Wireguard 简介

WireGuard 是一款极其简单但快速且现代的 VPN，采用最先进的加密技术。它的目标是比 IPsec 更快、更简单、更精简、更有用，同时避免令人头疼的问题。它的性能远高于 OpenVPN。

WireGuard 被设计为通用 VPN，可在嵌入式接口和超级计算机上运行，适合许多不同的情况。它最初针对 Linux 内核发布，现在已跨平台（Windows、macOS、BSD、iOS、Android）且可广泛部署。它目前正在大力开发中，但它可能已被视为业内最安全、最易于使用且最简单的 VPN 解决方案。

2020 年 3 月，该软件的 Linux 版本达到了稳定的生产版本，并被纳入 Linux 5.6 内核，并向后移植到一些 Linux 发行版中的早期 Linux 内核。 Linux 内核组件根据 GNU 通用公共许可证 (GPL) 版本 2 获得许可；其他实现均遵循 GPLv2 或其他免费/开源许可证。

官方网站：https://www.wireguard.com/

项目地址：https://github.com/WireGuard

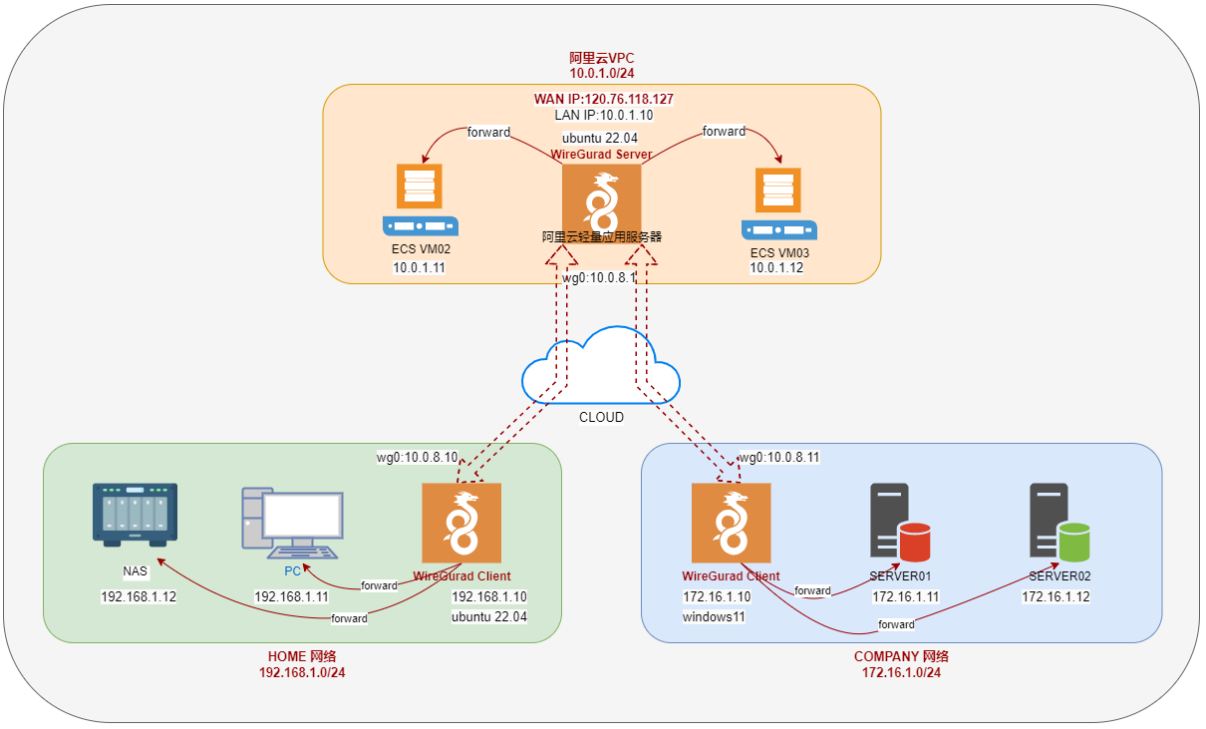
组网规划示例

准备三个网络环境，在阿里云上创建一台轻量云服务器，并且具有公网IP地址，作为wireguard server端。在其他网络环境部署一个 WireGuard 客户端，负责转发流量到所在本地子网。

节点名称	IP地址	操作系统	网络环境	网络地址段
wireguard server	WAN IP：120.76.118.127 LAN IP：10.0.1.10	Ubuntu 22.04	ALIYUN 网络	10.0.1.0/24
wireguard client01	192.168.1.10	Ubuntu 22.04	HOME 网络	192.168.1.0/24
wireguard client02	172.16.1.10	Windows 11	COMPANY 网络	172.16.1.0/24

说明：WireGuard 组网所使用的隧道网络IP 地址段不能与任何本地网络相冲突。本示例为 `10.0.8.0/24`。

整体组网架构图如下：



WireGuard中心辐射型组网

组网后效果：任意网络下的设备之间能够通过内网地址互相访问。

防火墙配置：阿里云防火墙需开通端口 `51820/UDP` 端口。

配置wireguard server端

安装wireguard server

在阿里云轻量服务器 `wireguard server` 上安装wireguard，作为中继服务器端。

```
apt update -y
apt install -y wireguard
```

在中继服务器端开启IP转发，实现不同客户端跨网段互联互通和正确路由。

```
cat <<EOF | sudo tee /etc/sysctl.d/wireguard.conf
net.ipv4.ip_forward = 1
EOF
sudo sysctl --system
```

