删除点的操作；
体验了主机节点的模拟 ping 的过程，观察到包的发送的动画演示；

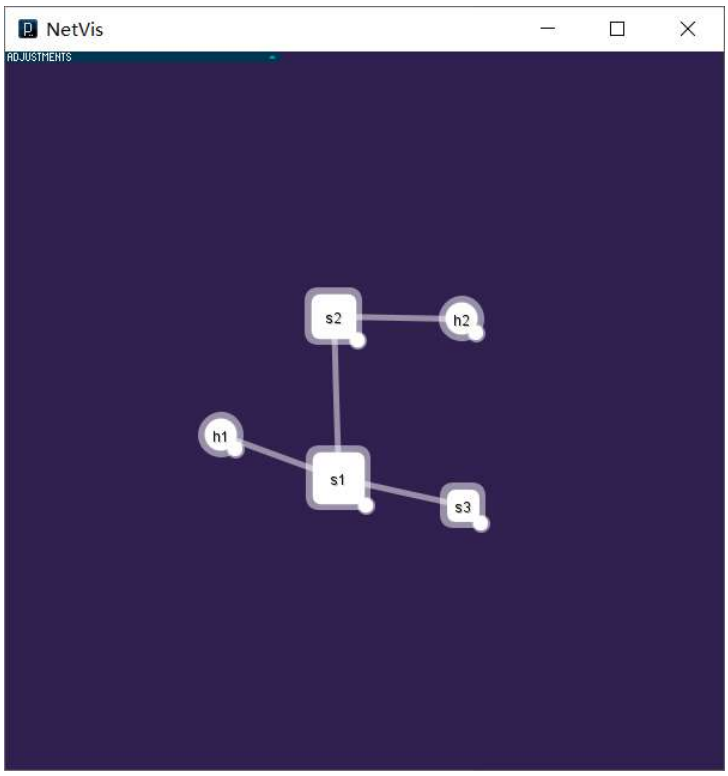
第二步：写学习型交换机

根据老师给的伪代码可以理解到学习型交换机工作的原理:

```python
class LearningSwitch (Entity):
    def __init__(self):
        # Add your code here!
        self.ports= {}

    def handle_rx (self, packet, port):
        # Add your code here!
        packetDest = packet.dst;
        if not self.ports.has_key(packet.src):
            self.ports[packet.src] = port;

        if self.ports.has_key(packetDest):
            self.send(packet, self.ports[packetDest])
        else:
            self.send(packet, port, flood=True)
        # raise NotImplementedError
```

编写好交换机之后，可以将 switch 从集线器 hub 换成 learningswitch;





进行 ping 操作可以实现，学习型交换机完成;

第三步：距离向量路由算法编写

通过 flood 进行建表，然后查表进行包的转发。RIPRouter 需要处理三个类型的包：DiscoverPacket， RoutingUpdate, others。

当收到 DiscoverPacket 类型的包，表示此节点和邻居是相连的，需要将包的 src 存放到路由表中。DiscoverPacket 只是相邻节点进行发包。

当收到 RoutingUpdate 类型的包时，如果这个包的 src，以前没有收到，这时需要把包的 src 存放到路由表中，如果路由表中有该 src 的记录，需要比较路径的长度，如果长度比路由表中存的小，需要更新路由表,然后已有的路由表里面的信息 flood 出去。

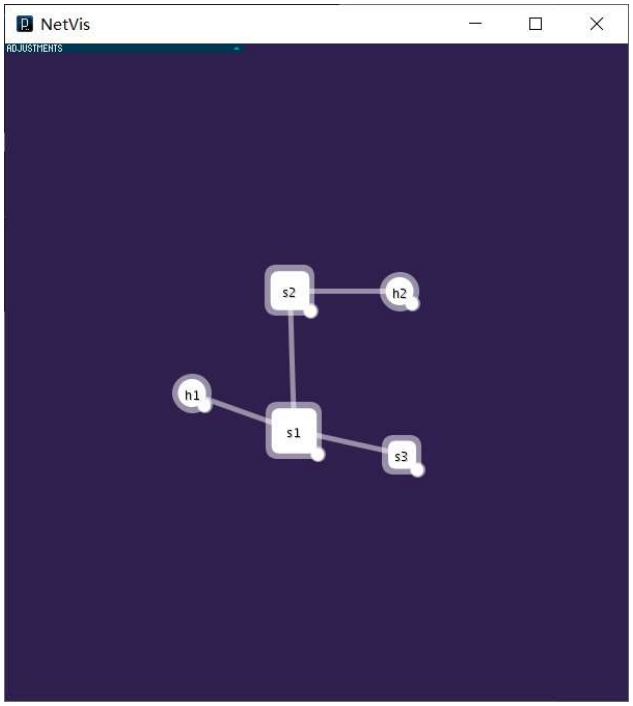
当收到 other 类型的包时，数据包直接根据目的地址进行转发。

```
if hasattr(packet, 'is_link_up'): # DISCOVERY PACKET!
    self.ports[packet.src] = port #how to get to my neighbours
    if packet.is_link_up:
        self.routingTable[packet.src] =  {packet.src: 1}
    else:
        self.routingTable.pop(packet.src)
```

