**电子科技大学**

**实**

**验**

**报**

**告**

学生姓名：李昊霖

学号：2019270102007

课程名称：计算机网络基础

指导教师：杨宁

日期： 2021 年 6月 22日

实验项目名称： SDN实验3

报告评分： 教师签字：

**一、实验概要**

实验目的:

学习使用opendaylight控制器以及postman工具构建sdn网络，下发流表项以实现不同的操作。

实验内容：

1.安装OpenDaylight 控制器

2.创建ovs网桥

3.连接网桥与odl并测试

4.使用Yang UI工具下发流表项

5.使用postman工具下发流表项

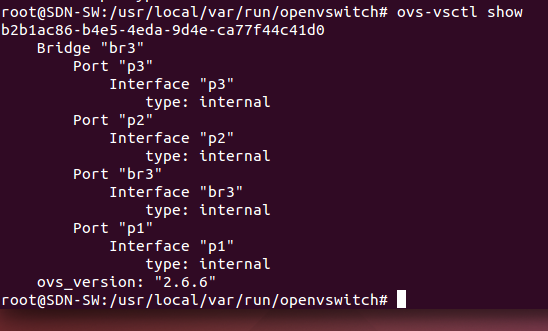
6.回答实验指导书中的问题

**二、实验数据及分析结果**

**记录：**

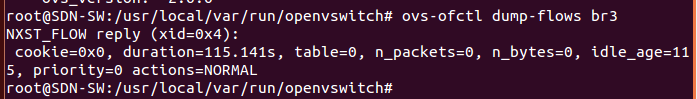
实验3.2：

1. ovs-vsctl show 命令及其结果截图



2. 查看 br3 流表项的 ovs-ofctl 命令及其结果截图

命令：ovs-ofctl dump-flows br3



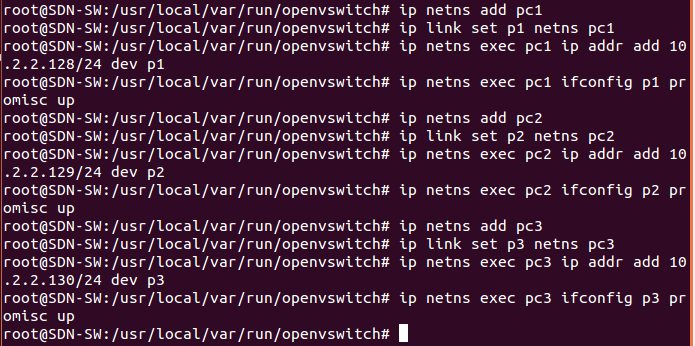
3. 创建 pc2 和 pc3 并连接网桥 br3 的命令

# ip netns add pc2/3

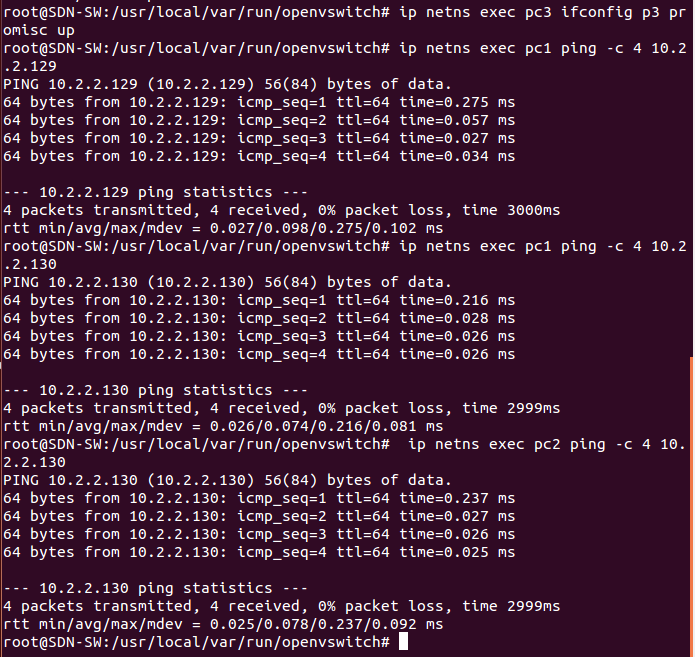
# ip link set p2/3 netns pc2/3

# ip netns exec pc2/3 ip addr add 10.2.2.129（130）/24 dev p2/3

# ip netns exec pc2/3 ifconfig p2/3 promisc up

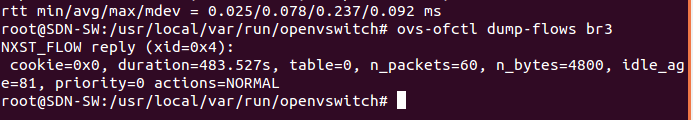


4. 以上三个命令及其结果截图



5. 查看 br3 流表项的 ovs-ofctl 命令及其结果截图

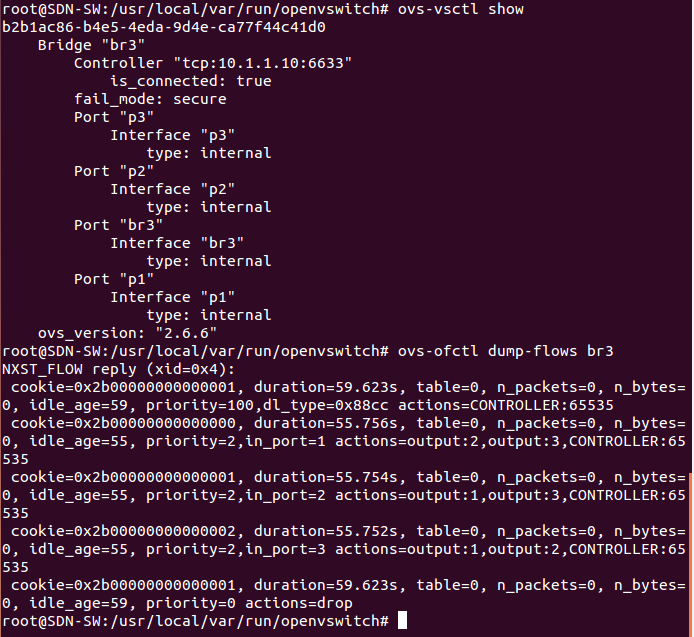
命令：ovs-ofctl dump-flows br3



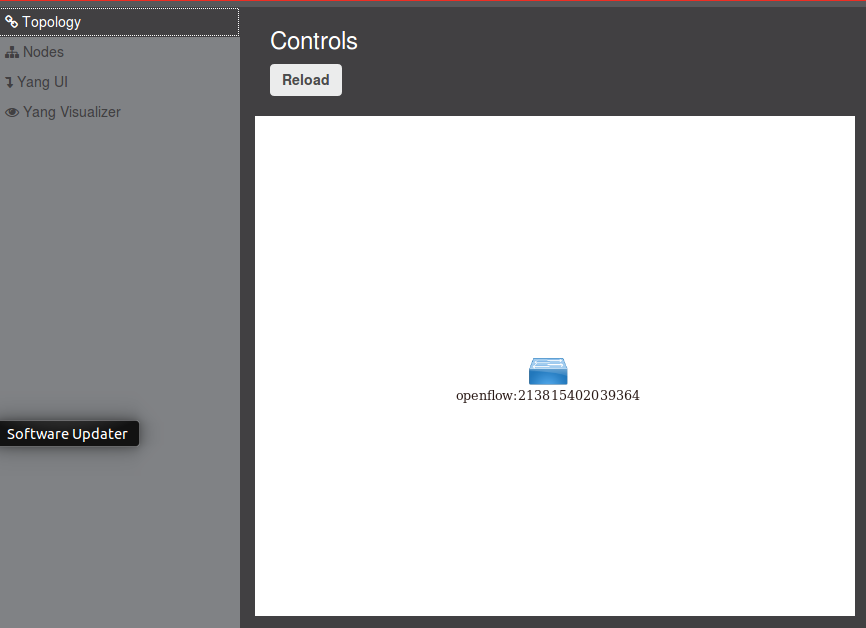
实验3.3：

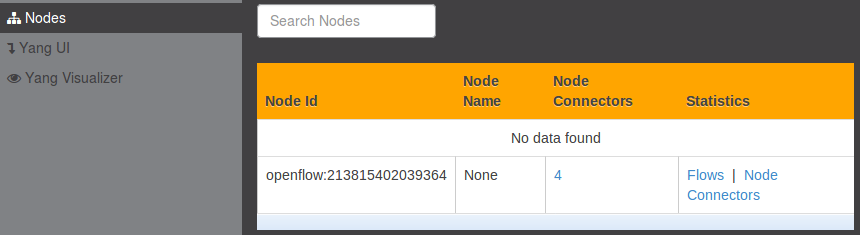
7. SDN-SW 虚拟机上的 ovs-vsctl show 命令及其结果截图，以及查看 br3

