**第 三 次 实 验 报 告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 课程名称 | **网络安全实验** | | | | |
| 学生姓名 | 应晓宇 | 学号 | 2021302181140 | 指导老师 | 陈治宏 |
| 专业 | 网安 | 班级 | 7 | 实验时间 | 24.4.3 |

1. **实验介绍**
   * + 1. **实验名称：防火墙实验**
       2. **实验任务：**

任务一 了解Iptables防火墙基础；

任务二 掌握Iptables主机防火墙常用配置和审计功能；

任务三 了解虚拟网络中常用软件；

任务四 利用虚拟网络构建网络防火墙实验场景；

任务五 使用Iptables实战网络防火墙NAT、网络控制；

任务六 Iptables实战公有云二层防火墙。

* + - 1. **实验目的：**

了解Linux防火墙iptables的基础概念，掌握其基本命令，会进行简单的规则的增删查。

掌握IPtables工作中的常用策略。

掌握iptables的审计策略。

通过Iptables工作中的常用策略以及审计策略的学习和使用，具备丰富的主机安全防护的能力。

了解openswitch与namespace的概念。

掌控Linux虚拟网络设备的使用。

掌握SDN交换机openvswitch的安装与使用。

掌握利用openswitch与netns来搭建虚拟网络环境。

掌握iptables nat的基础知识与常用配置的操作。

掌握简单的网络防火墙的访问控制操作。

掌握企业公有云中二层防火墙实现的一种方法

通过以上工具和知识的学习和使用，能够融会贯通，掌握iptables防火墙配置的原理和方法，掌握自主学习和实践企业网络防火墙的搭建与配置策略，使所处工作环境处于安全的状态。

* + - 1. **实验工具：**

openvswitch

ovs-vsctl (openvswitch软件提供)

iptables

brctl

ip (包含ip link、ip netns、ip addr等)

tcpdump

ethtool

route

sysctl

modprobe

ipset

* + - 1. **实验环境：**



1. **实验内容**
2. **任务一**

**【任务描述】**

本实验任务基于个人主机操作环境，主要完成以下内容：

了解Iptables的基本概念与相应基础知识，熟悉使用语法。

Iptables规则的增删改查的简单使用。

通过完成本实验任务，要求学生了解iptables的基础概念，掌握使用语法，会进行简单规则的增，删，改，查；为完成后续防火墙实验任务奠定坚实的基础。

**【实验目标】**

了解IPtables的基本概念与基础知识，熟悉使用语法。

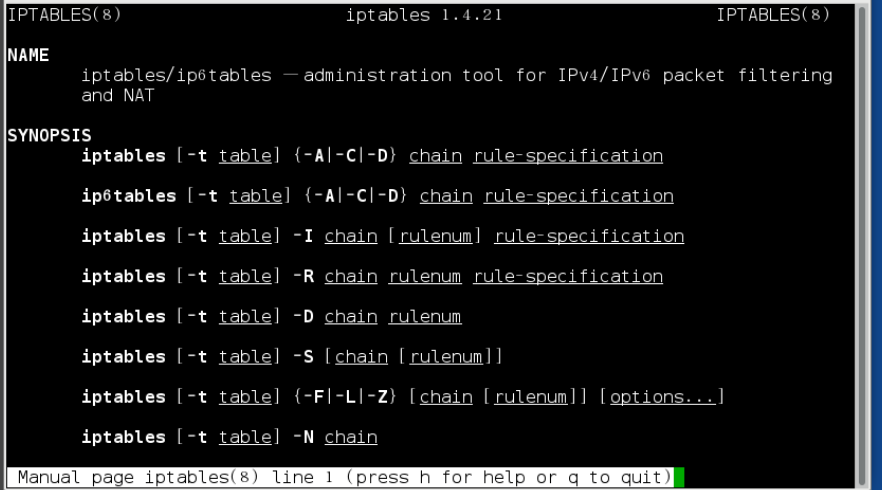
会进行简单规则的增删查。

**【实验工具】**

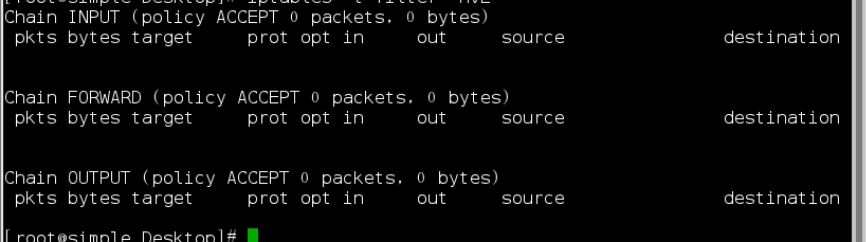
**iptables**

**【操作步骤】**

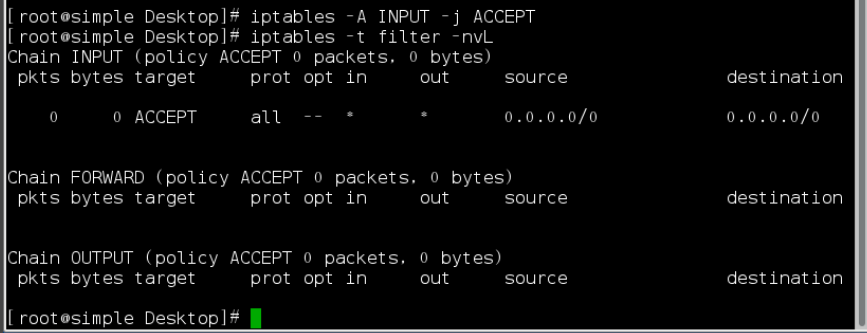
1. 使用man手册



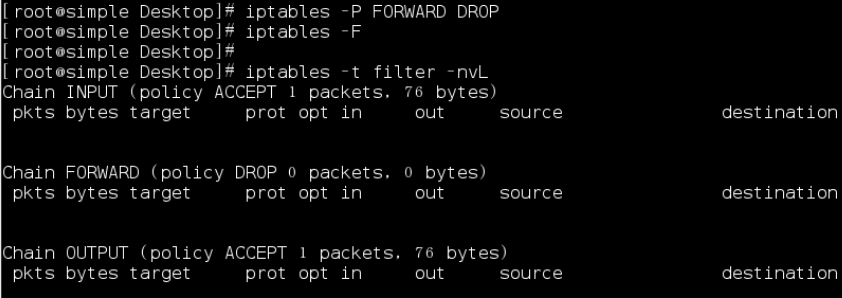
2.查看filter表规则

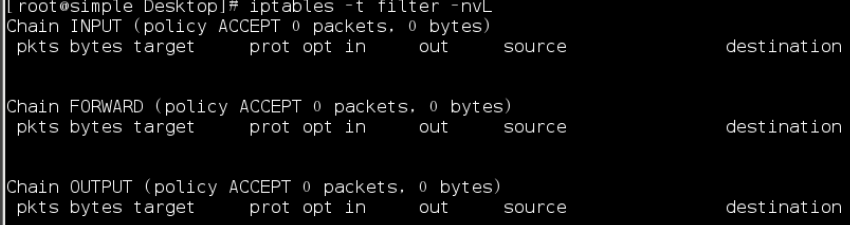


3.添加规则,允许所有网络访问本机

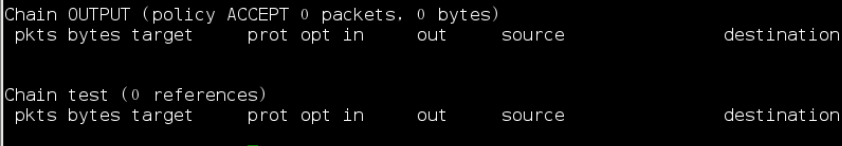


4.为FORWARD链添加默认规则





6.添加自定链test



7.删除自定链test



**2.任务二**

【**任务描述**】

本实验任务基于个人主机操作环境，主要完成以下内容：

iptables的常用语法的实际操作。

进行Iptables的特殊参数，自定义策略的的操作。

启用Iptables日志记录的功能，配置相应的策略，进行操作，验证。

通过完成本实验任务，要求学生掌握Iptables主机防火墙的常用配置已审计策略，具体包括：Iptables的常用语法，状态监测，特殊参数，自定义策略的使用，以及启用日志记录的功能进行相应的策略配置，具备更为夯石的防火墙配置使主机安全的能力。

【**实验目标**】

掌握Iptables工作中的常用策略。

掌握审计策略。

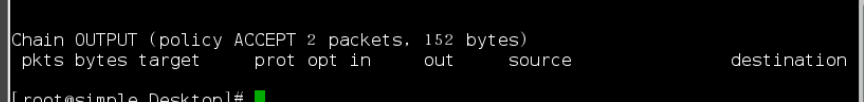
通过Iptables工作中的常用策略以及审计策略的学习和使用，具备丰富的主机安全防护的能力。

【**实验工具**】

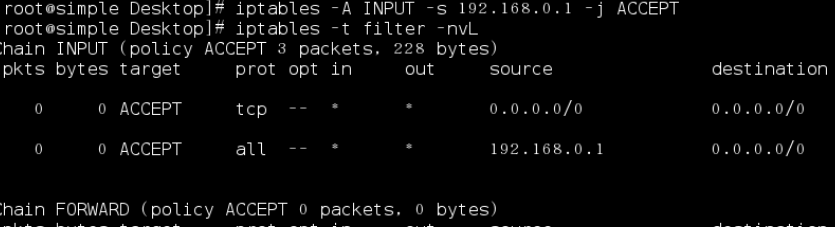
**Iptables**

【**操作步骤**】

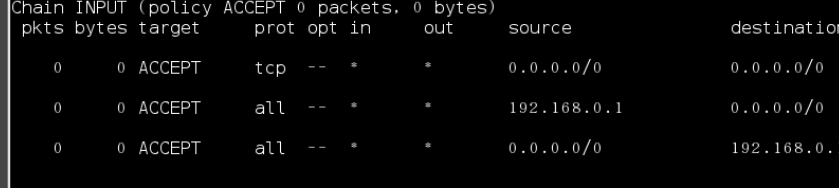
1.匹配指定协议iptables -A INPUT -p tcp -j ACCEPT



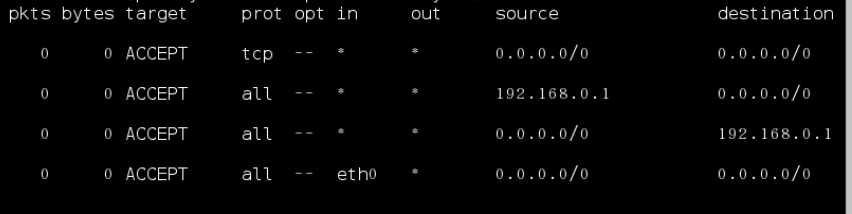
2.以IP源地址匹配包iptables -A INPUT -s 192.168.0.1 -j ACCEPT



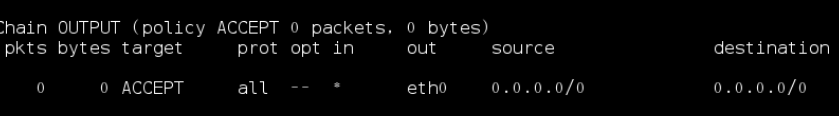
3.以IP目的地址匹配包iptables -A INPUT -d 192.168.0.1 -j ACCEPT



