信息安全原理设计报告

（实验二）

姓　　名：王润霖

学　　号：2053182

专　　业：信息安全

二〇二三年三月

目 录

第一部分　实验内容和分工

1.1 实验题目

基于 linux 搭建一个基本 ipsec-vpn 原型

1.选择一个开源程序库，总结一下 linux 下 ipsec 网关程序模块构架及其相互关系

（1）总结配置 ike 的方法与参数

（2）总结 l2tp-ipsec 的配置方法与参数

（3）总结 ipsec 模式选择及参数配置方法

2.搭建一个 gate2gate ipsec vpn 网关对（gate 后 lan 为私有网络）。

1.2 实验环境

CentOS 7 环境、Libreswan 开源程序库

1.3 小组分工

任务点	内容	完成情况
1	总结一下 linux 下 ipsec 网关程序模块构架及其相互关系	√
2	总结配置 ike 的方法与参数	√
3	总结 l2tp-ipsec 的配置方法与参数	√
4	总结 ipsec 模式选择及参数配置方法	√
5	搭建一个 gate2gate ipsec vpn 网关对（gate 后 lan 为私有网络）	√

学号	姓名	主要任务点	贡献率
2052338	鲍宇轩	1、2、3、4	50%
2053182	王润霖	1、3、4、5	50%

我们基于开源的 libreswan 搭建了 l2tp over IPsec 的 VPN，并验证了其正确性，主要方法由王润霖总结（任务点 3），并通过查阅资料阅读源码总结了 ipsec 网关程序模块及相互关系（任务点 1）。

截至 2023 年 3 月 24 日：

鲍宇轩阅读开源代码相关文档等，学习参数含义、可能取值以及配置文件的格式、配置方法等，进行了一系列的修改和尝试，并印证已成功配置相应参数。（任务点 2,4）。

王润霖总结了 l2tp over IPsec 的 VPN 搭建方法（任务点 3），对参数进行了一定配置并取得成功（任务点 4），并且在 strongswan 的基础上研究了搭建 gate2gate ipsec vpn 网关的方法，有一定的进展和收获（任务点 5）。

更新 2023 年 3 月 26 日：

成功基于开源的 libreswan 搭建了 l2tp over IPsec 的 VPN，并通过多种方法验证了其正确性！

更新 2023 年 3 月 27 日：

在 3 月 26 日成功搭建符合要求的 IPsec VPN 基础上，重新完成任务点 1-4 的参数配置工作，对于每一个任务均进行了实验，并验证了其正确性。

第二部分　实验过程及总结

为了便于描述和验证试验，本报告中，首先完成搭建 gate2gate ipsec vpn 网关对，并验证正确；再进行 ike、l2tp、ipsec 参数的配置和分析；最后总结 linux 下 ipsec 网关程序模块构架及其相互关系。

2.搭建一个 gate2gate ipsec vpn 网关对（gate 后 lan 为私有网络）

2.1 写在前面

本次实验先后尝试了多种方法，包括 0-2 个本地电脑、0-4 个虚拟机、0-4 个云主机的多种可能的排列组合，先后尝试了 OpenSwan、StrongSwan、Libreswan 等多种开源程序库，先后尝试了 Windows10、CentOS7.8、Ubuntu16 等多种环境，先后尝试了运行 Project1 主程序、wireshark 抓包等多种辅助验证方法，先后进行了脚本搭建和手工配置搭建操作。然而，在虚拟机上完成本次实验操作非常困难，在 2 本地+2 云主机的实验中也遇到了不少困难，这可能是由于本地和云主机的体质并不完全一样导致的。

最终，确定使用 4 个云主机的实验方法，**成功搭建了 gate to gate ipsec vpn 网关对，并成功进行了正确性验证试验**（包括验证 gate to gate 搭建成功、验证 vpn 经过了 ipsec 加密）。

本次实验的本个小题，得到了 2053636-朱祉同在实验思路上的帮助。

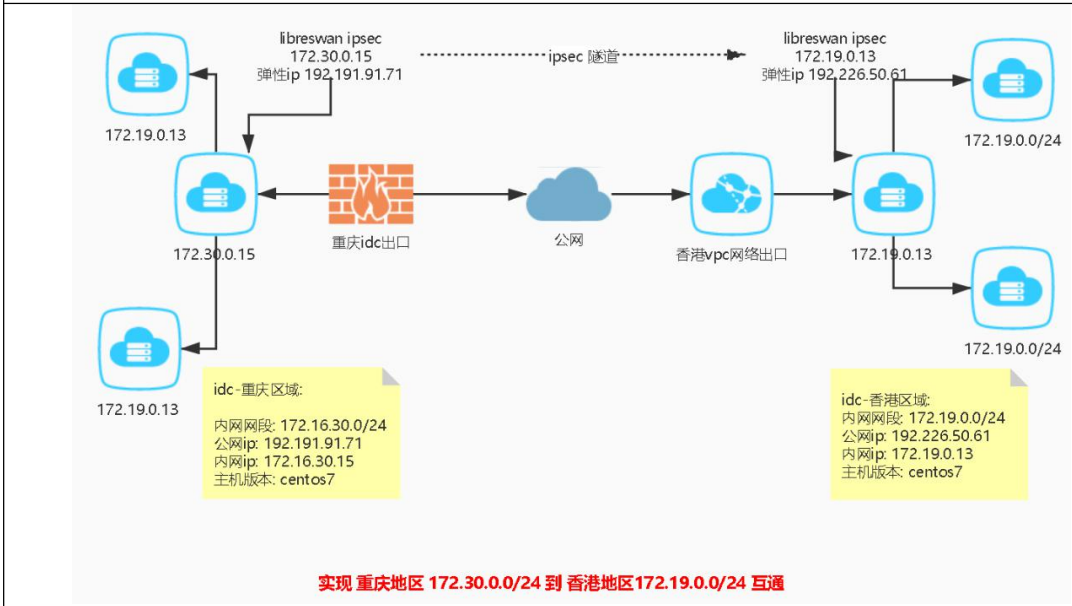
2.2 实验材料

4 个阿里云 ECS 实例（1 vCPU、2 GiB、I/O 优化、ecs.n4.small），CentOS 7.8 镜像，Libreswan 开源程序库。

实验网络结构：

Left Net=========Left Gate----------Right Gate=========Right Net

Internet



*实际 ip 以下面的实验材料细目表为准。

实验材料细目表：

编号*	地区	公网 ip	私有 ip	性质
LG	上海	47.103.98.35	172.28.129.83	网关

LN	上海	-	172.28.129.84	内网主机
RG	南京	47.122.20.94	172.26.238.168	网关
RN	南京	-	172.26.238.170	内网主机

*LG：Left Gate，RN：Right Net，其余同理。

（图：实验材料-2 个上海地区云主机、2 个南京地区云主机）

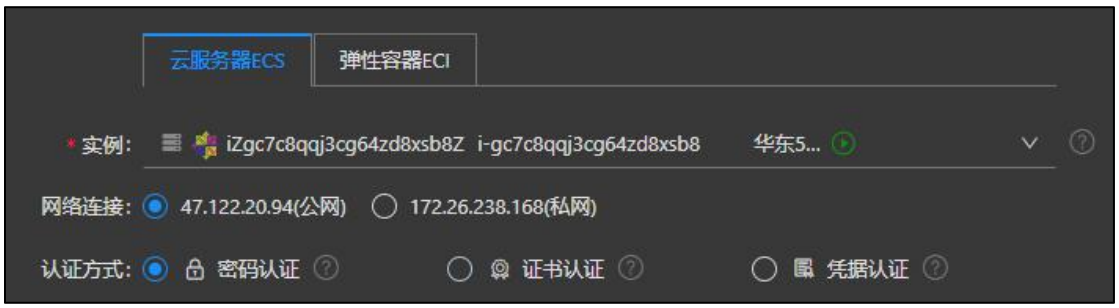
2.3 搭建 Gate to Gate 网关

2.3.1 观察实验前的网络连接情况

此处以 RG、RN 作为演示，LG、LN 的配置同理。

RG 可以连接外网*。

*可以 ping www.baidu.com.

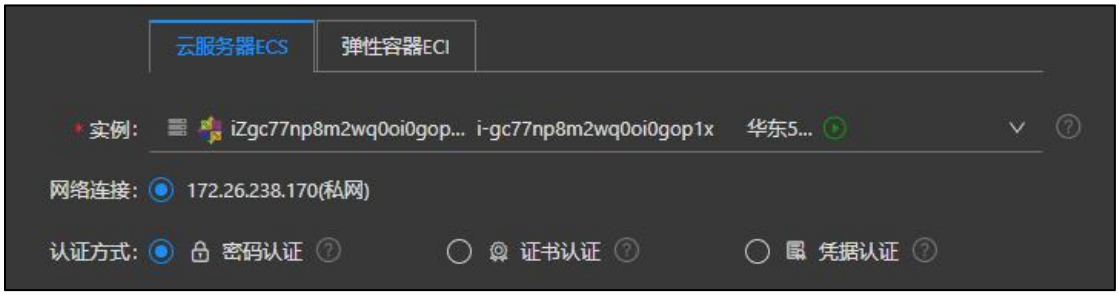


```
[root@iZgc7c8qqj3cg64zd8xsb8Z ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.26.238.168  netmask 255.255.0.0  broadcast 172.26.255.255
        inet6 fe80::216:3eff:fe01:1aa7  prefixlen 64  scopeid 0x20<link>
        ether 00:16:3e:01:1a:a7  txqueuelen 1000  (Ethernet)
        RX packets 73667  bytes 101220648 (96.5 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12097  bytes 1482441 (1.4 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
[root@iZgc7c8qqj3cg64zd8xsb8Z ~]# ping www.baidu.com
PING www.a.shifen.com (180.101.50.242) 56(84) bytes of data.
64 bytes from 180.101.50.242 (180.101.50.242): icmp_seq=1 ttl=49 time=15.1 ms
64 bytes from 180.101.50.242 (180.101.50.242): icmp_seq=2 ttl=49 time=15.1 ms
64 bytes from 180.101.50.242 (180.101.50.242): icmp_seq=3 ttl=49 time=15.1 ms
64 bytes from 180.101.50.242 (180.101.50.242): icmp_seq=4 ttl=49 time=15.1 ms
64 bytes from 180.101.50.242 (180.101.50.242): icmp_seq=5 ttl=49 time=15.1 ms
64 bytes from 180.101.50.242 (180.101.50.242): icmp_seq=6 ttl=49 time=15.1 ms
64 bytes from 180.101.50.242 (180.101.50.242): icmp_seq=7 ttl=49 time=15.1 ms
^C
--- www.a.shifen.com ping statistics ---
```

RN 无法连接外网*。

*无法 ping www.baidu.com，100% packet loss.



```
[root@iZgc77np8m2wq0oi0gop1xZ ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.26.238.170  netmask 255.255.0.0  broadcast 172.26.255.255
        inet6 fe80::216:3eff:fe01:d5a  prefixlen 64  scopeid 0x20<link>
        ether 00:16:3e:01:0d:5a  txqueuelen 1000  (Ethernet)
        RX packets 73275  bytes 101156024 (96.4 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12819  bytes 1522709 (1.4 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@iZgc77np8m2wq0oi0gop1xZ ~]# ping www.baidu.com
PING www.a.shifen.com (180.101.50.242) 56(84) bytes of data.
^C
--- www.a.shifen.com ping statistics ---
86 packets transmitted, 0 received, 100% packet loss, time 84999ms
```

2.3.2 将主机 RG 配置成网关服务器[1]

