WireGuard, le VPN simple et efficace

CHRISTOPHE BROCAS

LES LOGICIELS DE SÉCURITÉ PEUVENT ÊTRE COMPLEXES À COMPRENDRE ET LOURDS À METTRE EN ŒUVRE. CELA PEUT NOUS FREINER DANS L'USAGE ET LE DÉPLOIEMENT D'ARCHITECTURES SÉCURISÉES. MAIS LE LOGICIEL WIREGUARD VEUT NOUS RÉCONCILIER AVEC LES SOLUTIONS DE SÉCURITÉ. IL A POUR BUT DE CRÉER UN ACCÈS VPN PERFORMANT ENTRE DEUX MACHINES ET NE PROPOSE QUE PEU DE CHOIX DE CONFIGURATION. EN EFFET, SON DÉVELOPPEUR, JASON A. DONENFELD A PRIS LE PARTI DE N'IMPLÉMENTER QU'UN JEU RESTREINT DE PROTOCOLES CRYPTOGRAPHIQUES, TOUS À L'ÉTAT DE L'ART, ET DE NE FOURNIR À L'UTILISATEUR QU'UNE SIMPLE INTERFACE RÉSEAU. CES CHOIX PERMETTENT DE MASQUER BEAUCOUP DE LA COMPLEXITÉ SOUS-JACENTE DU VPN DERRIÈRE CETTE ABSTRACTION. ET EN PLUS DE CETTE SIMPLICITÉ DE CONFIGURATION, WIREGUARD AFFIRME AUSSI APPORTER UN HAUT NIVEAU DE PERFORMANCE. JE VOUS PROPOSE DE VÉRIFIER ENSEMBLE SI CETTE PROMESSE EST BIEN TENUE !

1. UN VPN, DIANTRE, MAIS POURQUOI FAIRE ?

Excellente question ! En effet, nous entendons souvent parler de VPN, mais avant d'en installer un, voyons si nous en avons vraiment besoin.

Avant d'être un logiciel, un *Virtual Private Network* (VPN) est une architecture réseau permettant d'établir une communication chiffrée et sécurisée entre deux machines et au-delà de ces machines, entre les deux réseaux de part et d'autre de ce tunnel chiffré. Cette architecture est souvent basée sur le modèle client-serveur, mais pas tout le temps.

Les cas d'usage les plus fréquents sont les suivants :

- l'usage professionnel : après s'être connecté au Wi-Fi de son hôtel, un professionnel en déplacement va se connecter au serveur VPN de son entreprise. Cela va établir, entre son ordinateur et le serveur en question, un tunnel chiffré dans lequel tout le trafic réseau sortant de la machine du professionnel transitera. La personne en question sera alors sûre qu'aucune information sortant de sa machine ne sera accessible à quelqu'un qui scruterait le réseau (requêtes DNS, trafic web non sécurisé, messages électroniques en clair, etc.). La connexion VPN lui permet ainsi d'accéder aux services et serveurs disponibles sur son réseau interne d'entreprise (serveurs intranet, applicatifs métiers, messagerie, etc.), mais aussi à son proxy de navigation web.
- l'usage pour le consommateur : au-delà de la communication sécurisée décrite dans le paragraphe précédent, un VPN peut vous aider à accéder à des services inaccessibles depuis votre domicile ou votre smartphone en France. En effet, il faut comprendre que si votre trafic web passe au travers d'un serveur VPN localisé par exemple aux États-Unis, alors, l'adresse IP que verront les services web auxquels vous accéderez sera celle du serveur VPN. Et dans le cas d'un serveur VPN aux USA, ces services web verront que votre adresse IP est américaine. Cela peut, par exemple, vous servir à avoir accès à des services uniquement accessibles aux machines américaines.
- l'usage pour le citoyen ou le militant : nous avons compris qu'après avoir établi la communication entre votre poste et le serveur VPN, tout votre trafic réseau part de manière chiffrée vers ce serveur. Ainsi, plus aucune inspection ni manipulation du trafic n'est possible entre ces deux machines. Dans le cas d'un citoyen d'un pays autoritaire, voire pire, si le serveur est hors de portée de l'État en question, le citoyen peut être sûr qu'il ne pourra pas être surveillé ni censuré par une surveillance de masse étatique. Des attaques ciblées peuvent toujours être opérées contre votre ordinateur ou votre smartphone, mais dans ce cas, nous sortons du modèle de menace auquel répond le VPN. En un mot, si vous êtes dans ce cas, courez !