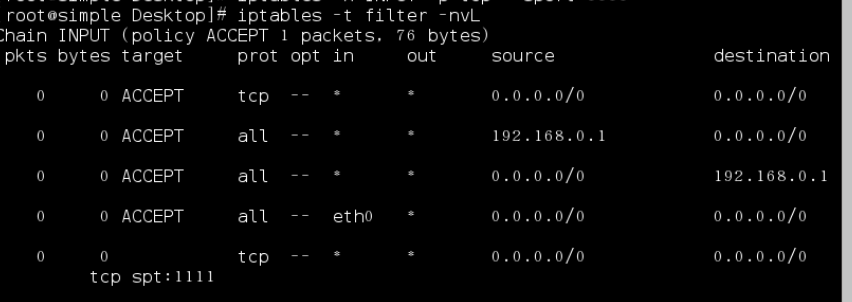
4.以包进入本地使用的网络接口匹配包iptables -A INPUT -i eth0 -j ACCEPT



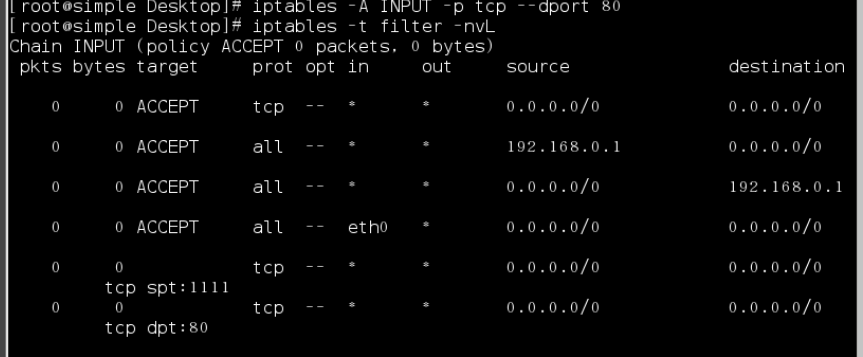
5.以包离开本地所使用的网络接口来匹配包 iptables -A OUTPUT -o eth0 -j ACCEPT



6.匹配通信源端口iptables -A INPUT -p tcp --sport 1111



7.匹配通信源端口iptables -A INPUT -p tcp --dport 80

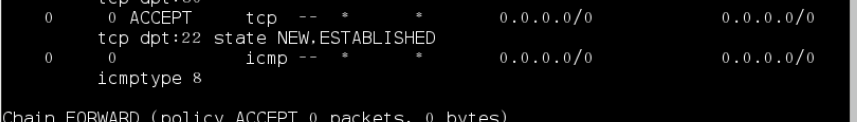


8.状态检测-m state --state {NEW,ESTATBLISHED,INVALID,RELATED},指定检测哪种状态iptables -A INPUT -p tcp --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT



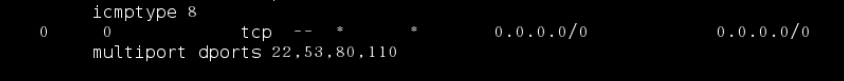
9.特殊参数

--icmp-type 指定ICMP的类型编号iptables -A INPUT -p icmp --icmp-type 8



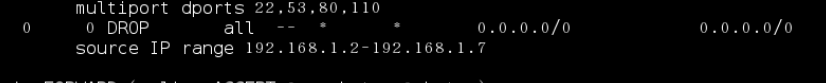
-m multiport 指定多端口号--sport --dport --ports

iptables -A INPUT -p tcp -m multiport --dport 22,53,80,110



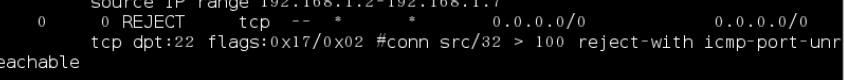
-m iprange 指定IP段--src-range ip-ip --dst-range ip-ip

iptables -A INPUT -m iprange --src-range 192.168.1.2-192.168.1.7 -j DROP



-m connlimit 连接限定--comlimit-above # 限定大连接个数

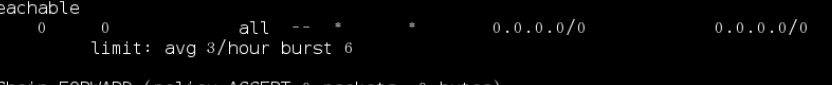
iptables -A INPUT -p tcp --syn --dport 22 -m connlimit --connlimit-above 100 -j REJECT



-m limit 现在连接速率，也就是限定匹配数据包的个数

--limit # 指定速率 --limit-burst # 峰值速率，最大限定

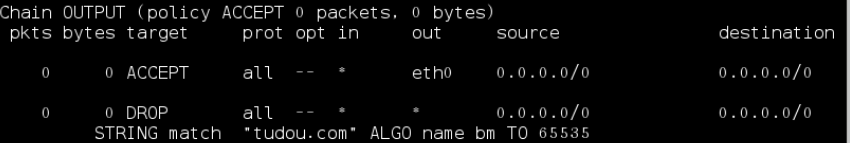
iptables -A INPUT -m limit --limit-burst 6



-m string 按字符串限定

--algo bm|kmp # 指定算法bm或kmp|--string “STRING” # 指定字符串本身

iptables -A OUTPUT -m string --string “tudou.com” --algo bm -j DROP



1. iptables常用的操作语法

| 功能 | 参数 | 语法 | 说明 |
| --- | --- | --- | --- |
| 添加规则 | A | iptables -A INPUT -p tcp -j ACCEPT | 在INPUT链后加入允许所有TCP请求 |
| 删除规则 | -D | iptables -D INPUT -p tcp -j ACCEPT | 在INPUT链中删除对应规则 |
| 自定义链重命名 | -E | iptables -E A B | 将自定义链A重命名为B，原来的名字在前，新名字在后 |
| 清空规则 | -F | iptables -F INPUT | 清空INPUT链规则, 如果不加链名则是清除当前表所有规则 |
| 插入规则 | -I | iptables -I INPUT 1 -p tcp -j ACCEPT | 在INPUT链内的某个位置插入规则，如果序号为1或没有序号，规则会被插入到的头部 |
| 显示规则 | -L | iptables -L INPUT | 显示INPUT链的所有规则，如果没有指定链，则显示指定表中的所有链。精确输出可用-n和-v等参数 |
| 新建自定义链 | -N | iptables -N A | 在指定表新建链A,不可以同名 |
| 默认策略 | -P | iptables -P INPUT DROP | 指定INPUT链的默认策略为DROP, 可选策略ACCEPT、DROP、REJECT、REDIRECT |
| 替换规则 | -R | iptables -R INPUT 1 -p udp -j ACCETP | 替换INPUT链中的第1条策略 |
| 删除用户自定义链 | -X | iptables -X A | 删除指定表中的自定义链A |
| 计数器归零 | -Z | iptables -Z | 清空指定表下指定链（如未指定则认为所有链）的所有计数器归零 |

11.iptables日志记录

Linux下单独记录Iptables日志,编辑/etc/syslog.conf文件，加入一行

# echo “kern.warning /var/log/iptables.log” >>/etc/rsyslog.conf

重启rsyslog服务生效：

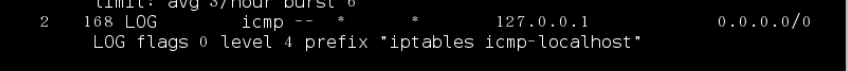
# systemctl restart rsyslog

(1) 配置iptables策略

# iptables -A INPUT -s 127.0.0.1 -p icmp -j LOG --log-prefix “iptables icmp-localhost “



(2) 验证规则 # ping -c 1 127.0.0.1，有两个数据包匹配



1. 查看日志 tail -f /var/log/iptables.log

(4) iptables日志字段解释(编号21后为未用到的字段)

Jan 12 18:24:05 [localhost] kernel: iptables icmp-localhost IN=lo OUT= MAC=00:00:00:00:00:00:00:00:00:00:00:00:08:00 SRC=127.0.0.1 DST=127.0.0.1 LEN=84 TOS=0x00 PREC=0x00 TTL=64 ID=8718 PROTO=ICMP TYPE=0 CODE=0 ID=9413 SEQ=1

(5) 日志策略

a、 获取所有TCP日志

# iptables -A INPUT -p tcp -j LOG --log-prefix “iptables TCP”

b、 获取所有UDP日志

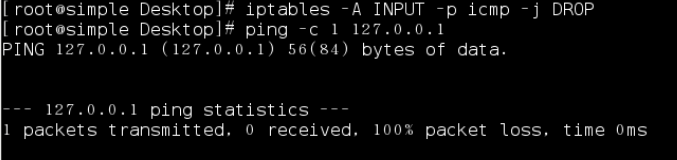
# iptables -A INPUT -p udp -j LOG --log-prefix “iptables UDP”

c、 获取ssh的日志

# iptables -A INPUT -p tcp --dport 22 -j LOG --log-prefix “iptables SSH”

12.自定义策略

(1)禁止ping 127.0.0.1



(2) 状态为已连接的放行

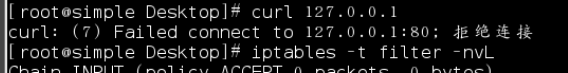
# iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

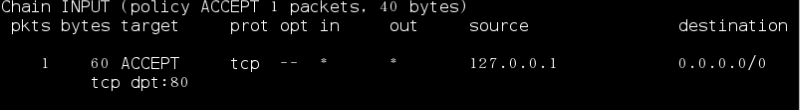


(3)只允许本机访问80

# iptables -A INPUT -p tcp --src 127.0.0.1 --dport 80 -j ACCEPT







(4) 利用扩展模块limit，可以实现DoS攻击防范

# iptables -A INPUT -p tcp --dport 80 -m limit --limit 25/minute --limit-burst 100 -j ACCEPT



**3.任务三**

【**任务描述**】

本实验在个人主机操作环境中虚拟网络的使用：openswitch与namespace，模拟多个网络设备，将不同类型的网络应用隔离。为下一任务搭建网络环境实验场景做环境准备。

通过完成本实验任务，要求学生掌握openswitch与namespace的使用，会构建虚拟网络环境。

【**实验目标**】

了解openswitch与namespace的概念。

掌控Linux虚拟网络设备的使用。

掌握SDN交换机openvswitch的安装与使用。

【**实验工具**】

Openswitch

ovs-vsctl(openvswitch软件提供)

IP(包含ip link、ip netns、ip addr等)

tcpdump

ethtool

【**操作步骤**】

1. 概述

（1）Open vSwitch

Open vSwitch（简称为 OVS）是由 Nicira Networks 主导的，运行在虚拟化平台（例如 KVM，Xen）上的虚拟交换机。在虚拟化平台上，OVS 可以为动态变化的端点提供 2 层交换功能，很好的控制虚拟网络中的访问策略、网络隔离、流量监控等等。

OVS 遵循 Apache 2.0 许可证, 能同时支持多种标准的管理接口和协议。OVS 也提供了对 OpenFlow 协议的支持，用户可以使用任何支持 OpenFlow 协议的控制器对 OVS 进行远程管理控制。

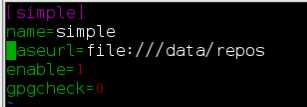
（2）网络名称空间netns

netns是在linux中提供网络虚拟化的一个项目，使用netns网络空间虚拟化可以在本地虚拟化出多个网络环境。netns可以让一台机器上模拟多个网络设备，是网络虚拟化的重要组成，将不同类型的网络应用隔离。