下面，我们将主机 RG 配置成网关服务器，注意这里的操作在 RG 上。

开启转发功能并使其生效。

```
# 开启转发
echo "net.ipv4.ip_forward = 1" >> /etc/sysctl.conf
# 生效
sysctl -p
```

```
[root@iZgc7c8qqj3cg64zd8xsb8Z ~]# # 开启转发
[root@iZgc7c8qqj3cg64zd8xsb8Z ~]# echo "net.ipv4.ip_forward = 1" >> /etc/sysctl.conf
[root@iZgc7c8qqj3cg64zd8xsb8Z ~]# # 生效
[root@iZgc7c8qqj3cg64zd8xsb8Z ~]# sysctl -p
vm.swappiness = 0
kernel.sysrq = 1
net.ipv4.neigh.default.gc_stale_time = 120
net.ipv4.conf.all.rp_filter = 0
net.ipv4.conf.default.rp_filter = 0
net.ipv4.conf.default.arp_announce = 2
net.ipv4.conf.lo.arp_announce = 2
net.ipv4.conf.all.arp_announce = 2
net.ipv4.tcp_max_tw_buckets = 5000
net.ipv4.tcp_syncookies = 1
net.ipv4.tcp_max_syn_backlog = 1024
net.ipv4.tcp_synack_retries = 2
net.ipv4.ip_forward = 1
```

配置 iptables 做 SNAT，指定具体的网段*；然后查看 iptables。

*指定整个网段做 SNAT，转发到 RG 的内网 ip。

```
# 配置 iptables 做 SNAT，指定具体的网段
iptables -t nat -I POSTROUTING -s 172.26.238.0/24 -j SNAT --to-source 172.26.238.168
# 查看 iptables
iptables -L -t nat
```

```
[root@iZgc7c8qqj3cg64zd8xsb8Z ~]# iptables -t nat -I POSTROUTING -s 172.26.238.0/24 -j SNAT --to-source 172.26.23
8.168
[root@iZgc7c8qqj3cg64zd8xsb8Z ~]# iptables -L -t nat
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
SNAT       all  --  172.26.238.0/24      anywhere             to:172.26.238.168
```

2.3.3 配置主机 RN[2]

打开 ifcfg-eth0。

这里英文提示无法编辑。“If you do not want clout-init generated automatically, you can disable it in /etc/cloud/cloud.cfg”

cat /etc/sysconfig/network-scripts/ifcfg-eth0

```
[root@iZgc77np8m2wq0oi0gop1xZ ~]# cat /etc/sysconfig/network-scripts/ifcfg-eth0
# Created by cloud-init on instance boot automatically, do not edit.
# If you don't want cloud-init genrated automatically,you can disable it in /etc/cloud/cloud.cfg
# For more information, please refer to: https://help.aliyun.com/document_detail/57803.html
#
BOOTPROTO=dhcp
DEVICE=eth0
ONBOOT=yes
STARTMODE=auto
TYPE=Ethernet
USERCTL=no
```

修改/etc/cloud/cloud.cfg，自定义网络配置。

vim /etc/cloud/cloud.cfg

```
[root@iZgc77np8m2wq0oi0gop1xZ ~]# vim /etc/cloud/cloud.cfg
```

```
# The top level settings are used as module
# and system configuration.

# A set of users which may be applied and/or used by various modules
# when a 'default' entry is found it will reference the 'default_user'
# from the distro configuration specified below
users:
   - default

user:
    name: root
    lock_passwd: False

# If this is set, 'root' will not be able to ssh in and they
# will get a message to login instead as the default $user
disable_root: false

# This will cause the set+update hostname module to not operate (if true)
preserve_hostname: false

manage_etc_hosts: localhost
```

```
datasource_list: [ AliYun ]

# Example datasource config
datasource:
    AliYun:
        support_xen: false
        timeout: 5
        max_wait: 300
#       metadata_urls: [ 'blah.com' ]

# The modules that run in the 'init' stage
cloud_init_modules:
 - migrator
 - source-address
 - pip-source
 - seed_random
 - bootcmd
 - write-files
"/etc/cloud/cloud.cfg" 98L, 2069C
```

按 i 进入编辑模式，在 Example datasource config 之前增加 disabled 配置。

```
network:
  config: disabled
```

```
# This will cause the set+update hostname module to not operate (if true)
preserve_hostname: false

manage_etc_hosts: localhost

network:
  config: disabled

datasource_list: [ AliYun ]

# Example datasource config
```

编辑完成后，按 Esc 键退出编辑模式，然后输入:wq 并回车，保存退出文件。

```
:wq
```

```
 - seed_random
 - bootcmd
:wq
```

再次打开 ifcfg-eth0，并编辑该文件。

```
vim /etc/sysconfig/network-scripts/ifcfg-eth0
```

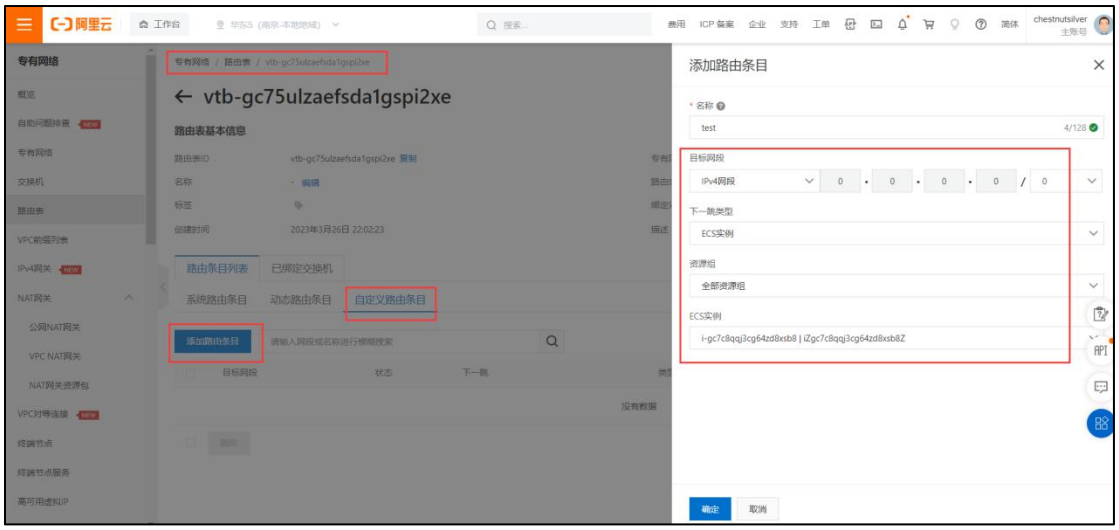
```
[root@iZgc77np8m2wq0oi0gop1xZ ~]# vim /etc/sysconfig/network-scripts/ifcfg-eth0
```

```
IPADDR=172.26.238.170
NETMASK=255.255.255.0
GATEWAY=172.26.238.168
BROADCAST=172.26.238.255
```

```
>_ 5. root@iZgc77np8m2wq0oi0gop1xZ:~ ×
# Created by cloud-init on instance boot automatically, do not edit.
# If you don't want cloud-init genrated automatically,you can disable it in /etc/cloud/cloud.cfg
# For more information, please refer to: https://help.aliyun.com/document_detail/57803.html
#
BOOTPROTO=dhcp
DEVICE=eth0
ONBOOT=yes
STARTMODE=auto
TYPE=Ethernet
USERCTL=no
IPADDR=172.26.238.170
NETMASK=255.255.255.0
GATEWAY=172.26.238.168
BROADCAST=172.26.238.255
```

2.3.4 添加路由表信息[3]

添加路由表信息。



2.3.5 测试网关配置

测试：主机 RN 可以 ping www.baidu.com，网关配置结束。

```
>_ 7. root@iZgc77np8m2wq0oi0gop1xZ:~
Last login: Mon Mar 27 01:46:08 2023 from 100.104.207.144

Welcome to Alibaba Cloud Elastic Compute Service !

[root@iZgc77np8m2wq0oi0gop1xZ ~]# ping www.baidu.com
PING www.a.shifen.com (180.101.50.242) 56(84) bytes of data.
64 bytes from 180.101.50.242 (180.101.50.242): icmp_seq=1 ttl=48 time=15.5 ms
64 bytes from 180.101.50.242 (180.101.50.242): icmp_seq=2 ttl=48 time=15.3 ms
64 bytes from 180.101.50.242 (180.101.50.242): icmp_seq=3 ttl=48 time=15.4 ms
64 bytes from 180.101.50.242 (180.101.50.242): icmp_seq=4 ttl=48 time=15.3 ms
64 bytes from 180.101.50.242 (180.101.50.242): icmp_seq=5 ttl=48 time=15.3 ms
^C
--- www.a.shifen.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4003ms
rtt min/avg/max/mdev = 15.308/15.397/15.526/0.178 ms
```

2.4 在网关（LG、RG）上配置 IPsec

此阶段（3.4）仅在两个网关主机（LG、RG）配置即可。

2.4.1 安装并查看 IPsec 配置

安装 epel-release、libreswan、xl2tpd，查看 IPsec 配置相关内容。

```
yum install -y epel-release
yum install -y libreswan
yum install -y xl2tpd
```

```
rpm -ql libreswan|grep -E -v "share|libe"
```

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# rpm -ql libreswan|grep -E -v "share|libe"
/etc/ipsec.conf
/etc/ipsec.d
/etc/ipsec.d/policies
/etc/ipsec.d/policies/block
/etc/ipsec.d/policies/clear
/etc/ipsec.d/policies/clear-or-private
/etc/ipsec.d/policies/portexcludes.conf
/etc/ipsec.d/policies/private
/etc/ipsec.d/policies/private-or-clear
/etc/ipsec.secrets
/etc/pam.d/pluto
/etc/prelink.conf.d
/etc/prelink.conf.d/libreswan-fips.conf
/usr/lib/systemd/system/ipsec.service
/usr/lib64/fipscheck/pluto.hmac
/usr/sbin/ipsec
/var/log/pluto/peer
/var/run/pluto
```

2.4.2 配置内核参数

修改/etc/sysctl.conf 文件。

开启路由转发（ip_forward）、关闭源路由验证（rp_filter）、关闭 ICMP 重定向（send_redirects、accept_redirects）。

```
vim /etc/sysctl.conf
```

```
# ipsec
## 开启路由转发
net.ipv4.ip_forward = 1
## 关闭源路由验证、关闭 ICMP 重定向
net.ipv4.conf.all.accept_redirects = 0
net.ipv4.conf.all.rp_filter = 0
net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.default.accept_redirects = 0
net.ipv4.conf.default.rp_filter = 0
net.ipv4.conf.default.send_redirects = 0
net.ipv4.conf.lo.accept_redirects = 0
net.ipv4.conf.lo.rp_filter = 0
net.ipv4.conf.lo.send_redirects = 0
```

```
>_ 2. root@iZuf6bb85jtec7klo4e99vZ:~ ×
vm.swappiness = 0
kernel.sysrq = 1

net.ipv4.neigh.default.gc_stale_time = 120

# see details in https://help.aliyun.com/knowledge_detail/39428.html
net.ipv4.conf.all.rp_filter = 0
net.ipv4.conf.default.rp_filter = 0
net.ipv4.conf.default.arp_announce = 2
net.ipv4.conf.lo.arp_announce = 2
net.ipv4.conf.all.arp_announce = 2

# see details in https://help.aliyun.com/knowledge_detail/41334.html
net.ipv4.tcp_max_tw_buckets = 5000
net.ipv4.tcp_syncookies = 1
net.ipv4.tcp_max_syn_backlog = 1024
net.ipv4.tcp_synack_retries = 2


# ipsec
## 开启路由转发
net.ipv4.ip_forward = 1
## 关闭源路由验证、关闭ICMP重定向
net.ipv4.conf.all.accept_redirects = 0
net.ipv4.conf.all.rp_filter = 0
net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.default.accept_redirects = 0
net.ipv4.conf.default.rp_filter = 0
net.ipv4.conf.default.send_redirects = 0
net.ipv4.conf.lo.accept_redirects = 0
net.ipv4.conf.lo.rp_filter = 0
net.ipv4.conf.lo.send_redirects = 0
~
-- INSERT --
```

加载内核参数，检验我们刚才的配置。