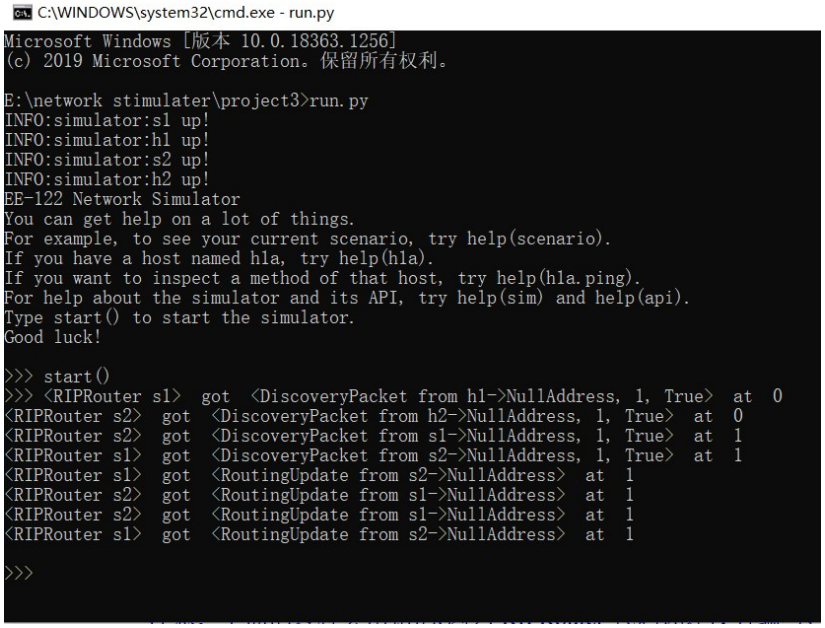
```
elif hasattr(packet, 'paths'): # UPDATE PACKET!

    print "Routing update is ", packet.str_routing_table();
    if not self.routingTable.has_key(packet.src): #if its not a neighbour, fuck it
        pass
    else: #its a neighbour, deal with it
        for dest in packet.all_dests():
            if packet.get_distance(dest) == 100:
                self.routingTable[packet.src][dest] = 100
            else:
                throughSrc = self.minCosts[packet.src][1] + packet.get_distance(dest)
                #if not self.routingTable[packet.src].has_key(dest) or  throughSrc < self.routingT
                self.routingTable[packet.src][dest] = throughSrc

    self.lastPackets[packet.src] = packet
```

