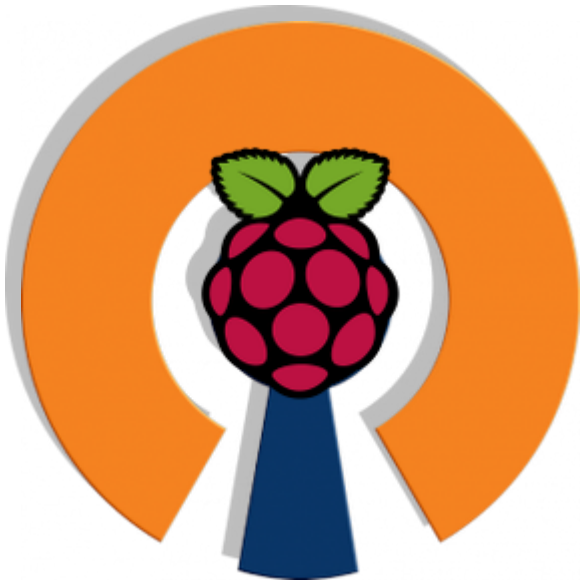


Par El Bai Ilyasse, INSTA**.**

Jeudi 22 avril 2021

Sommaire :

Projet :

Annexes :

Nous allons voir comment installer le VPN WireGuard sur notre système d'exploitation, ici un Linux Ubuntu 20.04.

1-Définition :

Un VPN est un système permettant de créer un lien direct entre des ordinateurs distants, qui isole leurs échanges du reste du trafic se déroulant sur des réseaux de télécommunication publics.

Linux Ubuntu est un système d'exploitation utilisé par des millions de machines avec une interface simple, intuitive, et sécurisée. Elle est la distribution Linux la plus consultée, et le système d'exploitation le plus utilisé sur les systèmes Cloud ainsi que sur les serveurs informatiques.
Linux Ubuntu est développé, commercialisé et maintenu pour les ordinateurs (version Desktop), serveurs (version Server) et les objets connectés (version Core).

WireGuard est une application gratuite et open source avec un protocole de communication qui implémente des techniques de réseau privé virtuel (VPN) pour créer des connexions point à point sécurisées dans des configurations routées ou pontées. Il est exécuté en tant que module à l'intérieur du noyau Linux, et vise de meilleures performances et des économies d'énergie. Développé et publié par Jason A. Donenfeld sous la licence publique générale GNU (GPL) version 2. La version Linux du logiciel a atteint une version de production stable et a été incorporée dans la version du noyau Linux fin mars 2020.

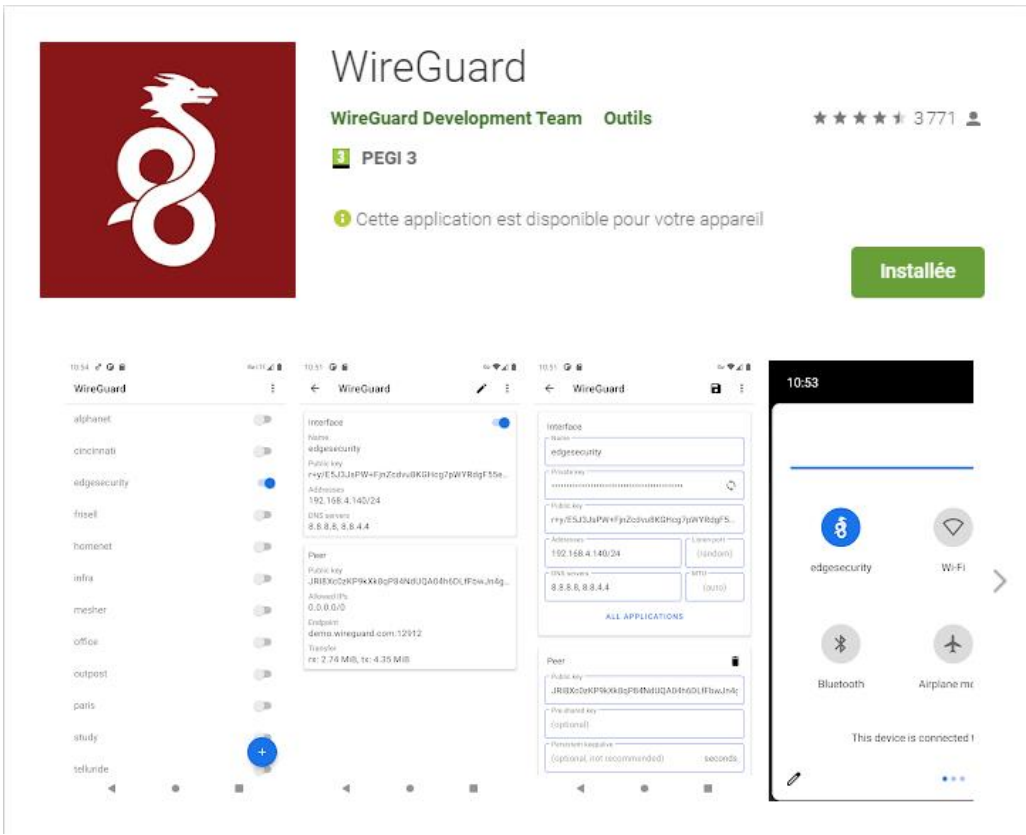
PIVPN est un outil développé et maintenu à jour par une communauté de passionnés, PiVPN comporte plusieurs intérêts qui le destinent aux novices comme aux experts : entièrement paramétrable, cet outil a été conçu pour être déployé sur Raspberry Pi, mais fonctionne sur n'importe quel serveur Debian VPS. Parmi ses fonctionnalités, on profite d'une clé de chiffrement à courbe elliptique jusqu'à 512 bits, d'une prise en charge de la keychain d'iOS, de serveurs DNS multiples et personnalisés, d'une intégration à Bitwarden ou encore du support de Pi-Hole. Enfin, il est possible d'utiliser des noms de domaines personnalisés si vous optez pour le protocole OpenVPN.

2-Prérequis :

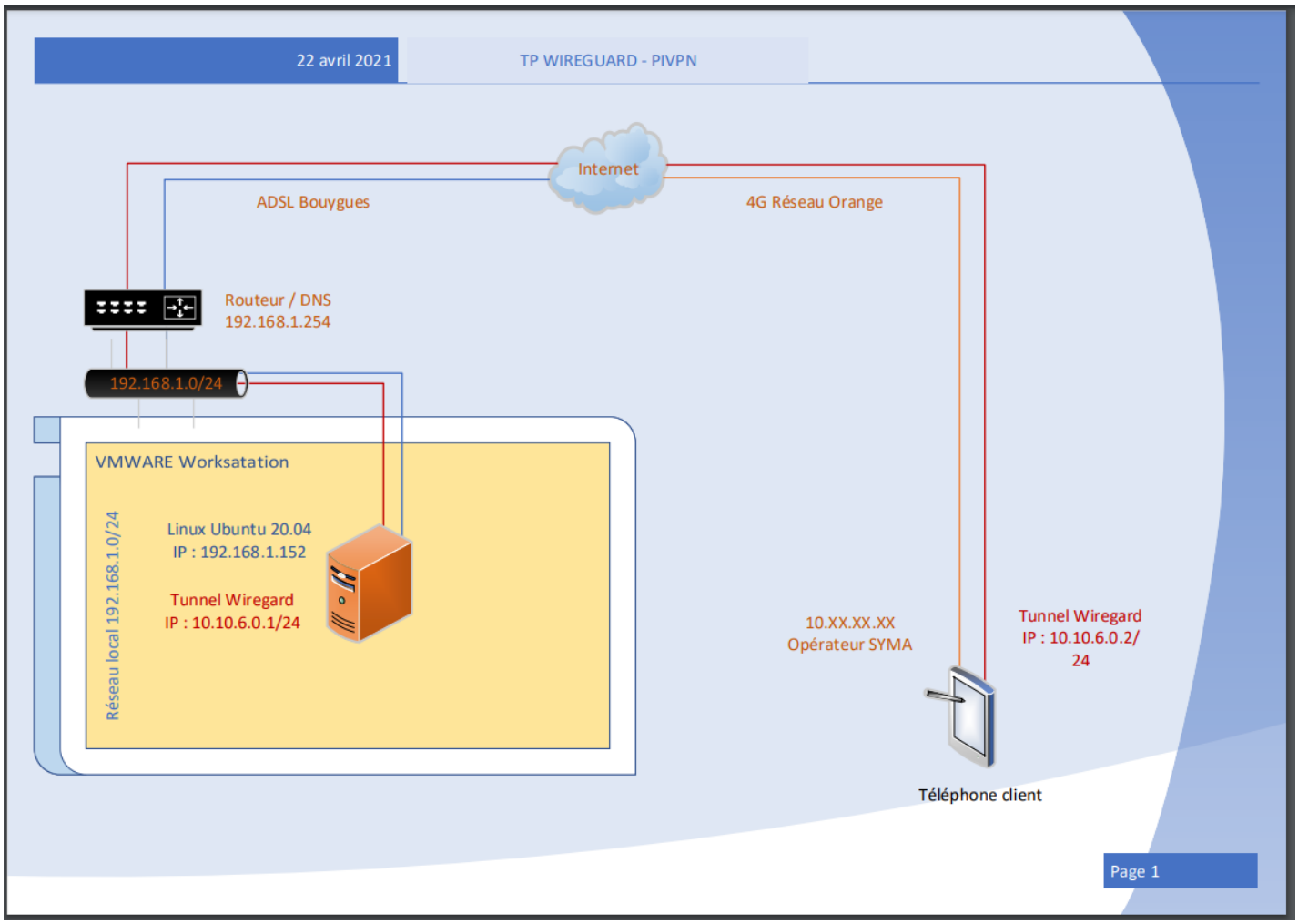
Nous allons installer une image de Linux Ubuntu en 20.04 en version 64 bits avec une sortie internet bridgé vers notre réseau local en 192.168.1.0/24 avec la passerelle 192.168.1.254. Ce système d'exploitation aura une adresse ip fixe en 192.168.1.152 car ce sera notre serveur VPN. Il faut créer une règle pouvant laisser le trafic circulé vers votre serveur VPN ici 192.168.1.152 sur le port que vous allez choisir (par défaut c'est le 51820 qui va être utilisé pour l'exemple) n'hésiter pas en pratique à choisir un port atypique afin d'éviter les attaques sur les ports par défauts.

Cela a été effectuer sur une box operateur Bouygues Télécom :
Nous autorisons bien le trafic depuis n'importe quel IP sur le port 51820 vers notre machine Linux Ubuntu sur le même port. Si vous connaissez les adresses IP de vos équipements distant vous pouvez filtrer les IP externe afin de ne laissez passer que vos équipements.