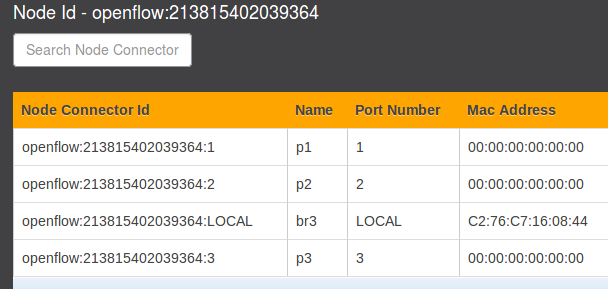
流表项的 ovs-ofctl 命令及其结果截图

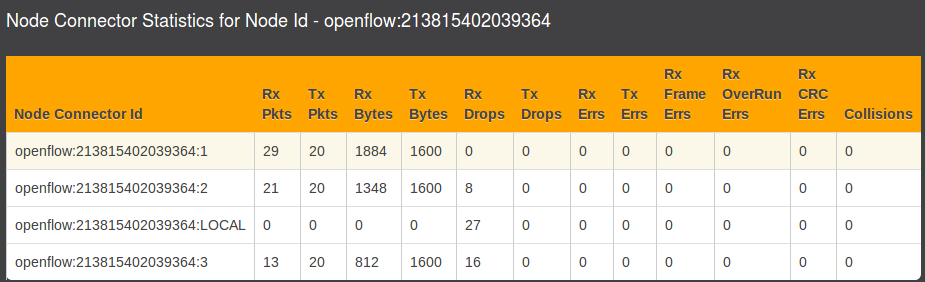


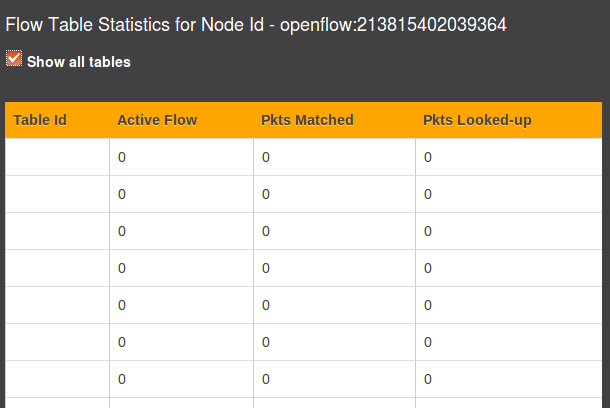
8. SDN-Controller 虚拟机上的 Topology 页面截图、Nodes 页面截图、Node Connectors 列中详细信息截图、Statistics 列中的 Flows 详细信息截图（选中“Show all tables”，第一屏的截图即可）、Statistics 列中的 Node Connectors 详细信息截图



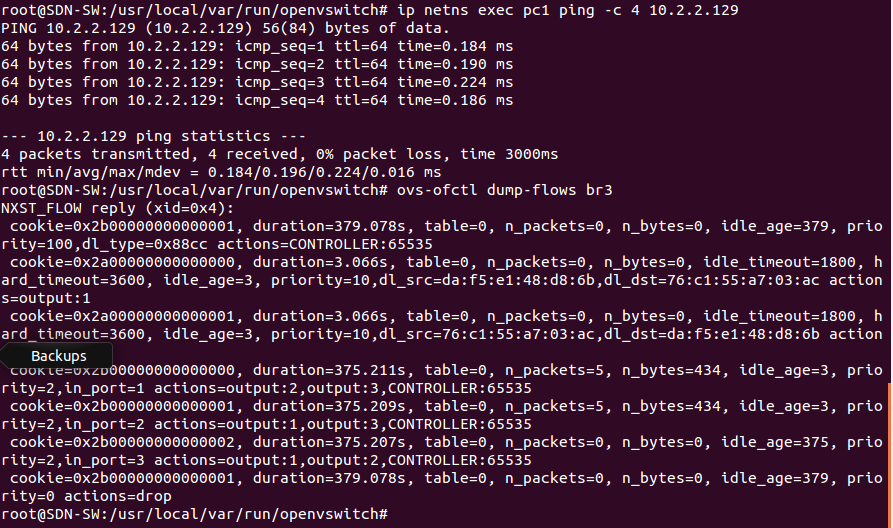




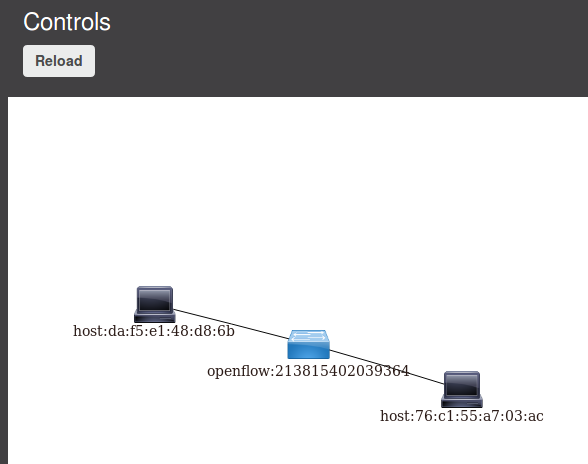




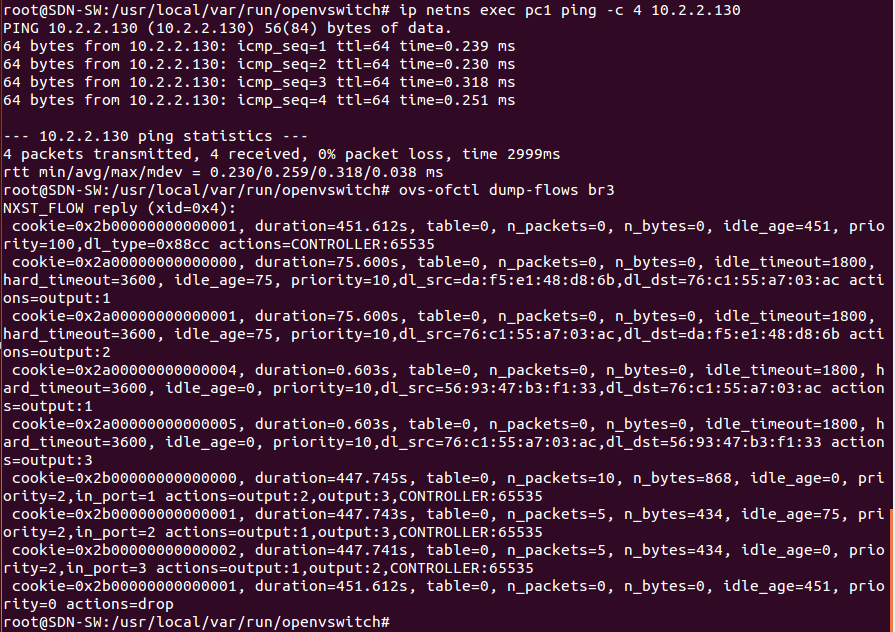
9. SDN-SW 虚拟机上的 ping 命令及其结果截图、查看 br3 流表项的 ovs-ofctl 命令及其结果截图