```
sysctl -p
```

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# sysctl -p
vm.swappiness = 0
kernel.sysrq = 1
net.ipv4.neigh.default.gc_stale_time = 120
net.ipv4.conf.all.rp_filter = 0
net.ipv4.conf.default.rp_filter = 0
net.ipv4.conf.default.arp_announce = 2
net.ipv4.conf.lo.arp_announce = 2
net.ipv4.conf.all.arp_announce = 2
net.ipv4.tcp_max_tw_buckets = 5000
net.ipv4.tcp_syncookies = 1
net.ipv4.tcp_max_syn_backlog = 1024
net.ipv4.tcp_synack_retries = 2
net.ipv4.ip_forward = 1
net.ipv4.conf.all.accept_redirects = 0
net.ipv4.conf.all.rp_filter = 0
net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.default.accept_redirects = 0
net.ipv4.conf.default.rp_filter = 0
net.ipv4.conf.default.send_redirects = 0
net.ipv4.conf.lo.accept_redirects = 0
net.ipv4.conf.lo.rp_filter = 0
net.ipv4.conf.lo.send_redirects = 0
```

启动 ipsec 服务，验证内核配置。

```
systemctl start ipsec
systemctl status ipsec
```

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# systemctl start ipsec
[root@iZuf6bb85jtec7klo4e99vZ ~]# systemctl status ipsec
● ipsec.service - Internet Key Exchange (IKE) Protocol Daemon for IPsec
   Loaded: loaded (/usr/lib/systemd/system/ipsec.service; disabled; vendor preset: disabled)
   Active: active (running) since Mon 2023-03-27 02:19:27 CST; 750ms ago
     Docs: man:ipsec(8)
           man:pluto(8)
           man:ipsec.conf(5)
  Process: 12565 ExecStartPre=/usr/sbin/ipsec --checknflog (code=exited, status=0/SUCCESS)
  Process: 12560 ExecStartPre=/usr/sbin/ipsec --checknss (code=exited, status=0/SUCCESS)
  Process: 12129 ExecStartPre=/usr/libexec/ipsec/_stackmanager start (code=exited, status=0/SUCCESS)
  Process: 12127 ExecStartPre=/usr/libexec/ipsec/addconn --config /etc/ipsec.conf --checkconfig (code=exited, status=0/SUCCESS)
 Main PID: 12579 (pluto)
   Status: "Startup completed."
    Tasks: 3
   Memory: 3.1M
   CGroup: /system.slice/ipsec.service
           └─12579 /usr/libexec/ipsec/pluto --leak-detective --config /etc/ipsec.conf --nofork

Mar 27 02:19:27 iZuf6bb85jtec7klo4e99vZ pluto[12579]: adding interface lo/lo 127.0.0.1:500
Mar 27 02:19:27 iZuf6bb85jtec7klo4e99vZ pluto[12579]: adding interface lo/lo 127.0.0.1:4500
Mar 27 02:19:27 iZuf6bb85jtec7klo4e99vZ pluto[12579]: adding interface lo/lo ::1:500
Mar 27 02:19:27 iZuf6bb85jtec7klo4e99vZ pluto[12579]: | setup callback for interface lo:500 fd 19
Mar 27 02:19:27 iZuf6bb85jtec7klo4e99vZ pluto[12579]: | setup callback for interface lo:4500 fd 18
Mar 27 02:19:27 iZuf6bb85jtec7klo4e99vZ pluto[12579]: | setup callback for interface lo:500 fd 17
Mar 27 02:19:27 iZuf6bb85jtec7klo4e99vZ pluto[12579]: | setup callback for interface eth0:4500 fd 16
Mar 27 02:19:27 iZuf6bb85jtec7klo4e99vZ pluto[12579]: | setup callback for interface eth0:500 fd 15
Mar 27 02:19:27 iZuf6bb85jtec7klo4e99vZ pluto[12579]: loading secrets from "/etc/ipsec.secrets"
Mar 27 02:19:27 iZuf6bb85jtec7klo4e99vZ pluto[12579]: no secrets filename matched "/etc/ipsec.d/*.secrets"
```

通过 ipsec verify 查看 ipsec 状态。

除密钥处于未设置状态外，其他均为 OK 状态。密钥稍后设置。

```
ipsec verify
```

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# ipsec verify
Verifying installed system and configuration files

Version check and ipsec on-path                         [OK]
Libreswan 3.25 (netkey) on 3.10.0-1127.19.1.el7.x86_64
Checking for IPsec support in kernel                    [OK]
 NETKEY: Testing XFRM related proc values
         ICMP default/send_redirects                    [OK]
         ICMP default/accept_redirects                  [OK]
         XFRM larval drop                               [OK]
Pluto ipsec.conf syntax                                 [OK]
Two or more interfaces found, checking IP forwarding    [OK]
Checking rp_filter                                      [OK]
Checking that pluto is running                          [OK]
 Pluto listening for IKE on udp 500                     [OK]
 Pluto listening for IKE/NAT-T on udp 4500              [OK]
 Pluto ipsec.secret syntax                              [OK]
Checking 'ip' command                                   [OK]
Checking 'iptables' command                             [OK]
Checking 'prelink' command does not interfere with FIPS [OK]
Checking for obsolete ipsec.conf options                [OBSOLETE KEYWORD]
warning: could not open include filename: '/etc/ipsec.d/*.conf'
```

2.4.3 查看端口开放状态

IPsec 需要开放的端口：500、4500、50、51。

需要开放到自身和自身发送的如下报文：

（1）目的端口为 500 和 4500 的 UDP 报文。 IKE 协商的初始端口使用的是 500，完成 NAT-T 能力检测和 NAT 网关探测后，封装 ISAKMP 消息的 UDP 端口号被修改为 4500，后续协商及数据传输都使用这个端口。

（2）AH 协议（IP 协议号为 51）：提供数据源认证、数据完整性校验和防报文重放功能，它能保护通信免受篡改，但不能防止窃听，适合用于传输非机密数据。AH 的工作原理是在每一个数据包上添加一个身份验证报文头，此报文头插在标准 IP 包头后面，对数据提供完整性保护。可选择的认证算法有 MD5（Message Digest）、SHA-1（Secure Hash Algorithm）等。

ESP 协议（IP 协议号为 50）：提供加密、数据源认证、数据完整性校验和防报文重放功能。ESP 的工作原理是在每一个数据包的标准 IP 包头后面添加一个 ESP 报文头，并在数据包后面追加一个 ESP 尾。与 AH 协议不同的是，ESP 将需要保护的用户数据进行加密后再封装到 IP 包中，以保证数据的机密性。常见的加密算法有 DES、3DES、AES 等。同时，作为可选项，用户可以选择 MD5、SHA-1 算法保证报文的完整性和真实性。

（3）业务报文根据内层报文进行具体的域间策略设置。

查看端口开放状态，发现 4500、500 已经开放。

```
netstat -unlp
```

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# netstat -unlp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
udp        0      0 127.0.0.1:323           0.0.0.0:*                           533/chronyd
udp        0      0 127.0.0.1:4500          0.0.0.0:*                           12579/pluto
udp        0      0 172.28.129.83:4500      0.0.0.0:*                           12579/pluto
udp        0      0 127.0.0.1:500           0.0.0.0:*                           12579/pluto
udp        0      0 172.28.129.83:500       0.0.0.0:*                           12579/pluto
udp        0      0 0.0.0.0:68              0.0.0.0:*                           760/dhclient
udp6       0      0 ::1:323                 :::*                                533/chronyd
udp6       0      0 ::1:500                 :::*                                12579/pluto
```

配置安全组规则，再次检查 4500、500 端口应当处于开放状态。

关闭防火墙。

```
systemctl disable firewalld
```

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# systemctl disable firewalld
```

安装 nmap，LG、RG 互相扫描对方的公网 ip，确认并验证 500、4500 端口的开放状态。

LG 扫描 RG（在 LG 命令行）：

```
yum install nmap
```

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# yum install nmap
```

```
nmap -sU 47.122.20.94 -p 500,4500 -Pn
```

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# nmap -sU 47.122.20.94 -p 500,4500 -Pn

Starting Nmap 6.40 ( http://nmap.org ) at 2023-03-27 02:41 CST
Nmap scan report for 47.122.20.94
Host is up.
PORT     STATE         SERVICE
500/udp  open|filtered isakmp
4500/udp open|filtered nat-t-ike

Nmap done: 1 IP address (1 host up) scanned in 3.19 seconds
```

RG 扫描 LG（在 RG 命令行）：

```
yum install nmap
```

```
[root@iZgc7c8qqj3cg64zd8xsb8Z ~]# yum install nmap
```

```
nmap -sU 47.103.98.35 -p 500,4500 -Pn
```

```
[root@iZgc7c8qqj3cg64zd8xsb8Z ~]# nmap -sU 47.103.98.35 -p 500,4500 -Pn

Starting Nmap 6.40 ( http://nmap.org ) at 2023-03-27 02:43 CST
Nmap scan report for 47.103.98.35
Host is up.
PORT     STATE         SERVICE
500/udp  open|filtered isakmp
4500/udp open|filtered nat-t-ike

Nmap done: 1 IP address (1 host up) scanned in 3.04 seconds
```

500、4500 端口均处于 open 状态。端口配置成功。

2.4.4 配置 myvpn.conf 文件

创建/etc/ipsec.d/myvpn.conf 文件，配置 ipsec。

注意：

（1）源端：①leftsourceip：本地内网 ip；②leftsubnet：本地子网网段；③leftid：

本地公网 ip。

（2）目标端：①rightsourceip、rightid：公网 ip；②rightsubnet：目标内网网段。

说明：

Libreswan 不使用术语 “source”（来源） 或 “destination”（目的）。相反，它用术语 “left”（左边） 和 “right”（右边） 来代指终端（主机）。虽然大多数管理员用 “left” 表示本地主机，“right” 表示远程主机，但是这样可以再大多数情况下在两个终端上使用相同的配置。

由于我们的服务器使用的是 vpc 网络,采用静态 nat 的形式,在配置 left 和 right 时，本端的 ip 需要使用内网 ip,或 %defaultroute。left 和 right 是两端的 ip 地址,而 leftid 和 rightid 为代号 id。

配置 LG：

vim /etc/ipsec.d/myvpn.conf

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# vim /etc/ipsec.d/myvpn.conf
```

```
conn myvpn
    ### phase 1 ###
    # 指定认证类型预共享密钥
    authby = secret
    # 指定 ike 算法
    ike = 3des-sha1
    # 指定 ike
    keyexchange = ike

    ### phase 2 ###
    # 指定使用 esp
    phase2 = esp
    # 指定 phase2 的算法
    phase2alg = 3des-sha1
    # 指定是否压缩
    compress = no
    # 指定是否加密
    pfs = yes
    # 指定连接添加类型，start 为开机自启动
    auto = start
    # 指定模式类型为隧道模式
    type = tunnel

    left=%defaultroute
    leftsourceip=172.28.129.83
    leftsubnet = 172.28.0.0/16
    leftid = 47.103.98.35
    leftnexthop = %defaultroute
```

```
    right = 47.122.20.94
    rightsubnet = 172.26.0.0/16
    rightid = 47.122.20.94
    rightnexthop = %defaultroute
```

```
>_ 2. root@iZuf6bb85jtec7klo4e99vZ:~ ×
conn myvpn
  ### phase 1 ###
  # 指定认证类型预共享密钥
  authby = secret
  # 指定ike算法
  ike = 3des-sha1
  # 指定ike
  keyexchange = ike

  ### phase 2 ###
  # 指定使用esp
  phase2 = esp
  # 指定phase2的算法
  phase2alg = 3des-sha1
  # 指定是否压缩
  compress = no
  # 指定是否加密
  pfs = yes
  # 指定连接添加类型，start为开机自启动
  auto = start
  # 指定模式类型为隧道模式
  type = tunnel

  left=%defaultroute
  leftsourceip=172.28.129.83
  leftsubnet = 172.28.0.0/16
  leftid = 47.103.98.35
  leftnexthop = %defaultroute

  right = 47.122.20.94
  rightsubnet = 172.26.0.0/16
  rightid = 47.122.20.94
  rightnexthop = %defaultroute
```

配置 RG：

