# 实验三

1. 实验目的

使用Pyretic实现第二层Mac地址上的防火墙

1. 实验环境
2. VMWare虚拟机
3. 安装mininet virtual machine，这在实验二的时候介绍过了，就不重复介绍了。
4. 安装putty，安装putty的方法在实验二也介绍过了，也不重复介绍了。
5. 实验内容
6. 网络拓扑结构

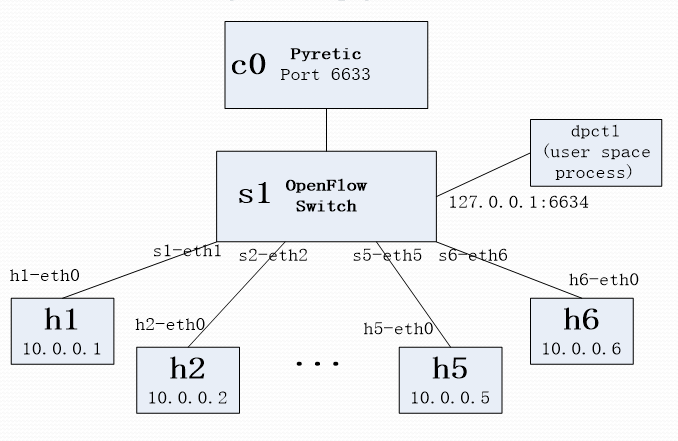
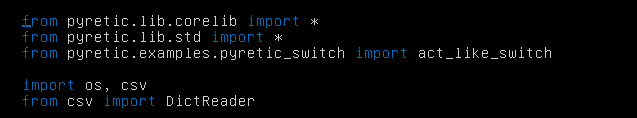


图1 网络拓扑结构

我们需要实现的网络拓扑结构如图1。这是一个简单的网络拓扑结构。一个OpenFlow交换机控制着6个主机，而交换机又由Pyretic控制器控制着。

1. 防火墙代码与相关解释
2. 引入头文件



1. 确定防火墙规则文件地址



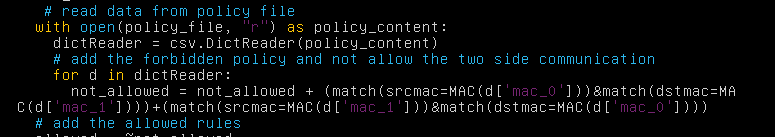
1. 主函数：
   1. 初始化防火墙规则为none



* 1. 读入防火墙规则文件的数据



* 1. 根据防火墙规则文件的数据确定不训练连接的网络链路



* 1. 确定允许连接的网络链路，它相当于不允许的链路取反



* 1. 输出当前的允许的链路连接规则

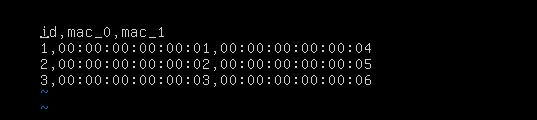


* 1. 将允许的规则传递act\_like\_switch()



我将这个代码保存在了$HOME/pyretic/pyretic/examples/pyretic\_firewall.py中

1. 防火墙规则



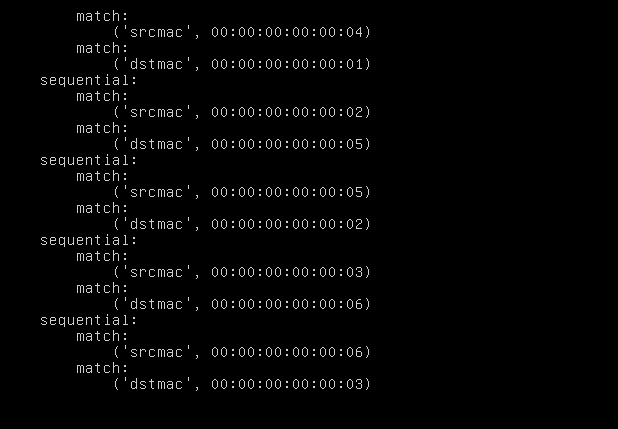
我制定的规则是节点1和节点4之间不能相互通信，节点2和节点5之间不同相互通信，节点3和节点6之间不同相互通信。我将这个防火墙规则文件保存在$HOME/pyrertic/pyretic/examples/firewall-policies.csv中。