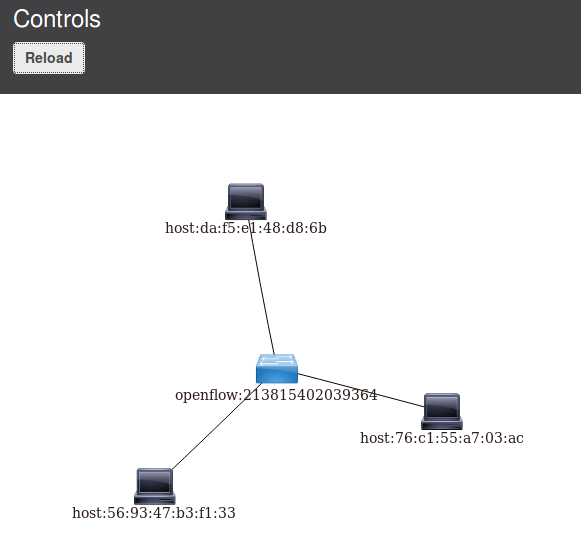
10. SDN-Controller 虚拟机上的 Topology 页面截图。



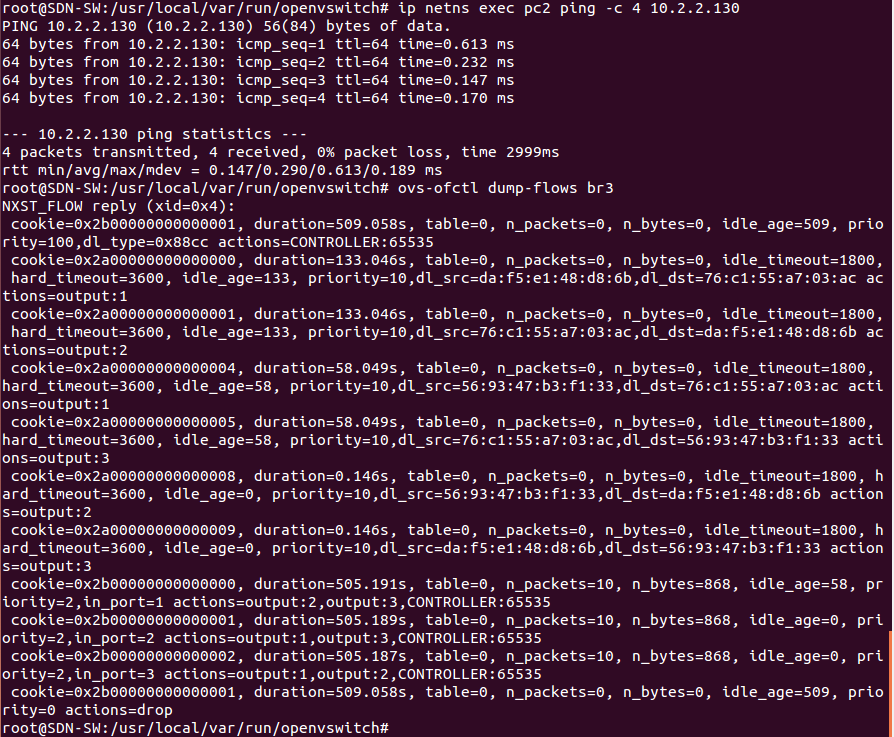
11. SDN-SW 虚拟机上的 ping 命令及其结果截图、查看 br3 流表项的 ovsofctl 命令及其结果截图



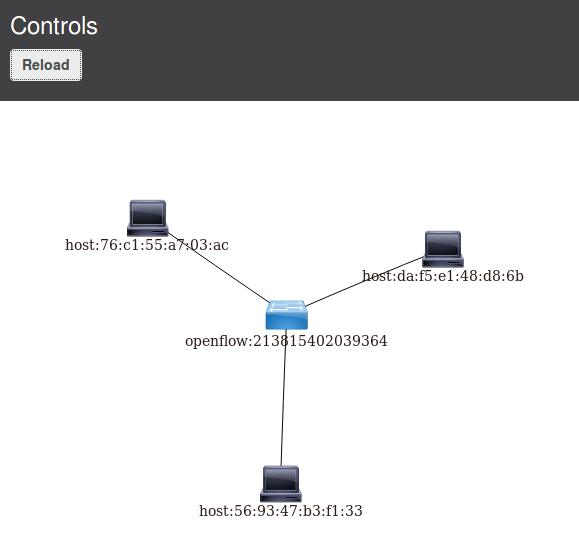
12. SDN-Controller 虚拟机上的 Topology 页面截图



13. SDN-SW 虚拟机上的 ping 命令及其结果截图、查看 br3 流表项的 ovsofctl 命令及其结果截图



14. SDN-Controller 虚拟机上的 Topology 页面截图



Q1、步骤 1 中 br3 的流表项与实验 3.2 步骤 6 中的流表项有哪些不同？请尝试分析本步骤中每条流表项的作用（即匹配哪种特征的分组？做出什么动作？）。

答：步骤 1 中 br3 有5条流表项，实验 3.2 步骤 6中仅有一条normal流表项，表示按照传统交换机的2层或者3层进行转发处理，优先级为0.