```
vim /etc/ipsec.d/myvpn.conf
```

```
[root@iZgc7c8qqj3cg64zd8xsb8Z ~]# vim /etc/ipsec.d/myvpn.conf
```

```
conn myvpn
    ### phase 1 ###
    # 指定认证类型预共享密钥
    authby = secret
    # 指定 ike 算法
    ike = 3des-sha1
    # 指定 ike
    keyexchange = ike

    ### phase 2 ###
    # 指定使用 esp
    phase2 = esp
```

```
# 指定 phase2 的算法
phase2alg = 3des-sha1
# 指定是否压缩
compress = no
# 指定是否加密
pfs = yes
# 指定连接添加类型，start 为开机自启动
auto = start
# 指定模式类型为隧道模式
type = tunnel

left=%defaultroute
leftsourceip=172.26.238.168
leftsubnet = 172.26.0.0/16
leftid = 47.122.20.94
leftnexthop = %defaultroute

right = 47.103.98.35
rightsubnet = 172.28.0.0/16
rightid = 47.103.98.35
rightnexthop = %defaultroute
```

```
>_ 6. root@iZgc7c8qqj3cg64zd8xsb8Z:~ ×
conn myvpn
  ### phase 1 ###
  # 指定认证类型预共享密钥
  authby = secret
  # 指定ike算法
  ike = 3des-sha1
  # 指定ike
  keyexchange = ike

  ### phase 2 ###
  # 指定使用esp
  phase2 = esp
  # 指定phase2的算法
  phase2alg = 3des-sha1
  # 指定是否压缩
  compress = no
  # 指定是否加密
  pfs = yes
  # 指定连接添加类型，start为开机自启动
  auto = start
  # 指定模式类型为隧道模式
  type = tunnel

  left=%defaultroute
  leftsourceip=172.26.238.168
  leftsubnet = 172.26.0.0/16
  leftid = 47.122.20.94
  leftnexthop = %defaultroute

  right = 47.103.98.35
  rightsubnet = 172.28.0.0/16
  rightid = 47.103.98.35
  rightnexthop = %defaultroute
```

2.4.5 设置预共享密钥 PSK

在 LG、RG 创建文件/etc/ipsec.d/myvpn.secrets，存放预共享密钥。

注意使用相同的预共享密钥

```
vim /etc/ipsec.d/myvpn.secrets
```

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# vim /etc/ipsec.d/myvpn.secrets
```

预共享密钥遵循以下格式：

源 ip 目的 ip : PSK "key"

0.0.0.0 表示所有 ip，可以省略。

```
: PSK "2053182"
```

```
>_ 2. root@iZuf6bb85jtec7klo4e99vZ:~ ×
: PSK "2053182"
```

查看 LG、RG 预共享密钥，检验密钥设置是否相同。

```
cat /etc/ipsec.d/myvpn.secrets
```

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# cat /etc/ipsec.d/myvpn.secrets
: PSK "2053182"
```

```
[root@iZgc7c8qqj3cg64zd8xsb8Z ~]# cat /etc/ipsec.d/myvpn.secrets
: PSK "2053182"
```

2.4.6 建立 Gate to Gate IPsec vpn 连接！

在 LG、RG 开启日志服务，便于查看相关信息。

vim /etc/ipsec.conf

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# vim /etc/ipsec.conf
```

logfile=/var/log/pluto.log

（取消注释）

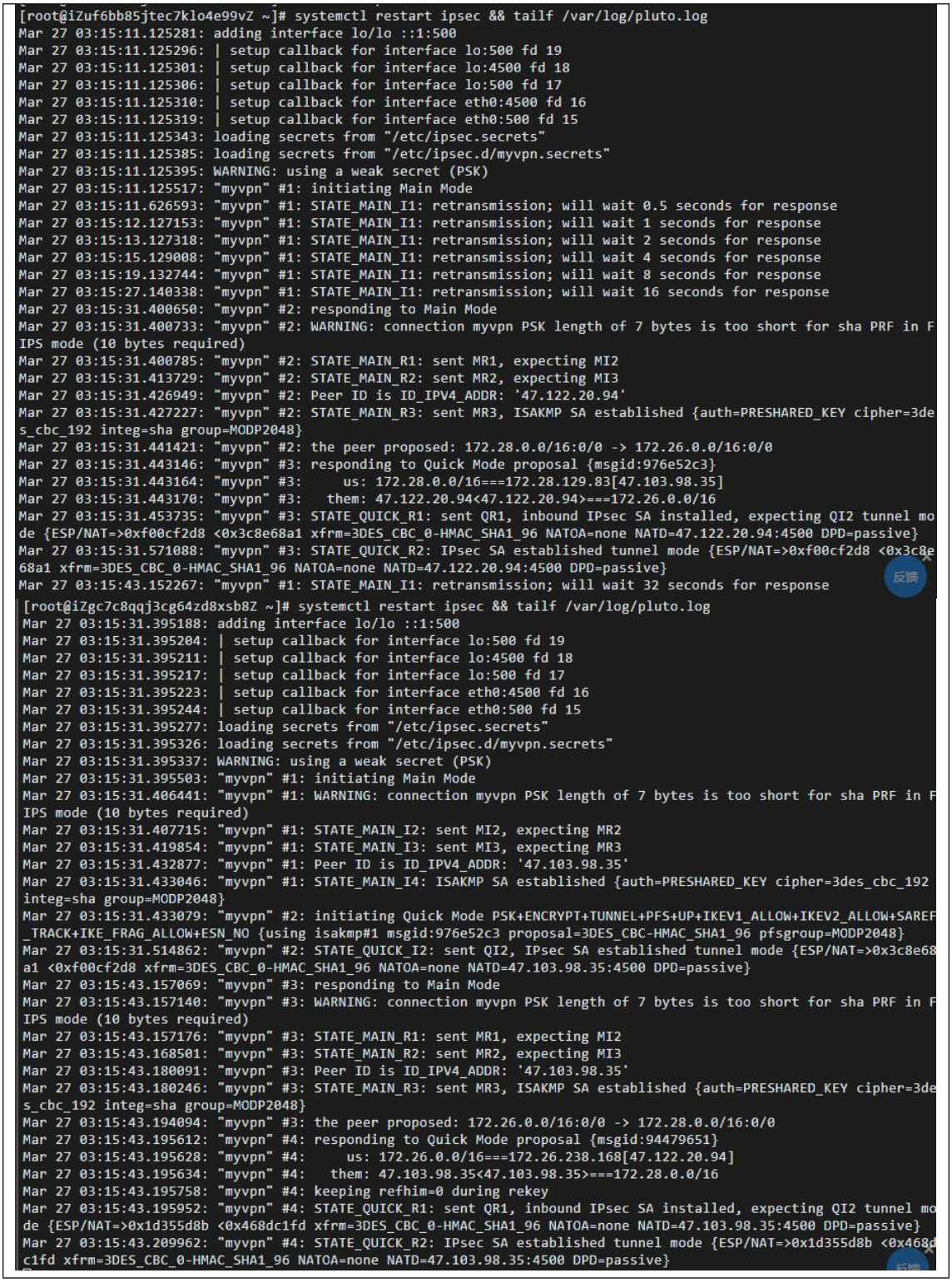
```
# /etc/ipsec.conf - Libreswan IPsec configuration file
#
# see 'man ipsec.conf' and 'man pluto' for more information
#
# For example configurations and documentation, see https://libreswan.org/wiki/

config setup
        # Normally, pluto logs via syslog.
        logfile=/var/log/pluto.log
        #
        # Do not enable debug options to debug configuration issues!
        #
        # plutodebug="control parsing"
        # plutodebug="all crypt"
        plutodebug=none
        #
        # NAT-TRAVERSAL support
        # exclude networks used on server side by adding %v4:!a.b.c.0/24
        # It seems that T-Mobile in the US and Rogers/Fido in Canada are
        # using 25/8 as "private" address space on their wireless networks.
        # This range has never been announced via BGP (at least up to 2015)
        virtual_private=%v4:10.0.0.0/8,%v4:192.168.0.0/16,%v4:172.16.0.0/12,%v4:25.0.0.0/8,%v4:100.64.0.0/10,%v6:
fd00::/8,%v6:fe80::/10

# if it exists, include system wide crypto-policy defaults
# include /etc/crypto-policies/back-ends/libreswan.config

# It is best to add your IPsec connections as separate files in /etc/ipsec.d/
include /etc/ipsec.d/*.conf
```

在 LG、RG 重启网络服务，建立 IPsec vpn，同时打印 vpn 连接过程。

IPsec SA established tunnel mode # 看到日志为建立隧道成功

systemctl restart ipsec && tailf /var/log/pluto.log

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# systemctl restart ipsec && tailf /var/log/pluto.log
[root@iZgc7c8qqj3cg64zd8xsb8Z ~]# systemctl restart ipsec && tailf /var/log/pluto.log
```



查看 IPsec 连接状态。

看到 ISAKMP SA established（第一阶段）、IPsec SA established（第二阶段），至此，成功建立 ipsec vpn!

ipsec auto --status
[root@iZuf6bb85jtec7klo4e99vZ ~]# ipsec auto --status

```
000 Connection list:
000
000 "myvpn": 172.28.0.0/16===172.28.129.83[47.103.98.35]---172.28.143.253...47.122.20.94<47.122.20.94>===172.26.0
.0/16; erouted; eroute owner: #4
000 "myvpn":     oriented; my_ip=172.28.129.83; their_ip=unset; my_updown=ipsec _updown;
000 "myvpn":   xauth us:none, xauth them:none,  my_username=[any]; their_username=[any]
000 "myvpn":   our auth:secret, their auth:secret
000 "myvpn":   modecfg info: us:none, them:none, modecfg policy:push, dns:unset, domains:unset, banner:unset, cat
:unset;
000 "myvpn":   labeled_ipsec:no;
000 "myvpn":   policy_label:unset;
000 "myvpn":   ike_life: 3600s; ipsec_life: 28800s; replay_window: 32; rekey_margin: 540s; rekey_fuzz: 100%; keyi
ngtries: 0;
000 "myvpn":   retransmit-interval: 500ms; retransmit-timeout: 60s;
000 "myvpn":   initial-contact:no; cisco-unity:no; fake-strongswan:no; send-vendorid:no; send-no-esp-tfc:no;
000 "myvpn":   policy: PSK+ENCRYPT+TUNNEL+PFS+UP+IKEV1_ALLOW+IKEV2_ALLOW+SAREF_TRACK+IKE_FRAG_ALLOW+ESN_NO;
000 "myvpn":   conn_prio: 16,16; interface: eth0; metric: 0; mtu: unset; sa_prio:auto; sa_tfc:none;
000 "myvpn":   nflog-group: unset; mark: unset; vti-iface:unset; vti-routing:no; vti-shared:no; nic-offload:auto;
000 "myvpn":   our idtype: ID_IPV4_ADDR; our id=47.103.98.35; their idtype: ID_IPV4_ADDR; their id=47.122.20.94
000 "myvpn":   dpd: action:hold; delay:0; timeout:0; nat-t: encaps:auto; nat_keepalive:yes; ikev1_natt:both
000 "myvpn":   newest ISAKMP SA: #1; newest IPsec SA: #4;
000 "myvpn":   IKE algorithms: 3DES_CBC-HMAC_SHA1-MODP2048, 3DES_CBC-HMAC_SHA1-MODP1536
000 "myvpn":   IKE algorithm newest: 3DES_CBC_192-HMAC_SHA1-MODP2048
000 "myvpn":   ESP algorithms: 3DES_CBC-HMAC_SHA1_96
000 "myvpn":   ESP algorithm newest: 3DES_CBC_000-HMAC_SHA1_96; pfsgroup=<Phase1>
```