1. 实验结果
2. 启动控制器

输入 pyretic.py -v high pyretic.examples.pyretic\_firewall

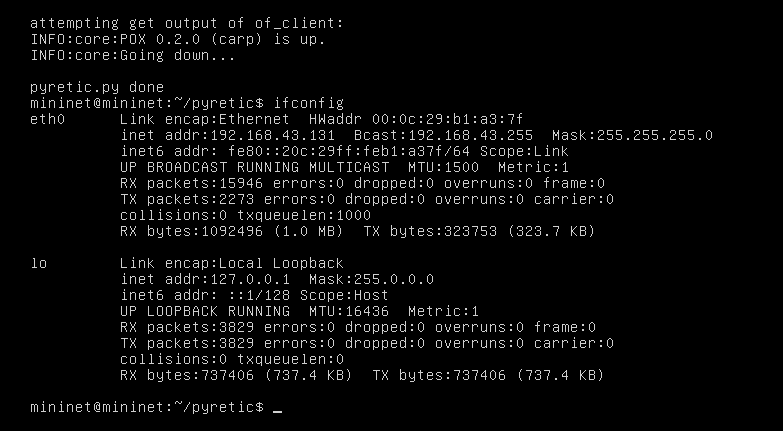


得到如下的执行结果：



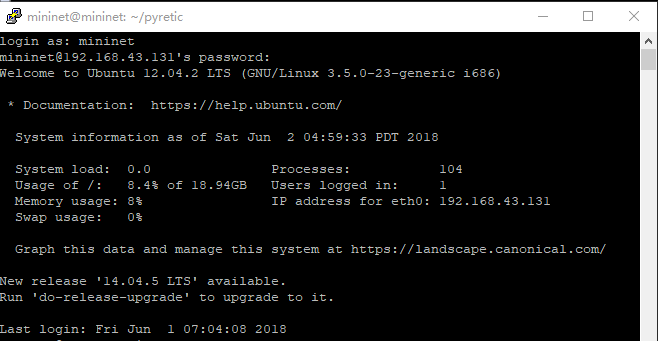
这是允许的链路规则，说明我们的防火墙代码已经生效了。

1. 创建拓扑
   1. 首先先打开putty，将其连接到我的虚拟机上。连接虚拟机需要ip地址，所以我先用ifconfig查看了我的ip地址。我当前的ip地址如下：

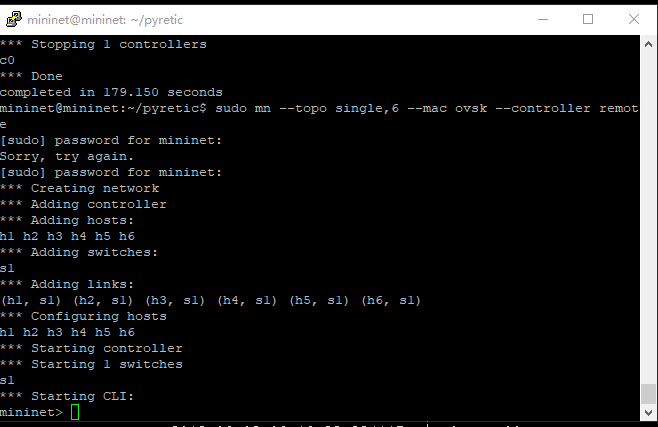


从图中的eth0的inet addr可以看出我的ip地址是192.168.43.131

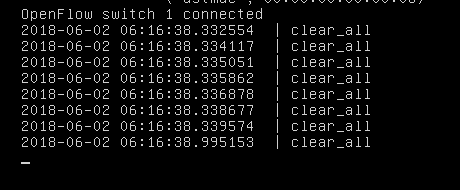
* 1. 通过putty连接上虚拟机，然后输入账号密码登陆。登陆后的结果如下图：