本步骤中 第一条流表项优先级为100，dl\_type=0x88cc表示链路层发现协议，action为controller表示将packet封装并转发给控制器。

第二条流表项OpenFlow 端口2输入的数据包输出到端口1，3；第三条流表项表示将端口1输入的数据包输出到其它端口；第四条流表项表示将端口3输入的数据包输出到其它端口。三者优先级均为2。

最后一条流表项的action为drop，但优先级为0。

Q2、相比于步骤 1，步骤 2 的 br3 新增了哪几条流表项？这些流表项是通过

哪种类型的 OpenFlow 报文生成的？请分析本步骤中新增的每条流表项的作用

（即匹配哪种特征的分组？做出什么动作？）。

答：新增两条流表项。

由sdn培训课程2可知，modify-state类型的报文用来操作流表，故流表项应为modify-state报文（controller to switch报文的子类型）生成的。

流表项优先级=10，空闲超时时间为1800s，匹配以太帧首部源和目的MAC地址字段，action分别为输出到端口2，1。在两个host之间建立连接。

Q3、相比于步骤 1，步骤 2 的 ODL 拓扑图中新增了哪些拓扑信息？ODL 是通

过哪种类型的 OpenFlow 报文获得这些拓扑信息的？

答：新增了两个host，其mac地址匹配流表项中的两个地址。

由sdn培训课程2可知，asynchronous-configuration类型的报文用于通告消息或转发报文到控制器，这里ODL应通过packet-in类型报文获得拓扑信息。