生成服务器端密钥

在 `wireguard server` 服务器端生成服务器私钥和公钥

```
cd /etc/wireguard/
umask 077
wg genkey | tee server.key | wg pubkey > server.key.pub
```

显示生成的中继服务器端密钥

```
root@wireguard-server:/etc/wireguard# ls
server.key  server.key.pub
```

生成客户端(client1)密钥

在 `wireguard server` 服务器端生成 `HOME网络环境` 的 `client01` 私钥，并通过私钥生成公钥

```
wg genkey | tee client1.key | wg pubkey > client1.key.pub
```

显示生成的 `client01` 密钥

```
root@wireguard-server:/etc/wireguard# ls client1*
client1.key  client1.key.pub
```

生成客户端(client2)密钥

在 `wireguard server` 服务器端生成 `COMPANY网络环境` 的 `client02` 私钥，并通过私钥生成公钥

```
wg genkey | tee client2.key | wg pubkey > client2.key.pub
```

显示生成的 `client02` 密钥

```
root@wireguard-server:/etc/wireguard# ls client2*
client2.key  client2.key.pub
```

当前 `/etc/wireguard/` 目录下有以下密钥文件

```
root@wireguard-server:~# ls /etc/wireguard/
client1.key  client1.key.pub  client2.key  client2.key.pub  server.key
server.key.pub
```

创建服务器配置文件

在 `wireguard server` 服务器端为wireguard中继服务器创建配置文件。

查看服务器网卡名称，名称为`eth0`，记录该名称，需要与下面配置文件 `PostUp` 和 `PostDown` 中的参数一致。

```
root@wireguard-server:/etc/wireguard# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group
default qlen 1000
    link/ether 00:16:3e:18:9b:50 brd ff:ff:ff:ff:ff:ff
    altname enp0s5
    altname ens5
    inet 10.0.1.10/20 metric 100 brd 10.0.1.255 scope global dynamic eth0
       valid_lft 315119772sec preferred_lft 315119772sec
    inet6 fe80::216:3eff:fe18:9b50/64 scope link
       valid_lft forever preferred_lft forever
```

在 `wireguard server` 服务器端创建wireguard配置文件：

```
cd /etc/wireguard
cat >/etc/wireguard/wg0.conf<<EOF
[Interface]
PrivateKey = $(cat server.key)
Address = 10.0.8.1/24
ListenPort = 51820
PostUp   = iptables -A FORWARD -i %i -j ACCEPT; iptables -A FORWARD -o %i -j
ACCEPT; iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
PostDown = iptables -D FORWARD -i %i -j ACCEPT; iptables -D FORWARD -o %i -j
ACCEPT; iptables -t nat -D POSTROUTING -o eth0 -j MASQUERADE

[Peer]
PublicKey =  $(cat client1.key.pub)
AllowedIPs = 10.0.8.10/32,192.168.1.0/24

[Peer]
PublicKey =  $(cat client2.key.pub)
AllowedIPs = 10.0.8.11/32,172.16.1.0/24
EOF
```

查看创建后的配置文件

```
root@wireguard-server:~# cat /etc/wireguard/wg0.conf
[Interface]
PrivateKey = kND88sthrLmvK9eRUefaV82rsNRGFXoEzq+biqbsDU4=
Address = 10.0.8.1/24
ListenPort = 51820
```

```
PostUp   = iptables -A FORWARD -i %i -j ACCEPT; iptables -A FORWARD -o %i -j
ACCEPT; iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
PostDown = iptables -D FORWARD -i %i -j ACCEPT; iptables -D FORWARD -o %i -j
ACCEPT; iptables -t nat -D POSTROUTING -o eth0 -j MASQUERADE

[Peer]
PublicKey =  SEkxxA17Rb/eDFyO7SsVeUcD/OosYlVWcgL9IHnUAEU=
AllowedIPs = 10.0.8.10/32,192.168.1.0/24

[Peer]
PublicKey =  Uurn5z46FnntquzoGoy/5OZABcak3XaK/NhAodu8CAU=
AllowedIPs = 10.0.8.11/32,172.16.1.0/24
```

参数说明：

1. `[Interface]` 表示这一部分是服务器端接口的配置。
2. `PrivateKey` 是服务器的私钥，用于识别和加密通信。通常私钥保存在 `server.key` 文件中，然后在配置文件中读取。
3. `Address` 指定了服务器在 WireGuard 虚拟网络中`wg0`接口的 IP 地址，这里是 `10.0.8.1`。
4. `PostUp` 和 `PostDown` 是在 WireGuard 接口启动和停止时执行的 iptables 规则。这些规则允许转发 WireGuard 接口的流量，并在服务器的物理网卡 `eth0` 上进行 NAT 伪装，以允许客户端访问互联网。注意修改 `PostUP`及`PostDown`参数的接口名称，与实际匹配。
5. `ListenPort` 指定了 WireGuard 服务器监听的 UDP 端口，这里是 `51820`。
6. `[Peer]` 表示这一部分是客户端的配置。
7. `PublicKey` 是客户端的公钥，用于识别和加密与该客户端的通信。这里从 `client1.key.pub` 文件中读取。
8. `AllowedIPs` 指定了允许该对等节点（peer）发送过来的 VPN 流量中的源地址范围。这里是对端两个客户端的接口IP地址 `10.0.8.10/32`和 `10.0.8.11/32`，以及两个客户端本地的网段 `192.168.1.0/24`和 `172.16.1.0/24`。