En résumé, le VPN peut vous permettre :

- d'accéder à Internet avec la certitude que votre trafic réseau n'est ni observable ni modifiable entre votre machine et votre serveur VPN ;
- d'accéder de manière sécurisée à votre réseau d'entreprise et donc à des services et des ressources non directement raccordés à Internet ;
- et de pouvoir cacher votre adresse IP réelle. Cette caractéristique n'est pas une assurance tout risque, mais un premier pas vers plus de discrétion. Si tel est votre objectif, l'usage de réseau de type Tor et de navigateurs durcis (blocage du JavaScript, etc.) vous permettra d'avoir plus d'assurance sur ce dernier point.

Maintenant que vous avez tout cela en main, pensez-vous être intéressé par la découverte d'un VPN simple et efficace ? Oui ? Alors, jetons un regard curieux et enthousiaste sur WireGuard.

2. WIREGUARD : LES CONCEPTS

WireGuard, avant d'être un outil, est tout d'abord un protocole de communication. Son auteur l'a décrit et documenté de manière ouverte.

L'objectif est de faciliter la création d'implémentations sur de multiples

plateformes, dans des langages différents et avec des choix d'architecture qui peuvent différer de l'implémentation de référence. Par exemple, un développeur peut choisir de en pas utiliser de module noyau et de s'orienter vers une implémentation uniquement en espace utilisateur.

Décrivons rapidement quelques points clés de WireGuard :

- il s'agit d'un protocole de communication de niveau 3 (même niveau qu'IP) ;
- il utilise UDP comme transport ;
- il peut établir une communication de pair à pair entre les deux machines établissant le lien VPN tout comme une implémentation de type client-serveur ;
- l'authentification des paquets émis par les machines repose sur le principe des clés SSH (chaque machine dispose d'un jeu de clés privée et publique) ;
- le protocole est discret : aucun paquet non authentifié ne recevra de réponse ;
- il implémente un jeu restreint d'algorithmes cryptographiques qui sont à l'état de l'art. Quand ceux-ci seront mis en défaut par une attaque, ils seront changés pour des alternatives plus sécurisées ;
- un point important est que le protocole ne comporte aucune phase de négociation sur ces algorithmes entre les deux machines. Ainsi, on coupe court à tout risque d'attaque de type « négociation à la baisse » vers des algorithmes moins sûrs ;
- enfin, WireGuard est basé sur l'implémentation d'une interface réseau de type WireGuard (i.e. wg0). Du coup, tout ce que vous faites habituellement sur les interfaces réseau (filtrage, routage) est faisable sur l'interface réseau dédiée au VPN et fournie par WireGuard.

3. WIREGUARD : LA MISE EN ŒUVRE

La configuration de notre test est basée sur un serveur core i3 tournant sous Debian 9 hébergé dans un datacenter OVH en France et sur un ordinateur portable tournant sous Ubuntu 19.04 raccordé à Internet via la fibre. Je précise cela pour donner le contexte technique des mesures de performance données dans le dernier paragraphe de l'article.

3.1 Réseau

Lorsque le VPN sera activé sur l'ordinateur portable, nous allons choisir de router tout le trafic sortant de l'ordinateur au travers du tunnel VPN.

Voici ce que cela donne au travers d'un schéma :

Il est à noter que ce n'est pas du tout une obligation et on pourrait router dans ce tunnel que le trafic à destination des adresses privées des machines appartenant au VPN.

En effet, il est important de comprendre qu'un VPN est un réseau privé virtuel au sein duquel tous les ordinateurs qui y évoluent ont une adresse IP privée appartenant à ce VPN. Pour notre architecture VPN, nous allons mettre en place la plage d'adresses IP privées suivante : 10.200.200.0 → 10.200.200.255, soit le sous-réseau 10.200.200.0/24.