Q4、相比于步骤 2，步骤 3 的 br3 新增了哪几条流表项？这些流表项是通过

哪种类型的 OpenFlow 报文生成的？请分析本步骤中新增的每条流表项的作用

（即匹配哪种特征的分组？做出什么动作？）。

答：新增两条流表项。modify-state报文生成。

作用为匹配以太帧首部源和目的MAC地址字段，action分别为输出到端口3，1。新建一条两个host之间的连接。

Q5、相比于步骤 2，步骤 3 的 ODL 拓扑图中新增了哪些拓扑信息？ODL 是通

过哪种类型的 OpenFlow 报文获得这些拓扑信息的？

答：新增了1个host。ODL应通过packet-in类型报文获得信息。

Q6、相比于步骤 3，步骤 4 的 br3 新增了哪几条流表项？这些流表项是通过

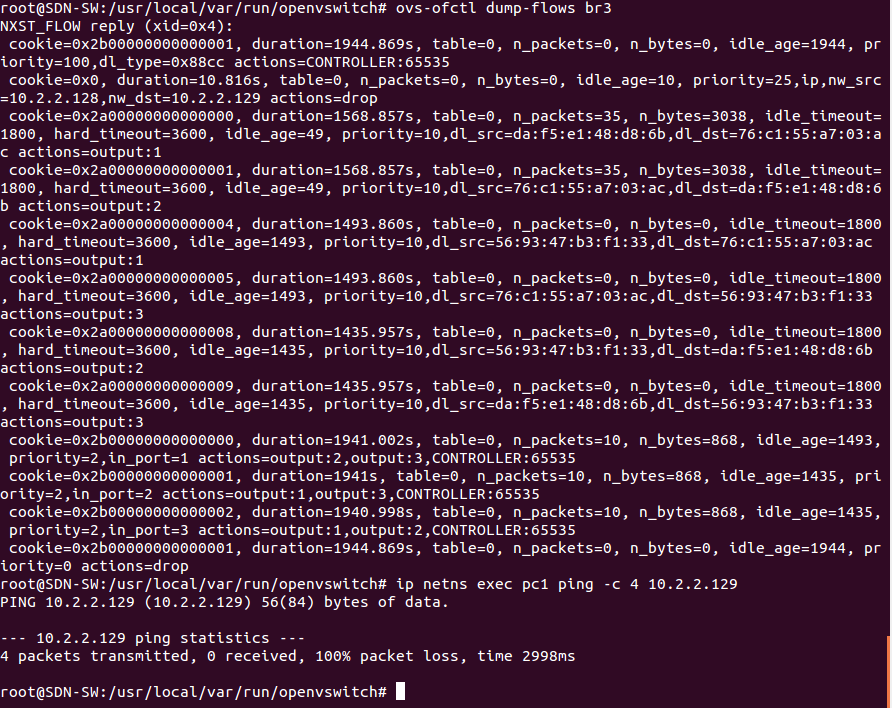
哪种类型的 OpenFlow 报文生成的？请分析本步骤中新增的每条流表项的作用

（即匹配哪种特征的分组？做出什么动作？）。

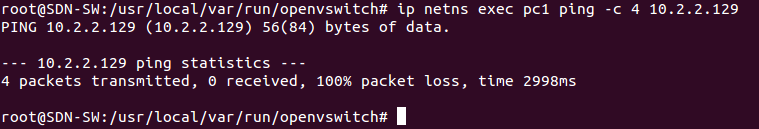