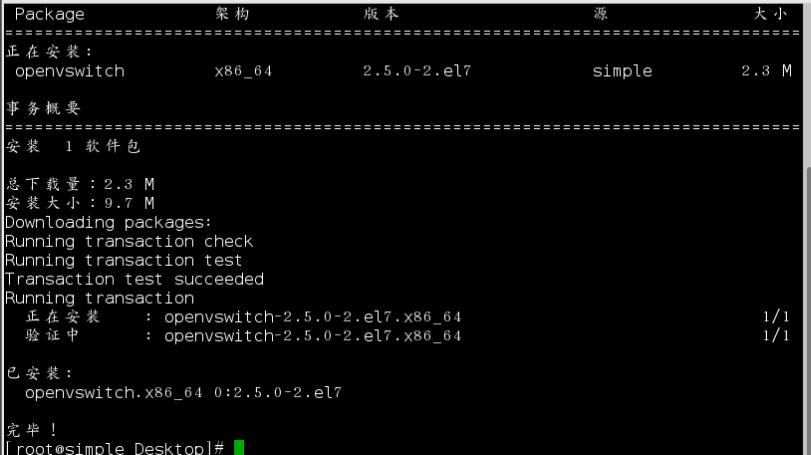
一个net namespace拥有独立的独立的网卡空间，路由表，ARP表，ip地址表，iptables等。

2.软件安装

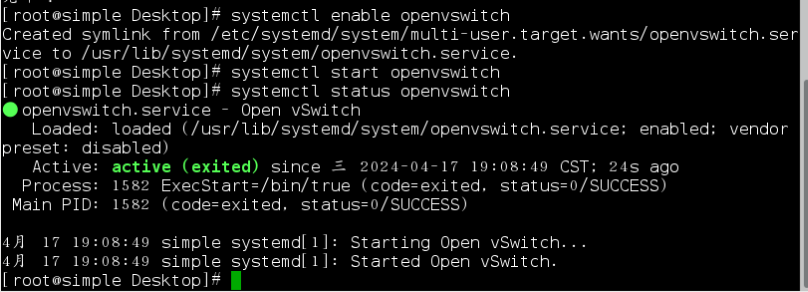
修改yum源



安装yum -y install openvswitch

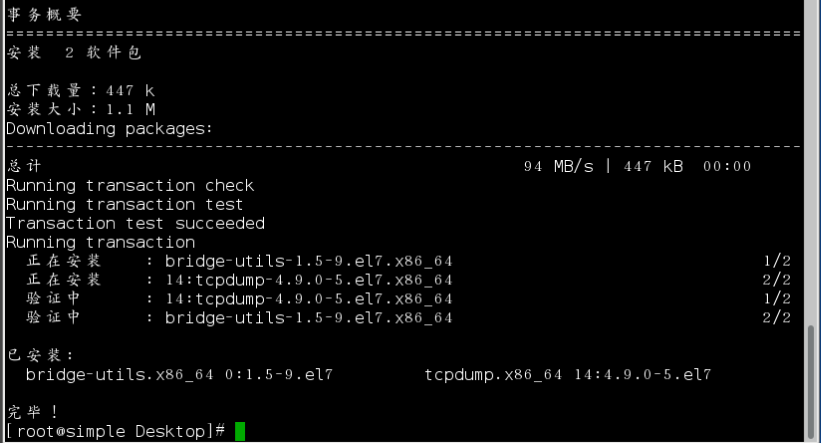


启动openvswitch并设置自启



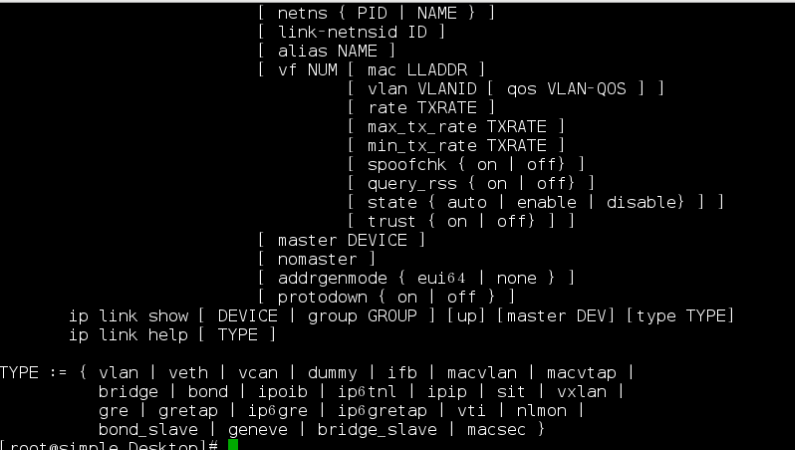
安装抓包工具tcpdump,网络配置工具bridge-utils

# yum -y install tcpdump bridge-utils



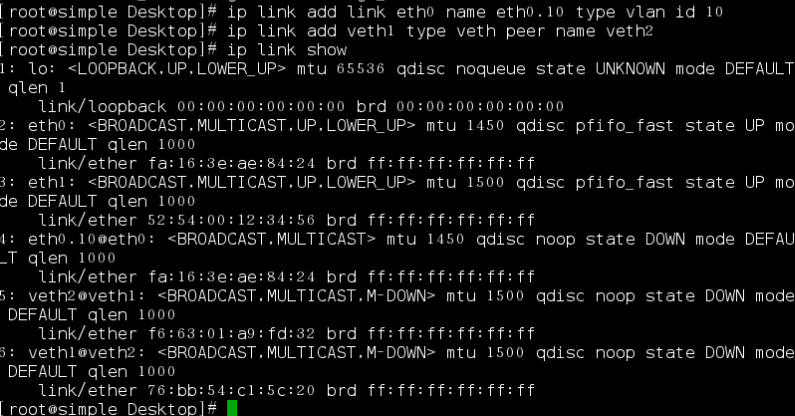
3.ip link使用

(1) 查看ip link帮助 # ip link help



(2) 新建网络接口# ip link add link eth0 name eth0.10 type vlan id 10

# ip link add veth1 type veth peer name veth2

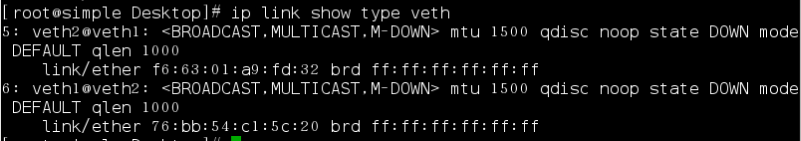


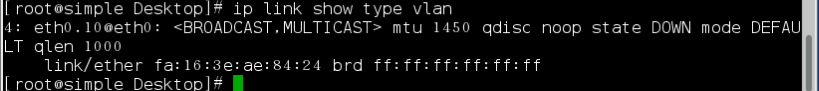
(3) 查看网络接口

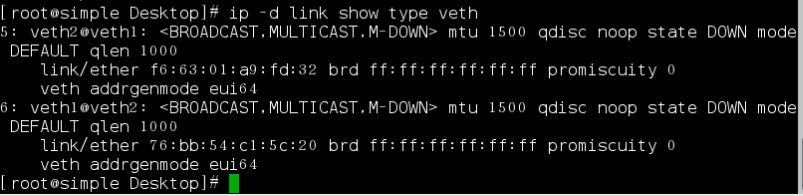
# ip link show # ip link show type veth # ip link show type vlan

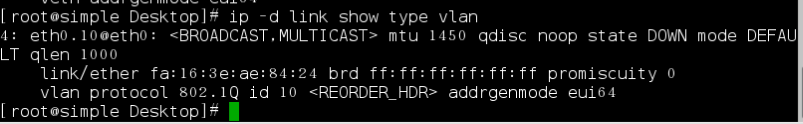
# ip -d link show type veth # ip -d link show type vlan

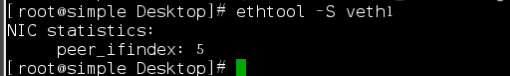
# ethtool -S veth1





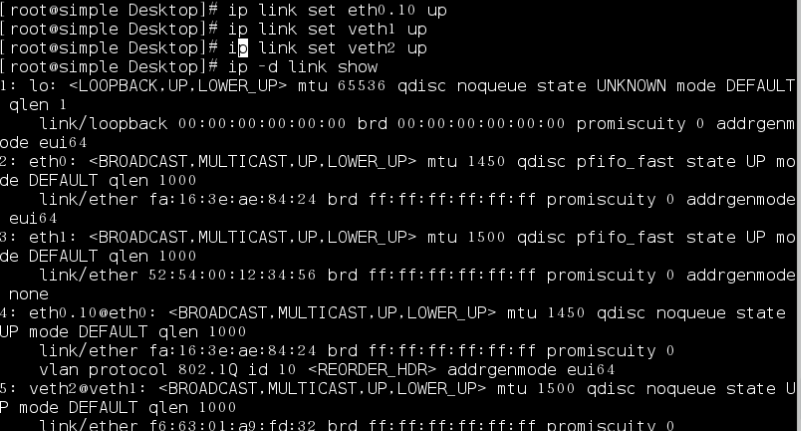






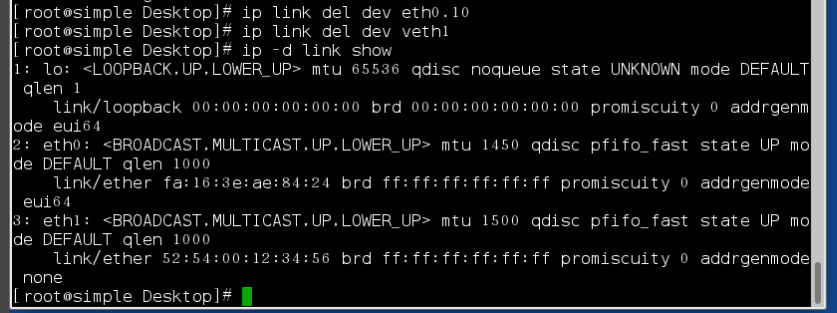
(4) 使接口UP

# ip link set eth0.10 up # ip link set veth1 up # ip link set veth2 up



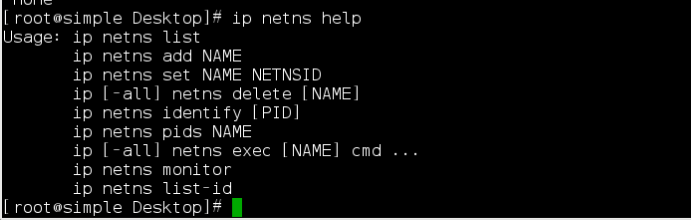
(5) 删除接口

# ip link del dev eth0.10 # ip link del dev veth1



4、ip netns使用

(1) 查看ip netns帮助 # ip netns help



(2) 创建一个名为test的namespace# ip netns add test



(3) 查看所有namespace ip netns list 或 ip netns

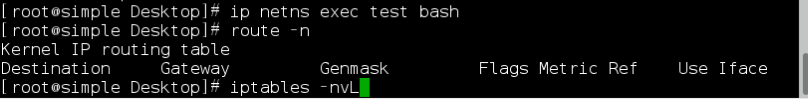


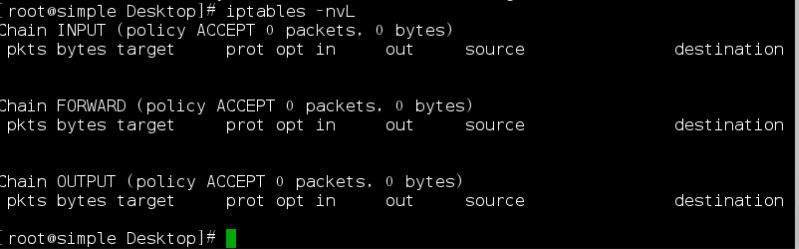
(4) 查看名为test的namespace # ip netns exec test ip addr show