Le but de cet exercice sera de connecter un mobile Android a notre serveur VPN via l'application WireGuard disponible sur le Play Store de Google, disponible aussi sur iOS, Windows et Linux.
Pour avoir l'application Android : https://play.google.com/store/apps/details?id=com.wireguard.android



3-Schéma réseau du TP WIREGUARD – PIVPN :



4-Installation de PIVPN et Wireguard

Suite a l'instalation de votre système Linux et sa configuration :

```
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.152  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::557f:5e99:6cbd:db62  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:c6:be:21  txqueuelen 1000  (Ethernet)
        RX packets 133278  bytes 198404774 (198.4 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 104367  bytes 9235076 (9.2 MB)
        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 4107  bytes 353450 (353.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4107  bytes 353450 (353.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@ubuntu:/home/ielb#
```

Nous allons installer WireGuard sur notre machine Linux :
Avec les droit super utilisateur installer la commande « git » avec la commande apt install git

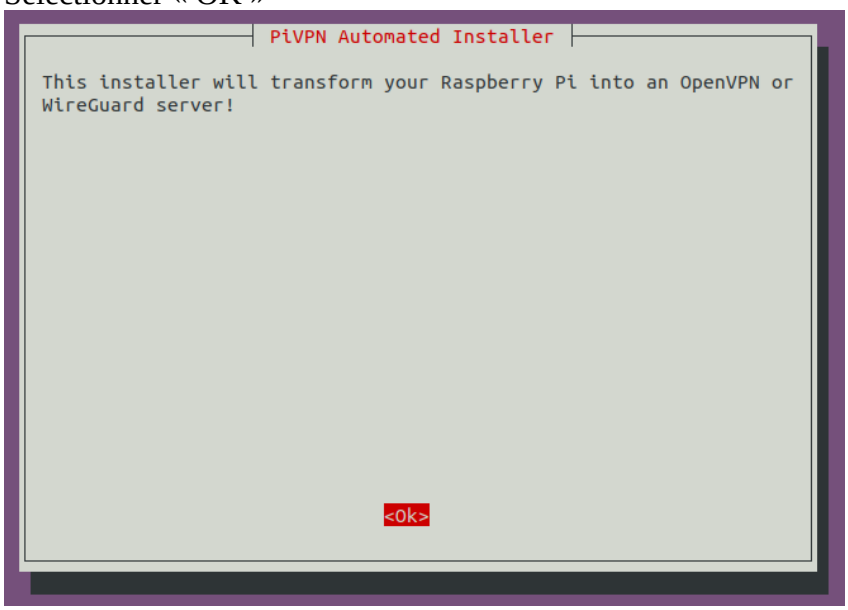
```
root@ubuntu:/home/ielb# apt install git
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  git-man liberror-perl
Suggested packages:
  git-daemon-run | git-daemon-sysvinit git-doc git-el git-email git-gui gitk gitweb
  git-cvs git-mediawiki git-svn
The following NEW packages will be installed:
  git git-man liberror-perl
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 5,468 kB of archives.
After this operation, 38.4 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://us.archive.ubuntu.com/ubuntu focal/main amd64 liberror-perl all 0.17029-1 [2
6.5 kB]
Get:2 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 git-man all 1:2.25.1-1
ubuntu3.1 [884 kB]
Get:3 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 git amd64 1:2.25.1-1ub
untu3.1 [4,557 kB]
Fetched 5,468 kB in 8s (726 kB/s)
Selecting previously unselected package liberror-perl.
(Reading database ... 182878 files and directories currently installed.)
Preparing to unpack .../liberror-perl_0.17029-1_all.deb ...
Unpacking liberror-perl (0.17029-1) ...
Selecting previously unselected package git-man.
Preparing to unpack .../git-man_1%3a2.25.1-1ubuntu3.1_all.deb ...
Unpacking git-man (1:2.25.1-1ubuntu3.1) ...
Selecting previously unselected package git.
Preparing to unpack .../git_1%3a2.25.1-1ubuntu3.1_amd64.deb ...
Unpacking git (1:2.25.1-1ubuntu3.1) ...
Setting up liberror-perl (0.17029-1) ...
Setting up git-man (1:2.25.1-1ubuntu3.1) ...
Setting up git (1:2.25.1-1ubuntu3.1) ...
Processing triggers for man-db (2.9.1-1) ...
root@ubuntu:/home/ielb#
```

Puis avec la commande « git clone https://github.com/pivpn/pivpn.git » aller chercher le scipt en ligne (En annexe il y a de fournit le script)

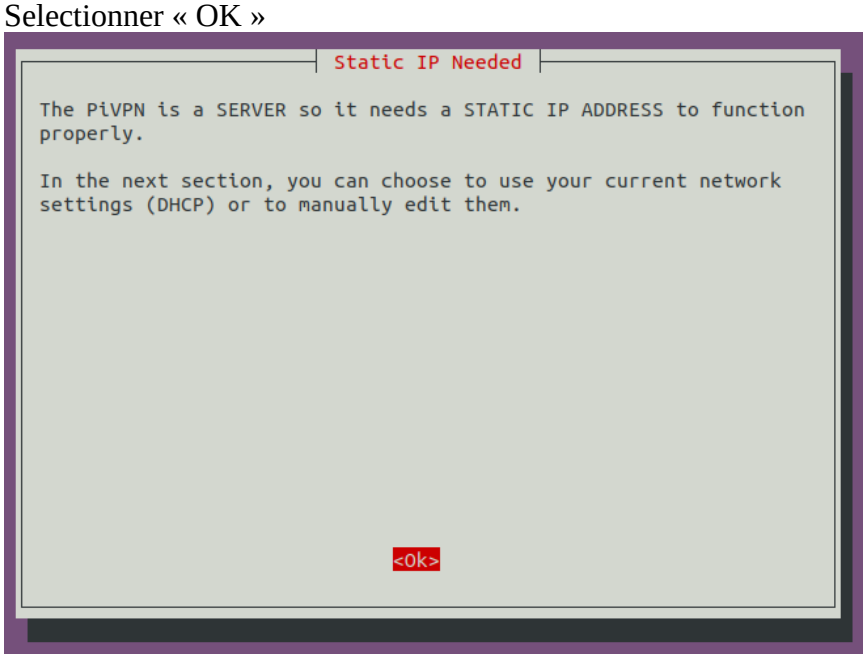
Et lancez le avec la commande sudo bash pivpn/auto_install/install.sh

```
root@ubuntu:/home/ielb# sudo bash pivpn/auto_install/install.sh
:::
::: You are root.
::: Hostname length OK
::: Verifying free disk space...
:::
::: Package Cache update is needed, running apt-get update -y ...
```

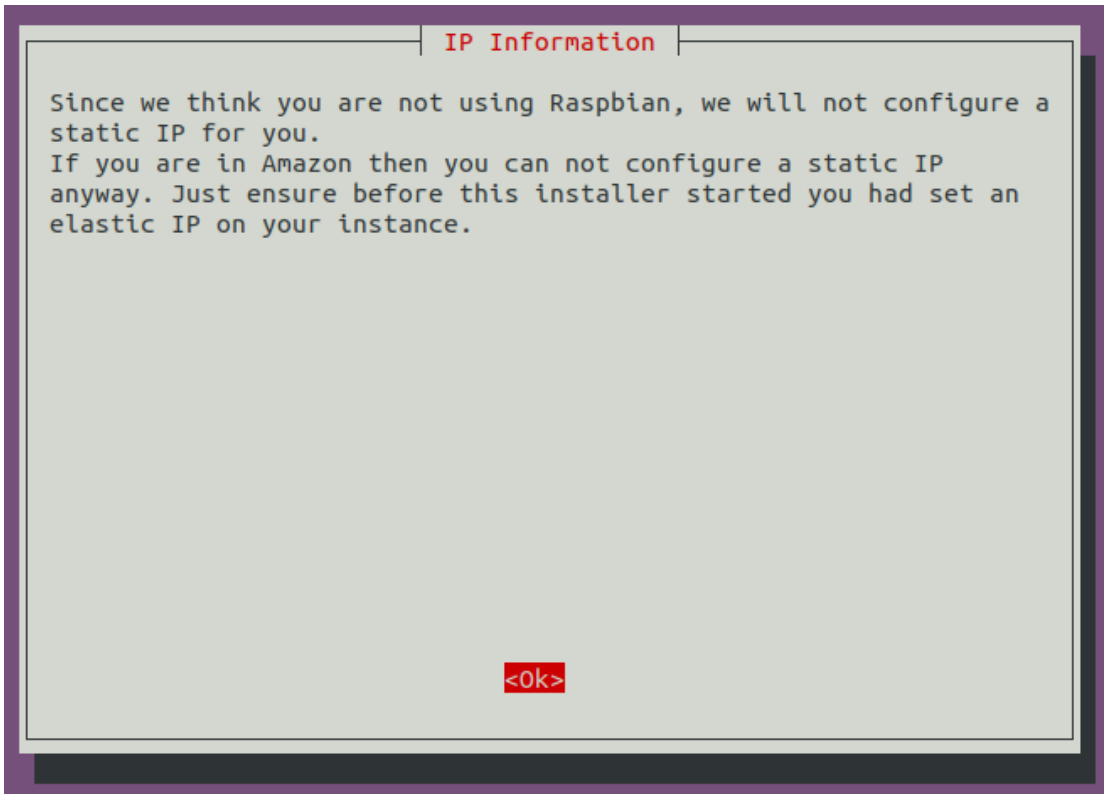
Ensuite le menu d'instalation apparait.
Selectionner « OK »



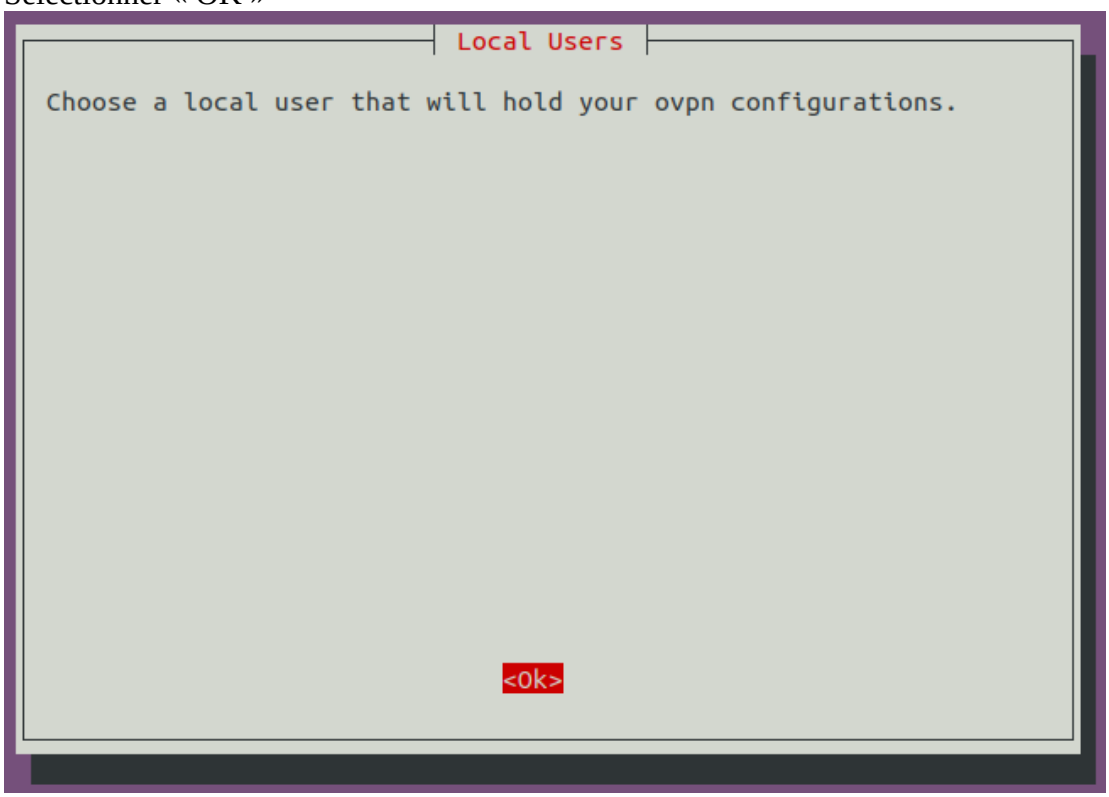
Selectionner « OK »



