wireguard??

wg-easy ??docker??

`docker-compose.yml`

```
services:
  wg-easy:
    container_name: wg-easy
    image: ghcr.io/wg-easy/wg-easy
    network_mode: bridge
    environment:
      - LANG=chs # or en, de ...
      - WG_HOST=43.136.104.74 #替换自己的服务器ip
      - PASSWORD=123456 # WebUI 登录密码（明文即可，容器会自动 bcrypt）
      #- PASSWORD_HASH=$$2a$$12$$S5L7nVgk6I7O/tdVLyVH0.QmD89RHJdKgHL8ayXOiGYwwFa4UmONC
#PASSWORD和这一个密码用其中一个
      - LANG=chs# 可选：中文界面
      - WG_DEFAULT_DNS=192.168.2.1,114.114.114.114,8.8.8.8,8.8.4.4
      - PORT=51820 #可更改
      - WG_DEFAULT_ADDRESS=10.9.0.x
      - WG_PORT=51821 # 可选：自定义 WireGuard 端口
      - WG_PRE_UP = iptables -t nat -F; iptables -F;  # Flush all rules
      #- WG_POST_UP=iptables -A FORWARD -i %i -j ACCEPT; iptables -A FORWARD -o %i -j ACCEPT;
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
      #- WG_POST_DOWN=iptables -D FORWARD -i %i -j ACCEPT; iptables -D FORWARD -o %i -j
ACCEPT; iptables -t nat -D POSTROUTING -o eth0 -j MASQUERADE
      - WG_ALLOWED_IPS=114.114.114.114,8.8.8.8,8.8.4.4,192.168.2.0/24,10.9.0.0/24


      - UI_TRAFFIC_STATS=true
      - UI_CHART_TYPE=3 # (0 Charts disabled, 1 # Line chart, 2 # Area chart, 3 # Bar chart)
      # - WG_ENABLE_ONE_TIME_LINKS=true
      - UI_ENABLE_SORT_CLIENTS=true
      # - WG_ENABLE_EXPIRES_TIME=true
    volumes:
```

```
      - /home/ubuntu/work/wireguard:/etc/wireguard
    ports:
      - 51820:51820/tcp
      - 51821:51821/udp
    cap_add:
      - NET_ADMIN
      - SYS_MODULE
    sysctls:
      - net.ipv4.conf.all.src_valid_mark=1
      - net.ipv4.ip_forward=1
    restart: unless-stopped
```

???????

PASSWORD_HASH

```
PASSWORD_HASH=$$2a$$12$$31H.ZEl74tEF98shuIWWxe2PTsljr3vEMRfU7HL8dPvNJTImcUgRq
```

这是密码为 123456的哈希值，参见 How_to_generate_an_bcrypt_hash.md 来了解如何获取这个哈希值，当你登录到 `wg-easy` 配置 web界面时， `123456` 将是你的密码。但现在的方法有一点问题，所以可以用以下命令代替：

```
sudo docker run -it ghcr.io/wg-easy/wg-easy /app/wgpw.sh 123456
```

如果上述命令的输出是：

```
PASSWORD_HASH='$2a$12$31H.ZEl74tEF98shuIWWxe2PTsljr3vEMRfU7HL8dPvNJTImcUgRq'
```

因为我们使用的是 `docker-compose.yaml` 文件，需要将每个 `$` 符号替换为两个 `$$` 符号。所以在 yaml中的 `PASSWORD_HASH` 应为：

```
PASSWORD_HASH=$$2a$$12$$31H.ZEl74tEF98shuIWWxe2PTsljr3vEMRfU7HL8dPvNJTImcUgRq
```

记得用一个更强的密码替换简单的 `123456` 。

关于我们使用的 IP和 IP范围的注释：

```
192.168.2.1 主机网络网关，请改为你的主机网络网关。
192.168.2.0/24 主机网络IP范围，请同样改为你的主机网络IP范围。
10.9.0.x 你的虚拟网络IP范围，除非你偏好其他的IP范围，否则不需要更改。
10.9.0.0/24 由 10.9.0.x 设定的虚拟网络IP范围。
```

????

WireGuard????

维基百科是这样描述的：

- WireGuard是一种实现加密虚拟专用网络（VPN）的通信协议和免费开源软件，其设计目标是易于使用、高速性能和低攻击面；
- 它旨在比IPsec和OpenVPN这两种常见的隧道协议具有更好的性能和更强大的功能；WireGuard协议通过UDP传递流量。

通过描述，我们知道了wireguard其实就是一个FAST（速度快）、MODERN（流行）、SECURE（安全）的VPN TUNNEL（VPN隧道）。

??????