答：新增两条流表项。modify-state报文生成。

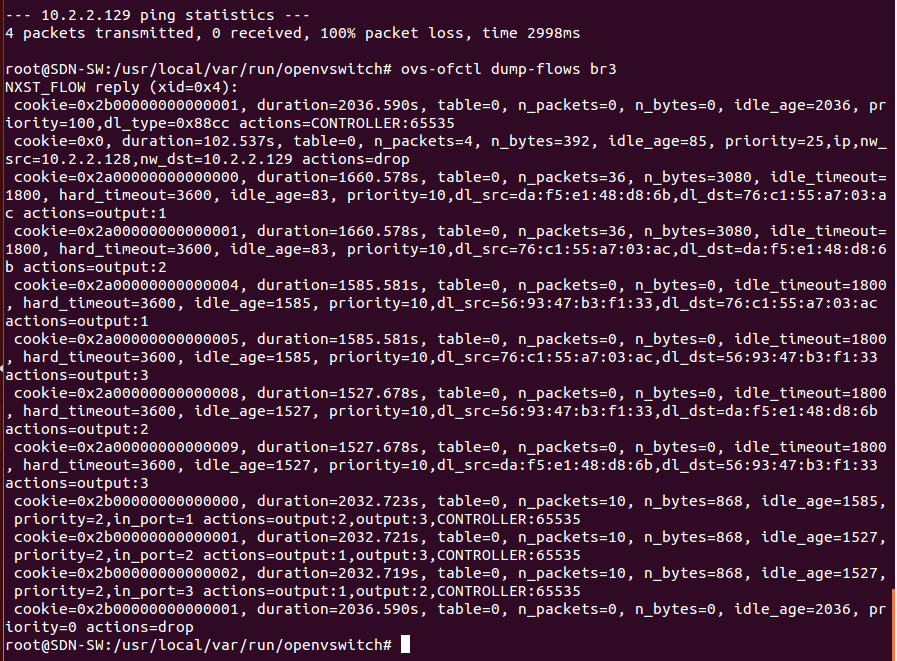
作用为匹配以太帧首部源和目的MAC地址字段，action分别为输出到端口2，3。再次两个host之间的连接，至此三个host之间已经两两建立起连接。

15. SDN-SW 虚拟机上查看 br3 流表项的 ovs-ofctl 命令及其结果截图



16. 以上命令及其结果截图

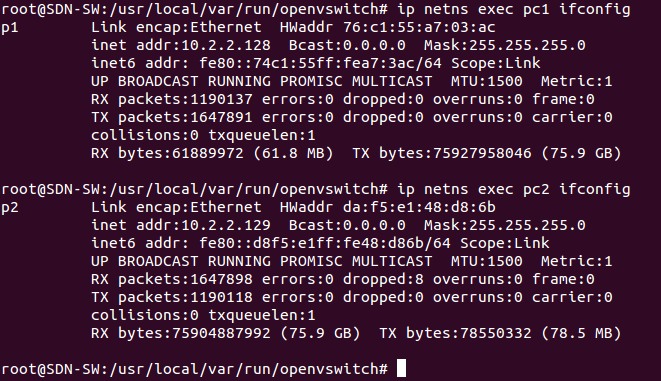




（ping测试后流表项）

请对比 ping 测试前后 br3 中各流表项的 n\_packets 值

Q7、步骤 3 的 PC1 ping PC2 时，PC1 发出的 ping 请求报文匹配 br3 上的哪一条流表项？br3 如何处理这些 ping 请求报文？请给出依据截图。

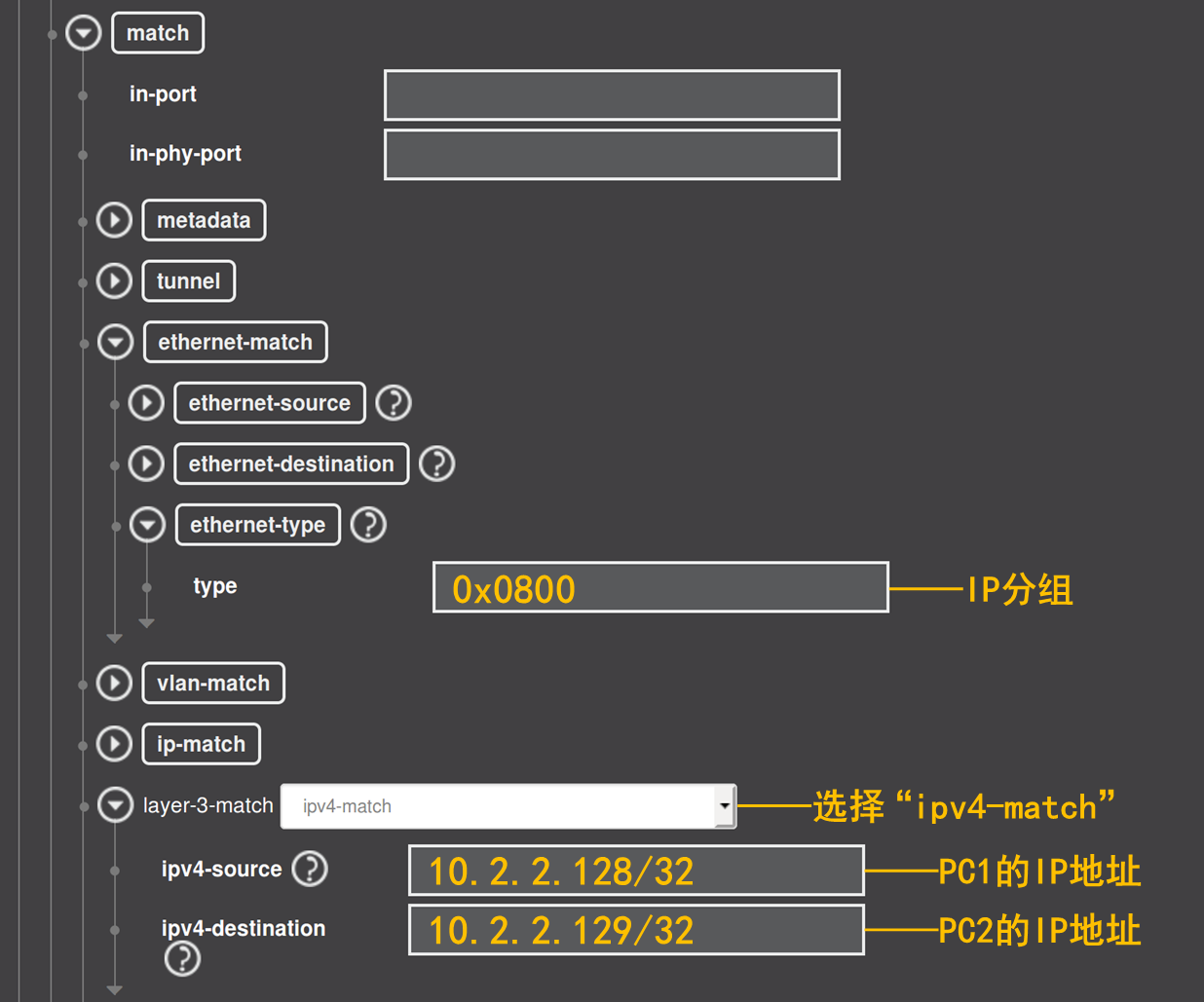
答：匹配br3中，目的ip地址为10.2.2.129，源ip地址为10.2.2.128，优先级为25，actions=drop的流表项