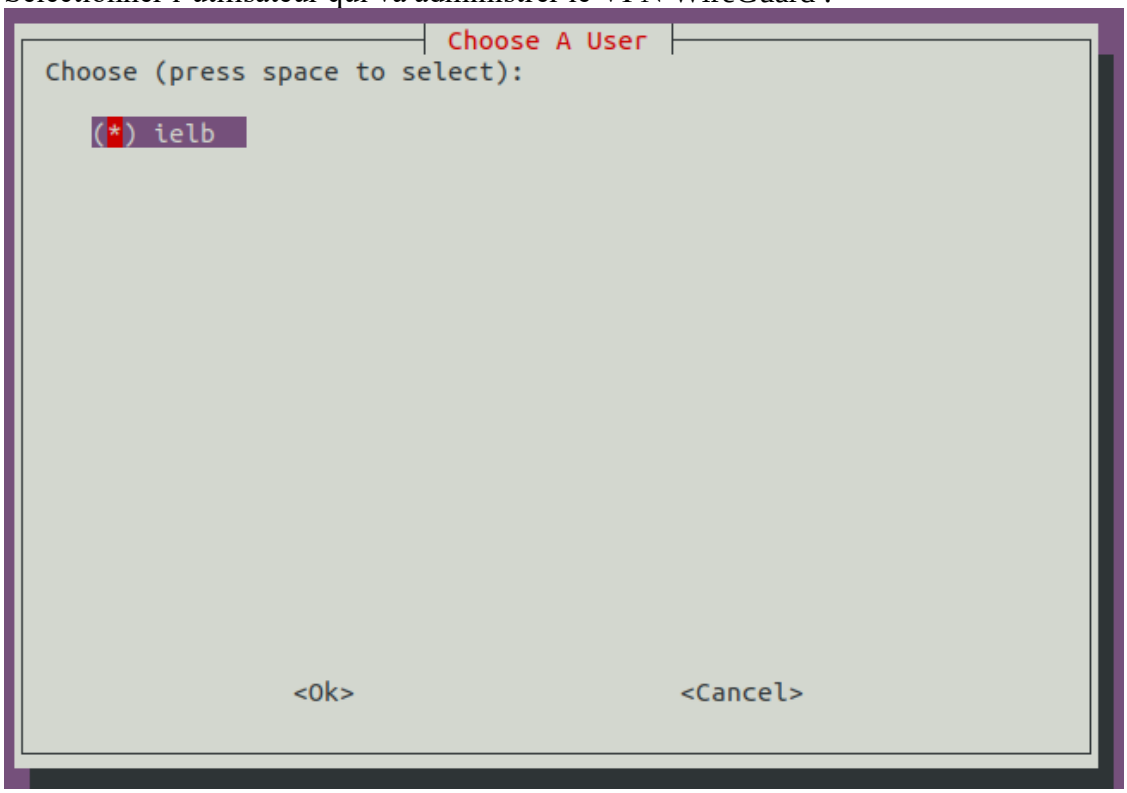
Selectionner « OK »



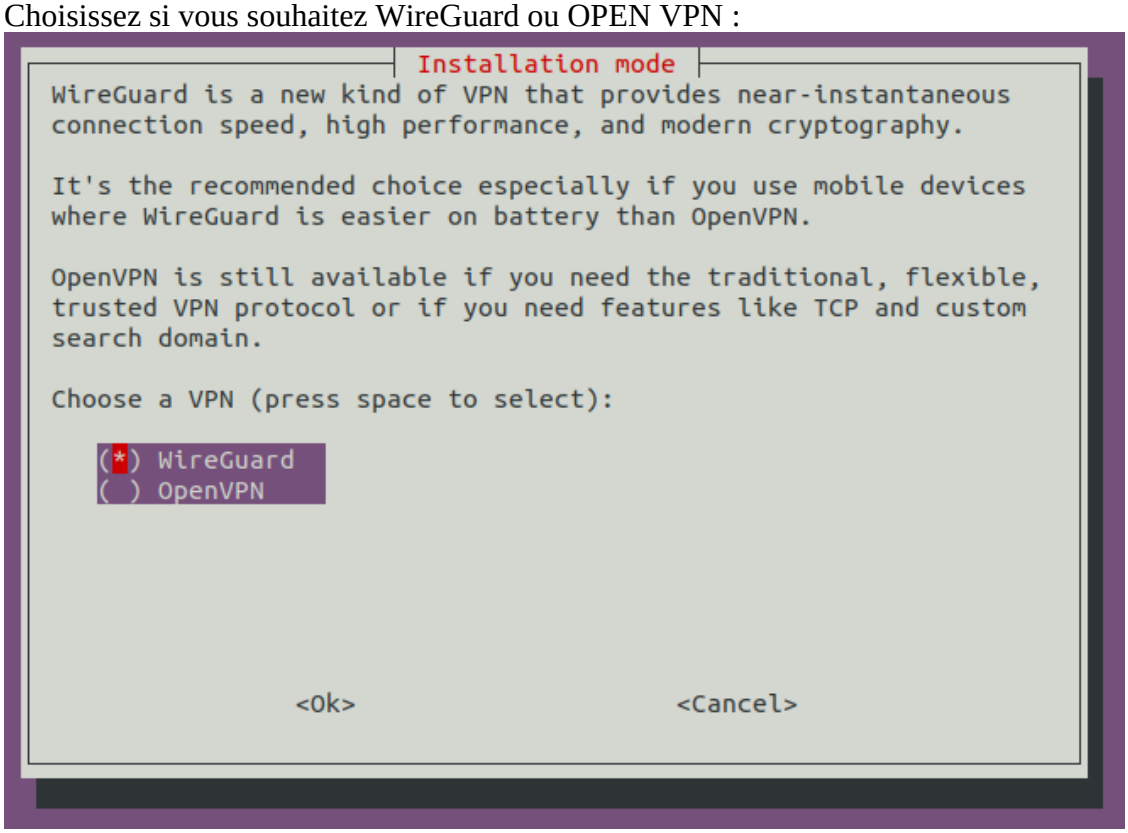
Selectionner « OK »



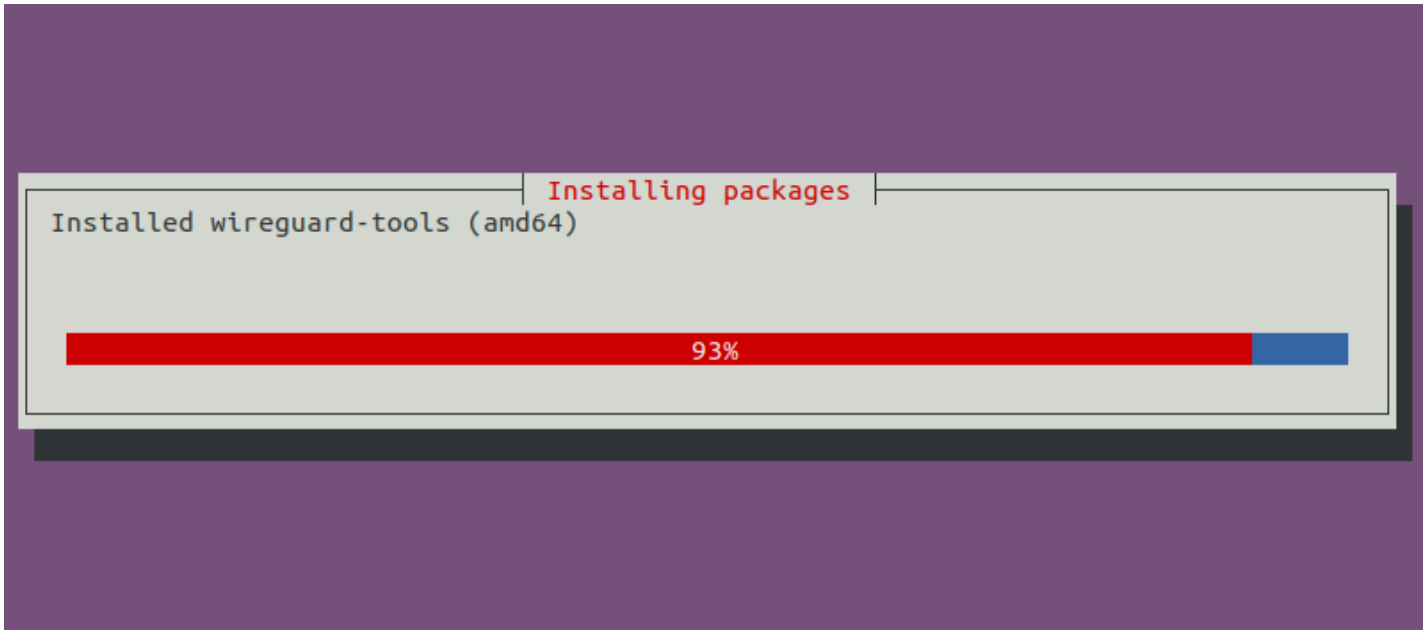
Selectionner l'utilisateur qui va administrer le VPN WireGuard :



Choisissez si vous souhaitez WireGuard ou OPEN VPN :

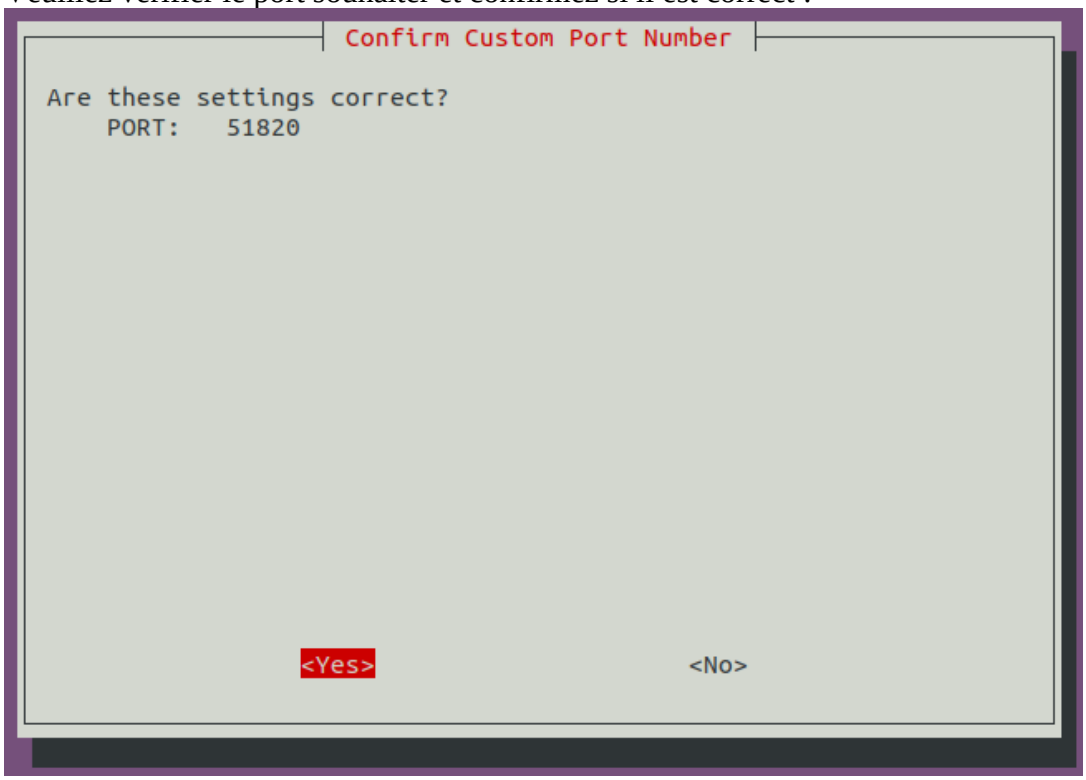


WireGuard est en cours d'installation :