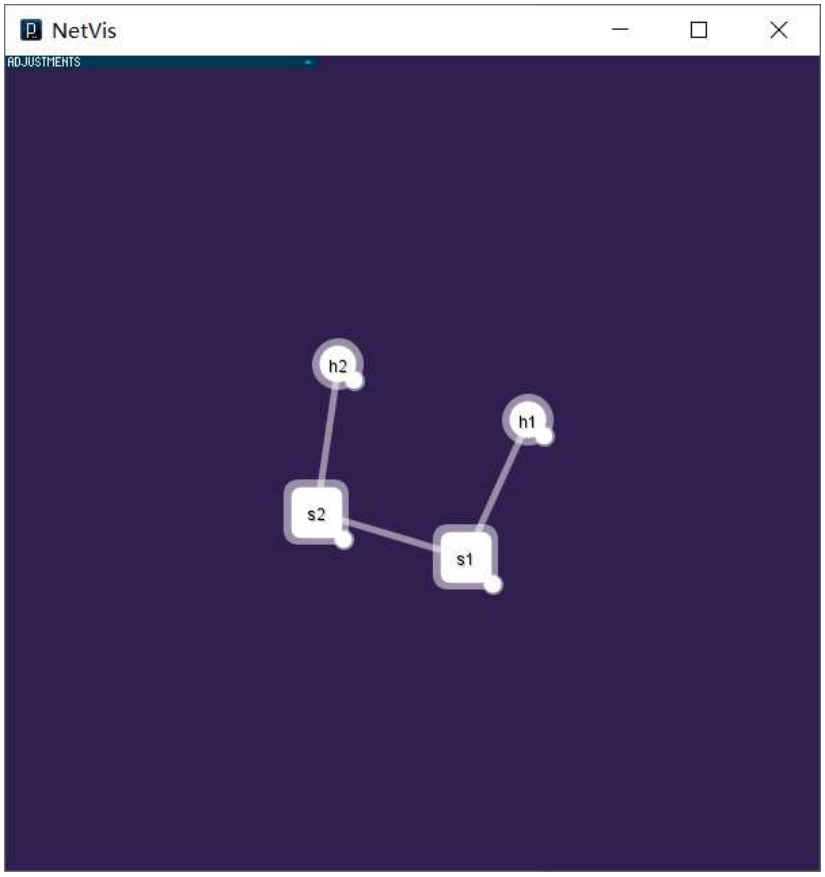
```
else: #DATA PACKET!
    if self.minCosts.has_key(packet.dst):
        print "Trying to send to ", packet.dst, " through ", self.minCosts[packet.dst]
        if self.minCosts[packet.dst][1] != 100:
            self.send(packet, self.ports[self.minCosts[packet.dst][0]]) # data packet, send it thr
```