启动wireguard

在服务器端启动wireguard服务

```
systemctl enable --now wg-quick@wg0
```

查看wireguard服务运行状态

```
root@wireguard-server:~# systemctl status wg-quick@wg0
● wg-quick@wg0.service - WireGuard via wg-quick(8) for wg0
     Loaded: loaded (/lib/systemd/system/wg-quick@.service; enabled; vendor
preset: enabled)
     Active: active (exited) since Mon 2024-04-15 11:16:51 CST; 43s ago
       Docs: man:wg-quick(8)
             man:wg(8)
             https://www.wireguard.com/
             https://www.wireguard.com/quickstart/
             https://git.zx2c4.com/wireguard-tools/about/src/man/wg-quick.8
             https://git.zx2c4.com/wireguard-tools/about/src/man/wg.8
```

```
    Process: 19783 ExecStart=/usr/bin/wg-quick up wg0 (code=exited,
status=0/SUCCESS)
   Main PID: 19783 (code=exited, status=0/SUCCESS)
        CPU: 42ms
Apr 15 11:16:51 wireguard-server systemd[1]: Starting WireGuard via wg-quick(8)
for wg0...
Apr 15 11:16:51 wireguard-server wg-quick[19783]: [#] ip link add wg0 type
wireguard
Apr 15 11:16:51 wireguard-server wg-quick[19783]: [#] wg setconf wg0 /dev/fd/63
Apr 15 11:16:51 wireguard-server wg-quick[19783]: [#] ip -4 address add
10.0.8.1/24 dev wg0
Apr 15 11:16:51 wireguard-server wg-quick[19783]: [#] ip link set mtu 1420 up dev
wg0
Apr 15 11:16:51 wireguard-server wg-quick[19783]: [#] ip -4 route add
192.168.1.0/24 dev wg0
Apr 15 11:16:51 wireguard-server wg-quick[19783]: [#] ip -4 route add
172.16.1.0/24 dev wg0
Apr 15 11:16:51 wireguard-server wg-quick[19783]: [#] iptables -A FORWARD -i wg0
-j ACCEPT; iptables -A FORWARD -o wg0 -j ACCEPT; iptables -t nat -A POSTROUTING -
o eth0 -j MASQUERADE
Apr 15 11:16:51 node71 systemd[1]: Finished WireGuard via wg-quick(8) for wg0.
```

或者使用命令启动停止wireguard，如果需要切换到使用 `wg-quick` 启停服务，需要先执行`systemctl stop wg-quick@wg0`，以免冲突。

```
wg-quick down wg0
wg-quick up wg0
```

通过上面的命令，我们启动了服务，并使其在启动时自动启动。要验证配置是否已应用，运行 `wg` 命令。生成的输出应显示有关 `wg0` 接口的信息：

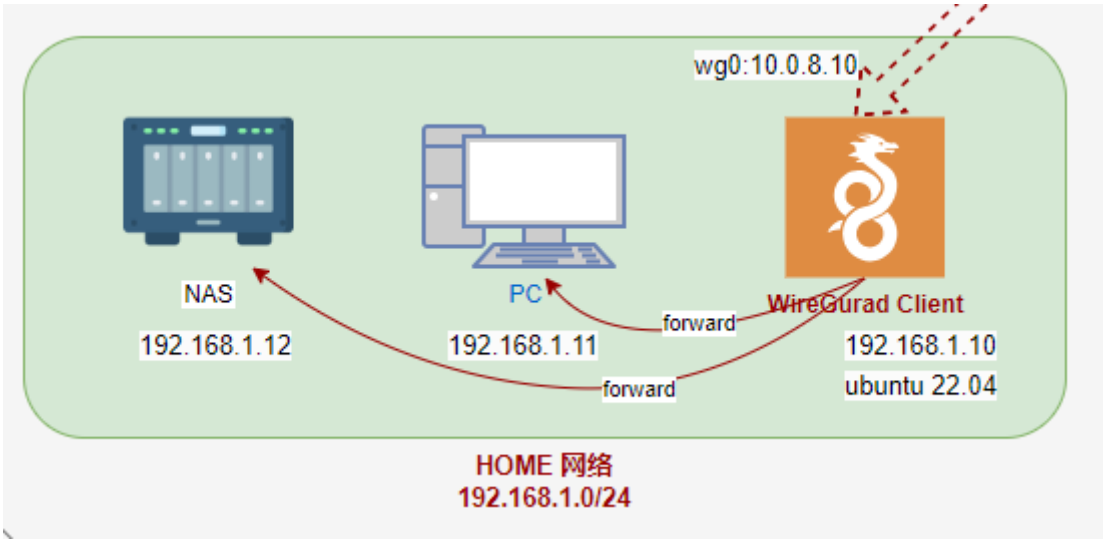
```
root@wireguard-server:~# wg show
interface: wg0
  public key: 1zKXJiO5yLakaRqbujM1e5VW5OnHqlq6vvGDBGVTS3k=
  private key: (hidden)
  listening port: 51820

peer: SEkxxA17Rb/eDFyO7SsVeUcD/OosYlVWcgL9IHnUAEU=
  allowed ips: 10.0.8.10/32, 192.168.1.0/24

peer: Uurn5z46FnntquzoGoy/5OZABcak3XaK/NhAodu8CAU=
  allowed ips: 10.0.8.11/32, 172.16.1.0/24
```