(5) 进入名为test的namespace,执行网络命令

# ip netns exec test bash # route -n # iptables -nvL



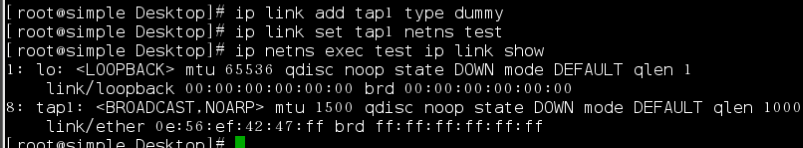


(6) 退出namespace # exit

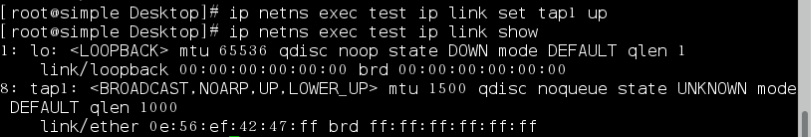


(7)给test 添加接口tap1 # ip link add tap1 type dummy

# ip link set tap1 netns test # ip netns exec test ip link show



(8) 启用tap1虚拟接口 # ip netns exec test ip link set tap1 up



(9) 给tap1虚拟接口配置IP

# ip netns exec test ip addr add dev tap1 192.168.0.1/24

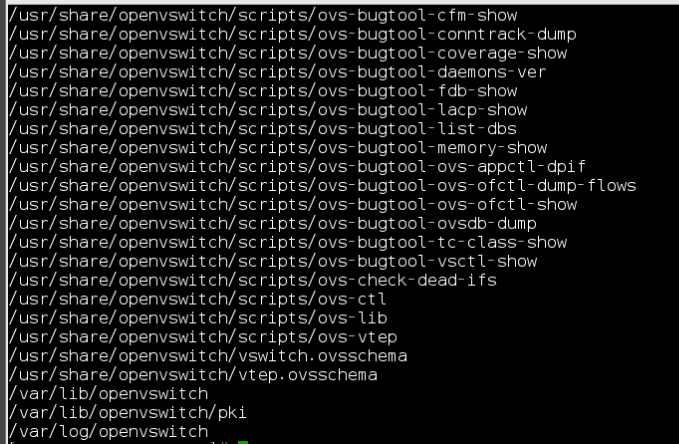


(10) 删除test namespace # ip netns del test



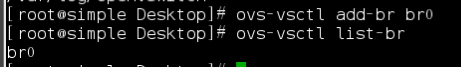
5、openvswitch使用

(1) 查看openvswitch安装的命令工具



(2)添加网桥br0 # ovs-vsctl add-br br0

(3)列出open vswitch中所有的网桥 # ovs-vsctl list-br



(4)判断网桥是否存在 # ovs-vsctl br-exists br0

(5)将网卡添加到网桥br0

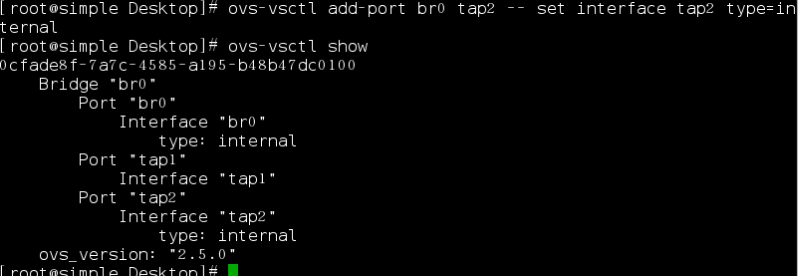
# ip link add tap1 type dummy # ovs-vsctl add-port br0 tap1

(6)查看open vswitch的网络状态 # ovs-vsctl show



(7)在网桥br0中新建openvswitch网口

# ovs-vsctl add-port br0 tap2 -- set interface tap2 type=internal



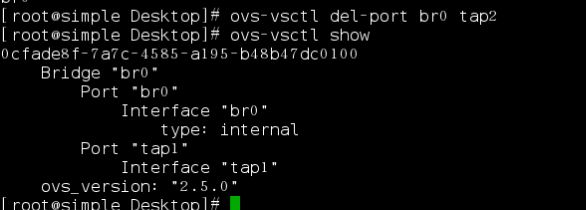
1. 列出网桥br0中所有端口 # ovs-vsctl list-ports br0



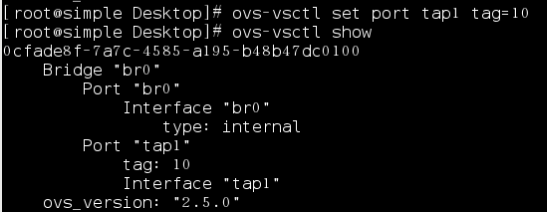
(9)列出所有连接到网卡tap2的网桥 # ovs-vsctl port-to-br tap2



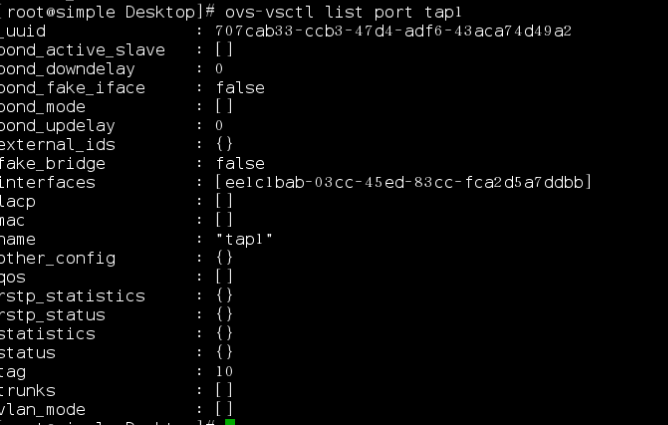
(10)删除网桥br0上的网口tap2 # ovs-vsctl del-port br0 tap2



(11)设置网口tap1的vlan tag为10 # ovs-vsctl set port tap1 tag=10

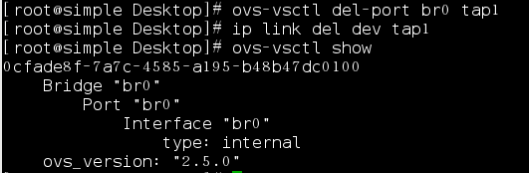


(12)查看网口tap1的属性 # ovs-vsctl list port tap1

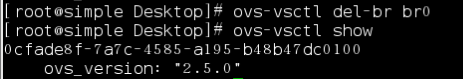


(13)从网桥br0删除网口tap1, 并从系统删除虚拟网口tap1

# ovs-vsctl del-port br0 tap1 # ip link del dev tap1



(14)删除网桥br0 # ovs-vsctl del-br br0



**4.任务四**

【**实验目标**】

Linux下如何利用工具组建虚拟网络环境。

【**实验工具**】

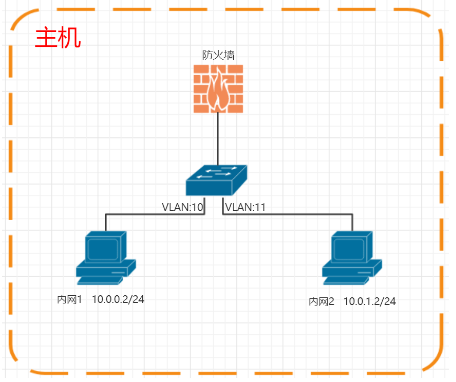
**ovs-vsctl (openvswitch软件提供)**

**ip (包含ip link、ip netns、ip addr等)**

**sysctl**

【**操作步骤**】

1、实验网络拓扑



2.环境搭建

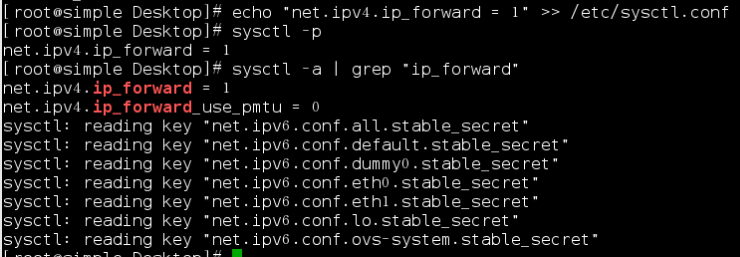
利用openvswitch创建tag为10和11的两个接口,结全netns模拟内网1、内网2。

(1)开启主机路由转发功能

\# echo “net.ipv4.ip\_forward = 1” >> /etc/sysctl.conf

\# sysctl -p

\# sysctl -a | grep “ip\_forward”

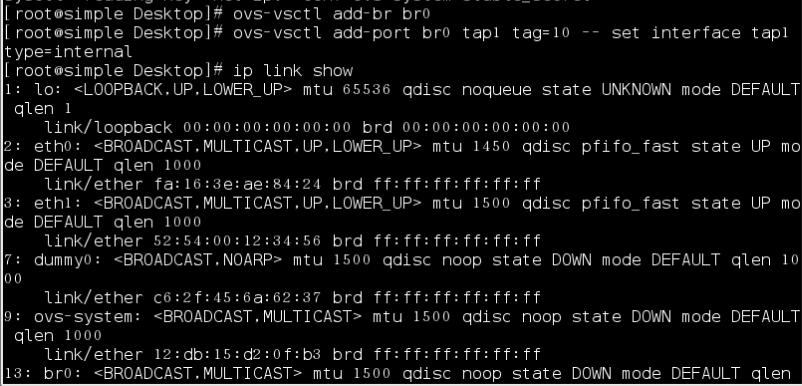


(2)创建tag为10的内网1

\# ovs-vsctl add-br br0

\# ovs-vsctl add-port br0 tap1 tag=10 -- set interface tap1 type=internal

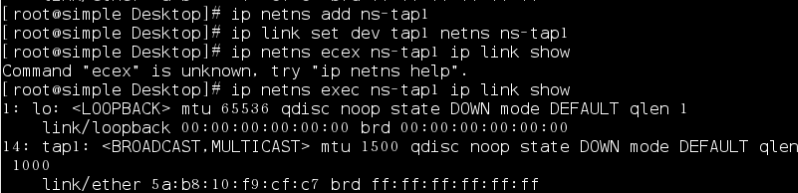
\# ip link show



\# ip netns add ns-tap1

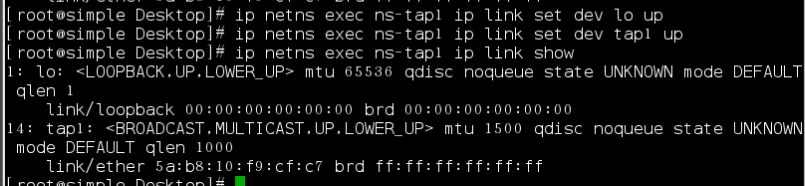
\# ip link set dev tap1 netns ns-tap1

\# ip netns exec ns-tap1 ip link show



\# ip netns exec ns-tap1 ip link set dev lo up

\# ip netns exec ns-tap1 ip link set dev tap1 up

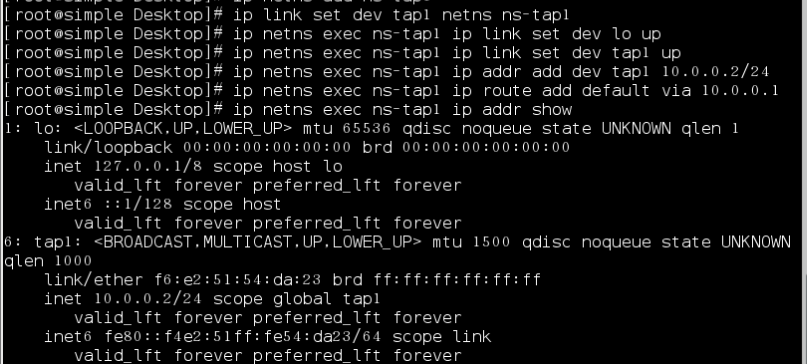


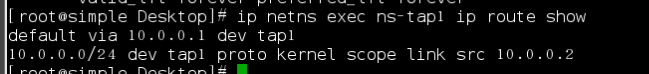
\# ip netns exec ns-tap1 ip addr add dev tap1 10.0.0.2/24