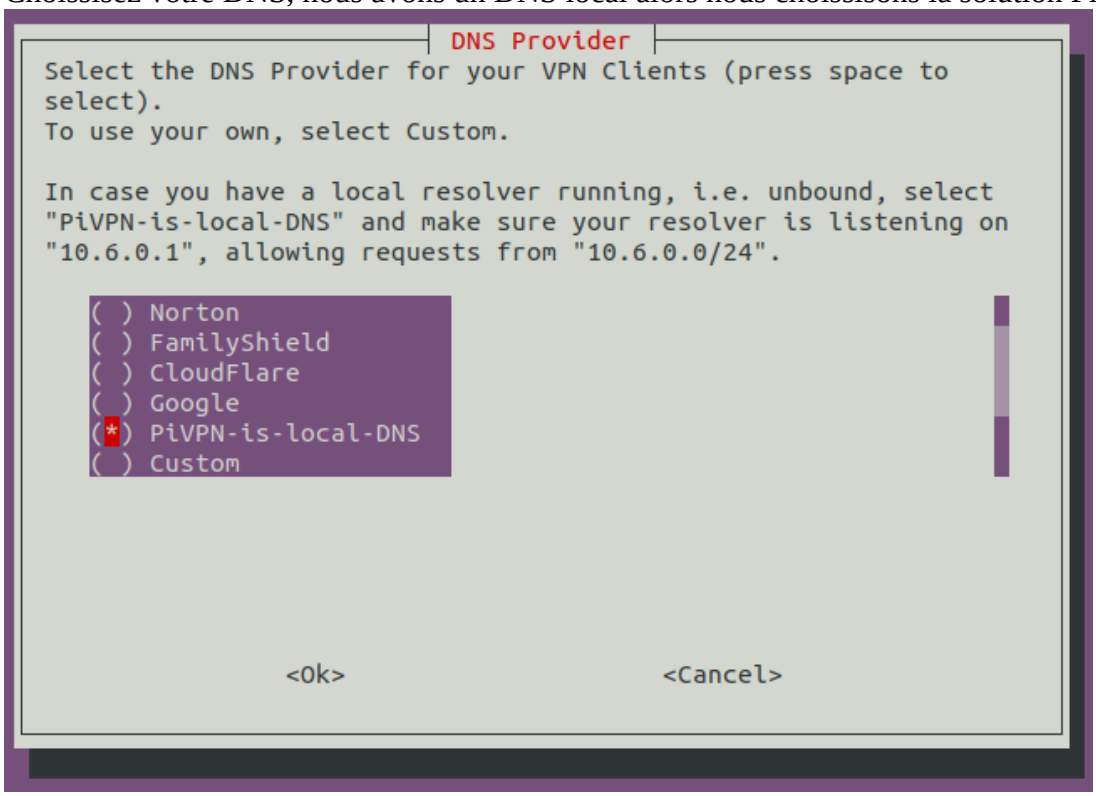


Nous allons selectionner le port sur lequel WireGuard va mettre autoriser les connection :
Par défaut il est régler a 51820.

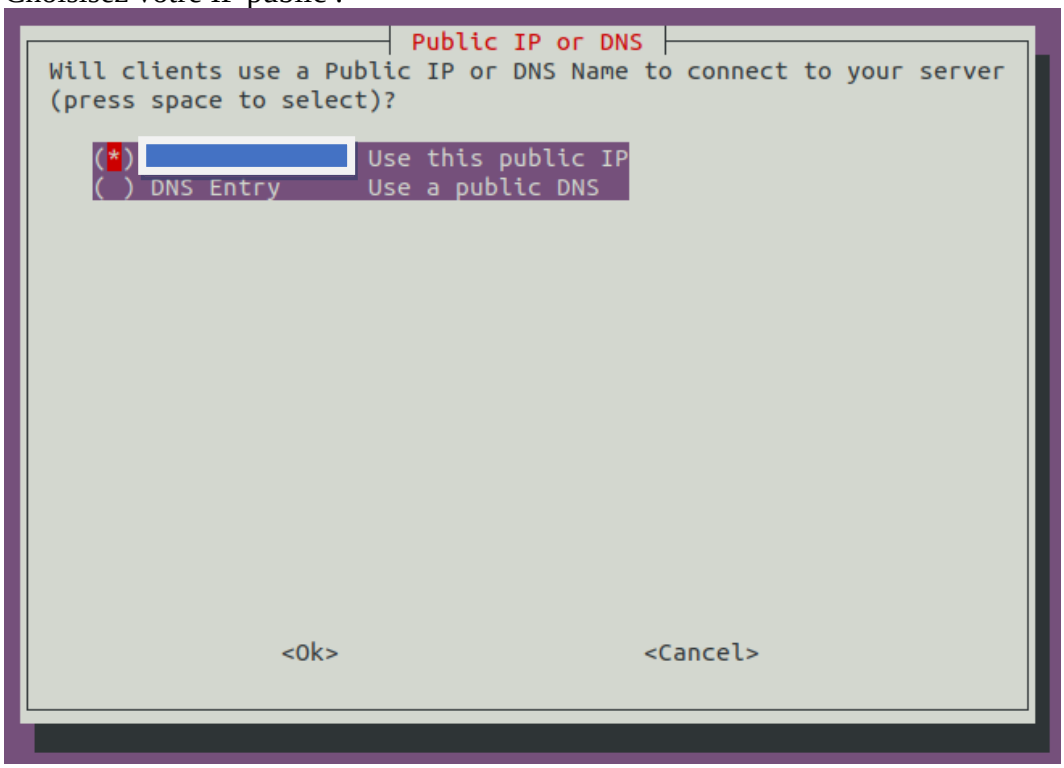
Veuillez verifier le port souhaiter et confirmez si il est correct :



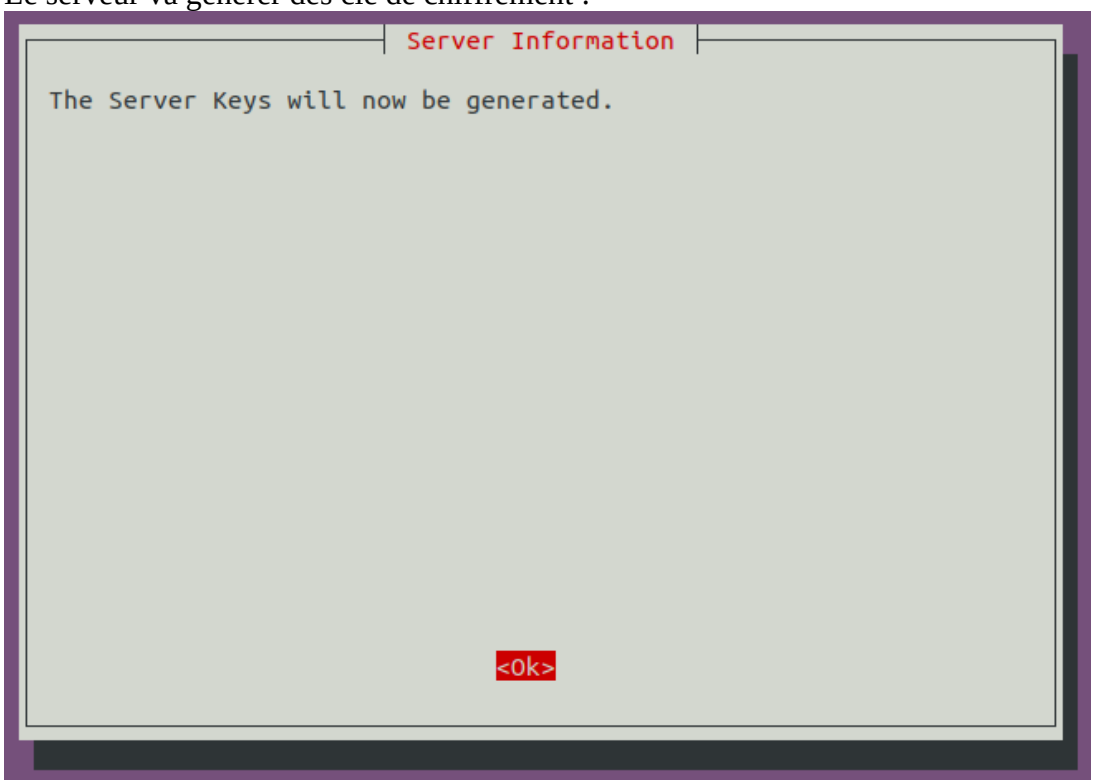
Choissisez votre DNS, nous avons un DNS local alors nous choissisons la solution PiVPN-is-local-DNS :



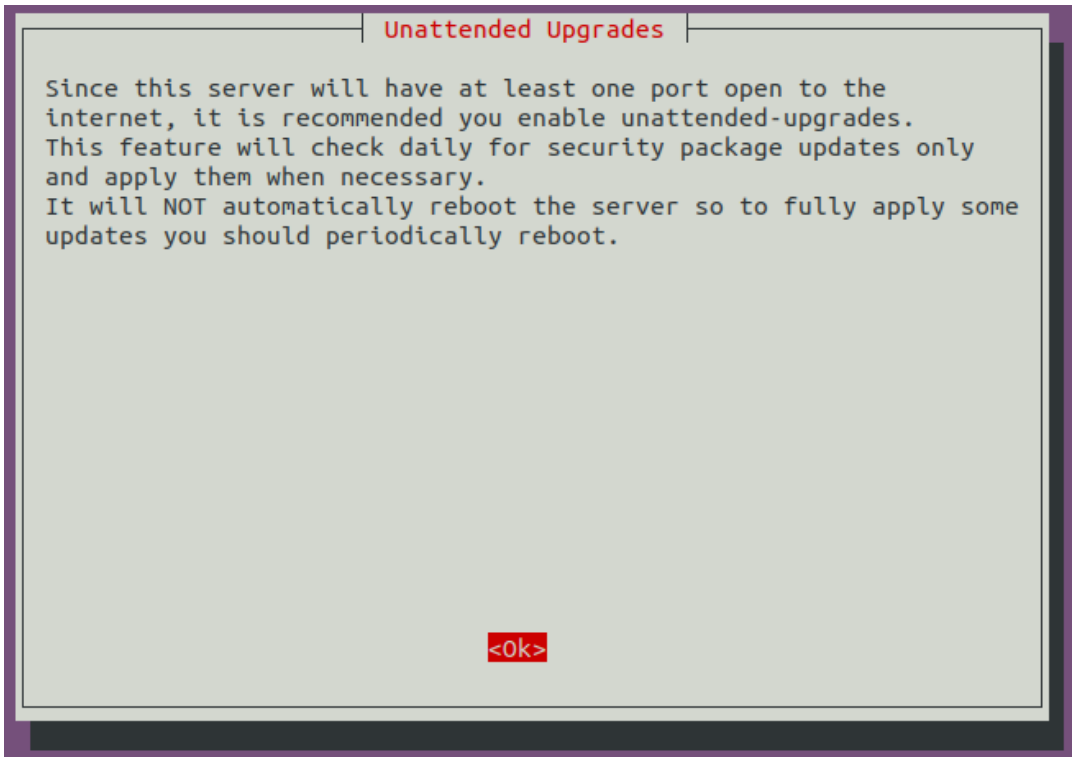
Choisisez votre IP public :