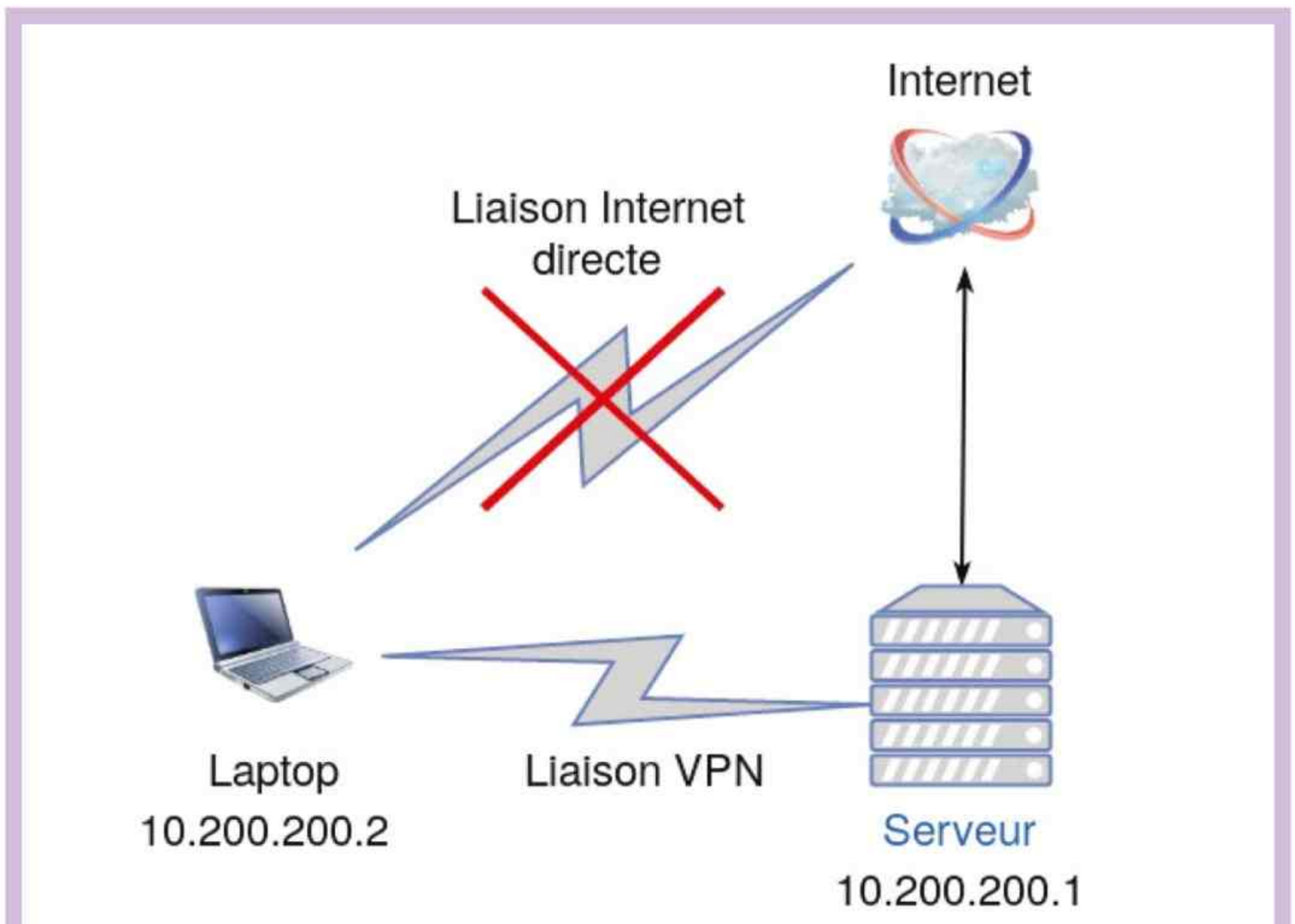

FIGURE 1. Architecture de notre VPN.

3.2 Configuration du serveur VPN

Nous allons commencer par installer WireGuard sur notre serveur. Il s'agit dans notre cas d'un serveur sous Debian 9 (aka Stretch). Il est à noter que toutes les modifications qui vont être apportées au système et à la configuration de WireGuard exigent que vous ayez les droits d'administration sur votre système (sudo ou élévation de privilèges de type *root*).

NOYAU LINUX : UN PETIT POINT D'ATTENTION

Côté implémentation sous Linux, il faut attirer votre attention sur un point particulier : WireGuard implémente la majeure partie de son protocole en environnement noyau. Cela nécessite donc d'installer un module noyau. Et ce module, malgré les efforts de son développeur principal, n'est pas encore incorporé au sein des modules standards du noyau Linux.

Cela a un impact sur votre mise en œuvre en tant qu'utilisateur : il faut que les entêtes de votre noyau Linux soient installés sur les machines où vous installerez WireGuard afin de pouvoir procéder à la compilation du module WireGuard.