\# ip netns exec ns-tap1 ip route add default via 10.0.0.1

\# ip netns exec ns-tap1 ip addr show

\# ip netns exec ns-tap1 ip route show



(3)创建tag为11的内网2，方法同上

\# ovs-vsctl add-port br0 tap2 tag=11 -- set interface tap2 type=internal

\# ip netns add ns-tap2

\# ip link set dev tap2 netns ns-tap2

\# ip netns exec ns-tap2 ip link show



\# ip netns exec ns-tap2 ip link set dev lo up

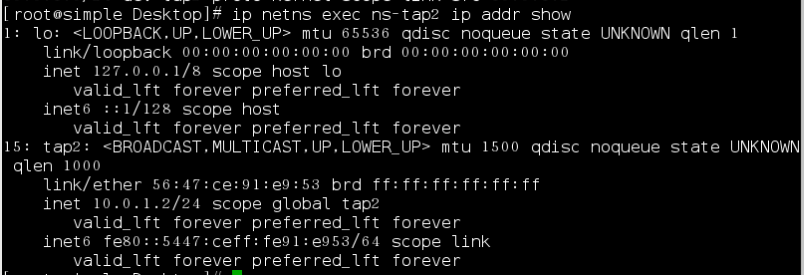
\# ip netns exec ns-tap2 ip link set dev tap2 up

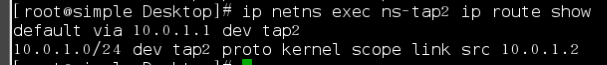
\# ip netns exec ns-tap2 ip addr add dev tap2 10.0.1.2/24

\# ip netns exec ns-tap2 ip route add default via 10.0.1.1

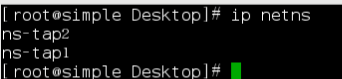
\# ip netns exec ns-tap2 ip addr show

\# ip netns exec ns-tap2 ip route show

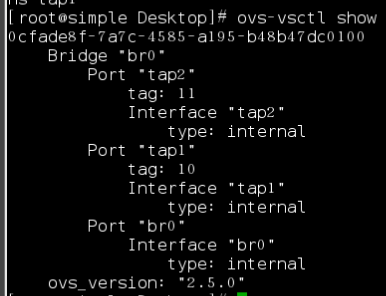




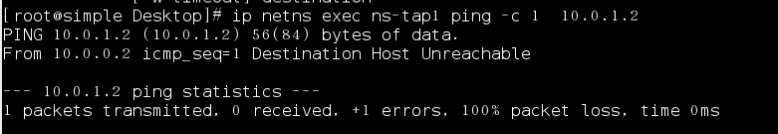
(4)查看网络内网1 与 内网2的连通性 # ip netns



# ovs-vsctl show



# ip netns exec ns-tap1 ping -c 1 10.0.1.2



测试两个内网是不通的

(5)、netns模拟器由器实现内网1与内网2互通

\# ovs-vsctl add-port br0 r1 tag=10 -- set interface r1 type=internal

\# ovs-vsctl add-port br0 r2 tag=11 -- set interface r2 type=internal



\# ip netns add router

\# ip link set dev r1 netns router

\# ip link set dev r2 netns router

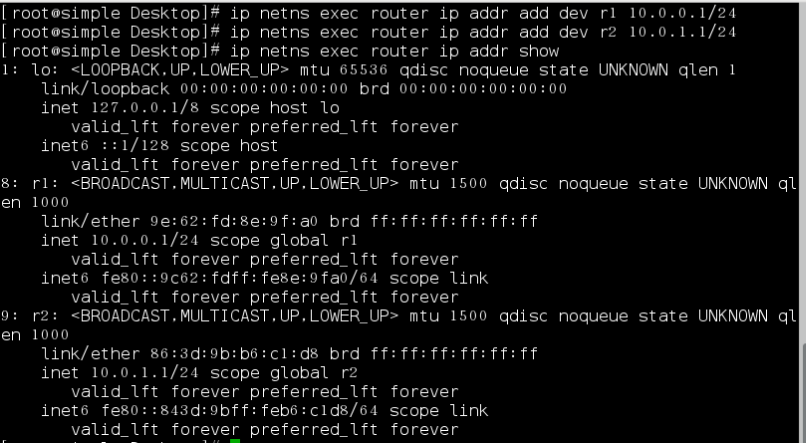
\# ip netns exec router ip link set dev lo up

\# ip netns exec router ip link set dev r1 up

\# ip netns exec router ip link set dev r2 up

\# ip netns exec router ip addr add dev r1 10.0.0.1/24

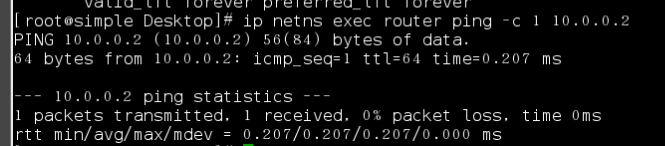
\# ip netns exec router ip addr add dev r2 10.0.1.1/24

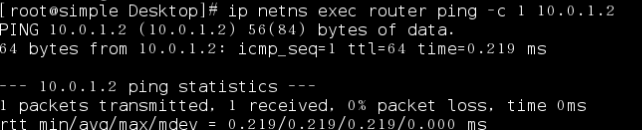


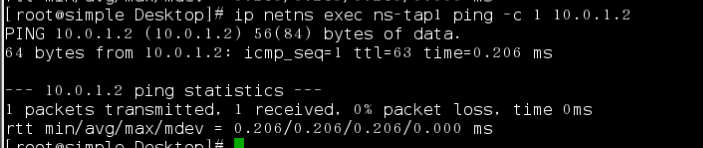
\# ip netns exec router ping -c 1 10.0.0.2

\# ip netns exec router ping -c 1 10.0.1.2

\# ip netns exec ns-tap1 ping -c 1 10.0.1.2







在路由器内ping两个内网都是通的, 在内网1内ping内网2也是通的,实验环境搭建成功。

**5.任务五**

【**任务描述**】

本实验任务在任务四操作完成的基础上，主要完成以下内容：

了解iptables nat基础知识与常用配置，

进行网络防火墙访问控制的实战。

通过完成本实验任务，要求学生掌握企业中简单的iptables防火墙的应用。

【**实验目标**】

掌握iptables nat的基础知识与常用配置的操作。

掌握简单的网络防火墙的访问控制操作。

可以利用iptables加固企业网络。

【**实验工具**】

**iptables**

**ip (包含ip link、ip netns、ip addr等)**

【**操作步骤**】

1、iptables nat基础知识

nat表需要的三个链:

PREROUTING:在数据包到达防火墙时进行路由判断之前的规则，作用是是否改变目的地址或者目的端口;

POSTROUTING:在数据包离开防火墙时进行路由判断，是否要改变源地址、源端口等;

INPUT:改变访问目的为主机的数据包源地址;

OUTPUT:改变主机发出去的数据包目的地址;

动作选项：

REDIRECT: 将数据包重定向到其它或其它主机的某个端口;

SNAT: 源地址转换，改变数据包的源地址;

DNAT: 目的地址转换，改变数据包的目的地址;

MASQUERADE: ip智能伪装;

注意点:

PRERROUTING: DNAT、REDIRECT （路由之前）只支持-i，不支持-o。在作出路由之前，对目的地址进行修改;

POSTROUTING: SNAT、MASQUERADE （路由之后）只支持-o，不支持-i。在作出路由之后，对源地址进行修改;

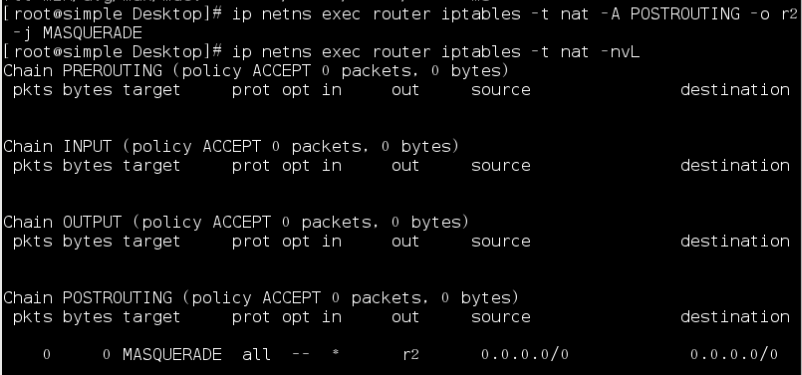
OUTPUT: DNAT、REDIRECT （本机）DNAT和REDIRECT规则用来处理来自NAT主机的出站数据包;

INPUT: SNAT （本机）SNAT规则用来修改目的为本机的源地址;

2、nat常用配置

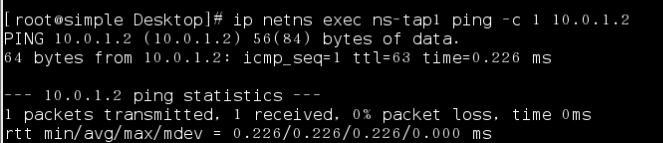
(1) 将源转换成路由器router的r2接口地址

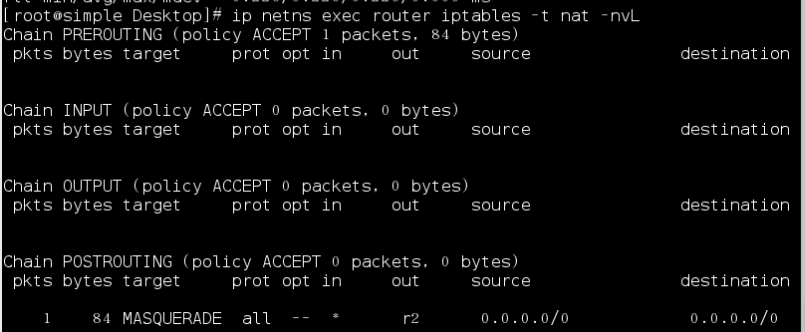
# ip netns exec router iptables -t nat -A POSTROUTING -o r2 -j MASQUERADE



\# ip netns exec ns-tap1 ping -c 1 10.0.1.2

\# ip netns exec router iptables -t nat -nvL

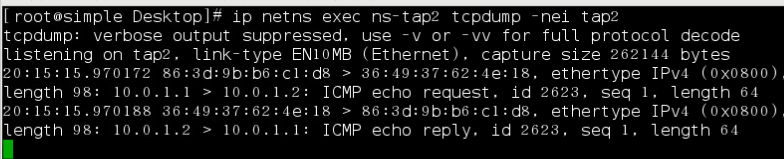




(2) 打开两个终端利用tcpdump抓包分析

a、# ip netns exec ns-tap1 ping -c 1 10.0.1.2

b、# ip netns exec ns-tap2 tcpdump -nei tap2



(3) 配置SNAT

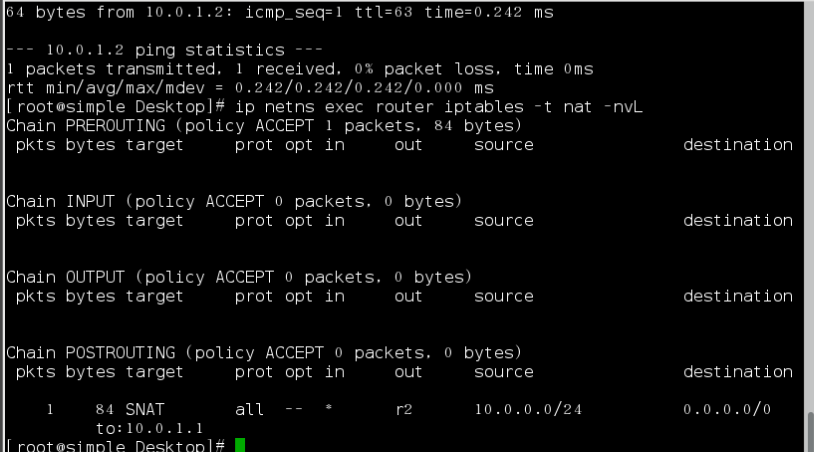
\# ip netns exec router iptables -t nat -F