Le serveur va generer des clé de chiffrement :

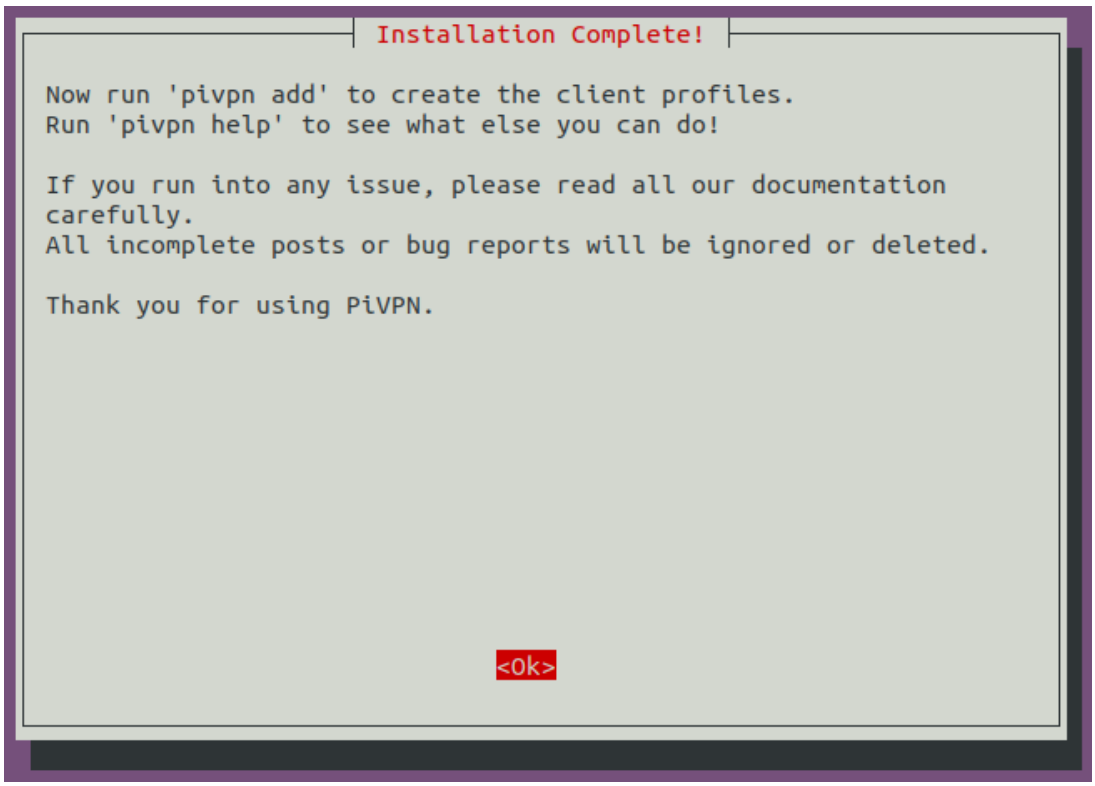


WireGuard souhaite activer la verification de mise a jour de sécurité et de les telecharger automatiquement comme ce serveur sera connecter a internet mais pour appliquer les mise a jour se sera a chaque redemarrage manuel de la machine.



Le serveur demande si vous souhaitez activer la verification de mise a jour de sécurité et de le telechargement automatique :

L'installation est effectuer.



Veuillez redemarrer le système.

5-Ajout de la configuration d'un client VPN sur le serveur

Nous allons passer a la configutaion de Wireguard afin de créer notre tunnel VPN.
Une fois redémarré, tout se passe avec la commande « pivpn », on va ajouter un client avec la commande suivante : pivpn -a

On choisi le nom de notre premier client qui va se connecter au VPN, ici Z6-PRO car c'est un mobile Android de la marque Lenovo et le model Z6-PRO.
Une clé et une config est generer et WireGuard redemarrer pour prendre cela en compte :

```
root@ubuntu:/home/ielb# pivpn -a
Enter a Name for the Client: Z6-PRO
::: Client Keys generated
::: Client config generated
::: Updated server config
::: WireGuard reloaded
================================================================================
::: Done! Z6-PRO.conf successfully created!
::: Z6-PRO.conf was copied to /home/ielb/configs for easy transfer.
::: Please use this profile only on one device and create additional
::: profiles for other devices. You can also use pivpn -qr
::: to generate a QR Code you can scan with the mobile app.
================================================================================
root@ubuntu:/home/ielb#
```

Dans le dossier « configs » on peut verifier les configuration créer, ici Z6-PRO.conf et on peut l'afficher pour le lire :

```
root@ubuntu:/home/ielb# ls configs/
Z6-PRO.conf
root@ubuntu:/home/ielb# cat configs/Z6-PRO.conf
[Interface]
PrivateKey = QChzeuPYqsipNF8UtWarakVbS80NvYAMMOEKuobV80k=
Address = 10.6.0.2/24
DNS = 10.6.0.1

[Peer]
PublicKey = npQ9spkDup/YqnkBu9T92ZYINZkllqmDueYMzqnxKCg=
PresharedKey = nxjyGwmLZUBy1e5sKZO2E0n/U0pDqaCrvWjMV5S5T5o=
Endpoint = 89.80.156.131:51820
AllowedIPs = 0.0.0.0/0, ::0/0
root@ubuntu:/home/ielb#
```

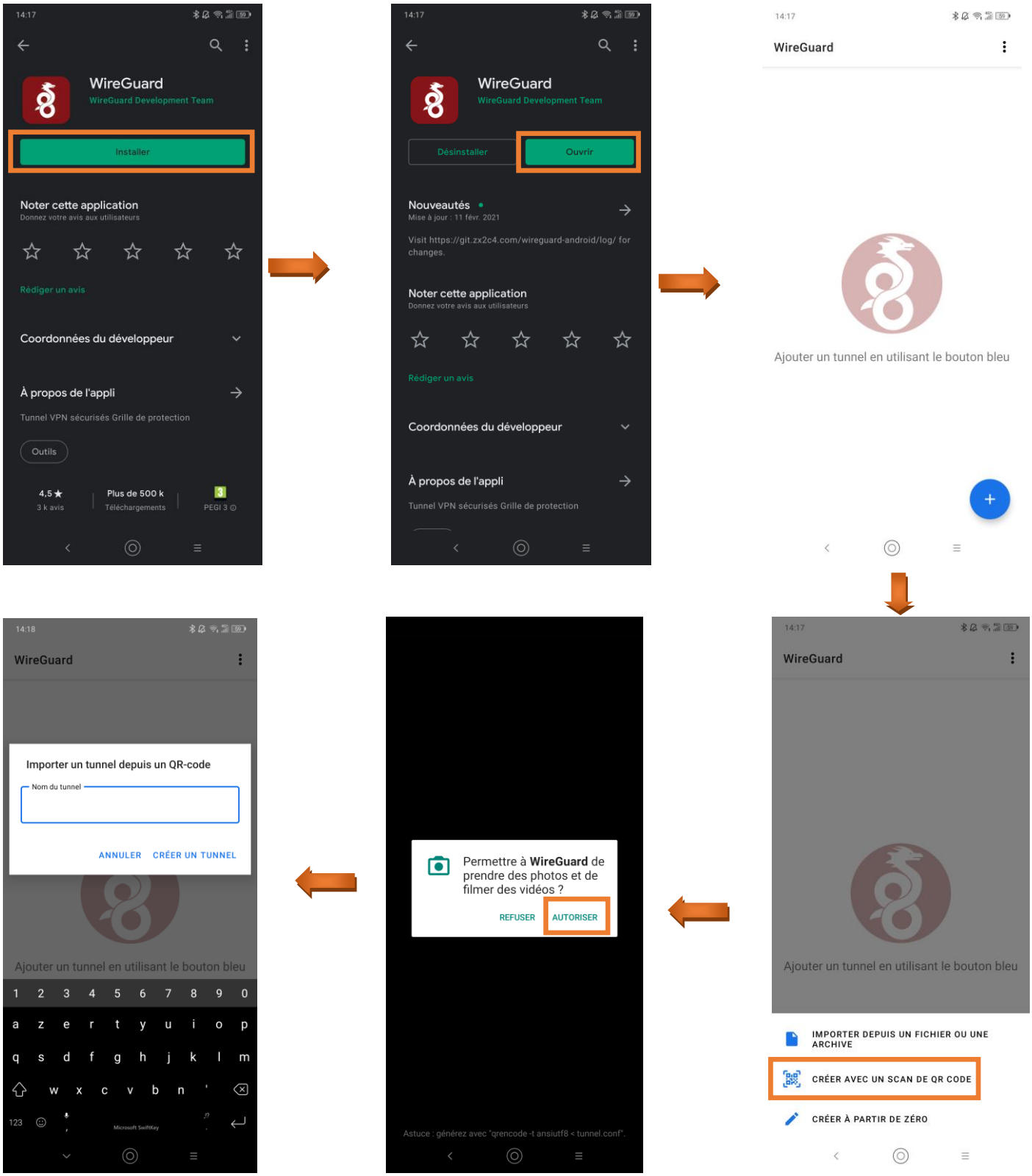
Nous allons connecter notre mibile Android a notre VPN WireGuard maintenant.
Veuillez générez un QR Code à scanner avec le client, entrez la commande : pivpn -qr
Choissez le client que vous souhaiter connecter auVPN pour generer son QR code unique :

```
root@ubuntu:/home/ielb# pivpn -qr
::  Client list  ::
1) Z6-PRO
Please enter the Index/Name of the Client to show: 1
::: Showing client Z6-PRO below
```

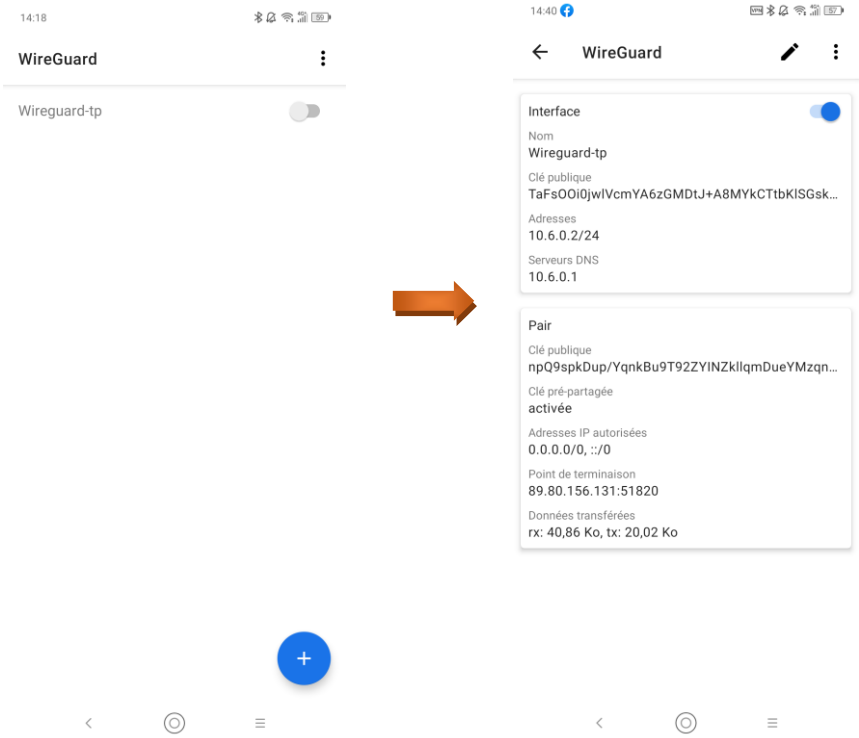
```
root@ubuntu:/home/ielb#
```

6-Configuration du client Android

Telecharger l'application WireGuard sur votre mobile (voir prerequis)