```
000 Total IPsec connections: loaded 1, active 1
000
000 State Information: DDoS cookies not required, Accepting new IKE connections
000 IKE SAs: total(2), half-open(0), open(0), authenticated(2), anonymous(0)
000 IPsec SAs: total(2), authenticated(2), anonymous(0)
000
000 #1: "myvpn":4500 STATE_MAIN_I4 (ISAKMP SA established); EVENT_SA_REPLACE in 2423s; newest ISAKMP; lastdpd=-1s
(seq in:0 out:0); idle; import:admin initiate
000 #2: "myvpn":4500 STATE_MAIN_R3 (sent MR3, ISAKMP SA established); EVENT_SA_REPLACE in 3134s; lastdpd=-1s(seq
in:0 out:0); idle; import:not set
000 #3: "myvpn":4500 STATE_QUICK_R2 (IPsec SA established); EVENT_SA_REPLACE in 28334s; isakmp#2; idle; import:no
t set
000 #3: "myvpn" esp.f00cf2d8@47.122.20.94 esp.3c8e68a1@172.28.129.83 tun.0@47.122.20.94 tun.0@172.28.129.83 ref=0
 refhim=0 Traffic: ESPin=0B ESPout=0B! ESPmax=4194303B
000 #4: "myvpn":4500 STATE_QUICK_I2 (sent QI2, IPsec SA established); EVENT_SA_REPLACE in 27864s; newest IPSEC; e
route owner; isakmp#1; idle; import:admin initiate
000 #4: "myvpn" esp.468dc1fd@47.122.20.94 esp.1d355d8b@172.28.129.83 tun.0@47.122.20.94 tun.0@172.28.129.83 ref=0
 refhim=0 Traffic: ESPin=0B ESPout=0B! ESPmax=4194303B
000
000 Bare Shunt list:
000
```

2.4.7 验证成功建立 Gate to Gate IPsec vpn 连接！

在 LG、RG 查看连接状态，出现对应连接！

ifconfig -a
[root@iZuf6bb85jtec7klo4e99vZ ~]# ifconfig -a

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# ifconfig -a
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.28.129.83  netmask 255.255.240.0  broadcast 172.28.143.255
        inet6 fe80::216:3eff:fe1f:6234  prefixlen 64  scopeid 0x20<link>
        ether 00:16:3e:1f:62:34  txqueuelen 1000  (Ethernet)
        RX packets 115133  bytes 148496822 (141.6 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 28192  bytes 3956443 (3.7 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ip_vti0: flags=128<NOARP>  mtu 1480
        tunnel   txqueuelen 1000  (IPIP Tunnel)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
[root@iZgc7c8qqj3cg64zd8xsb8Z ~]#  ifconfig -a
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.26.238.168  netmask 255.255.0.0  broadcast 172.26.255.255
        inet6 fe80::216:3eff:fe01:1aa7  prefixlen 64  scopeid 0x20<link>
        ether 00:16:3e:01:1a:a7  txqueuelen 1000  (Ethernet)
        RX packets 113773  bytes 148332906 (141.4 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 23070  bytes 3548576 (3.3 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ip_vti0: flags=128<NOARP>  mtu 1480
        tunnel   txqueuelen 1000  (IPIP Tunnel)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

在 LG、RG 保持 VPN 开启状态，在 RN ping LN，在 LN 通过 wireshark，成功抓到 IPsec vpn 数据包！（需要首先安装 wireshark）

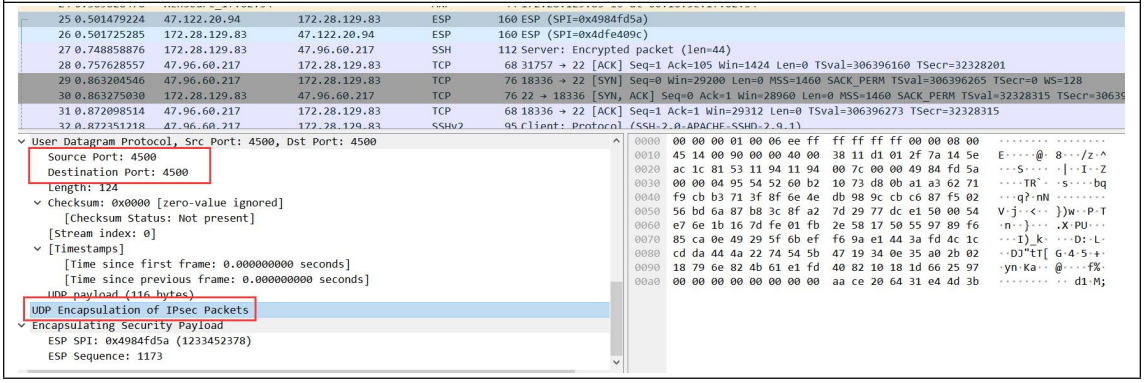
```
yum -y install wireshark
```

```
tshark -i any host 172.28.129.84 -c 100 # LN
ping 172.28.129.84 # RN
systemctl restart ipsec && tailf /var/log/pluto.log # LG
systemctl restart ipsec && tailf /var/log/pluto.log # RG
```

```
[root@iZgc77np8m2wq0oi0gop1xZ ~]# ping 172.28.129.84
PING 172.28.129.84 (172.28.129.84) 56(84) bytes of data.
64 bytes from 172.28.129.84: icmp_seq=1 ttl=62 time=13.3 ms
64 bytes from 172.28.129.84: icmp_seq=2 ttl=62 time=13.0 ms
64 bytes from 172.28.129.84: icmp_seq=3 ttl=62 time=13.0 ms
64 bytes from 172.28.129.84: icmp_seq=4 ttl=62 time=13.0 ms
64 bytes from 172.28.129.84: icmp_seq=5 ttl=62 time=13.0 ms
64 bytes from 172.28.129.84: icmp_seq=6 ttl=62 time=13.0 ms
64 bytes from 172.28.129.84: icmp_seq=7 ttl=62 time=13.0 ms
64 bytes from 172.28.129.84: icmp_seq=8 ttl=62 time=13.0 ms
64 bytes from 172.28.129.84: icmp_seq=9 ttl=62 time=13.0 ms
64 bytes from 172.28.129.84: icmp_seq=10 ttl=62 time=13.0 ms
64 bytes from 172.28.129.84: icmp_seq=11 ttl=62 time=13.1 ms
64 bytes from 172.28.129.84: icmp_seq=12 ttl=62 time=13.0 ms
```

```
220 24.912204869 172.26.238.168 -> 172.28.129.84 ICMP 100 Echo (ping) request  id=0x30ed, seq=1/256, ttl=62
221 24.912243508 172.28.129.84 -> 172.26.238.168 ICMP 100 Echo (ping) reply    id=0x30ed, seq=1/256, ttl=64 (request in 220)
222 24.912402382 172.28.129.83 -> 172.28.129.84 ICMP 128 Redirect             (Redirect for host)
223 25.249707289 172.28.129.84 -> 100.104.100.1 SSH 672 Encrypted response packet len=604
224 25.257422599 100.104.100.1 -> 172.28.129.84 TCP 68 lofr-lm > ssh [ACK] Seq=61 Ack=26329 Win=1424 Len=0 TSval=1737210935 TSecr=30378547
225 25.749636208 172.28.129.84 -> 100.104.100.1 SSH 336 Encrypted response packet len=268
226 25.757347863 100.104.100.1 -> 172.28.129.84 TCP 68 lofr-lm > ssh [ACK] Seq=61 Ack=26597 Win=1424 Len=0 TSval=1737211435 TSecr=30379047
227 25.913204924 172.26.238.168 -> 172.28.129.84 ICMP 100 Echo (ping) request  id=0x30ed, seq=2/512, ttl=62
228 25.913241032 172.28.129.84 -> 172.26.238.168 ICMP 100 Echo (ping) reply    id=0x30ed, seq=2/512, ttl=64 (request in 227)
229 25.913315259 172.28.129.83 -> 172.28.129.84 ICMP 128 Redirect             (Redirect for host)
230 26.249721085 172.28.129.84 -> 100.104.100.1 SSH 672 Encrypted response packet len=604
231 26.257433275 100.104.100.1 -> 172.28.129.84 TCP 68 lofr-lm > ssh [ACK] Seq=61 Ack=27201 Win=1424 Len=0 TSval=1737211935 TSecr=30379547
232 26.914351780 172.26.238.168 -> 172.28.129.84 ICMP 100 Echo (ping) request  id=0x30ed, seq=3/768, ttl=62
233 26.914380857 172.28.129.84 -> 172.26.238.168 ICMP 100 Echo (ping) reply    id=0x30ed, seq=3/768, ttl=64 (request in 232)
234 26.914453768 172.28.129.83 -> 172.28.129.84 ICMP 128 Redirect             (Redirect for host)
235 26.999368790 172.28.129.84 -> 100.104.100.1 SSH 448 Encrypted response packet len=380
236 27.007069574 100.104.100.1 -> 172.28.129.84 TCP 68 lofr-lm > ssh [ACK] Seq=61 Ack=27581 Win=1424 Len=0 TSval=1737212684 TSecr=30380297
237 27.500255870 172.28.129.84 -> 100.104.100.1 SSH 560 Encrypted response packet len=492
238 27.507944420 100.104.100.1 -> 172.28.129.84 TCP 68 lofr-lm > ssh [ACK] Seq=61 Ack=28073 Win=1424 Len=0 TSval=1737213185 TSecr=30380798
239 27.915076497 172.26.238.168 -> 172.28.129.84 ICMP 100 Echo (ping) request  id=0x30ed, seq=4/1024, ttl=62
240 27.915105210 172.28.129.84 -> 172.26.238.168 ICMP 100 Echo (ping) reply    id=0x30ed, seq=4/1024, ttl=64 (request in 239)
```

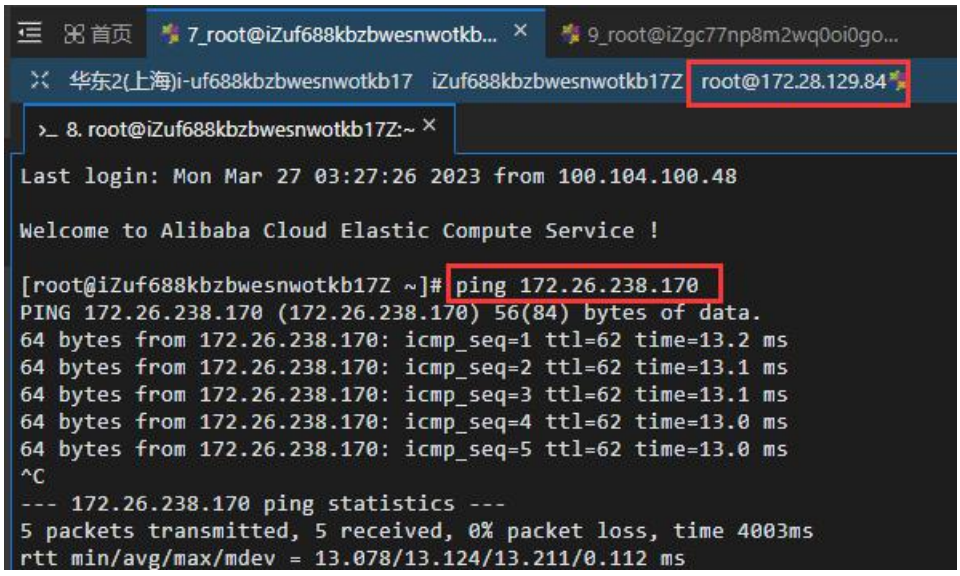
```
tshark -i any -w /test/test.pcap host 172.28.129.83 && esp -c 100
```



LN、RN 两端 ping 通！

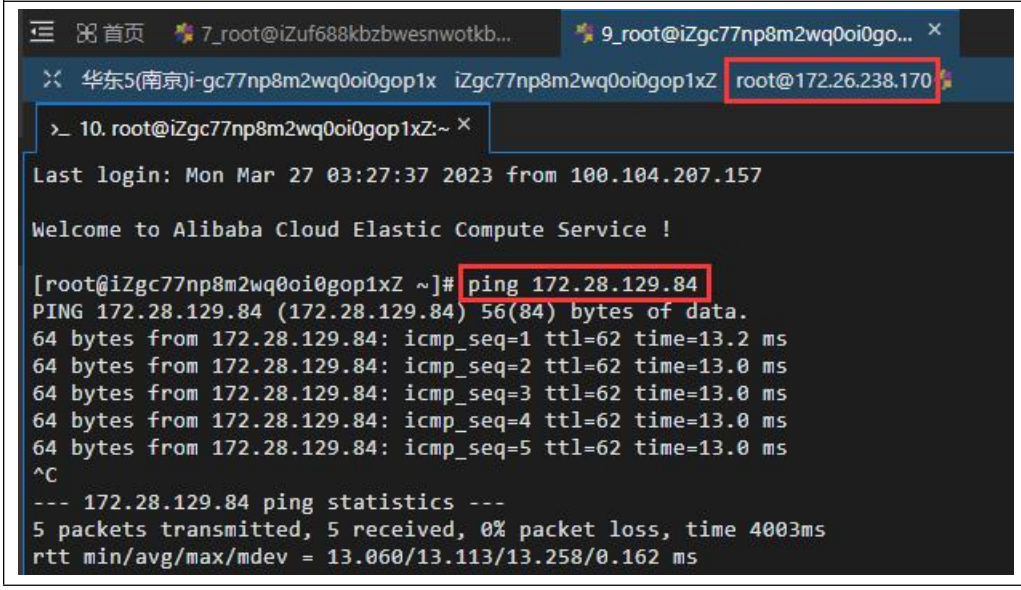
LN：