1. 需要有公网的ip地址（这些地址可以直接在腾讯云、阿里云等厂商购买）；
2. 如果Linux内核版本<5.6，可能需要首先更新内核（本文下面有内核升级教程）；
3. `WireGuard` 的安装和使用条件非常苛刻，对内核版本要求极高，不仅如此，在不同的系统中，内核，内核源码包，内核头文件必须存在且这三者版本要一致。所以一般不建议在生成环境中安装，除非你对自己的操作很有把握。`Red Hat`、`CentOS`、`Fedora` 等系统的内核，内核源码包，内核头文件包名分别为 `kernel`、`kernel-devel`、`kernel-headers`，`Debian`、`Ubuntu` 等系统的内核，内核源码包，内核头文件包名分别为 `kernel`、`linux-headers`。

如果这三者任一条件不满足的话，则无法测试和安装 `Wiregurad`！

目前 `WireGuard` 已经被合并到 `Linux 5.6` 内核中了，如果你的内核版本 >= `5.6`，就可以用上原生的 `WireGuard` 了，只需要安装 `wireguard-tools` 即可。例如，对于 `Ubuntu 20.04` 来说，它的内核版本是 `5.4`，虽然小于 `5.6`，但经过我的测试发现它已经将 `WireGuard` 合并到了内核中，我们只需要安装 `wireguard-tools` 即可。

??WireGuard

教程中使用的系统为 **`Ubuntu 24.04`**。

第0步：切换到root用户

```
sudo su
```

第一步：开启内核转发：

```
echo 1 > /proc/sys/net/ipv4/ip_forward
echo "net.ipv4.ip_forward = 1" >> /etc/sysctl.conf
sysctl -p
```

第二步：安装 WireGuard

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

第三步：验证是否安装成功

```
wg --version
```

正常来说应该可以看到类似下面的输出：

```
wireguard-tools v1.0.20210914 - https://git.zx2c4.com/wireguard-tools/
```

另外一种验证方式：

```
modprobe wireguard && lsmod | grep wireguard
```

输出如下：

```
wireguard              86016  0
curve25519_x86_64      36864  1 wireguard
libchacha20poly1305    16384  1 wireguard
libblake2s             16384  1 wireguard
ip6_udp_tunnel         16384  1 wireguard
udp_tunnel             24576  1 wireguard
libcurve25519_generic    49152  2 curve25519_x86_64,wireguard
```

??WireGuard

```
cd /etc/wireguard
```

?????

1、生成服务器私钥并将其保存在 `/etc/wireguard/server_private.key`：

```
wg genkey | sudo tee /etc/wireguard/server_private.key
```

输出如下：

```
UOO//MO2GCC+5hHOz91YCP60/Zv/cnSskEH2j4eRPXo=
```

2、生成服务器公钥并将其保存在 `/etc/wireguard/server_public.key`：

```
cat /etc/wireguard/server_private.key | wg pubkey | sudo tee /etc/wireguard/server_public.key
```

输出如下：

```
W+l7Uapd98bsNhN1g3Hs4iTCfKzcV03KNwhDPFgzqR4=
```

3、查看可用的服务器网络接口并验证主接口的详细信息

```
ip a
```

输出如下：

```
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 noprefixroute
       valid_lft forever preferred_lft forever
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default
qlen 1000
    link/ether 56:00:04:f8:7f:67 brd ff:ff:ff:ff:ff:ff
    inet 192.0.2.161/24 metric 100 brd 192.0.2.255 scope global dynamic enp1s0
       valid_lft 56853sec preferred_lft 56853sec
    inet6 2a05:0000:0000:000:5400:4ff:0000:7f67/64 scope global dynamic mngtmpaddr
noprefixroute
       valid_lft 2591775sec preferred_lft 604575sec
    inet6 2a05:0000:0000:000:5400:4ff:0000:7f67/64 scope link
       valid_lft forever preferred_lft forever
```

根据描述信息可知道 `enp1s0` 是主公网接口，IP地址为 `192.0.2.161`。`WireGuard` 通过主公网接口将网络请求转换并转发到 Internet。

4、生成服务端配置文件

```
echo "[Interface]
PrivateKey = $(cat server_private.key)
Address = 10.8.0.1/24
#如果你的服务器主网卡名称不是 enp1s0 ，那么请修改下面防火墙规则中最后的 enp1s0 为你的主网卡名称。
PostUp = ufw route allow in on wg0 out on enp1s0
PostUp = iptables -t nat -I POSTROUTING -o enp1s0 -j MASQUERADE
PreDown = ufw route delete allow in on wg0 out on enp1s0
PreDown = iptables -t nat -D POSTROUTING -o enp1s0 -j MASQUERADE
ListenPort = 51820
DNS = 8.8.8.8
MTU = 1420
"|sed '/^#/d;/^\s*$/d' > wg0.conf
```

`ListenPort`为端口号，可以设置成自己想使用的端口。

?????(??????????????)