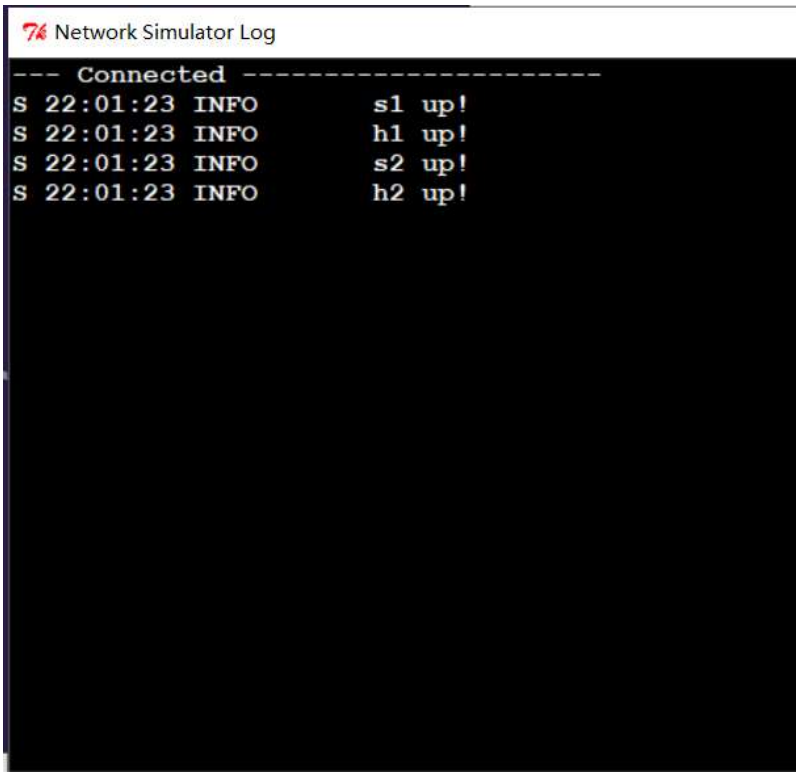
编写好路由器之后，将路由器作为 switch，进行调用：



第四步：进行 log 信息查看



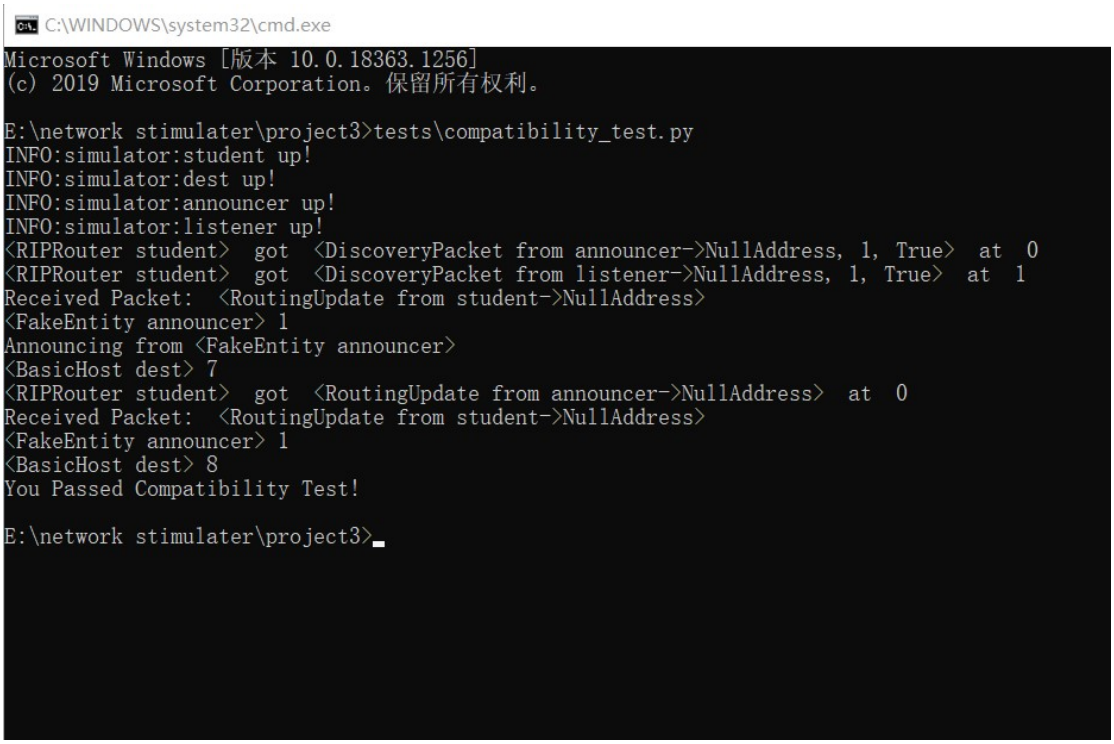


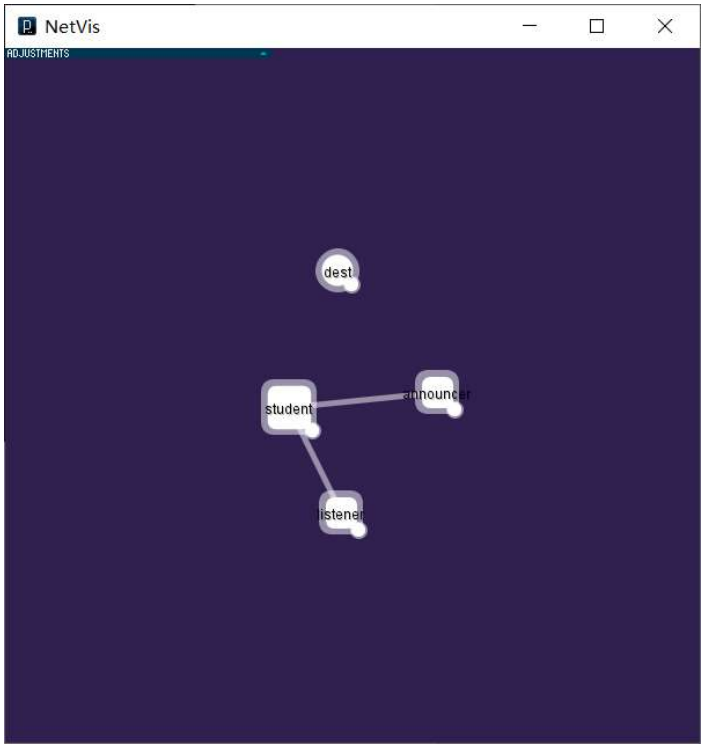


当程序运行起来之后就会自动建立两个主机 h1、h2；两个路由节点 s1、s2；
Log 消息中可以看到；

第五步：测试