中继服务器机器的 WireGuard 接口与两个远端对等体 `peer` 建立了加密隧道。

配置wireguard 客户端

客户端配置（client1）

client01 位于HOME网络环境，操作系统为ubuntu 22.04，为linux服务器。



在`HOME网络`的`wireguard client01`上安装 wireguard

```
apt update -y
apt install -y wireguard
```

开启IP转发，实现不同客户端跨网段互联互通和正确路由。

```
cat <<EOF | sudo tee /etc/sysctl.d/wireguard.conf
net.ipv4.ip_forward = 1
EOF
sudo sysctl --system
```

linux客户端1配置

```
cat >/etc/wireguard/wg0.conf<<EOF
[Interface]
PrivateKey = QPxOgSjOj5X+vibvoenLOaT5SKdHAPVMsWMh/bDIqls=
Address = 10.0.8.10/32
PostUp   = iptables -A FORWARD -i %i -j ACCEPT; iptables -A FORWARD -o %i -j
ACCEPT; iptables -t nat -A POSTROUTING -o ens33 -j MASQUERADE
PostDown = iptables -D FORWARD -i %i -j ACCEPT; iptables -D FORWARD -o %i -j
ACCEPT; iptables -t nat -D POSTROUTING -o ens33 -j MASQUERADE

[Peer]
PublicKey = 1zKXJiO5yLakaRqbujM1e5VW5OnHqlq6vvGDBGVTS3k=
AllowedIPs = 10.0.8.0/24,10.0.1.0/24,172.16.1.0/24
Endpoint = 120.76.118.127:51820
PersistentKeepalive = 25
EOF
```

参数说明：

1. `[Interface]`表示这一部分是客户端接口的配置。
2. `PrivateKey`是客户端的私钥，用于识别和加密通信。此处为client1的私钥`/etc/wireguard/client1.key`。
3. `Address`指定了客户端在 WireGuard 虚拟网络中的 IP 地址，这里是`10.0.8.10`。这个 IP 地址应与服务器端的`AllowedIPs`设置相匹配。

4. `PostUp` 和 `PostDown` 是在 WireGuard 接口启动和停止时执行的 iptables 规则。这些规则允许转发 WireGuard 接口的流量，并在服务器的物理网卡 `ens33` 上进行 NAT 伪装，以允许客户端访问互联网。注意修改 `PostUP` 及 `PostDown` 参数的接口名称，与实际匹配。
5. `[Peer]` 表示这一部分是服务器端的配置。
6. `PublicKey` 是服务器server端的公钥，用于识别和加密与服务器端的通信，此处为 `/etc/wireguard/server.key.pub`。
7. `AllowedIPs` 指定了允许该对等节点（peer）发送过来的 VPN 流量中的源地址范围。这里是 wireguard 隧道网络的地址 `10.0.8.0/24`，以及阿里云网络和COMPANY网络的本地地址段 `10.0.1.0/24` 和 `172.16.1.0/24`。
8. `Endpoint` 指定了服务器端的公网 IP 地址和监听端口，这里是 `120.76.118.127:51820`。客户端将通过这个端点连接到服务器端。将公网服务器作为唯一的 Peer，通过设置 PersistentKeepalive 来进行连接的保活。

在客户端端启动wireguard服务

```
systemctl enable --now wg-quick@wg0
```

查看wireguard端口当前状态

```
root@wireguard-server:~# wg
interface: wg0
  public key: 1zKXJi05yLakaRqbujM1e5VW5OnHqlq6vvGDBGVTS3k=
  private key: (hidden)
  listening port: 51820

peer: SEkxxA17Rb/eDFyO7SsVeUcD/OosYlVWcgL9IHnUAEU=
  endpoint: 219.134.225.139:20748
  allowed ips: 10.0.8.10/32, 192.168.1.0/24
  latest handshake: 47 seconds ago
  transfer: 55.66 KiB received, 54.29 KiB sent

peer: Uurn5z46FnntquzoGoy/5OZABcak3XaK/NhAodu8CAU=
  allowed ips: 10.0.8.11/32, 172.16.1.0/24
```

测试 `client01` 与中继服务器wireguard-server端口连通性

```
root@client01:~# ping 10.0.8.1 -c 4
PING 10.0.8.1 (10.0.8.1) 56(84) bytes of data.
64 bytes from 10.0.8.1: icmp_seq=1 ttl=64 time=5.77 ms
64 bytes from 10.0.8.1: icmp_seq=2 ttl=64 time=4.30 ms
64 bytes from 10.0.8.1: icmp_seq=3 ttl=64 time=4.56 ms
64 bytes from 10.0.8.1: icmp_seq=4 ttl=64 time=4.83 ms

--- 10.0.8.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 4.296/4.862/5.768/0.555 ms
```

测试 `client01` 与中继服务器本地子网(`阿里云VPC`)其他设备的连通性，例如 `ECS vm02`

```
root@client01:~# ping 10.0.1.11 -c 4
PING 10.0.1.11 (10.0.1.11) 56(84) bytes of data.
64 bytes from 10.0.1.11: icmp_seq=1 ttl=63 time=4.86 ms
64 bytes from 10.0.1.11: icmp_seq=2 ttl=63 time=4.95 ms
64 bytes from 10.0.1.11: icmp_seq=3 ttl=63 time=5.06 ms
64 bytes from 10.0.1.11: icmp_seq=4 ttl=63 time=4.97 ms

--- 10.0.1.11 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 4.863/4.960/5.062/0.070 ms
```