\# ip netns exec router iptables -t nat -A POSTROUTING -s 10.0.0.0/24 -o r2 -j SNAT --to 10.0.1.1



\# ip netns exec ns-tap1 ping -c 1 10.0.1.2

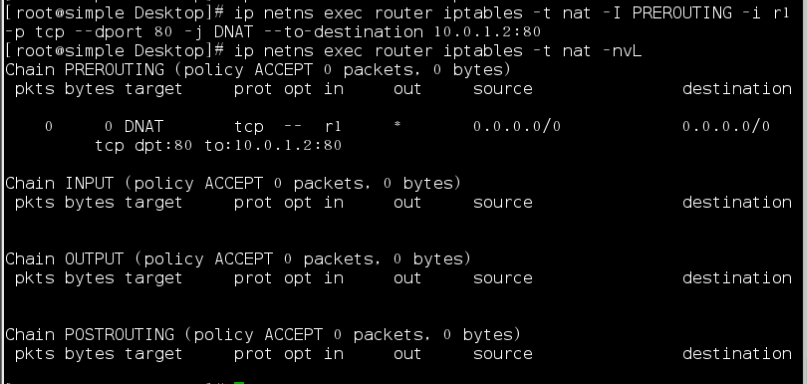
\# ip netns exec router iptables -t nat -nvL



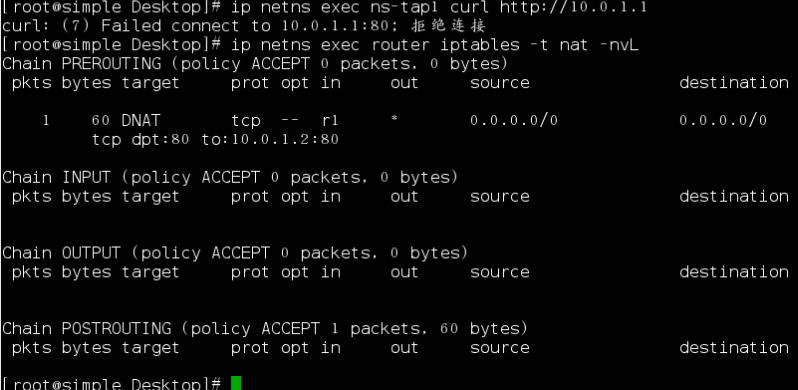
(4) 配置DNAT

\# ip netns exec router iptables -t nat -F

\# ip netns exec router iptables -t nat -I PREROUTING -i r1 -p tcp —dport 80 -j DNAT --to-destination 10.0.1.2:80



# ip netns exec ns-tap1 curl http://10.0.1.1

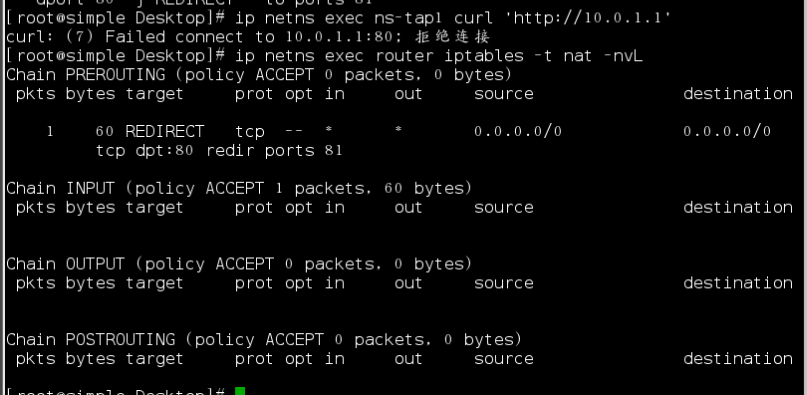


(5) 重定向

\# ip netns exec router iptables -t nat -F

\# ip netns exec router iptables -t nat -I PREROUTING -p tcp —dport 80 -j REDIRECT --to-ports 81

\# ip netns exec ns-tap1 curl `http://10.0.1.1`



3、网络防火墙

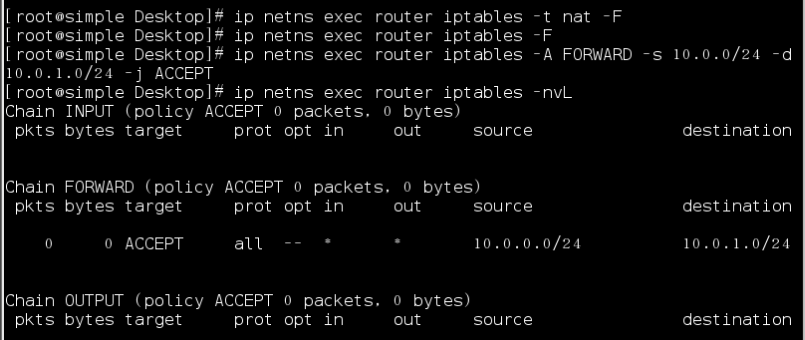
(1) 允许内网１访问内问２

\# ip netns exec router iptables -t nat -F

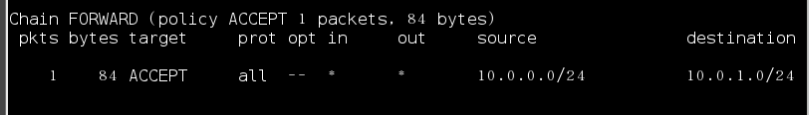
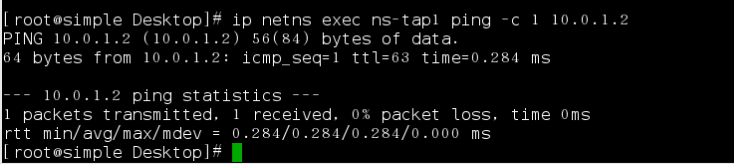
\# ip netns exec router iptables -F

\# ip netns exec router iptables -A FORWARD -s 10.0.0/24 -d 10.0.1.0/24 -j ACCEPT

\# ip netns exec router iptables -nvL



# ip netns exec ns-tap1 ping -c 1 10.0.1.2



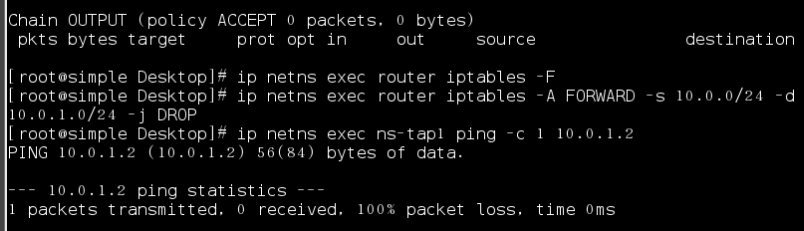
有数据包匹配说明规则生效

(2) 拒绝内网１访问内网２

\# ip netns exec router iptables -F

\# ip netns exec router iptables -A FORWARD -s 10.0.0/24 -d 10.0.1.0/24 -j DROP

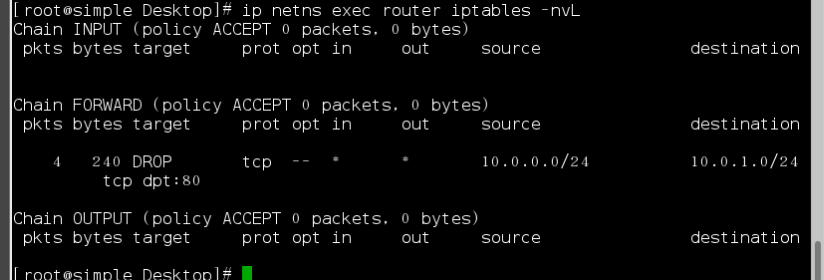
\# ip netns exec ns-tap1 ping -c 1 10.0.1.2



(3) 拒绝内网１访问内网２的80服务

\# ip netns exec router iptables -F

\# ip netns exec router iptables -A FORWARD -s 10.0.0/24 -d 10.0.1.0/24 -p tcp --dport 80 -j DROP