Q8、如果是 PC2 ping PC1，其 ping 报文交互结果与步骤 3 有哪些不同？请说明理由。

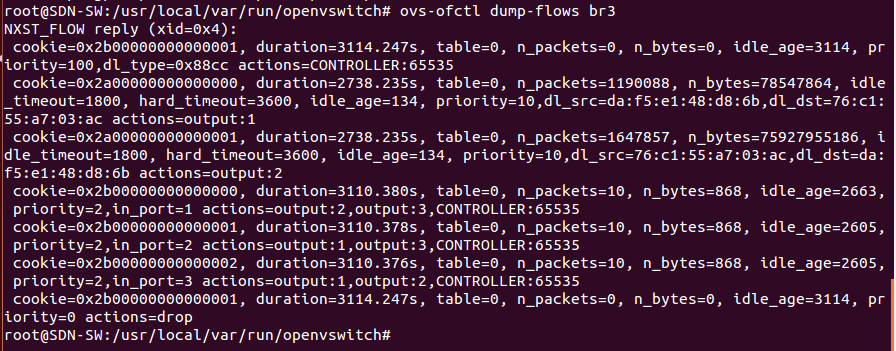
答：交互结果为上图中两条匹配mac地址的流表项顺序交换，内容相同。由于从pc1到pc2的数据drop，所以无论是pc1 ping pc2还是反之都无法ping通。

Q9、如何能让 PC2 ping PC1 的报文交互结果与步骤 3 一样

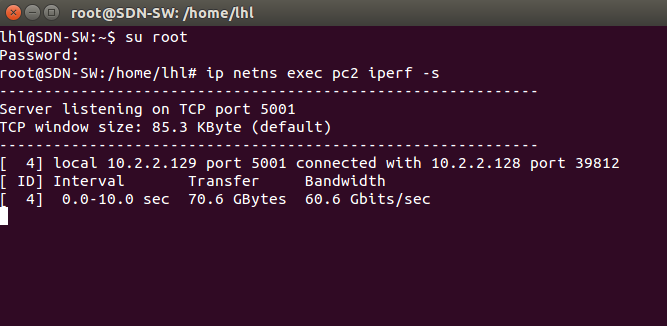
答：将下发的流表项的source和destination交换，如下图source改为pc2的ip地址，destination改为PC1的ip地址。

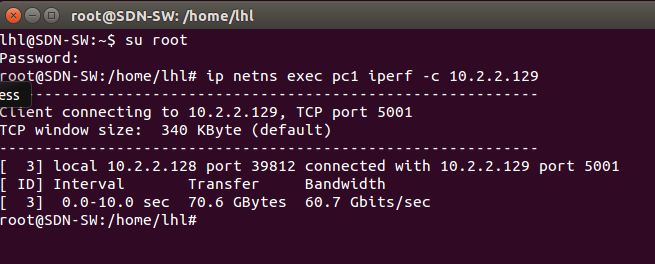


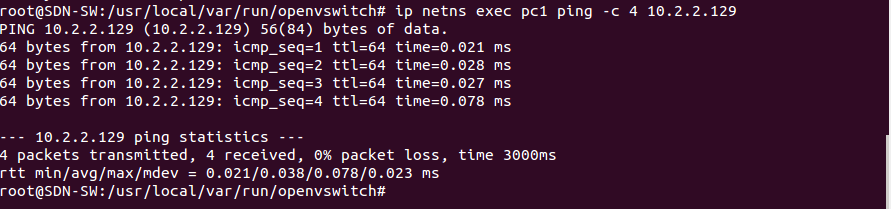
17. SDN-SW 虚拟机上查看 br3 流表项的 ovs-ofctl 命令及其结果截图