Nous avons appeler notre tunnel Wireguard-tp et nous activons notre connexion :



7-Vérification de la connexion

Avec la commande pivpn list on vois la liste des client disponible et leurs clé public.
Avec la commande pivpn clients on vois chaque client et leurs derniere connexion, leurs IP et debit utilisé.

```
root@ubuntu:/home/ielb# pivpn list
::: Clients Summary :::
Client      Public key                                    Creation date
Z6-PRO      TaFsOOi0jwlVcmYA6zGMDtJ+A8MYkCTtbKlSGskYsFU=      25 Mar 2021, 06:06, PDT
::: Disabled clients :::
root@ubuntu:/home/ielb# pivpn clients
::: Connected Clients List :::
Name        Remote IP               Virtual IP      Bytes Received      Bytes Sent      Last Seen
Z6-PRO      192.168.1.254:38561     10.6.0.2        92KiB               138KiB          Mar 25 2021 - 06:40:19
::: Disabled clients :::
root@ubuntu:/home/ielb#
```

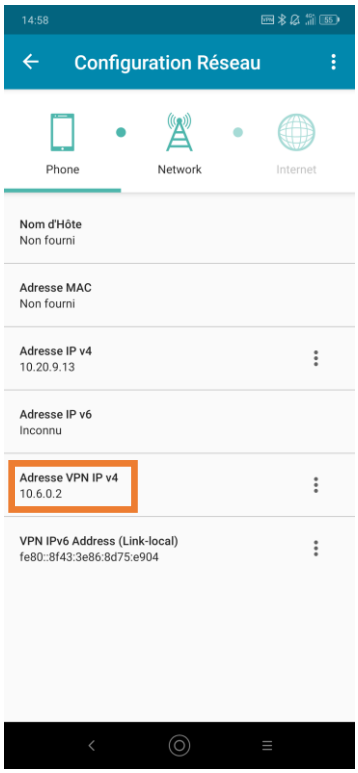
8-Test de la connexion du serveur vers le mobile

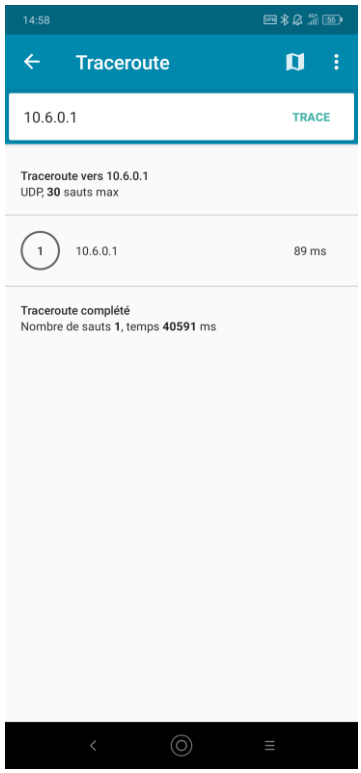
Nous testons la connexion de notre mobile sur le VPN :
Du serveur vers le client :
Le serveur est en 10.6.0.1/24 et le mobile android en 10.6.0.2/24.

```
root@ubuntu:/home/ielb# ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.152  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::557f:5e99:6cbd:db62  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:c6:be:21  txqueuelen 1000  (Ethernet)
        RX packets 33514  bytes 10939177 (10.9 MB)
        RX errors 0  dropped 1  overruns 0  frame 0
        TX packets 34427  bytes 8609006 (8.6 MB)
        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 10220  bytes 1074612 (1.0 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 10220  bytes 1074612 (1.0 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wg0: flags=209<UP,POINTOPOINT,RUNNING,NOARP>  mtu 1420
        inet 10.6.0.1  netmask 255.255.255.0  destination 10.6.0.1
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 1000  (UNSPEC)
        RX packets 1792  bytes 219848 (219.8 KB)
        RX errors 48  dropped 0  overruns 0  frame 48
        TX packets 1132  bytes 217600 (217.6 KB)
        TX errors 0  dropped 48 overruns 0  carrier 0  collisions 0

root@ubuntu:/home/ielb# ping 10.6.0.2
PING 10.6.0.2 (10.6.0.2) 56(84) bytes of data.
64 bytes from 10.6.0.2: icmp_seq=1 ttl=64 time=60.2 ms
64 bytes from 10.6.0.2: icmp_seq=2 ttl=64 time=84.4 ms
64 bytes from 10.6.0.2: icmp_seq=3 ttl=64 time=105 ms
64 bytes from 10.6.0.2: icmp_seq=4 ttl=64 time=20.6 ms
64 bytes from 10.6.0.2: icmp_seq=5 ttl=64 time=44.8 ms
^C
--- 10.6.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4008ms
rtt min/avg/max/mdev = 20.639/63.063/105.310/29.598 ms
root@ubuntu:/home/ielb#
root@ubuntu:/home/ielb#
root@ubuntu:/home/ielb#
root@ubuntu:/home/ielb# traceroute 10.6.0.2
traceroute to 10.6.0.2 (10.6.0.2), 30 hops max, 60 byte packets
 1  10.6.0.2 (10.6.0.2)  117.422 ms  118.017 ms  118.016 ms
root@ubuntu:/home/ielb#
```

9-Test de la connexion du mobile vers le serveur et internet

Du client vers le serveur :



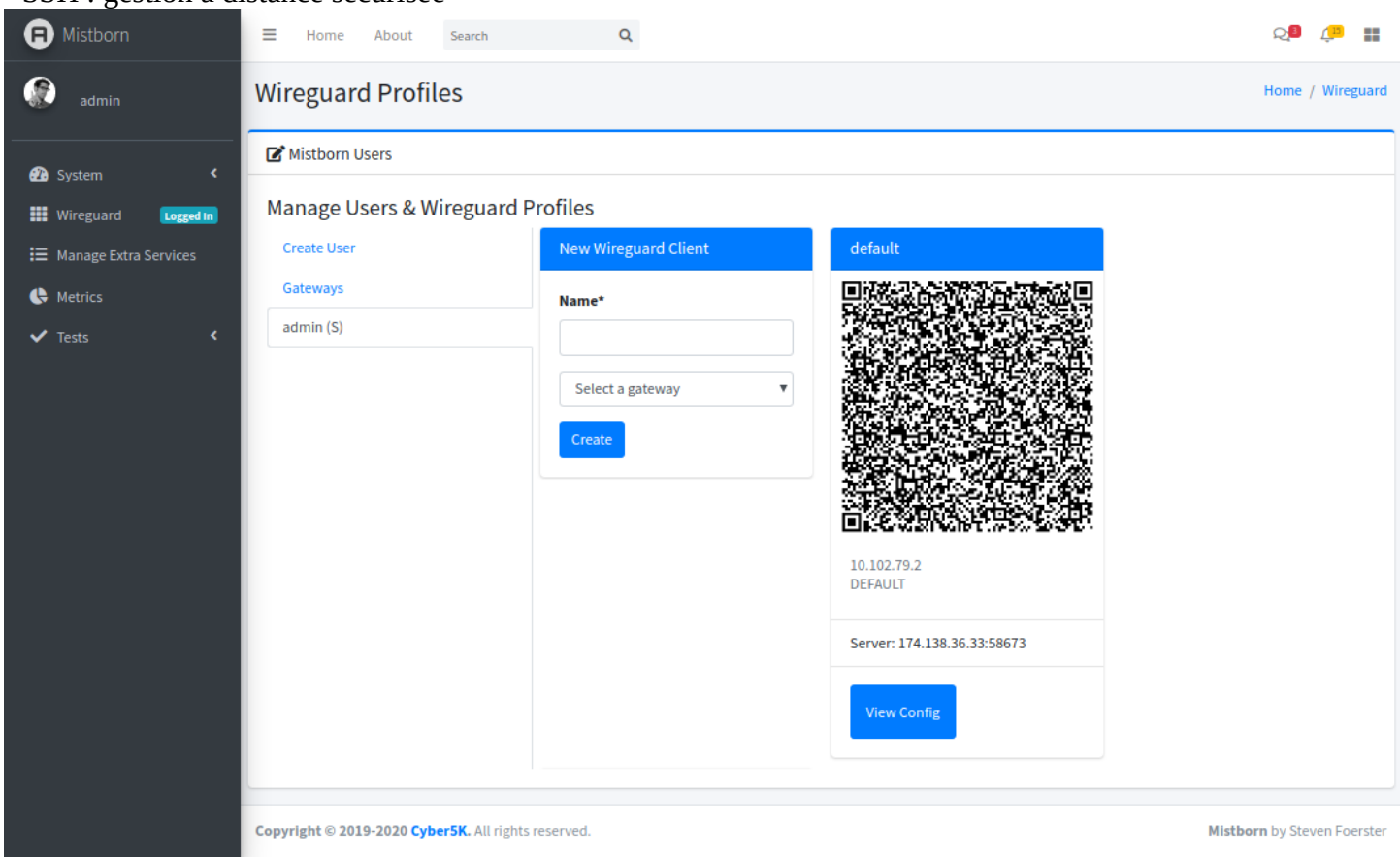


Le client mobile est bien connecter a notre Tunnel VPN et bien à internet.