**6.任务六**

【**任务描述**】

本实验任务在前五个实验掌握的基础上，主要完成以下内容：

利用前面学习到的虚拟化技术搭建复杂的网络实验场景；

进一步学习虚拟交换机openvwitch网络配置；

云计算中一种二层防火墙的实现；

利用ipset动态加载防火墙策略

巩固防火墙、虚拟网络知识

通过完成本实验任务，要求学生掌握企业公有云中二层防火墙实现的一种方法。可以控制同网段虚拟机间的通信,也可以防止用户随意修改IP或者MAC地址。

【**实验目标**】

利用虚拟化技术搭建复杂的网络实验场景

掌握云计算中一种二层防火墙的实验方法

进一步掌握iptables的使用

【**实验工具**】

**ovs-vsctl (openvswitch软件提供)**

**IP (包含ip link、ip netns、ip addr等)**

**brctl**

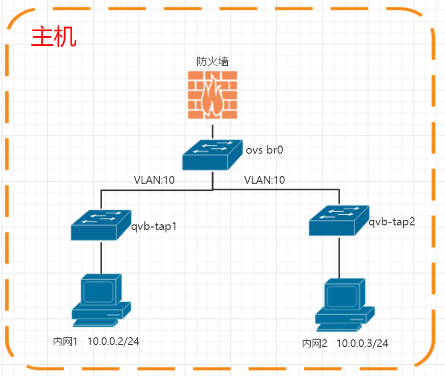
**modprobe**

**ipset**

【**操作步骤**】

1、实验网络修改

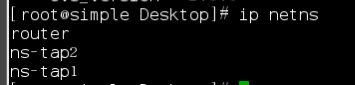
(1) 实验拓扑



(2) 查看上个实验环境

# ovs-vsctl show # ip netns

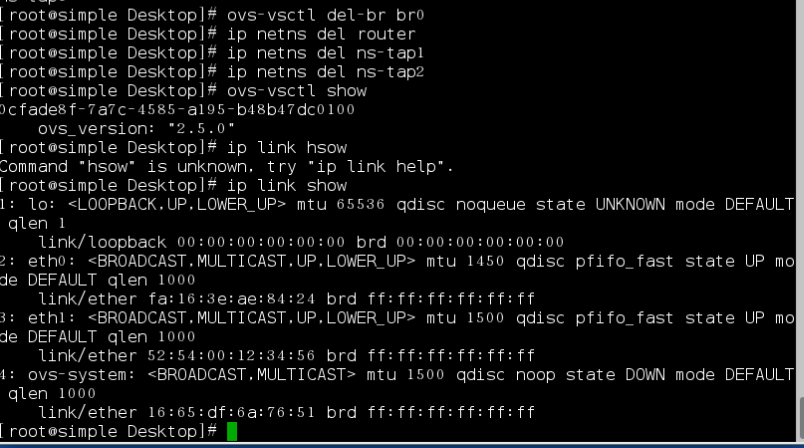




(3) 清空配置

# ovs-vsctl del-br br0 # ip netns del router

# ip netns del ns-tap1 # ip netns del ns-tap2



(4) 搭建实验环境

a、新建网桥br0

# ovs-vsctl add-br br0

b、配置内网1

# ip link add qvo-tap1 type veth peer name qvb-tap1

# ip link set qvb-tap1 up

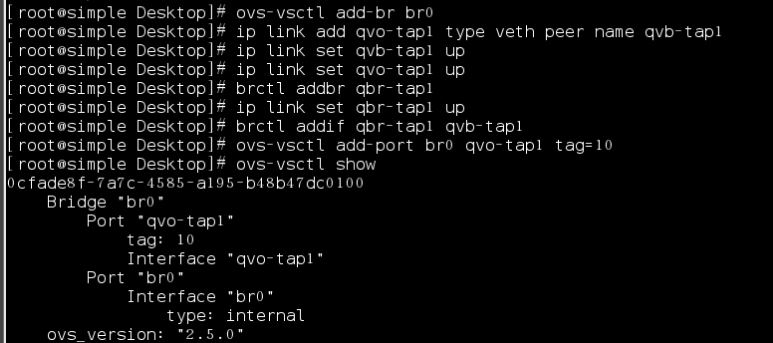
# ip link set qvo-tap1 up

# brctl addbr qbr-tap1

# ip link set qbr-tap1 up

# brctl addif qbr-tap1 qvb-tap1

# ovs-vsctl add-port br0 qvo-tap1 tag=10



# ip link add tap1 type veth peer name tap11

# brctl addif qbr-tap1 tap11

# ip link set tap11 up

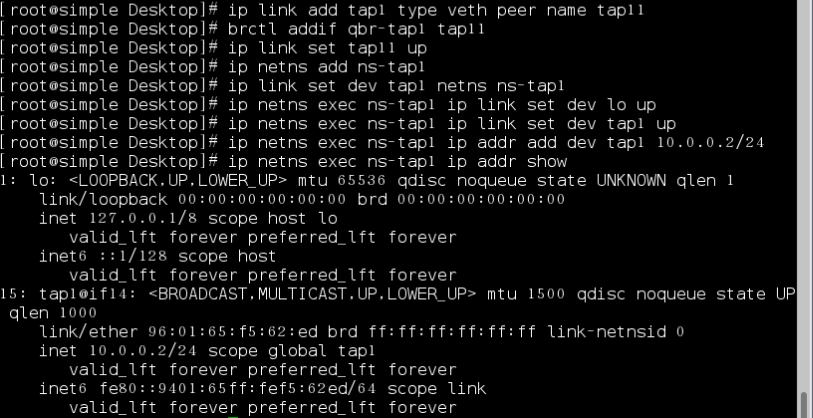
# ip netns add ns-tap1

# ip link set dev tap1 netns ns-tap1

# ip netns exec ns-tap1 ip link set dev lo up

# ip netns exec ns-tap1 ip link set dev tap1 up

# ip netns exec ns-tap1 ip addr add dev tap1 10.0.0.2/24



c、配置内网2

# ip link add qvo-tap2 type veth peer name qvb-tap2

# ip link set qvb-tap2 up

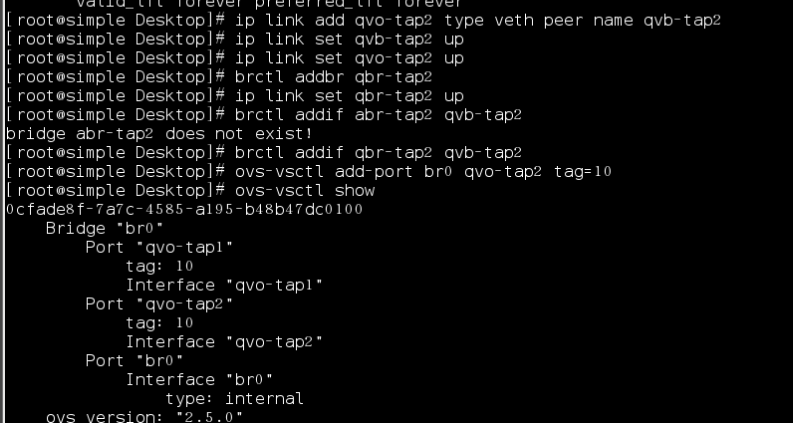
# ip link set qvo-tap2 up

# brctl addbr qbr-tap2

# ip link set qbr-tap2 up

# brctl addif qbr-tap2 qvb-tap2

# ovs-vsctl add-port br0 qvo-tap2 tag=10



# ip link add tap2 type veth peer name tap22

# brctl addif qbr-tap2 tap22

# ip link set tap22 up

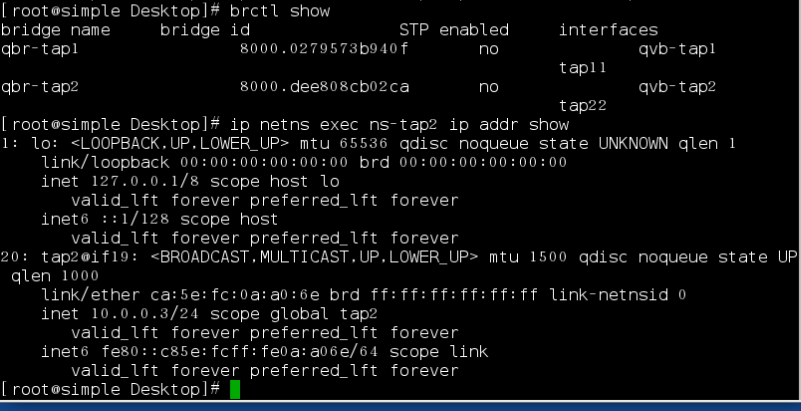
# ip netns add ns-tap2

# ip link set dev tap2 netns ns-tap2

# ip netns exec ns-tap2 ip link set dev lo up

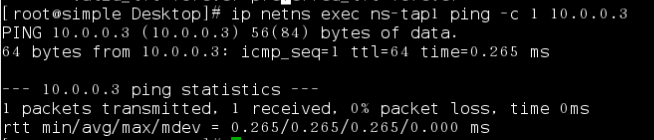
# ip netns exec ns-tap2 ip link set dev tap2 up

# ip netns exec ns-tap2 ip addr add dev tap2 10.0.0.3/24



d、测试二层同网段通信

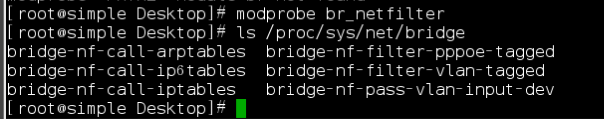
# ip netns exec ns-tap1 ping -c 1 10.0.0.3



(5) 加载内核参数

# modprobe br\_netfilter

# ls /proc/sys/net/bridge



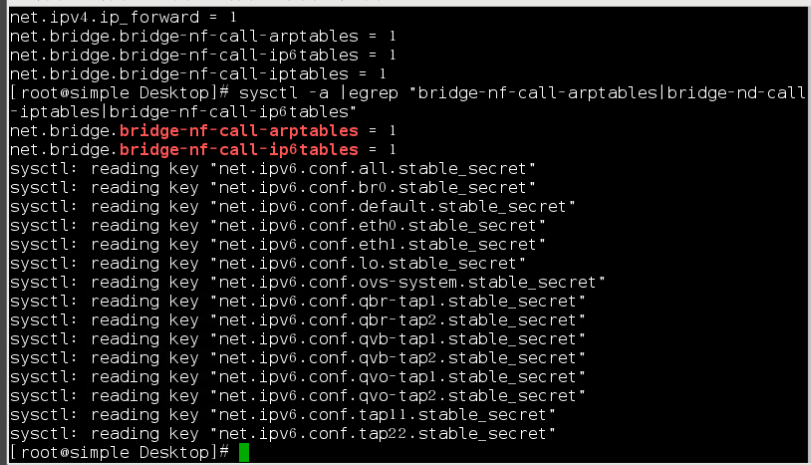
# echo “net.bridge.bridge-nf-call-arptables = 1” >> /etc/sysctl.conf

# echo “net.bridge.bridge-nf-call-ip6tables = 1” >> /etc/sysctl.conf

# echo “net.bridge.bridge-nf-call-iptables = 1” >> /etc/sysctl.conf

# sysctl -p

# sysctl -a | egrep “bridge-nf-call-arptables|bridge-nf-call-iptables|bridge-nf-call-ip6tables”

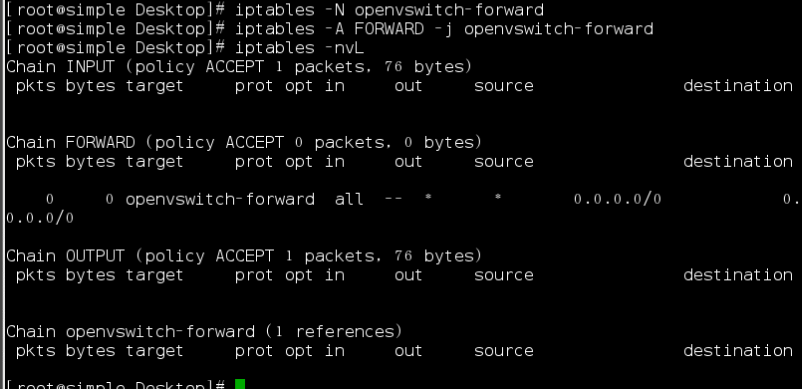


(6) 二层防火墙配置

a、把FORWARD链所有流量导入自定义链

# iptables -N openvswitch-forward

# iptables -A FORWARD -j openvswitch-forward



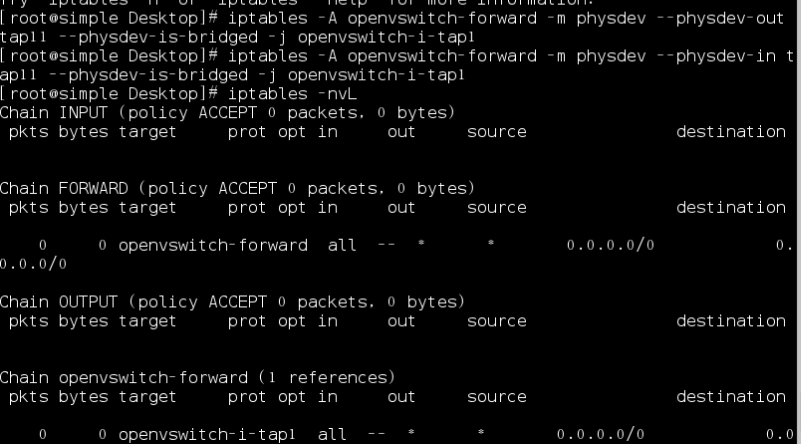
b、添加内网1 in、out方向链表，将匹配到的physdev流量倒入进出链表

# iptables -N openvswitch-i-tap1

# iptables -N openvswitch-o-tap1

# iptables -A openvswitch-forward -m physdev --physdev-out tap11 --physdev-is-bridged -j openvswitch-i-tap1

# iptables -A openvswitch-forward -m physdev --physdev-in tap11 --physdev-is-bridged -j openvswitch-o-tap1



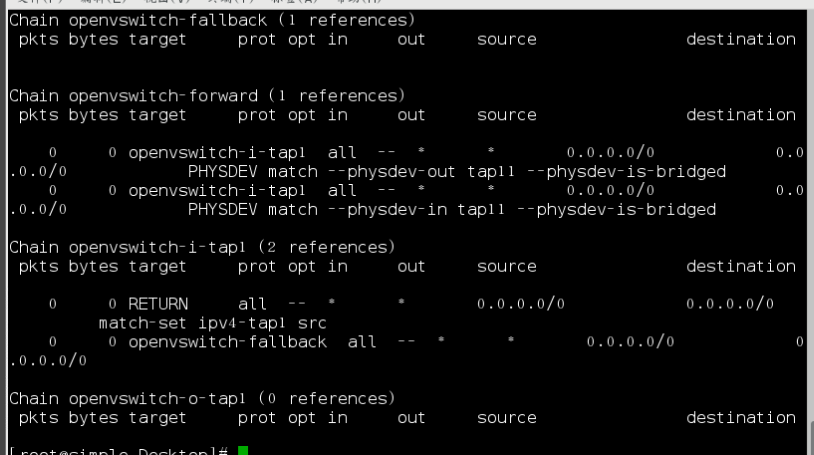
c、添加内网1 in方向规则, 并将没有匹配的流量导入新链表，过滤源地址使用ipset管理