在中继服务器`wireguard-server`上测试访问`HOME网络`中的`wireguard client`

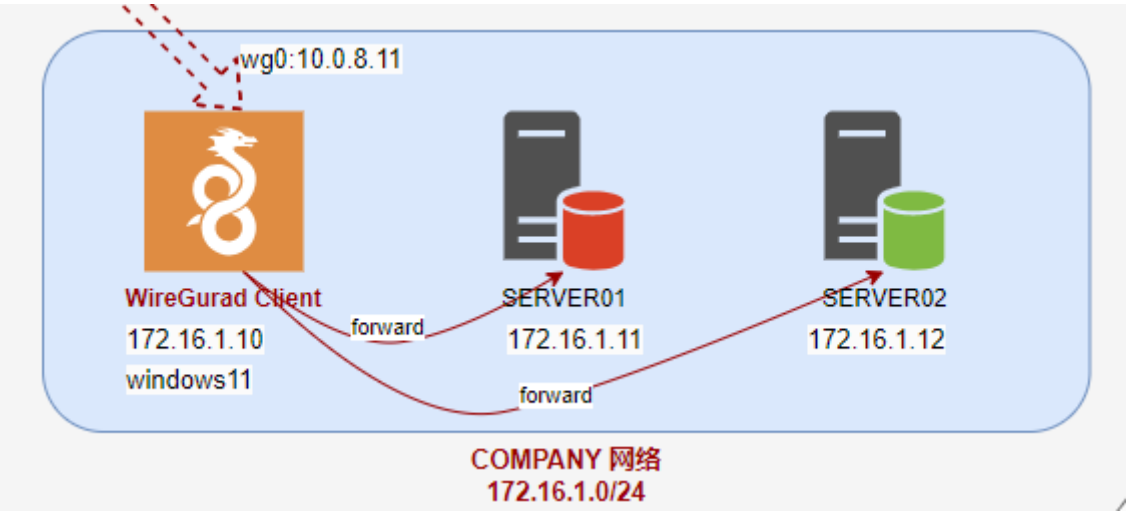
```
root@wireguard-server:~# ping 10.0.8.10 -c 4
PING 10.0.8.10 (10.0.8.10) 56(84) bytes of data.
64 bytes from 10.0.8.10: icmp_seq=1 ttl=64 time=24.7 ms
64 bytes from 10.0.8.10: icmp_seq=2 ttl=64 time=27.7 ms
64 bytes from 10.0.8.10: icmp_seq=3 ttl=64 time=26.2 ms
64 bytes from 10.0.8.10: icmp_seq=4 ttl=64 time=24.6 ms

--- 10.0.8.10 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 24.617/25.791/27.665/1.257 ms
```

在中继服务器上测试访问`HOME网络`中的其他设备，例如NAS设备

```
root@wireguard-server:~# ping 192.168.1.12 -c 4
PING 192.168.1.12 (192.168.1.12) 56(84) bytes of data.
64 bytes from 192.168.1.12: icmp_seq=1 ttl=63 time=83.7 ms
64 bytes from 192.168.1.12: icmp_seq=2 ttl=63 time=193 ms
64 bytes from 192.168.1.12: icmp_seq=3 ttl=63 time=111 ms
64 bytes from 192.168.1.12: icmp_seq=4 ttl=63 time=136 ms

--- 192.168.1.12 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3003ms
rtt min/avg/max/mdev = 83.678/130.733/192.572/40.158 ms
```