* 1. 创建一个6个节点，一个交换机的网络。

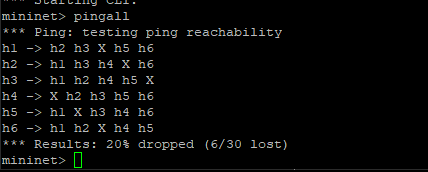


* 1. 此时可以发现在虚拟机上，出现了以下信息：



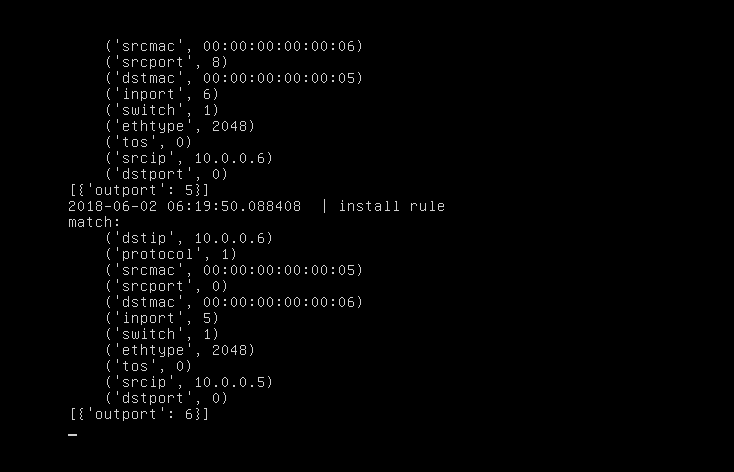
这说明网络拓扑正式建立，网络拓扑中的控制机也连接上了。

1. Pingall测试
   1. 我们在mininet中进行pingall测试，可以得到如下的结果：



* 1. 同时我们可以发现在虚拟机会显示如下的信息：

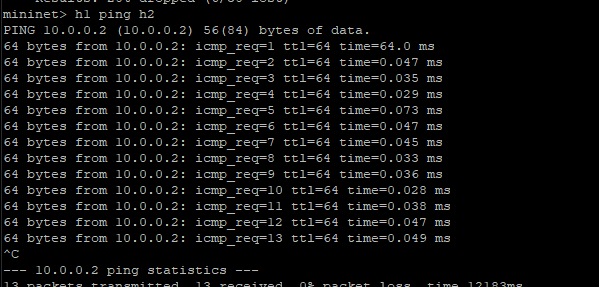
下面是部分信息截图

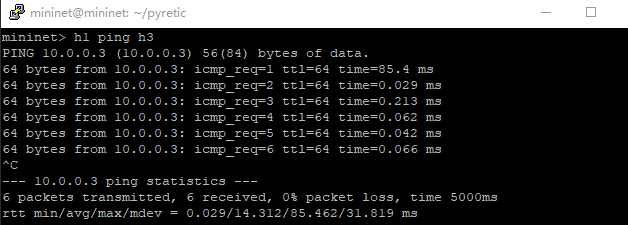


1. 单个交换机Ping测试

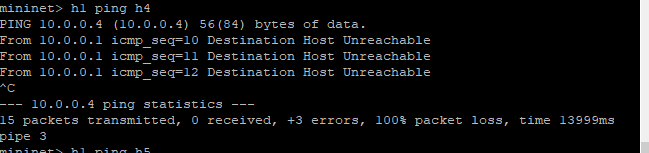
以节点1对其他节点的连通性为例，进行测试：

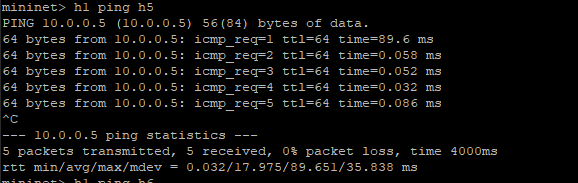
节点1与节点2：

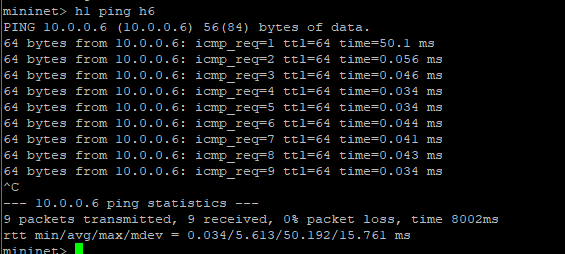


节点1与节点3：  


节点1与节点4：



节点1与节点5：  


节点1与节点6：  


从以上的结果图，我们可以很容易的看出节点1和节点2，3，5，6之间是连通的，而与节点4是不连通的。

1. 总结
2. 从实验结果上我们可以看出我们防火墙生效了。因为在pingall的时候，节点1与节点4、节点2与节点5、节点3与节点6是两两不连通的，而其余的节点是相互联通的。这和我们设定的防火墙规则是符合的。同时从节点1的单节点ping测试，我们也可以直观的看出防火墙的效果。
3. 本次实验使用了Pyretic控制器，使用控制器实现了Mac层上的防火墙相关的策略。也很容易从本次Project中看出，使用SDN网络实现防火墙策略的管理比传统网络更加容易。
4. 在做本次实验的时候有一个需要注意的就是，我们在写防火墙规则的时候，每一列之间要用逗号空格，因为Python csv的DictReader只能识别这种格式。我刚开始使用tab相隔，结果代码不能执行。