# iptables -N openvswitch-fallback

# ipset create ipv4-tap1 hash:net

# iptables -A openvswitch-i-tap1 -m set --match-set ipv4-tap1 src -j RETURN

# iptables -A openvswitch-i-tap1 -j openvswitch-fallback

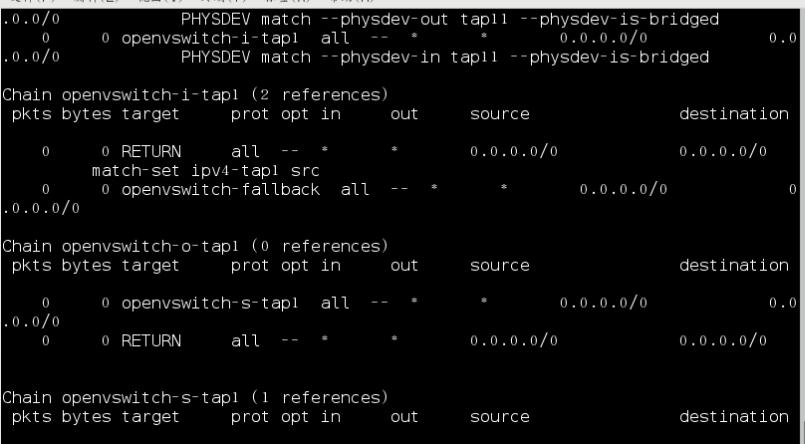


d、添加内网1 out方向规则，将所有流量导入新的链过滤

# iptables -N openvswitch-s-tap1

# iptables -A openvswitch-o-tap1 -j openvswitch-s-tap1

# iptables -A openvswitch-o-tap1 -j RETURN



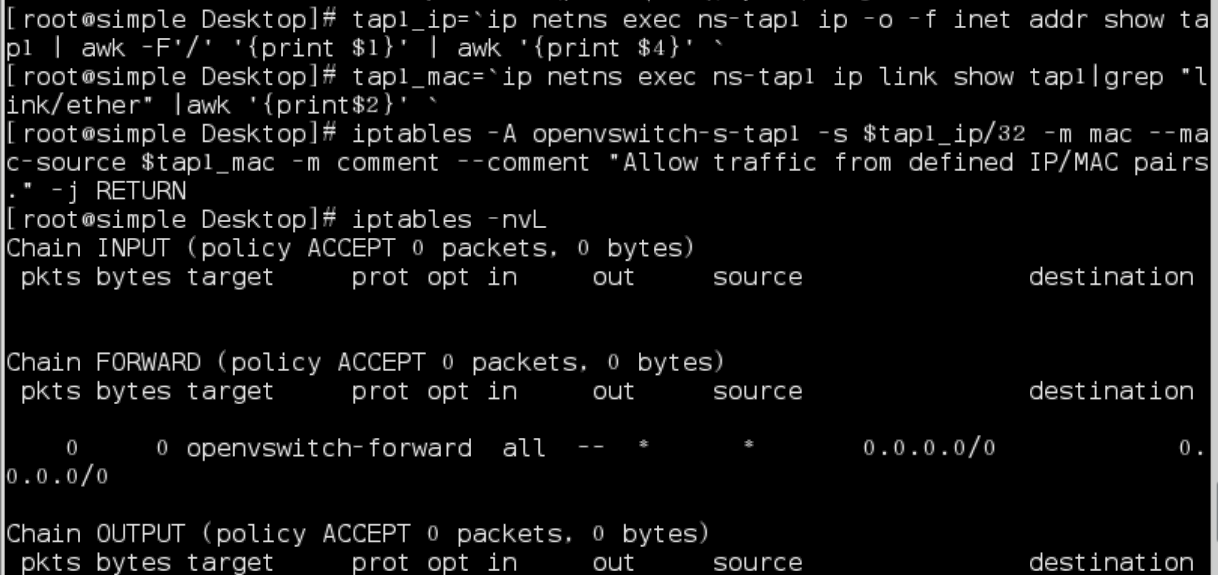
e、添加内网1安全规则

# tap1\_ip=\`ip netns exec ns-tap1 ip -o -f inet addr show tap1 | awk -F’/‘ ‘{print $1}’|awk ‘{print $4}’\`

# tap1\_mac=\`ip netns exec ns-tap1 ip link show tap1 | grep “link/ether”|awk ‘{print $2}’\`

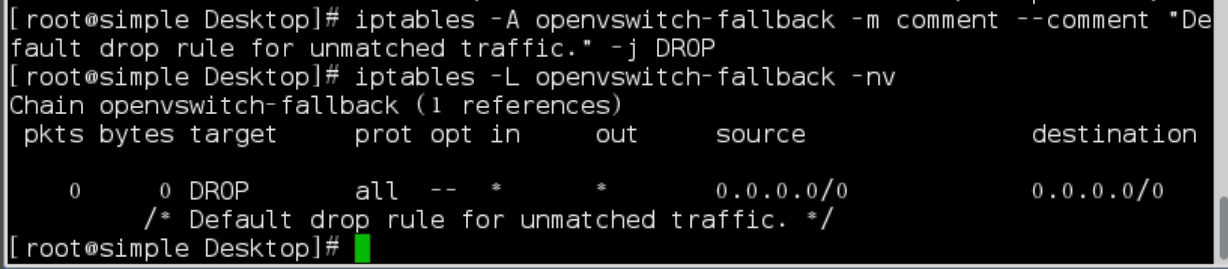
# iptables -A openvswitch-s-tap1 -s $tap1\_ip/32 -m mac --mac-source $tap1\_mac -m comment --comment “Allow traffic from defined IP/MAC pairs.” -j RETURN

# iptables -A openvswitch-s-tap1 -j DROP



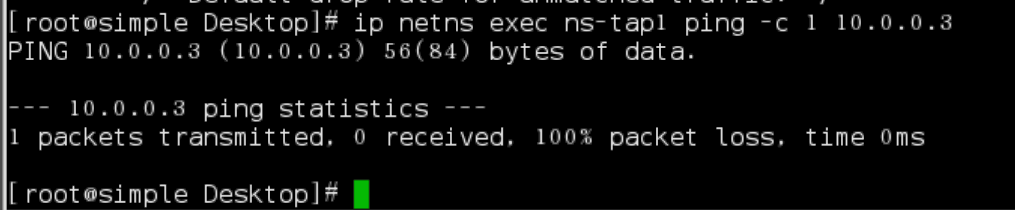
f、拒绝未匹配的流量

# iptables -A openvswitch-fallback -m comment --comment “Default drop rule for unmatched traffic.” -j DROP



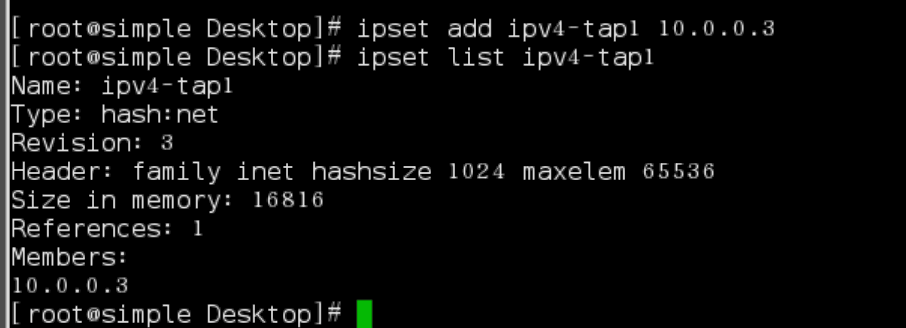
g、测试内网1与内网2的网络连通性

# ip netns exec ns-tap1 ping -c 1 10.0.0.3



可以看到已经不通了, 二层防火墙策略已经生效

h、ipset添加源地址内网2,验证连通性



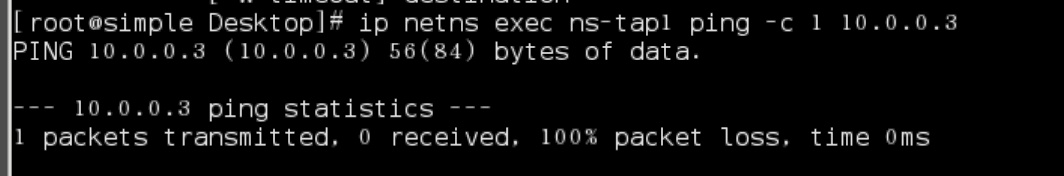
ipset是iptables的扩展,可以想像它是一个IP地址集合,可以动态的加载iptables规则的地址集, 有效提升iptables的查找效率

i、验证内网1能否修改IP、MAC

# ip netns exec ns-tap1 ip addr del dev tap1 10.0.0.2/24

# ip netns exec ns-tap1 ip addr add dev tap1 10.0.0.4/24

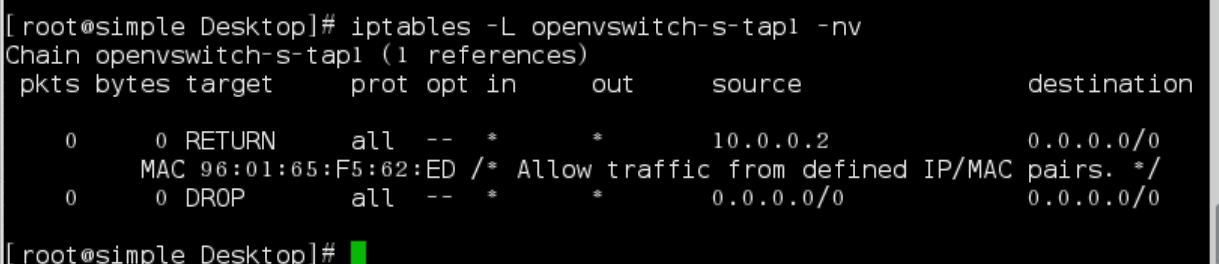




# ip netns exec ns-tap1 ip addr del dev tap1 10.0.0.4/24

# ip netns exec ns-tap1 ip addr add dev tap1 10.0.0.2/24

# ip netns exec ns-tap1 ip link set dev tap1 address 16:f7:55:f5:d7:ac



三、实验总结

在完成以上几个任务后，我对网络安全和虚拟网络的理解有了显著提升。首先，在了解和掌握了Iptables防火墙的基础后，我意识到防火墙在网络安全中的重要性。Iptables的配置和使用不仅可以保护主机免受恶意攻击，还能够实现网络流量的控制和审计功能。掌握了常用配置和审计功能后，我能够更好地应对各种网络安全挑战，确保主机的安全性和网络的稳定性。其次，通过了解虚拟网络中常用的软件和工具，我对虚拟化技术有了更深入的了解。虚拟网络的灵活性和可扩展性使得网络架构更具弹性，能够更好地适应不同的应用场景和需求。进而，在利用虚拟网络构建网络防火墙实验场景的过程中，我深刻体会到了理论知识与实践操作的结合。通过实验和模拟网络攻击，我更好地理解了防火墙的工作原理和实际应用。在任务五和任务六中，我通过使用Iptables实战网络防火墙的NAT和网络控制功能，以及搭建公有云二层防火墙，进一步加强了Iptables强大功能的认识和应用能力。这些实战经验不仅增强了我的技能，还提升了对网络安全的实际应用能力。

总的来说，通过这几个任务，我不仅学到了丰富的知识，还掌握了实用的技能，这将对我的职业发展和网络安全意识产生深远的影响。我期待在今后的工作和学习中能够进一步运用和发展这些知识。