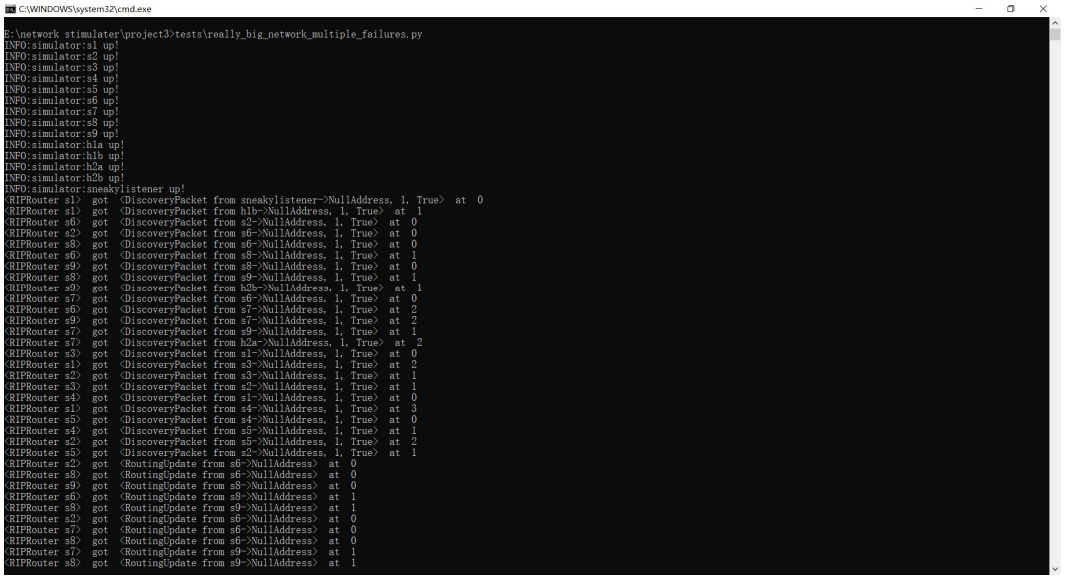
在命令行中调用测试用例：tests*.py
compatibility_test 测试：

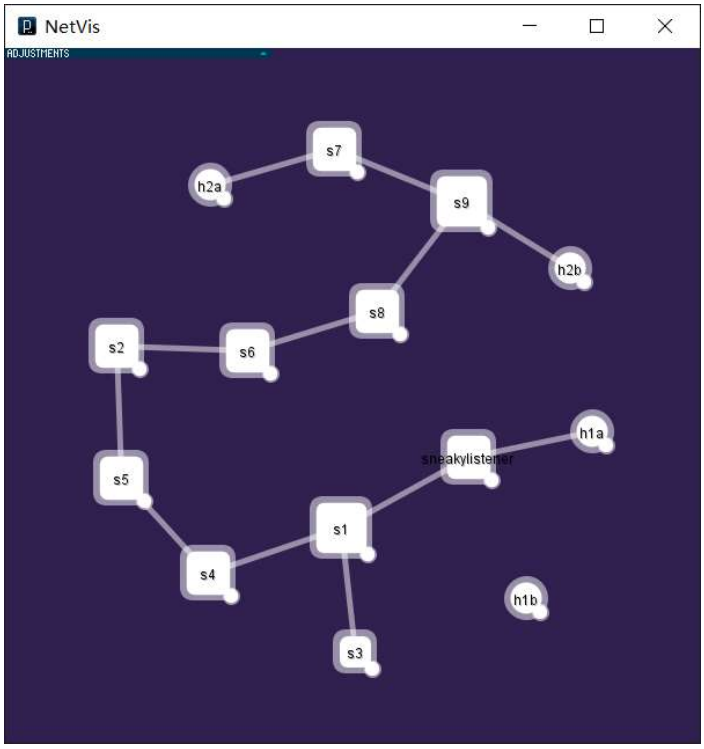




测试通过；

really_big_network_multiple_failures 测试：




NetVis
ADJUSTMENTS
s7
h2a
s9
h2b
s8
s2
s6
h1a
sneakylistener
s5
s1
s4
h1b
s3

完成测试；