```
ping 172.26.238.170
```



RN：

ping 172.28.129.84



THAT IS A "WOW MOMENT"! CONGRATULATIONS!

3. ike、l2tp、ipsec 参数配置

本小题的实验思路是在已经建立完成 Gate to Gate IPsec vpn 的基础上，通过查阅 Libreswan 开源程序库的官方文档[4]，了解各种参数的含义和作用；然后，修改 ike、l2tp、ipsec 的各种参数，观察实验结果；最后，通过 wireshark 验证有效性和正确性。

IPsec VPN 基础参数如下图所示：

IPSec SA 生成方式	手动指定生成	IKE 协商生成	IKE SA 协商模式	主模式、野蛮模式
			IKE SA 加密方式	DES、3DES、AES
			IKE SA 验证方式	MD5-HMAC、SHA-HMAC
			IKE SA 密钥生成方式	DH1、DH2、DH5
			IKE SA 认证方式	预共享密钥认证、数字证书认证
			IKE SA 身份标识	IP、FQDN、USER-FQDN、证书 DN
			IKE SA 生命周期	60 秒到 86400 秒（缺省 86400 秒）
IPSec SA 安全协议	AH、ESP			
IPSec SA 封装模式	传输模式、隧道模式			
IPSec SA 加密方式	DES、3DES、AES			
IPSec SA 验证方式	MD5-HMAC、SHA-HMAC			
IPSec SA 生命周期	0 或者 120 秒到 86400 秒（缺省 3600 秒）、0 或 2560KB 到 536870912KB（缺省 4608000KB）			

3.1 总结配置 ike 的方法与参数

3.1.1 ike 背景知识

IPSec 中通信双方建立的连接叫做安全关联（IPSec SA），双方通过参数协商完成 IPSec SA 建立后，通过 IPSec SA 传输加密的数据报文进行通信。所以两个对等体间要想通过 IPSec VPN 通信，首先要建立 IPSec SA。在进行 IPSec SA 建立时对等体间要进行 IPSec SA 参数协商，两端参数相同时才会建立成功。

ike 包括许多参数，要想在两个站点之间安全的传输 IP 数据流，它们之间必须要先进行协商，协商它们之间所采用的加密算法，封装技术以及密钥。这个协商过程是通过 IKE 来完成的，IKE 协商分两个阶段运行：

第一阶段：建立 ISAKMP SA 协商的是以下信息：

1、对等体之间采用何种方式做认证，是预共享密钥还是数字证书。

2、双方使用哪种加密算法

3、双方使用哪种 HMAC 方式，是 MD5 还是 SHA

4、双方使用哪种 Diffie-Hellman 密钥组

5、使用哪种协商模式（主模式或主动模式）

6、还要协商 SA 的生存期

第二阶段：建立 IPsec SA 协商的是以下信息：

1、双方使用哪种封装技术，AH 还是 ESP

2、双方使用哪种加密算法

3、双方使用哪种 HMAC 方式，是 MD5 还是 SHA

4、使用哪种传输模式，是隧道模式还是传输模式

5、还要协商 SA 的生存期

我们可以首先查看在上一问已经建立好的 Gate to Gate IPsec vpn 的参数，这种参数已经是一个可行的 case 样例。

```
ipsec auto --status
```

```
config setup options:

configdir=/etc, configfile=/etc/ipsec.conf, secrets=/etc/ipsec.secrets, ipsecdir=/etc/ipsec.d
nssdir=/etc/ipsec.d, dumpdir=/run/pluto, statsbin=unset
dnssec-rootkey-file=/var/lib/unbound/root.key, dnssec-trusted=<unset>
sbindir=/usr/sbin, libexecdir=/usr/libexec/ipsec
pluto_version=3.25, pluto_vendorid=OE-Libreswan-3.25
nhelpers=-1, uniqueids=yes, dnssec-enable=yes, perpeerlog=no, logappend=yes, logip=yes, shuntlifetime=900s, xfrmlifetime=300s
ddos-cookies-threshold=50000, ddos-max-halfopen=25000, ddos-mode=auto
ikeport=500, ikebuf=0, msg_errqueue=yes, strictcrlpolicy=no, crlcheckinterval=0, listen=<any>, nflog-all=0
ocsp-enable=no, ocsp-strict=no, ocsp-timeout=2, ocsp-uri=<unset>
ocsp-trust-name=<unset>
ocsp-cache-size=1000, ocsp-cache-min-age=3600, ocsp-cache-max-age=86400, ocsp-method=get
secctx-attr-type=32001
debug:

nat-traversal=yes, keep-alive=20, nat-ikeport=4500
virtual-private (%priv):
- allowed subnets: 10.0.0.0/8, 192.168.0.0/16, 172.16.0.0/12, 25.0.0.0/8, 100.64.0.0/10, fd00::/8, fe80::/10
```

```
stats db_ops: {curr_cnt, total_cnt, maxsz} :context={0,29,64} trans={0,29,6936} attrs={0,29,4624}

Connection list:

"myvpn": 172.28.0.0/16===172.28.129.83[47.103.98.35]---172.28.143.253...47.122.20.94<47.122.20.94>===172.26.0.0/16; erouted; eroute owner: #49
"myvpn":     oriented; my_ip=172.28.129.83; their_ip=unset; my_updown=ipsec _updown;
"myvpn":   xauth us:none, xauth them:none,  my_username=[any]; their_username=[any]
"myvpn":   our auth:secret, their auth:secret
"myvpn":   modecfg info: us:none, them:none, modecfg policy:push, dns:unset, domains:unset, banner:unset, cat:unset;
"myvpn":   labeled_ipsec:no;
"myvpn":   policy_label:unset;
"myvpn":   ike_life: 3600s; ipsec_life: 28800s; replay_window: 32; rekey_margin: 540s; rekey_fuzz: 100%; keyingtries: 0;
"myvpn":   retransmit-interval: 500ms; retransmit-timeout: 60s;
"myvpn":   initial-contact:no; cisco-unity:no; fake-strongswan:no; send-vendorid:no; send-no-esp-tfc:no;
"myvpn":   policy: PSK+ENCRYPT+TUNNEL+PFS+UP+IKEV1_ALLOW+IKEV2_ALLOW+SAREF_TRACK+IKE_FRAG_ALLOW+ESN_NO;
"myvpn":   conn_prio: 16,16; interface: eth0; metric: 0; mtu: unset; sa_prio:auto; sa_tfc:none;
"myvpn":   nflog-group: unset; mark: unset; vti-iface:unset; vti-routing:no; vti-shared:no; nic-offload:auto;
"myvpn":   our idtype: ID_IPV4_ADDR; our id=47.103.98.35; their idtype: ID_IPV4_ADDR; their id=47.122.20.94
"myvpn":   dpd: action:hold; delay:0; timeout:0; nat-t: encaps:auto; nat_keepalive:yes; ikev1_natt:both
"myvpn":   newest ISAKMP SA: #47; newest IPsec SA: #49;
"myvpn":   IKE algorithms: 3DES_CBC-HMAC_SHA1-MODP2048, 3DES_CBC-HMAC_SHA1-MODP1536
"myvpn":   IKE algorithm newest: 3DES_CBC_192-HMAC_SHA1-MODP2048
"myvpn":   ESP algorithms: 3DES_CBC-HMAC_SHA1_96
"myvpn":   ESP algorithm newest: 3DES_CBC_000-HMAC_SHA1_96; pfsgroup=<Phase1>
```

```
Total IPsec connections: loaded 1, active 1

State Information: DDoS cookies not required, Accepting new IKE connections
IKE SAs: total(4), half-open(0), open(0), authenticated(4), anonymous(0)
IPsec SAs: total(7), authenticated(7), anonymous(0)

#3: "myvpn":4500 STATE_QUICK_R2 (IPsec SA established); EVENT_SA_REPLACE in 19688s; isakmp#2; idle; import:not set
#3: "myvpn" esp.f00cf2d8@47.122.20.94 esp.3c8e68a1@172.28.129.83 tun.0@47.122.20.94 tun.0@172.28.129.83 ref=0 refhim=0 Traffic: ESPin=0B ESPout=0B! ESPmax=4194303B
#4: "myvpn":4500 STATE_QUICK_I2 (sent QI2, IPsec SA established); EVENT_SA_REPLACE in 19218s; isakmp#1; idle; import:admin initiate
#4: "myvpn" esp.468dc1fd@47.122.20.94 esp.1d355d8b@172.28.129.83 tun.0@47.122.20.94 tun.0@172.28.129.83 ref=0 refhim=0 Traffic: ESPin=1KB ESPout=1KB! ESPmax=4194303B
#19: "myvpn":4500 STATE_QUICK_I2 (sent QI2, IPsec SA established); EVENT_SA_REPLACE in 22281s; isakmp#16; idle; import:admin initiate
#19: "myvpn" esp.5f344d9d@47.122.20.94 esp.98d2a22e@172.28.129.83 tun.0@47.122.20.94 tun.0@172.28.129.83 ref=0 refhim=0 Traffic: ESPin=0B ESPout=0B! ESPmax=4194303B
#20: "myvpn":4500 STATE_QUICK_I2 (sent QI2, IPsec SA established); EVENT_SA_REPLACE in 22685s; isakmp#18; idle; import:admin initiate
#20: "myvpn" esp.fb88d6b1@47.122.20.94 esp.4f5f14f2@172.28.129.83 tun.0@47.122.20.94 tun.0@172.28.129.83 ref=0 refhim=0 Traffic: ESPin=0B ESPout=0B! ESPmax=4194303B
#43: "myvpn":4500 STATE_MAIN_I4 (ISAKMP SA established); EVENT_SA_REPLACE in 725s; lastdpd=-1s(seq in:0 out:0); idle; import:admin initiate
#44: "myvpn":4500 STATE_MAIN_R3 (sent MR3, ISAKMP SA established); EVENT_SA_REPLACE in 1173s; lastdpd=-1s(seq in:0 out:0); idle; import:not set
#45: "myvpn":4500 STATE_QUICK_I2 (sent QI2, IPsec SA established); EVENT_SA_REPLACE in 25692s; isakmp#43; idle; import:admin initiate
#45: "myvpn" esp.968dc9fb@47.122.20.94 esp.328d9e43@172.28.129.83 tun.0@47.122.20.94 tun.0@172.28.129.83 ref=0 refhim=0 Traffic: ESPin=0B ESPout=0B! ESPmax=4194303B
#46: "myvpn":4500 STATE_MAIN_I4 (ISAKMP SA established); EVENT_SA_REPLACE in 757s; lastdpd=-1s(seq in:0 out:0); idle; import:admin initiate
#47: "myvpn":4500 STATE_MAIN_I4 (ISAKMP SA established); EVENT_SA_REPLACE in 607s; newest ISAKMP; lastdpd=-1s(seq in:0 out:0); idle; import:admin initiate
#48: "myvpn":4500 STATE_QUICK_I2 (sent QI2, IPsec SA established); EVENT_SA_REPLACE in 25849s; isakmp#46; idle; import:admin initiate
#48: "myvpn" esp.390ded66@47.122.20.94 esp.6d5cb635@172.28.129.83 tun.0@47.122.20.94 tun.0@172.28.129.83 ref=0 refhim=0 Traffic: ESPin=0B ESPout=0B! ESPmax=4194303B
#49: "myvpn":4500 STATE_QUICK_I2 (sent QI2, IPsec SA established); EVENT_SA_REPLACE in 25767s; newest IPSEC; eroute owner; isakmp#47; idle; import:admin initiate
#49: "myvpn" esp.c70dbb50@47.122.20.94 esp.28648461@172.28.129.83 tun.0@47.122.20.94 tun.0@172.28.129.83 ref=0 refhim=0 Traffic: ESPin=0B ESPout=0B! ESPmax=4194303B
```

3.1.2 ike 参数总结

（1）IKE SA 协商模式

在 RFC2409（The Internet Key Exchange）中规定，IKE 第一阶段的协商可以采用两种模式：主模式（Main Mode）和野蛮模式（Aggressive Mode）。

主模式被设计成将密钥交换信息与身份、认证信息相分离。这种分离保护了身份信息；交换的身份信息受已生成的 Diffie-Hellman 共享密钥的保护。但这增加了 3 条消息的开销。

野蛮模式则允许同时传送与 SA、密钥交换和认证相关的载荷。将这些载荷组合到一条消息中减少了消息的往返次数，但是就无法提供身份保护了。虽然野蛮模式存在一些功能限制，但可以满足某些特定的网络环境需求。

参数设置方法：

使用 IKEv1 野蛮模式还是主模式（默认）：