客户端配置（client2）

client02 位于COMPANY 网络环境，操作系统为 windows11，为windows服务器。



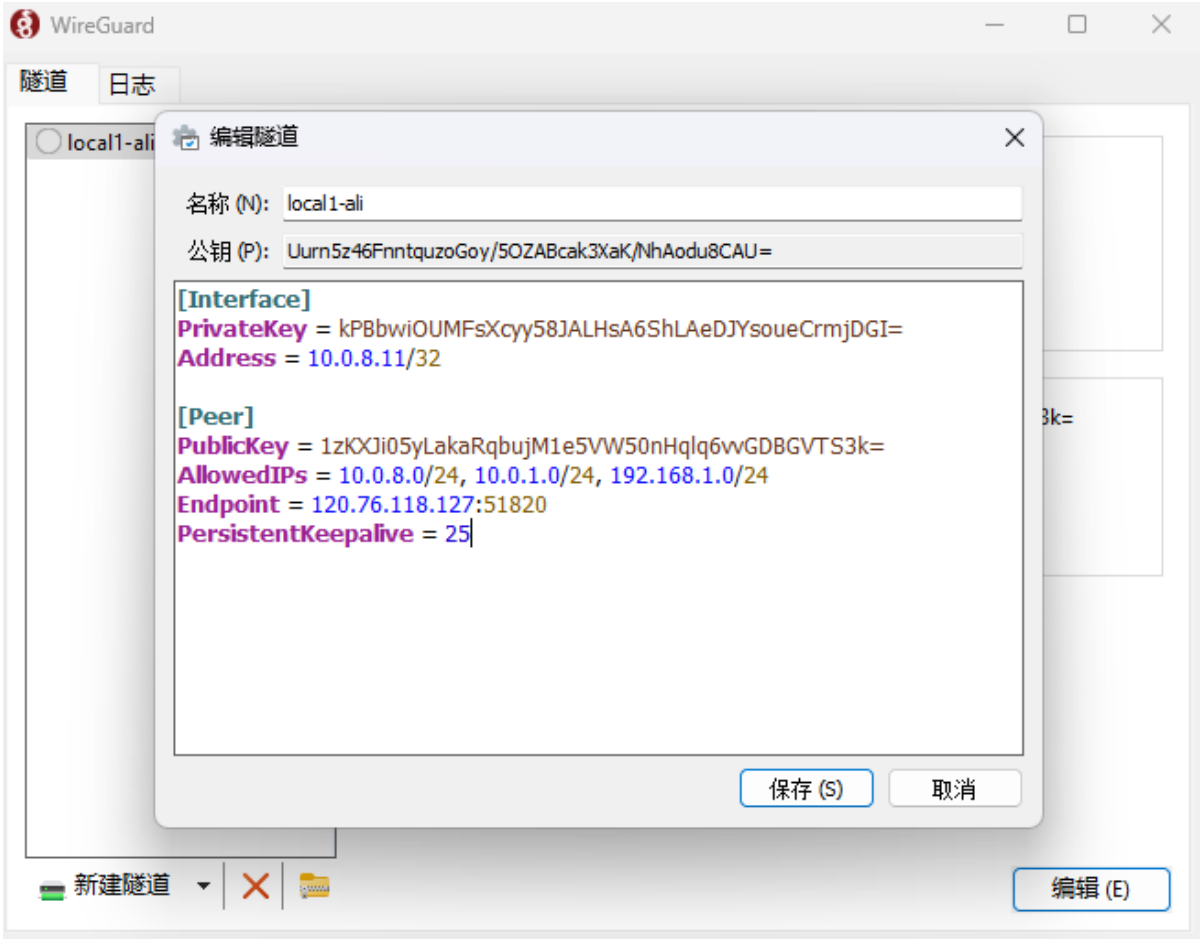
在 `COMPANY 网络` 的 client02 安装wireguard。客户端02为windows11机器，需要下载wireguard windows客户端。

windows客户端官方下载地址：https://www.wireguard.com/install/

下载安装后新建空隧道，粘贴以下配置，详见下面参数说明：

```
[Interface]
PrivateKey = kPBbwiOUMFsXcyy58JALHsA6ShLAeDJYsoueCrmjDGI=
Address = 10.0.8.11/32

[Peer]
PublicKey = 1zKXJi05yLakaRqbujM1e5VW50nHqlq6vvGDBGVTS3k=
AllowedIPs = 10.0.8.0/24, 10.0.1.0/24, 192.168.1.0/24
Endpoint = 120.76.118.127:51820
PersistentKeepalive = 25
```

配置示例



测试 `client02` 与中继服务器`wireguard-server`端口连通性

```
C:\Users\client02>ping 10.0.8.1

正在 Ping 10.0.8.1 具有 32 字节的数据:
来自 10.0.8.1 的回复: 字节=32 时间=26ms TTL=64
来自 10.0.8.1 的回复: 字节=32 时间=26ms TTL=64
来自 10.0.8.1 的回复: 字节=32 时间=28ms TTL=64
来自 10.0.8.1 的回复: 字节=32 时间=32ms TTL=64

10.0.8.1 的 Ping 统计信息:
    数据包: 已发送 = 4，已接收 = 4，丢失 = 0 (0% 丢失)，
往返行程的估计时间(以毫秒为单位):
    最短 = 26ms，最长 = 32ms，平均 = 28ms
```

测试 `client02` 与中继服务器本地子网(`阿里云VPC`)其他设备的连通性，例如`ECS vm02`

```
C:\Users\client02>ping 10.0.1.11

正在 Ping 10.0.1.11 具有 32 字节的数据:
来自 10.0.1.11 的回复: 字节=32 时间=61ms TTL=63
来自 10.0.1.11 的回复: 字节=32 时间=26ms TTL=63
来自 10.0.1.11 的回复: 字节=32 时间=29ms TTL=63
来自 10.0.1.11 的回复: 字节=32 时间=27ms TTL=63

10.0.1.11 的 Ping 统计信息:
    数据包: 已发送 = 4，已接收 = 4，丢失 = 0 (0% 丢失)，
往返行程的估计时间(以毫秒为单位):
    最短 = 26ms，最长 = 61ms，平均 = 35ms
```