10-Présentation de Mistborn

Nous allons proceder a l'installation d'une interface graphique pour la gestion du VPN avec Mistborn. Mistborn a commencé comme un projet passionnel pour un mari et un père protégeant sa famille. Certains membres de la famille ont insisté pour connecter leurs appareils à des réseaux WiFi publics gratuits. Nous avions besoin d'un moyen de sécuriser tous les appareils de la famille avec un VPN solide (Wireguard). Une fois que nous avons eu cela, nous voulions contrôler DNS pour bloquer les publicités sur tous les appareils et bloquer les sites Web malveillants sur tous les appareils de la famille. Ensuite, nous voulions des services de chat, de partage de fichiers et de discussion en ligne que nous pourrions utiliser pour nous-mêmes sans confier nos données à une grande entreprise de technologie. Et puis ... la domotique. Je sais que j'ajouterai plus de services, donc j'ai rendu cela facile à faire. Idéal pour les équipes qui:
- déteste les publicités Internet
- doivent être protégés contre les domaines Internet malveillants
- besoin de collaborer en toute sécurité
- nécessite une authentification multifacteur pour Wireguard
- souhaitent conserver la propriété exclusive de leurs données
- souhaitez accorder et révoquer facilement l'accès aux personnes et aux appareils via une interface Web simple
- veulent un accès Internet sécurisé où qu'ils soient
- souhaitez limiter ou arrêter les services de collecte de données
- voulez éviter d'être détecté / bloqué pour l'utilisation d'un proxy ou d'un service VPN

Mistborn dépend de ces technologies open source fondamentales:
- Docker : conteneurisation
- Wireguard : accès VPN sécurisé
- SSH : gestion à distance sécurisée



11-Installation de Mistborn

Nous allons télécharger le script de mistborn et l'installer via les commande :
git clone https://gitlab.com/cyber5k/mistborn.git
sudo -E bash ./mistborn/scripts/install.sh
Creez un mot de passe pour le compte admin par défaut :

```
ielb@ubuntu:~$ git clone https://gitlab.com/cyber5k/mistborn.git
Cloning into 'mistborn'...
remote: Enumerating objects: 296, done.
remote: Counting objects: 100% (296/296), done.
remote: Compressing objects: 100% (129/129), done.
remote: Total 1299 (delta 238), reused 198 (delta 158), pack-reused 1003
Receiving objects: 100% (1299/1299), 201.14 KiB | 246.00 KiB/s, done.
Resolving deltas: 100% (854/854), done.
ielb@ubuntu:~$ sudo -E bash ./mistborn/scripts/install.sh
Creating user: mistborn
mistborn already in sudoers
~ ~
~
Running as mistborn
  ____        _                  ____   _  __
 / ___|   _| |__   ___ _ __    | ___|| |/ /
| |  | | | | '_ \ / _ \ '__| |___ \| ' /
| |__| |_| | |_) |  __/ |      ___) | . \
 \____\__, |_.__/ \___|_|     |____/|_|\_\
      |___/
 __  __ _      _   _
|  \/  (_)___| |_| |__    ___  _ __ _ __
| |\/| | / __| __| '_ \ / _ \| '__| '_ \
| |  | | \__ \ |_| |_) | (_) | |  | | | |
|_|  |_|_|___/\__|_.__/ \___/|_|  |_| |_|

Cloning master branch from mistborn repo
Cloning into '/opt/mistborn'...
remote: Enumerating objects: 296, done.
remote: Counting objects: 100% (296/296), done.
remote: Compressing objects: 100% (129/129), done.
remote: Total 1299 (delta 238), reused 198 (delta 158), pack-reused 1003
Receiving objects: 100% (1299/1299), 201.14 KiB | 330.00 KiB/s, done.
Resolving deltas: 100% (854/854), done.
~ ~
Submodule 'modules/mistborn-cli' (https://gitlab.com/cyber5k/mistborn-cli.git) registered
 for path 'modules/mistborn-cli'
Cloning into '/opt/mistborn/modules/mistborn-cli'...
Submodule path 'modules/mistborn-cli': checked out '00986bcb7f945c611d13099672b4fb1e2d072
1f2'

(Mistborn) The default admin password may only container alphanumeric characters and _
(Mistborn) Set default admin password:
```

```
(Mistborn) The default admin password may only container alphanumeric characters and _
(Mistborn) Set default admin password:
(Mistborn) Password is accepted

MISTBORN_DEFAULT_PASSWORD is set

Install Cockpit (a somewhat resource-heavy system management graphical user interface --
NOT RECOMMENDED on Raspberry Pi)? [y/N]: y

Generating SSH keypair for mistborn
Hit:1 http://us.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://security.ubuntu.com/ubuntu focal-security InRelease [109 kB]
Get:3 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:4 http://us.archive.ubuntu.com/ubuntu focal-backports InRelease [101 kB]
Get:5 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [574 kB]
Get:6 http://us.archive.ubuntu.com/ubuntu focal-updates/main i386 Packages [443 kB]
Get:7 http://security.ubuntu.com/ubuntu focal-security/main i386 Packages [210 kB]
Get:8 http://security.ubuntu.com/ubuntu focal-security/main Translation-en [120 kB]
Get:9 http://security.ubuntu.com/ubuntu focal-security/main amd64 DEP-11 Metadata [24.3 k
B]
Get:10 http://security.ubuntu.com/ubuntu focal-security/main amd64 c-n-f Metadata [7,376
B]
Get:11 http://security.ubuntu.com/ubuntu focal-security/restricted amd64 Packages [148 kB
]
Get:12 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [889 kB]
39% [12 Packages 31.2 kB/889 kB 4%] [11 Packages 91.3 kB/148 kB 62%]
```

L'installation continu, elle prend plusieurs minutes.

```
/maintenance
 ---> Running in 02a9cf2e6f7b
Removing intermediate container 02a9cf2e6f7b
 ---> c4c5970d7074
Successfully built c4c5970d7074
Successfully tagged mistborn_production_postgres:latest
ubuntu already in /etc/hosts
address=/.mistborn/10.2.3.1
backup up original volumes folder
cleaning old docker volumes
mistborn_production_postgres_data
mistborn_production_postgres_data_backups
mistborn_production_traefik
Total reclaimed space: 0B
cleaning old wireguard services
/opt/mistborn /opt/mistborn
never connected
stoppping, disabling, and removing  wg0
Removed /etc/systemd/system/multi-user.target.wants/wg-quick@wg0.service.
/opt/mistborn
Created symlink /etc/systemd/system/multi-user.target.wants/Mistborn-base.service → /etc/
systemd/system/Mistborn-base.service.
Job for Mistborn-base.service failed because the control process exited with error code.
See "systemctl status Mistborn-base.service" and "journalctl -xe" for details.
root@ubuntu:/home/ielb# systemctl status Mistborn-base.service
● Mistborn-base.service - Mistborn Base
     Loaded: loaded (/etc/systemd/system/Mistborn-base.service; enabled; vendor preset: >
     Active: failed (Result: exit-code) since Fri 2021-03-26 07:20:35 PDT; 6s ago
    Process: 18677 ExecStartPre=/usr/local/bin/docker-compose -f /opt/mistborn/base.yml >
    Process: 18679 ExecStartPre=/usr/local/bin/docker-compose -f /opt/mistborn/base.yml >
    Process: 18688 ExecStartPre=/sbin/ip address add 10.2.3.1/30 dev ens33 (code=exited,>
    Process: 18689 ExecStartPre=/sbin/iptables -w -I DOCKER-USER -i ens33 -p udp --dport>
    Process: 18690 ExecStopPost=/sbin/iptables -D DOCKER-USER -i ens33 -p udp --dport 53>
    Process: 18691 ExecStopPost=/sbin/iptables -D DOCKER-USER -i ens33 -p tcp --dport 53>
    Process: 18692 ExecStopPost=/sbin/iptables -D DOCKER-USER -i ens33 -p tcp --dport 80>
    Process: 18693 ExecStopPost=/sbin/iptables -D DOCKER-USER -i ens33 -p tcp --dport 44>
    Process: 18694 ExecStopPost=/sbin/iptables -D DOCKER-USER -i ens33 -p tcp --dport 55>
    Process: 18695 ExecStopPost=/sbin/iptables -D OUTPUT -o ens33 -p udp --dport 53 -j M>
    Process: 18696 ExecStopPost=/sbin/ip6tables -D OUTPUT -p udp --dport 53 -j MISTBORN_>

Mar 26 07:20:35 ubuntu systemd[1]: Mistborn-base.service: Failed with result 'exit-code'.
Mar 26 07:20:35 ubuntu systemd[1]: Failed to start Mistborn Base.
Mar 26 07:20:35 ubuntu systemd[1]: Mistborn-base.service: Scheduled restart job, restart>
Mar 26 07:20:35 ubuntu systemd[1]: Stopped Mistborn Base.
Mar 26 07:20:35 ubuntu systemd[1]: Mistborn-base.service: Start request repeated too qui>
Mar 26 07:20:35 ubuntu systemd[1]: Mistborn-base.service: Failed with result 'exit-code'.
Mar 26 07:20:35 ubuntu systemd[1]: Failed to start Mistborn Base.

root@ubuntu:/home/ielb#
```

Une erreur apparait. Il semblerais que cette interface graphique sois incomptatible avec PiVPN a cause de la gestion des fichier.

12-ANNEXES : Script pivpn.git

Voici le script que vous aller chercher si vous voulez l'adapter a votre situation :
https://github.com/pivpn/pivpn/blob/master/scripts/pivpn

```
#!/bin/bash

# Must be root to use this tool
if [ $EUID -ne 0 ];then
                                                        if dpkg-query -s sudo &> /dev/null; then
        export SUDO="sudo"

                                                        else
                                                           echo "::: Please install sudo or run this as roo
                                                           exit 1
      fi
fi

scriptDir="/opt/pivpn"
uninstallServer(){
   $SUDO ${scriptDir}/uninstall.sh
   exit "$?"
}
backup(){
   $SUDO ${scriptDir}/backup.sh
   exit "$?"
}
showHelp(){
   echo "::: To pass off to the pivpn command for each protocol"
   echo ":::"
   echo "::: Usage: pivpn wg  <command> [option]"
   echo "::: Usage: pivpn ovpn <command> [option]"
   echo ":::"
   echo ":::  -h,  help           Show this help dialog"
   echo ":::  -u,  uninstall       Uninstall pivpn from your system!"
   echo ":::  -bk, backup          Backup VPN configs and user profiles"
   exit 0
}
if [ $# = 0 ]; then
   showHelp
fi
# Handle redirecting to specific functions based on arguments
case "$1" in
    wg                  ) "${scriptDir}/wireguard/pivpn.sh" "${@:2}";;
    ovpn                ) "${scriptDir}/openvpn/pivpn.sh"   "${@:2}";;
"-h"  | "help"            ) showHelp;;
"-u"  | "uninstall"         ) uninstallServer;;
"-bk" | "backup"            ) backup ;;
*                       ) showHelp;;
esac
```