```
aggressive = yes | no
```

（2）IKE SA 加密方式

IKE SA 使用对称加密算法对数据进行加密和解密，保证数据的安全性。常用的对称加密算法有 DES、3DES、AES 等，这三个加密算法的安全性由高到低依次是：AES、3DES、DES，

安全性高的加密算法实现机制复杂，运算速度慢。

参数设置方法：

设置 ike 加密方式：

```
ike = <cipher suites>
```

（3）IKE SA 验证方式

IKE SA 使用验证算法对报文完整性及来源合法性进行验证，常用的验证方式有 MD5-HMAC、SHA1-HMAC 等，是 HASH 算法和 HMAC 两种技术的结合。HASH 算法实现对报文进行完整性校验，常见的 HASH 算法有 MD5、SHA1 等，MD5 算法的计算速度比 SHA1 算法快，而 SHA1 算法的安全强度比 MD5 算法高。HMAC(Hash-based Message Authentication Code)是一种基于 HASH 算法和密钥进行消息认证的方法，实现对报文来源的合法性进行验证，可以与任何 HASH 算法捆绑使用。

（4）IKE SA 密钥生成方式

DH（Diffie-Hellman）是一种非对称密钥算法，双方可通过仅交换一些数据，即可计算出双方的密钥，并且第三方捕获了其中的数据也无法计算得出密钥。DH 产生的密钥用于数据报文加密及 HMAC 计算中。对等体两端 DH 组长度需指定为相同，常用的 DH 组长度有 768bit（DH1）、1024bit（DH2）、1536bit（DH5）。

（5）IKE SA 认证方式

在 IKE 对等体之间在进行身份认证时支持通过预共享密钥认证和数字证书认证两种方式来确认对方身份的合法性。预共享密钥认证配置比较简单，是目前比较常用的认证方式。数字证书认证相对复杂但安全性较高，对安全性有较高要求的场景建议使用数字证书认证。

参数设置方法：

设置两个网关应如何相互认证：

```
authby = pubkey | rsasig | ecdsasig | psk | secret | never | xauthpsk | xauthrsasig
```

（6）IKE SA 身份标识

在 IKE SA 协商中对等体双方需要使用相同类型的身份标识，常用的身份标识类型有 4 种，IP 地址、FQDN、USER-FQDN、证书 DN。

（7）IKE SA 生命周期

由于 IPSec SA 协商是建立在 IKE SA 基础上的，因此为节省协商 IPSec SA 的时间，一般 IKE SA 生命周期（60 秒到 86400 秒，缺省 86400 秒）比 IPSec SA 生命周期设置的长。当在进行 IKE SA 协商时，两端对等体设置的 IKE SA 生命周期不同不会造成 IKE SA 协商失败，而使用发送方设置的 IKE SA 生命周期。

参数设置方法：

设置 ike 生命周期：

```
ikelifetime = 3h | <time>
```

3.1.3 ike 参数配置方法

以修改 IKE SA 生命周期为例。

首先查看当前的参数。

```
vim /etc/ipsec.d/myvpn.conf
```

```
[root@iZgc7c8qqj3cg64zd8xsb8Z ~]# vim /etc/ipsec.d/myvpn.conf
```

```
ikelifetime=3h
```

```
# 指定ike生命周期
ikelifetime = 3h
```

3.1.4 验证 ike 参数配置方法正确性

ipsec auto --status 查看 ipsec vpn 状态。

```
ipsec auto --status
```

```
our auth:secret, their auth:secret
modecfg info: us:none, them:none, modecfg policy:push, dns:unset, domai
labeled_ipsec:no;
policy_label:unset;
ike_life: 10800s; ipsec_life: 28800s; replay_window: 32; rekey_margin:
retransmit_interval: 500ms; retransmit-timeout: 60s;
initial-contact:no; cisco-unity:no; fake-strongswan:no; send-vendorid:n
policy: PSK+ENCRYPT+TUNNEL+IKEV1_ALLOW+IKEV2_ALLOW+SAREF_TRACK+IKE_FRAG
conn_prio: 16,16; interface: eth0; metric: 0; mtu: unset; sa_prio:auto;
nflog-group: unset; mark: unset; vti-iface:unset; vti-routing:no; vti-s
our idtype: ID_IPV4_ADDR; our id=47.103.98.35; their idtype: ID_IPV4_AD
```

验证正确！3h=10800s！

3.2 总结 l2tp-ipsec 的配置方法与参数

3.2.1 ah/esp 背景知识

AH 和 ESP 是 IPSec 的两种安全协议，用于实现 IPSec 在身份认证和数据加密的安全机制。

AH 协议（Authentication Header，协议号 51），主要提供数据完整性确认、数据来源确认、防重放等安全特性。AH 通常使用 MD5-HMAC、SHA-HMAC 等验证算法实现数据完整性；

ESP 协议（Encapsulating Security Payload，协议号 50），主要提供数据完整性确认、数据加密、数据来源确认、防重放等安全特性。ESP 通常使用 DES、3DES、AES 等加密算法实现数据加密，使用 MD5-HMAC、SHA-HMAC 等验证算法实现数据完整性。ESP 协议相比 AH 协议多了支持数据加密、支持 NAT 穿越（NAT-T）这两大优势，是目前 IPSec VPN 较为常用的安全协议。

3.2.2 修改 esp 加密算法

首先查看当前的参数，发现为 3ecs-sha1，改为 null-sha1。

观察到，加密方法发生了改变。

```
vim /etc/ipsec.d/myvpn.conf
```

```
### phase 2 ###
# 指定使用esp
phase2 = esp
# 指定phase2的算法
phase2alg = null-sha1
```

```
systemctl restart ipsec && tailf /var/log/pluto.log
```

修改前：

```
10 bytes required)


group=MODP2048}
_FRAG_ALLOW+ESN_NO {using isakmp#1 msgid:9e5083b3 proposal=3DES_CBC-HMAC_SHA1_96 pfsgroup=MODP2048}
806e xfrm=3DES_CBC_0-HMAC_SHA1_96 NATOA=none NATD=47.103.98.35:4500 DPD=passive}

10 bytes required)
```

修改后：

```
x179f1429 <0xef9e3a95 xfrm=NULL_0-HMAC_SHA1_96 NATOA=none NATD=47.103.98.35:4500 DPD=passive}
_0-HMAC_SHA1_96 NATOA=none NATD=47.103.98.35:4500 DPD=passive}

ytes required)


p=MODP2048}
LOW+ESN_NO {using isakmp#3 msgid:7c49284f proposal=NULL-HMAC_SHA1_96 pfsgroup=no-pfs}
xfrm=NULL_0-HMAC_SHA1_96 NATOA=none NATD=47.103.98.35:4500 DPD=passive}
```

ipsec auto --status

```
"myvpn":   newest ISAKMP SA: #6; newest IPsec SA: #7;
"myvpn":   IKE algorithms: 3DES_CBC-HMAC_SHA1-MODP2048, 3DES_CBC-HMAC_SHA1-MODP1536
"myvpn":   IKE algorithm newest: 3DES_CBC_192-HMAC_SHA1-MODP2048
"myvpn":   ESP algorithms: NULL-HMAC_SHA1_96
"myvpn":   ESP algorithm newest: NULL_000-HMAC_SHA1_96; pfsgroup=<N/A>
```

3.2.3 验证修改 esp 加密算法正确性

让我们通过 wireshark，看一下是否已经取消掉 esp 加密！

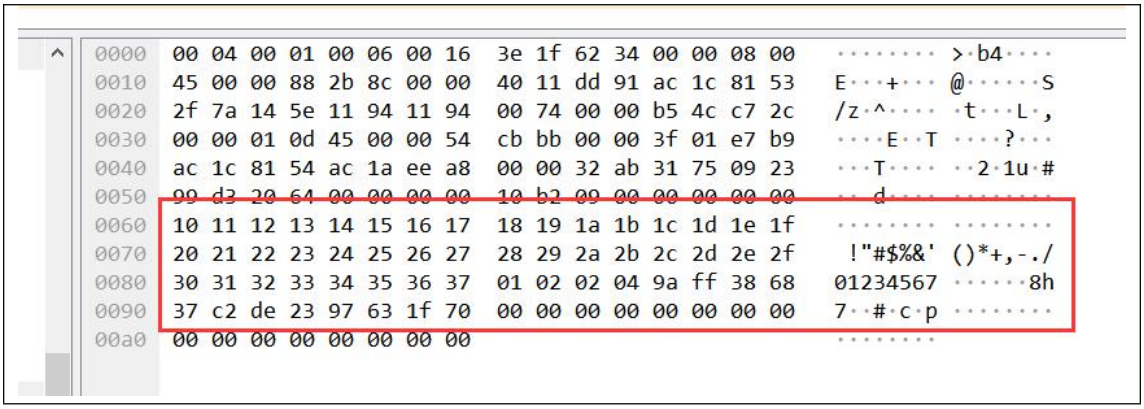
tshark -i any -w /test/test2.pcap host 172.28.129.83 && esp -c 100

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# tshark -i any -w /test/test2.pcap host 172.28.129.83 && esp -c 100
```