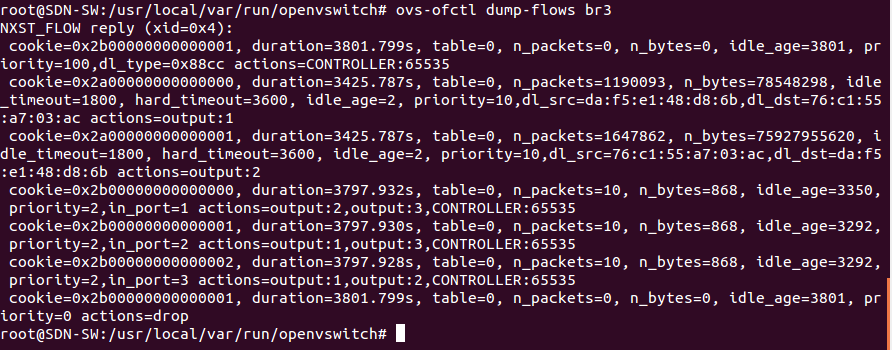
18. 以上命令及其结果截图







请对比 ipfer 测试和 ping 测试前后 br3 中各流表项的 n\_packets 值



（测试后流表项）

Q10、pc1 的 iperf 客户端发送给 pc2 的 iperf 服务器端的 TCP 性能测试数据匹配 br3 上的哪一条流表项？br3 如何处理这些测试数据？请给出依据截图。

答：TCP性能测试数据匹配br3上的第二条流表项

Q11、pc2 的 iperf 服务器端返回的测试统计反馈数据匹配 br2 上的哪一条流

表项？br3 如何处理这些测试统计反馈数据？请给出依据截图。

答：匹配上图中dl\_src= da:f5:e1:48:d8:6b的流表项，其源地址为PC2的mac地址，目的地址为PC1的mac地址，br3将对应数据输出至相应的端口1。iperf测试前后，n\_packet值分别为1190088，1190093，同样增加5个packet。

Q12、为什么 pc1 和 pc2 之间能进行正常的 iperf 测试，却不能进行 ping 通信？请详细说明。

答：L3流表项使pc1到pc2的数据被drop，而L4流表项仅使pc1 能作为iperf 客户端向 pc2 正常发起 TCP 性能测试，只有通过iperf 服务器端口号的数据包能正常接收。

**三、总结及心得体会**

通过本次实验我学习到了如何使用opendaylight控制器控制sdn网络，让我sdn网络架构，网络间通信有了更深刻的认识

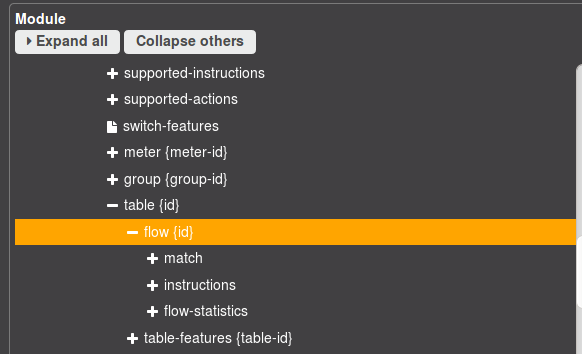
**四、对本实验过程及方法、手段的****改进建议**

实验过程中碰到的问题及解决方案：

实验指导书中的实验3.4-YANG UI操作示例和实验3.5-Postman操作示例产生的URL相同，即均为http://127.0.0.1:8181/restconf/config/opendaylight-inventory:nodes/node/openflow:xxxxxx/flow-node-inventory:table/0

因此OVS会认为这两次的PUT操作报文是对同一个URL对象的修改，而不是添加。

所以如果要添加，那么向此操作的URL必须不同。解决方法是在进行实验3.4时，YANG UI下发的流表项保持在table级（即指导书步骤），而在实验3.5中使用YANG UI生成代码时选择flow级的流表项（如下图），这样通过“show preview”得到的地址是流表级，两者url不同；或在3.4中使用flow级的流表项，而在3.5中生成table级的流表项对应代码。



改进建议：

如下图，实验中部分问题需要对比ping（iperf）测试前后的流表项，但实验未要求记录测试后的流表项，如果直接进行下一步会导致未记录流表项，相关问题难以回答，需要返回到某一步从新进行实验。

附：table级json消息代码

http://127.0.0.1:8181/restconf/config/opendaylight-inventory:nodes/node/openflow:213815402039364/flow-node-inventory:table/0

{

"table": [

{

"id": "0",

"flow": [

{

"id": "2",

"match": {

"ip-match": {

"ip-protocol": "6"

},

"ipv4-source": "10.2.2.128/32",

"ipv4-destination": "10.2.2.129/32",

"tcp-destination-port": "5001"

},

"instructions": {

"instruction": [

{

"order": "0",

"apply-actions": {

"action": [

{

"output-action": {

"output-node-connector": "2"

},

"order": "0"

}

]

}

}

]

},

"priority": "25",

"idle-timeout": "0",

"table\_id": "0"

}

]

}

]

}