配置本地转发

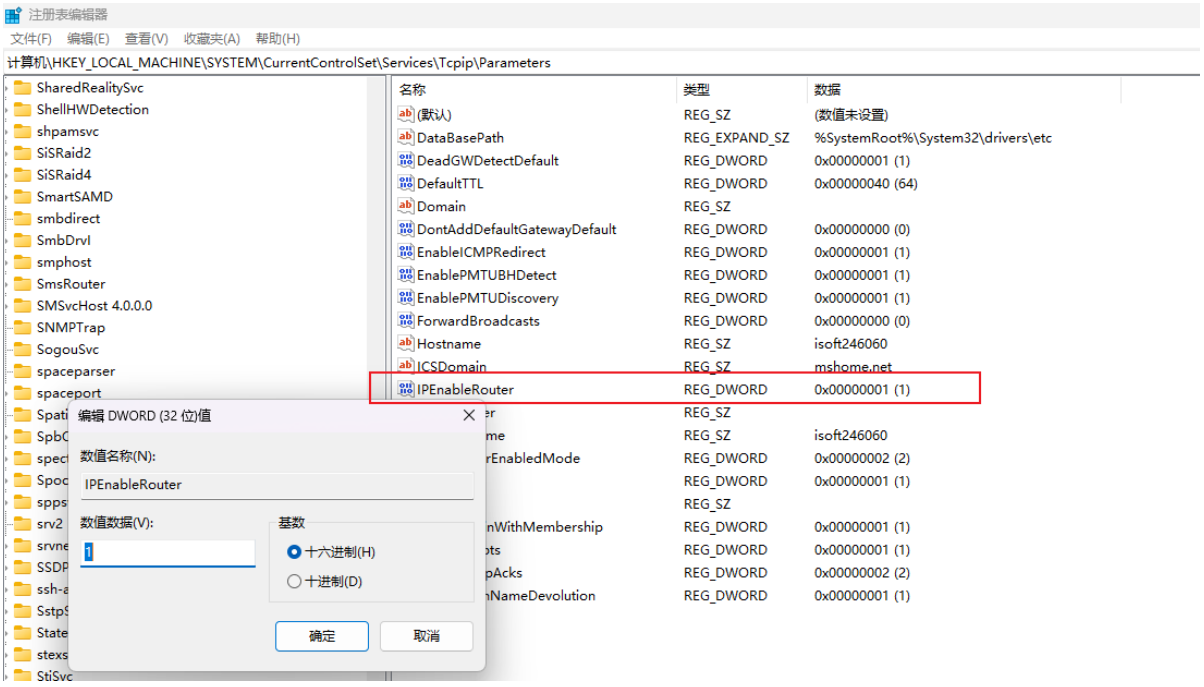
1.开启路由转发

配置转发，允许其他对等端通过 `wiregurad client` 转发实现访问本地子网其他设备。

1、首先开启window的路由转发功能，打开注册表编辑器并转到以下路径：

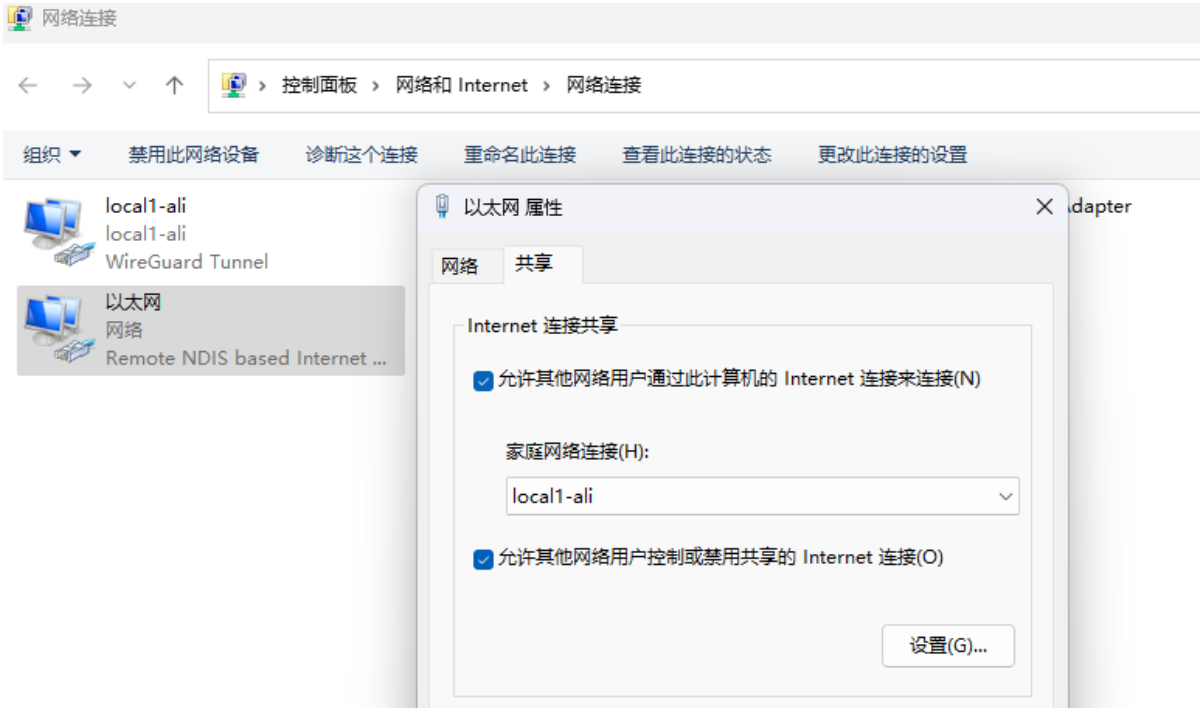
```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters
```

将`IPEnableRouter`的值修改为1，然后重启电脑。



2. 设置网络共享

选择你当前上网的网络适配器，右键，属性，选择“共享”，然后勾选“允许其他用户网络通过此计算机的Internet连接”，并从已有的网络连接中选择当前wireguard节点的名字。