13-ANNEXES : Script mistborn.git

Voici le script que vous aller chercher si vous voulez l'adapter a votre situation :
https://gitlab.com/cyber5k/mistborn/-/blob/master/scripts/install.sh

```
#!/bin/bash

set -e

export DEBIAN_FRONTEND=noninteractive

## ensure run as nonroot user
#if [ "$EUID" -eq 0 ]; then
MISTBORN_USER="mistborn"
if [ $(whoami) != "$MISTBORN_USER" ]; then
	echo "Creating user: $MISTBORN_USER"
	sudo useradd -s /bin/bash -d /home/$MISTBORN_USER -m -G sudo $MISTBORN_USER 2>/dev/null
|| true
	SCRIPTPATH="$( cd "$(dirname "$0")" ; pwd -P )"
	#echo "SCRIPTPATH: $SCRIPTPATH"
	FILENAME=$(basename -- "$0")
	#echo "FILENAME: $FILENAME"
	FULLPATH="$SCRIPTPATH/$FILENAME"
	#echo "FULLPATH: $FULLPATH"

	# SUDO
	case `sudo grep -e "^$MISTBORN_USER.*" /etc/sudoers >/dev/null; echo $?` in
	0)
	  echo "$MISTBORN_USER already in sudoers"
	  ;;
	1)
	  echo "Adding $MISTBORN_USER to sudoers"
	  sudo bash -c "echo '$MISTBORN_USER  ALL=(ALL) NOPASSWD: ALL' >> /etc/sudoers"
	  ;;
	*)
	  echo "There was a problem checking sudoers"
	  ;;
	esac

	# get git branch if one exists (default to master)
	pushd .
	cd $SCRIPTPATH
	GIT_BRANCH=$(git symbolic-ref --short HEAD || echo "master")
	popd

	sudo cp $FULLPATH /home/$MISTBORN_USER
	sudo chown $MISTBORN_USER:$MISTBORN_USER /home/$MISTBORN_USER/$FILENAME
	sudo SSH_CLIENT="$SSH_CLIENT"
MISTBORN_DEFAULT_PASSWORD="$MISTBORN_DEFAULT_PASSWORD"
GIT_BRANCH="$GIT_BRANCH"
```

```
MISTBORN_INSTALL_COCKPIT="$MISTBORN_INSTALL_COCKPIT" -i -u $MISTBORN_USER
bash -c "/home/$MISTBORN_USER/$FILENAME" # self-referential call
        exit 0
fi

echo "Running as $USER"

# banner
echo -e "  _____      _                 _____  _  __"
echo -e " / ____|    _| |__   ____ _ __  | ____|| |/ /"
echo -e "| |  | | | | '_ \ / _ \ '__| |___ \| ' /"
echo -e "| |__| |_| | |_) |  __/ |     ____) | . \ "
echo -e " \_____\__, |_.__/ \___|_|    |_____/|_|\_\ "
echo -e "       |___/"
echo -e " __  __ _      _   _"
echo -e "|  \/  (_)____| |_| |__   ___  _ __ _ __"
echo -e "| |\/| | / __| __| '_ \ / _ \| '__| '_ \ "
echo -e "| |  | | \__ \ |_| |_) | (_) | |  | | | |"
echo -e "|_|  |_|_|___/\__|_.__/ \___/|_|  |_| |_|"
echo -e ""

sudo rm -rf /opt/mistborn 2>/dev/null || true

# clone to /opt and change directory
echo "Cloning $GIT_BRANCH branch from mistborn repo"
sudo git clone https://gitlab.com/cyber5k/mistborn.git -b $GIT_BRANCH /opt/mistborn
sudo chown -R $USER:$USER /opt/mistborn
pushd .
cd /opt/mistborn
git submodule update --init --recursive

# MISTBORN_DEFAULT_PASSWORD
source ./scripts/subinstallers/passwd.sh

# Install Cockpit?
if [ -z "${MISTBORN_INSTALL_COCKPIT}" ]; then
    read -p "Install Cockpit (a somewhat resource-heavy system management graphical user interface -- NOT
RECOMMENDED on Raspberry Pi)? [y/N]: " MISTBORN_INSTALL_COCKPIT
    echo
    MISTBORN_INSTALL_COCKPIT=${MISTBORN_INSTALL_COCKPIT:-N}
fi

# SSH keys
if [ ! -f ~/.ssh/id_rsa ]; then
    echo "Generating SSH keypair for $USER"
    ssh-keygen -t rsa -b 4096 -N "" -m pem -f ~/.ssh/id_rsa -q

    # Authorized keys
    echo "from=\"172.16.0.0/12,192.168.0.0/16,10.0.0.0/8\" $(cat ~/.ssh/id_rsa.pub)" > ~/.ssh/authorized_keys
else
```

```
    echo "SSH key exists for $USER"
fi

# initial load update package list
sudo apt-get update

# install figlet
sudo -E apt-get install -y figlet

# get os and distro
source ./scripts/subinstallers/platform.sh


# iptables
echo "Setting up firewall (iptables)"
if [ ! -f "/etc/iptables/rules.v4" ]; then
    echo "Setting iptables rules..."
    ./scripts/subinstallers/iptables.sh
else
    echo "iptables rules exist. Leaving alone."
fi


# SSH Server
sudo -E apt-get install -y openssh-server
#sudo sed -i 's/#PasswordAuthentication.*/PasswordAuthentication yes/' /etc/ssh/sshd_config
#sudo sed -i 's/PasswordAuthentication.*/PasswordAuthentication yes/' /etc/ssh/sshd_config
#sudo sed -i 's/#PermitRootLogin.*/PermitRootLogin prohibit-password/' /etc/ssh/sshd_config
#sudo sed -i 's/PermitRootLogin.*/PermitRootLogin prohibit-password/' /etc/ssh/sshd_config
sudo sed -i 's/#Port.*/Port 22/' /etc/ssh/sshd_config
sudo sed -i 's/Port.*/Port 22/' /etc/ssh/sshd_config
sudo systemctl enable ssh
sudo systemctl restart ssh

# Additional tools fail2ban
sudo -E apt-get install -y dnsutils fail2ban

# Install kernel headers
if [ "$DISTRO" == "ubuntu" ] || [ "$DISTRO" == "debian" ]; then
    sudo -E apt install -y linux-headers-$(uname -r)
elif [ "$DISTRO" == "raspbian" ] || [ "$DISTRO" == "raspios" ]; then
    sudo -E apt install -y raspberrypi-kernel-headers
else
    echo "Unsupported OS: $DISTRO"
    exit 1
fi

# Wireugard
source ./scripts/subinstallers/wireguard.sh
```

```
# Docker
source ./scripts/subinstallers/docker.sh
sudo systemctl enable docker
sudo systemctl start docker

# Unattended upgrades
sudo -E apt-get install -y unattended-upgrades

# Cockpit
if [[ "$MISTBORN_INSTALL_COCKPIT" =~ ^([yY][eE][sS]|[yY])$ ]]
then
    # install cockpit
    source ./scripts/subinstallers/cockpit.sh

    # set variable (that will be available in environment)
    MISTBORN_INSTALL_COCKPIT=Y
fi

# Mistborn-cli (pip3 installed by docker)
figlet "Mistborn: Installing mistborn-cli"
sudo pip3 install -e ./modules/mistborn-cli

# Mistborn
# final setup vars

#IPV4_PUBLIC=$(ip -o -4 route show default | egrep -o 'dev [^ ]*' | awk '{print $2}' | xargs ip -4 addr show |
grep 'inet ' | awk '{print $2}' | grep -o "^[0-9.]*"  | tr -cd '\11\12\15\40-\176' | head -1) # tail -1 to get last
IPV4_PUBLIC="10.2.3.1"


# generate production .env file
#if [ ! -d ./.envs/.production ]; then
./scripts/subinstallers/gen_prod_env.sh "$MISTBORN_DEFAULT_PASSWORD"
#fi

# unattended upgrades
sudo cp ./scripts/conf/20auto-upgrades /etc/apt/apt.conf.d/
sudo cp ./scripts/conf/50unattended-upgrades /etc/apt/apt.conf.d/

sudo systemctl stop unattended-upgrades
sudo systemctl daemon-reload
sudo systemctl restart unattended-upgrades

# setup Mistborn services

#if [ "$DISTRO" == "debian" ] || [ "$DISTRO" == "raspbian" ]; then
#    # remove systemd-resolved lines
#    sudo sed -i '/.*systemd-resolved/d' /etc/systemd/system/Mistborn-base.service
#fi
```

```
sudo cp ./scripts/services/Mistborn-setup.service /etc/systemd/system/

# setup local volumes for pihole
sudo mkdir -p ../mistborn_volumes/
sudo chown -R root:root ../mistborn_volumes/
sudo mkdir -p ../mistborn_volumes/base/pihole/etc-pihole
sudo mkdir -p ../mistborn_volumes/base/pihole/etc-dnsmasqd
sudo mkdir -p ../mistborn_volumes/extra

# Traefik final setup (cockpit)
#cp ./compose/production/traefik/traefikv2.toml.template ./compose/production/traefik/traefik.toml

# setup tls certs
source ./scripts/subinstallers/openssl.sh
#sudo rm -rf ../mistborn_volumes/base/tls
#sudo mv ./tls ../mistborn_volumes/base/

# enable and run setup to generate .env
sudo systemctl enable Mistborn-setup.service
sudo systemctl start Mistborn-setup.service

# Download docker images while DNS is operable
sudo docker-compose -f base.yml pull || true
sudo docker-compose -f base.yml build

## disable systemd-resolved stub listener (creates symbolic link to /etc/resolv.conf)
if [ -f /etc/systemd/resolved.conf ]; then
    sudo sed -i 's/#DNSStubListener.*/DNSStubListener=no/' /etc/systemd/resolved.conf
    sudo sed -i 's/DNSStubListener.*/DNSStubListener=no/' /etc/systemd/resolved.conf
fi

## delete symlink if exists
if [ -L /etc/resolv.conf ]; then
    sudo rm /etc/resolv.conf
fi

## disable other DNS services
sudo systemctl stop systemd-resolved 2>/dev/null || true
sudo systemctl disable systemd-resolved 2>/dev/null || true
sudo systemctl stop dnsmasq 2>/dev/null || true
sudo systemctl disable dnsmasq 2>/dev/null || true

# hostname in /etc/hosts
sudo grep -qF "$(hostname)" /etc/hosts && echo "$(hostname) already in /etc/hosts" || echo "127.0.1.1
$(hostname) $(hostname)" | sudo tee -a /etc/hosts

# resolve all *.mistborn domains
echo "address=/.mistborn/10.2.3.1" | sudo tee ../mistborn_volumes/base/pihole/etc-dnsmasqd/02-lan.conf

# ResolvConf (OpenResolv installed with Wireguard)
```

```
#sudo sed -i "s/#name_servers.*/name_servers=$IPV4_PUBLIC/" /etc/resolvconf.conf
sudo sed -i "s/#name_servers.*/name_servers=10.2.3.1/" /etc/resolvconf.conf
sudo sed -i "s/name_servers.*/name_servers=10.2.3.1/" /etc/resolvconf.conf
#sudo sed -i "s/#name_servers.*/name_servers=127.0.0.1/" /etc/resolvconf.conf
sudo resolvconf -u 1>/dev/null 2>&1

echo "backup up original volumes folder"
sudo mkdir -p ../mistborn_backup
sudo chmod 700 ../mistborn_backup
sudo tar -czf ../mistborn_backup/mistborn_volumes_backup.tar.gz ../mistborn_volumes 1>/dev/null 2>&1

# clean docker
echo "cleaning old docker volumes"
sudo systemctl stop Mistborn-base || true
sudo docker-compose -f /opt/mistborn/base.yml kill
sudo docker volume rm -f mistborn_production_postgres_data 2>/dev/null || true
sudo docker volume rm -f mistborn_production_postgres_data_backups 2>/dev/null || true
sudo docker volume rm -f mistborn_production_traefik 2>/dev/null || true
sudo docker volume prune -f 2>/dev/null || true

# clean Wireguard
echo "cleaning old wireguard services"
sudo ./scripts/env/wg_clean.sh

# start base service
sudo systemctl enable Mistborn-base.service
sudo systemctl start Mistborn-base.service
popd

figlet "Mistborn Installed"
echo "Watch Mistborn start: sudo journalctl -xfu Mistborn-base"
echo "Retrieve Wireguard default config for admin: sudo mistborn-cli getconf"
```

14-ANNEXES : Commande de gestion PiVPN

Voici les commandes principales pour la gestion de l'outil WireGuard via pivpn :
Tapez « PiVPN » puis :

-a, add	Créer la configuration d'un nouveau client au VPN
-c, clients	Liste les clients connecter au serveur
-d, debug	Lance une session de débogage afin de détecter une erreur et sa source
-l, list	Liste tous les clients configurés
-qr, qrcode	Fait apparaitre le QR code lié à une configuration cliente pour les applications mobile.
-r, remove	Supprime un client
-off, off	Désactive un utilisateur
-on, on	Active un utilisateur
-h, help	Lance la page d'aide
-u, uninstall	Désinstalle PiVPN de votre système d'exploitation
-up, update	Met à jour les scripts de PiVPN
-bk, backup	Effectue une sauvegarde des configurations des client