1、生成客户端私钥:

```
wg genkey | sudo tee /etc/wireguard/client1_private.key
```

输出如下:

```
KBUxCUqNEJqN3DBO5xu2kiBQFT8Gv46Kkqu6OIKZu3Q=
```

2、生成客户端公钥:

```
cat /etc/wireguard/client1_private.key | wg pubkey | sudo tee
/etc/wireguard/client1_public.key
```

输出如下:

```
xZB9I6953ebGqWVLCR7L6yJw7YJi0shJ+Sub9gfUFVU=
```

3、生成客户端配置文件:

```
[Interface]
PrivateKey = +B1l4bteT0URxs47VL7mSUJ6Gjp2yrXsxuzGWQBPSUo=
Address = 10.8.0.3/24
```

```
DNS = 114.114.114.114, 8.8.8.8, 8.8.4.4

[Peer]
PublicKey = rxdt0em+q2ST/ZJwrwiozT5TPCzIyfZbFj/1TPsN02c=
PresharedKey = INzSFYALzwr7o4yiOSaAB4xkHSD9MZiv7HCwt204dAQ=
AllowedIPs = 10.9.0.0/24
Endpoint = 43.136.104.74:51821
```

通过上述配置，客户端设备可以与 `WireGuard` 服务器建立新的 VPN隧道，并通过隧道 IP地址 `10.8.0.2` 访问 internet。

4、打开 `WireGuard` 服务器配置，添加新的客户端

将 `xZB9I6953ebGqWVLCR7L6yJw7YJi0shJ+Sub9gfUFVU=` 替换为客户端公钥：

```
[Peer]
PublicKey = xZB9I6953ebGqWVLCR7L6yJw7YJi0shJ+Sub9gfUFVU=
AllowedIPs = 10.8.0.2/32
```

??`WireGuard`????????????

1、启动 `WireGuard`：

```
systemctl start wg-quick@wg0.service
```

或者 #启动 wg-quick up wg0 **注意： **文件名 wg0.conf意味着 wg0将是在客户端启动 WireGuard 连接的名字。可配置多个服务，根据 /etc/wireguard/目录下配置文件名称来， wg0.confwei wg0服务， wg1.confwei wg1服务。 2、配置 `WireGuard` 开机启动：

```
systemctl enable wg-quick@wg0.service
```

3、查看 `WireGuard` 状态：

```
systemctl status wg-quick@wg0.service
```

正确输出如下：

```
● wg-quick@wg0.service - WireGuard via wg-quick(8) for wg0
     Loaded: loaded (/usr/lib/systemd/system/wg-quick@.service; disabled; preset: enabled)
     Active: active (exited) since Wed 2024-06-12 04:47:53 UTC; 38s 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: 2384 ExecStart=/usr/bin/wg-quick up wg0 (code=exited, status=0/SUCCESS)
   Main PID: 2384 (code=exited, status=0/SUCCESS)
        CPU: 125ms
```

4、查看 `WireGuard` 虚拟网卡当前状态:

```
wg
```

或者

```
wg show wg0
```

输出如下:

```
interface: wg0
  public key: W+l7Uapd98bsNhN1g3Hs4iTCfKzcV03KNwhDPFgzqR4=
  private key: (hidden)
  listening port: 51820

peer: xZB9I6953ebGqWVLCR7L6yJw7YJi0shJ+Sub9gfUFVU=
  allowed ips: 10.8.0.2/32
```

5、停止 `WireGuard` 服务

```
wg-quick down wg0
```

????

多台机器互 `ping` 即可:

```
ping -c 4 10.8.0.1
```

能 ping通，说明 `WireGuard` 安装成功。

输出如下:

```
PING 10.8.0.1 (10.8.0.1) 56(84) bytes of data.
64 bytes from 10.8.0.1: icmp_seq=1 ttl=64 time=0.056 ms
64 bytes from 10.8.0.1: icmp_seq=2 ttl=64 time=0.076 ms
64 bytes from 10.8.0.1: icmp_seq=3 ttl=64 time=0.064 ms
64 bytes from 10.8.0.1: icmp_seq=4 ttl=64 time=0.065 ms

--- 10.8.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3049ms
rtt min/avg/max/mdev = 0.056/0.065/0.076/0.007 ms
```

?????

默认情况下，`Uncomplicated Firewall`（`UFW`）在`Ubuntu 24.04`服务器上是激活的。这时需要配置防火墙允许在配置中指定的`51820`端口上进行网络连接。此外，还需要在服务器上开启端口转发功能，以便通过`NAT`（`Network Address Translation`）将来自`WireGuard`接口地址的网络请求转发到internet。

查看防火墙状态:

```
ufw status
```

当`UFW`状态为`inactive`时，执行如下命令允许SSH端口，并开启防火墙。

```
ufw allow 22 && sudo ufw enable
```

允许`WireGuard`接口`UDP`端口`51820`通过防火墙：

```
ufw allow 51820/udp
```

重新加载`UFW`

```
ufw reload
```

再次查看`UFW`状态:

```
ufw status
```

输出如下:

```
Status: active
```

```
To                         Action      From

22/tcp                     ALLOW       Anywhere
51820/udp                  ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
51820/udp (v6)             ALLOW       Anywhere (v6)

Anywhere on enp1s0         ALLOW FWD   Anywhere on wg0
Anywhere (v6) on enp1s0    ALLOW FWD   Anywhere (v6) on wg0
```

配置 `iptables` 规则:

```
iptables -t nat -A POSTROUTING -s 10.8.0.0/24 -o enp1s0 -j MASQUERADE
```

注意将 `enp1s0` 改成自己的网卡。

保存规则:

```
iptables-save | sudo tee /etc/iptables/rules.v4
```

Revision #1
Created 2025-12-15 08:33:12 CST by eason
Updated 2025-12-15 08:33:35 CST by eason