注意：每次重启服务端时，都需要重新设置一次Internet连接共享！

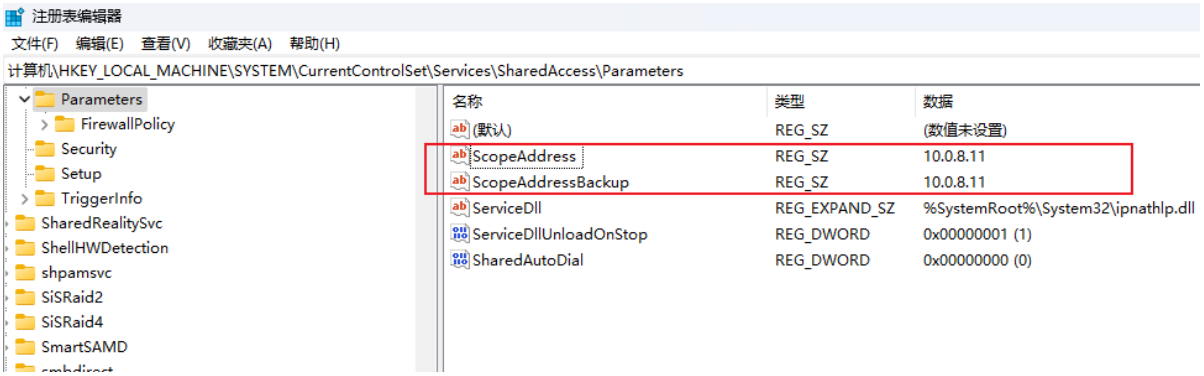
3. 更改默认 Internet 连接共享 IP

默认情况下，启用Internet共享(NAT)时，Windows会将适配器的IP地址更改为其他地址（以避免冲突），默认为`192.168.137.1`。但是本次使用共享网卡的wg0为固定IP`10.0.8.11`。因此需要修改注册表，修改IP地址。

打开注册表编辑器并转到以下路径：

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\SharedAccess\Parameters
```

然后只需将`ScopeAddress`和`StandaloneDHCPAddress`更改为wireguard指定的IP地址`10.0.8.11``。



测试 `client 01` 与 `HOME网络` 其他设备的连通性

```
C:\Users\client02>ping 192.168.1.12

正在 Ping 192.168.1.12 具有 32 字节的数据:
来自 192.168.1.12 的回复: 字节=32 时间=33ms TTL=62
来自 192.168.1.12 的回复: 字节=32 时间=44ms TTL=62
来自 192.168.1.12 的回复: 字节=32 时间=43ms TTL=62
来自 192.168.1.12 的回复: 字节=32 时间=32ms TTL=62

192.168.1.12 的 Ping 统计信息:
    数据包: 已发送 = 4，已接收 = 4，丢失 = 0 (0% 丢失)，
往返行程的估计时间(以毫秒为单位):
    最短 = 32ms，最长 = 44ms，平均 = 38ms
```

测试带宽

在 `wireguard-server` 服务端(阿里云 Ubuntu 主机)和 `client02` 客户端分别运行iperf3进行带宽测试。

服务端命令:

```
apt install -y iperf3
iperf3 -s -B 10.0.8.1
```

解释:

- `-s` 表示运行在服务器模式
- `-B 10.0.8.1` 将服务器端绑定到WireGuard虚拟IP 10.0.8.1

执行该命令后，iperf3 将在服务端监听客户端的连接请求。

客户端命令:

```
> ./iperf3.exe -c 10.0.8.1 -u -b 3M
```

解释:

- `-c 10.0.8.1` 指定连接服务器端的IP 10.0.8.1
- `-u` 使用UDP模式进行测试(也可使用TCP模式)
- `-b 3M` 设置目标带宽为3Mbps(可根据实际公网带宽调整)

先在服务端运行服务器命令，然后在客户端运行客户端命令，iperf3 将开始进行带宽测试。

测试过程中，服务端和客户端都会实时输出带宽、数据传输情况等统计数据。测试结束后，可以根据输出的最终结果判断实际带宽数值。

```
PS C:\Users\client02\Downloads\iperf-3.1.3-win64\iperf-3.1.3-win64> ./iperf3.exe
-c 10.0.8.1 -u -b 3M
Connecting to host 10.0.8.1, port 5201
[  4] local 10.0.8.11 port 53246 connected to 10.0.8.1 port 5201
[ ID] Interval           Transfer     Bandwidth       Total Datagrams
[  4]   0.00-1.00   sec   344 KBytes  2.81 Mbits/sec  43
[  4]   1.00-2.01   sec   368 KBytes  3.01 Mbits/sec  46
[  4]   2.01-3.00   sec   368 KBytes  3.02 Mbits/sec  46
[  4]   3.00-4.00   sec   368 KBytes  3.01 Mbits/sec  46
[  4]   4.00-5.00   sec   360 KBytes  2.95 Mbits/sec  45
[  4]   5.00-6.00   sec   368 KBytes  3.01 Mbits/sec  46
[  4]   6.00-7.00   sec   368 KBytes  3.02 Mbits/sec  46
[  4]   7.00-8.00   sec   368 KBytes  3.01 Mbits/sec  46
[  4]   8.00-9.01   sec   360 KBytes  2.95 Mbits/sec  45
[  4]   9.01-10.00  sec   368 KBytes  3.02 Mbits/sec  46
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Jitter    Lost/Total
Datagrams
[  4]   0.00-10.00  sec  3.55 MBytes  2.98 Mbits/sec  2.818 ms  0/455 (0%)
[  4] Sent 455 datagrams

iperf Done.
PS C:\Users\client02\Downloads\iperf-3.1.3-win64\iperf-3.1.3-win64>
```

测试实际带宽`Bandwidth`为`2.98 Mbits/sec`，与阿里云轻量服务器公网带宽基本一致。