修改前（有加密 test.pcap）：

```
0000  00 00 00 01 00 06 ee ff  ff ff ff ff 00 00 08 00   ........ ........
0010  45 14 00 90 00 00 40 00  38 11 d1 01 2f 7a 14 5e   E·····@· 8···/z·^
0020  ac 1c 81 53 11 94 11 94  00 7c 00 00 49 84 fd 5a   ···S···· ·|··I··Z
0030  00 00 04 95 54 52 60 b2  10 73 d8 0b a1 a3 62 71   ····TR`· ·s····bq
0040  f9 cb b3 71 3f 8f 6e 4e  db 98 9c cb c6 87 f5 02   ···q?·nN ........
0050  56 bd 6a 87 b8 3c 8f a2  7d 29 77 dc e1 50 00 54   V·j··<·· })w··P·T
0060  e7 6e 1b 16 7d fe 01 fb  2e 58 17 50 55 97 89 f6   ·n··}··· .X·PU···
0070  85 ca 0e 49 29 5f 6b ef  f6 9a e1 44 3a fd 4c 1c   ···I)_k· ···D:·L·
0080  cd da 44 4a 22 74 54 5b  47 19 34 0e 35 a0 2b 02   ··DJ"tT[ G·4·5·+·
0090  18 79 6e 82 4b 61 e1 fd  40 82 10 18 1d 66 25 97   ·yn·Ka·· @····f%·
00a0  00 00 00 00 00 00 00 00  aa ce 20 64 31 e4 4d 3b   ........ ·· d1·M;
```

修改后（无加密 test2.pcap）：



验证正确！esp 加密算法被成功修改了！

3.3 总结 ipsec 模式选择及参数配置方法

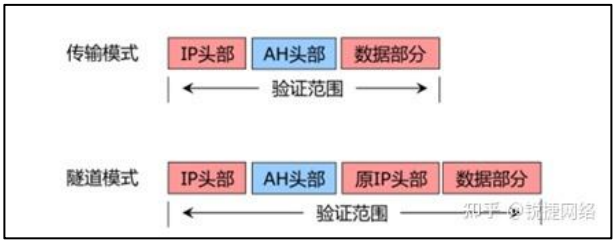
3.3.1 ipsec 封装模式背景知识

封装模式用于指定安全协议的封装位置，有传输模式和隧道模式两种：

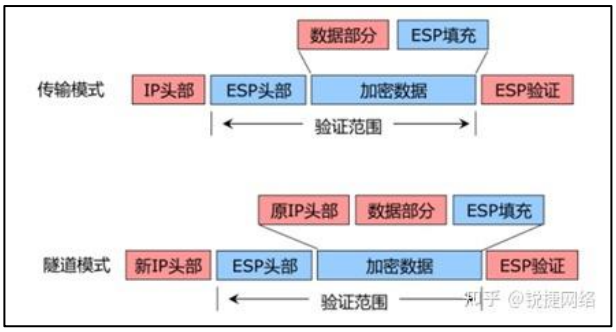
传输（Transport）模式下，AH 头或 ESP 头插入 IP 头和传输层协议之间，不改变原始报文头，IPSec 隧道的源和目的地址就是最终通信双方的源和目的地址，所以只能保护两个 IPSec 对等体之间相互通信。一般常用在使用 GRE over IPSec 或 L2TP over IPSec 协议的场景中，使用 IPSec 隧道保护 GRE 或 L2TP 对等体；

隧道（Tunnel）模式下，AH 头或 ESP 头插在原始 IP 头之前，并且新生成一个 IP 头放在 ESP 头或 AH 头之前，所以可以保护两个 IPSec 对等体背后两个网络之间进行通信。一般常用在站点间网络互通的场景，是较常用的封装模式。

AH 协议两种封装模式下的报文封装：



ESP 协议两种封装模式下的报文封装：



参数设置方法：

设置 ipsec 封装模式：

```
type = tunnel | transport | transport_proxy | passthrough | drop
```

3.3.2 ipsec 模式选择及参数配置方法

首先查看当前的参数，发现为隧道模式（Tunnel）。

vim /etc/ipsec.d/myvpn.conf

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# vim /etc/ipsec.d/myvpn.conf
```

```
  # 指定模式类型为隧道模式
  type = tunnel
```

systemctl restart ipsec && tailf /var/log/pluto.log

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# systemctl restart ipsec && tailf /var/log/pluto.log
Mar 27 05:48:30.080724: "myvpn" #3: responding to Quick Mode proposal {msgid:fa8546ff}
Mar 27 05:48:30.080742: "myvpn" #3:     us: 172.28.0.0/16===172.28.129.83[47.103.98.35]
Mar 27 05:48:30.080748: "myvpn" #3:   them: 47.122.20.94<47.122.20.94>===172.26.0.0/16
Mar 27 05:48:30.081230: "myvpn" #3: STATE_QUICK_R1: sent QR1, inbound IPsec SA installed, expecting QI2 tunnel mode {ES
Mar 27 05:48:30.187455: "myvpn" #3: STATE_QUICK_R2: IPsec SA established tunnel mode {ESP/NAT=>0xf539d931 <0x9d84d743 x
Mar 27 05:48:38.687245: "myvpn" #1: STATE_MAIN_I1: retransmission; will wait 32 seconds for response
Mar 27 05:48:38.697040: "myvpn" #1: WARNING: connection myvpn PSK length of 7 bytes is too short for sha PRF in FIPS m
Mar 27 05:48:38.698099: "myvpn" #1: STATE_MAIN_I2: sent MI2, expecting MR2
Mar 27 05:48:38.709475: "myvpn" #1: STATE_MAIN_I3: sent MI3, expecting MR3
Mar 27 05:48:38.722275: "myvpn" #1: Peer ID is ID_IPV4_ADDR: '47.122.20.94'
Mar 27 05:48:38.722404: "myvpn" #1: STATE_MAIN_I4: ISAKMP SA established {auth=PRESHARED_KEY cipher=3des_cbc_192 integ
Mar 27 05:48:38.722446: "myvpn" #4: initiating Quick Mode PSK+ENCRYPT+TUNNEL+PFS+UP+IKEV1_ALLOW+IKEV2_ALLOW+SAREF_TRACK
Mar 27 05:48:38.739271: "myvpn" #4: STATE_QUICK_I2: sent QI2, IPsec SA established tunnel mode {ESP/NAT=>0x3e6046fe <0x
^C
```

3.3.3 验证 ipsec 模式选择及参数配置正确性

通过修改/etc/ipsec.d/myvpn.conf 文件，将其改为传输模式（Transform），并再次检验。

vim /etc/ipsec.d/myvpn.conf

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# vim /etc/ipsec.d/myvpn.conf
```

```
  # 指定模式类型为传输模式
  type = transport
```

systemctl restart ipsec && tailf /var/log/pluto.log

```
[root@iZuf6bb85jtec7klo4e99vZ ~]# systemctl restart ipsec && tailf /var/log/pluto.log
Mar 27 06:02:47.829519: "myvpn" #3: STATE_QUICK_R2: IPsec SA established transport mode {ESP/NAT=>0xcaf5553c <0
Mar 27 06:02:48.011820: "myvpn" #1: STATE_MAIN_I1: retransmission; will wait 4 seconds for response
Mar 27 06:02:48.022148: "myvpn" #1: WARNING: connection myvpn PSK length of 7 bytes is too short for sha PRF in
Mar 27 06:02:48.023115: "myvpn" #1: STATE_MAIN_I2: sent MI2, expecting MR2
Mar 27 06:02:48.034455: "myvpn" #1: STATE_MAIN_I3: sent MI3, expecting MR3
Mar 27 06:02:48.047233: "myvpn" #1: Peer ID is ID_IPV4_ADDR: '47.122.20.94'
Mar 27 06:02:48.047341: "myvpn" #1: STATE_MAIN_I4: ISAKMP SA established {auth=PRESHARED_KEY cipher=3des_cbc_19
Mar 27 06:02:48.047373: "myvpn" #4: initiating Quick Mode PSK+ENCRYPT+PFS+UP+IKEV1_ALLOW+IKEV2_ALLOW+SAREF_TRAC
Mar 27 06:02:48.064022: "myvpn" #4: STATE_QUICK_I2: sent QI2, IPsec SA established transport mode {ESP/NAT=>0x4
^C
```

验证正确！成功改为了传输模式！

4. 总结 linux 下 ipsec 网关程序模块构架及其相互关系

4.1 IPsec 模块构架

linux 下 ipsec 的模块架构由如下五部分构成：

1. IKEv1、IKEv2 协议处理模块

负责协商安全会话的密钥，管理加密算法、身份验证和密钥交换协议，以及处理网络地址转换（NAT）等问题。

2. 加密和认证模块

包括对称密钥加密算法、公钥加密算法、哈希算法和数字签名算法等；管理证书和密钥，以确保安全通信的机密性和完整性。

3. 安全策略管理模块

管理安全策略，确定是否、如何保护流量以及如何管理访问控制。

4. IPsec 处理模块

实现 IPsec 协议主要功能，加密解密认证和数据完整性保护，处理 ipsec 的 SA，管理和流量选择。

5.网络接口处理模块

与操作系统交互，处理网络配置信息，负责处理数据包转发和重定向，确保流量通过正确的通道传输。

4.2 IPsec 模块间相互关系

IPSec 在 Linux 上有两种数据面，目前一般使用第二种 Linux 内核的 XFRM 框架，除非是很老的机器没有这个的时候会用第一种：

1.KLIPS ，很古老，2.6 版本以前的 Linux 版本使用。通过创建出虚拟 IPSec 接口，路由 IPSec 数据包，可以很方便添加 firewall 规则。

2.Linux 内核 XFRM/NETKEY（ipsec transform），执行具体的转发策略和封包解包，如果有 offload 还要执行 offload 提升速度。新版本内核也支持创建出虚拟 IPSec 接口，方便添加 firewall 规则。

libreswan 是目前使用较多的 IPSec 控制面实现，平常使用中需要掌握 libreswan 中的 pluto 和 whack 命令

1.pluto 命令是 IPsec IKE keying 守护进程，负责自动化 ipsec 之间的 SA 协商。

启动 pluto 守护进程命令：ipsec pluto

2.whack 命令是用户和 pluto 守护进程进行交互使用的命令

```
# 添加一条 ipsec 连接(vpn1 的具体配置写在文件中)
ipsec whack addconn vpn1 --config ipsec.config
# 允许 pluto 守护进程开始监听
ipsec whack --listen
# 初始化 ipsec 连接
ipsec whack --initiate --name vpn1
```

Linux 内核通过 XFRM 框架来支持 IPSec 的数据面实现，xfrm 框架支持 Linux 的网络 namespace。

5.心得收获

虽然经历许多困难，但好在很圆满地完成了实验任务。

通过这次实验，尤其深入理解了 IPsec SA 的结构和功能，对于 AH 和 ESP 的有了更多了解。特别是当 Gate to Gate IPsec VPN 成功建立并验证正确、Wireshark 验证参数修改正确的时候，确实令人惊喜。

遗憾的是，由于时间关系，AH 和 ESP 可能存在的组合数暂未实验，也是因为对课堂知识还需要进一步理解才能开展这样的探索。希望今后有机会完善。

第三部分　参考文献

1.《CentOS 7 配置成网关服务器》https://www.cnblogs.com/EasonJim/p/10206728.html

2.《安装 cloud-init》https://help.aliyun.com/document_detail/57803.html

3.《阿里云通过 EIP 实现 VPC 下的 SNAT 以及 DNAT》https://blog.csdn.net/xiaoxinla/article/details/118159253

4.《Libreswan ipsec.conf》https://libreswan.org/man/ipsec.conf.5.html
5.《站点间 IPSec VPN 网络技术深度解析》https://zhuanlan.zhihu.com/p/549488116
6.《使用 libreswan 搭建 ipsec 点对点隧道 实现两 idc 内网网段互通》https://blog.csdn.net/weixin_43423965/article/details/105071519
7.《详解 SLB、EIP、NAT 网关之间区别，合理选择云上公网入口》https://developer.aliyun.com/article/391631
8.《ipsec.conf 配置文件的一些参数说明》https://www.cnblogs.com/jianmuzi/p/16882584.html
9.《IPSEC 野蛮模式的详细介绍》http://www.51sjk.com/b33b132786/
10.《网关到底是什么求通俗易懂讲解》https://www.zhihu.com/question/362842680
11.《centos7 抓包（tcpdump）详解》https://blog.csdn.net/u012132164/article/details/127967362
12.《strongswan 搭建 IPSec 实验环境》https://huaweicloud.csdn.net/63566c01d3efff3090b5f240.html
13.《Libreswan Git 存储库》https://github.com/libreswan/libreswan/