Ce point est notable notamment si vous utilisez des serveurs OVH/Kimsufi où le kernel est souvent par défaut un noyau custom d'OVH sans headers disponibles. Avoir le noyau Linux de votre distribution préférée, les entêtes et DKMS installés seront la bonne configuration de départ.

```
# sudo apt-get install linux-headers-
$(uname -kernel-release) dkms
```

Modifions tout d'abord nos dépôts et rafraîchissons :

```
# sudo echo "deb http://deb.debian.org/
debian/ unstable main" > /etc/apt/sources.
list.d/unstable-wireguard.list
# sudo printf 'Package: *\nPin: release
a=unstable\nPin-Priority: 200\n' > /etc/apt/
preferences.d/limit-unstable
# sudo apt update
```

Puis installons WireGuard :

```
# sudo apt install wireguard-dkms
wireguard-tools
[...]
Setting up wireguard-dkms (0.0.20190905-1)
...
Loading new wireguard-0.0.20190905 DKMS
files...
Building for 4.9.0-9-amd64
Building initial module for 4.9.0-9-amd64
Done.
wireguard:
Running module version sanity check.
 - Original module
   - No original module exists within this
kernel
 - Installation
   - Installing to /lib/modules/4.9.0-9-
amd64/updates/dkms/
depmod...
DKMS: install completed.
#
```

Nous noterons la compilation et l'installation du module noyau de WireGuard.

Vérifions ensuite avec la commande lsmod que le module est bien chargé en mémoire :

```
# lsmod | grep wireguard
wireguard              208896  0
ip6 _ udp _ tunnel          16384  1
wireguard
udp _ tunnel               16384  1 wireguard
#
```

Passons ensuite à la configuration de notre serveur VPN WireGuard.

On se rend donc dans le répertoire /etc/wireguard.

MISE À JOUR DU NOYAU LINUX

Le fait que le noyau ne soit pas encore intégré au noyau Linux standard a un impact lors de l'application de vos mises à jour tant côté serveur que côté client. Quand vous mettrez à jour votre noyau, vous devrez réinstaller les entêtes, ce qui permettra la recompilation du module noyau de WireGuard. Vous n'avez plus qu'à relancer le module via la commande modprobe :

```
# sudo apt install linux-headers-$(uname –kernel-release)
# sudo modprobe wireguard
```

Puis effectuons les étapes de configuration suivantes :

1) Mettons en place le relais de paquets qui permet au serveur VPN de relayer vers l'Internet (ou son réseau local) les paquets venant de ses clients et de traiter correctement les paquets retours. Pour cela, on décommente la ligne ad-hoc dans le fichier /etc/sysctl.conf et on force la prise en compte immédiate de la modification :

```
# sed -i 's/^#net.ipv4.ip _ forward=1/net.ipv4.ip _ forward=1/' /etc/sysctl.conf
# sysctl -p
```

2) Ouverture du pare-feu : nous allons utiliser le port par défaut de WireGuard (51820) et nous allons configurer notre pare-feu pour laisser passer le trafic UDP vers ce port. Pour cela, nous utilisons l'interface de configuration iptables et nous contrôlons sa prise en compte :

```
# iptables -A INPUT -p udp --dport 51820 -j ACCEPT
# itables -L
Chain INPUT (policy DROP)
target     prot opt source          destination
ACCEPT     all  --  anywhere        anywhere        ctstate
RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere        anywhere        tcp dpt:ssh
ACCEPT     tcp  --  anywhere        anywhere        tcp dpt:http
ACCEPT     tcp  --  anywhere        anywhere        tcp dpt:https
ACCEPT     udp  --  anywhere        anywhere        udp dpt:51820
ACCEPT     icmp --  anywhere        anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source          destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source          destination
```

Bien entendu, vous veillerez à sauvegarder ces nouvelles règles iptables afin qu'elles persistent après le prochain redémarrage du système. Les commandes iptables-save et iptables-restore permettent de sauvegarder la configuration iptables courante dans un fichier et de la restaurer. Le paquet iptables-persistent permet de réutiliser ce fichier à la relance de votre machine. Pour cela, le paquet vous demandera lors de son installation les fichiers de sauvegarde de vos règles (en général, il s'agit de /etc/iptables/rules.v4 et rules.v6) afin de pouvoir les restaurer au redémarrage du système.

3) Génération des clés privée et publique du serveur.

Comme indiqué plus haut, l'authentification des paquets émis par les machines se fait au travers de clés privées et publiques. Il faut donc les générer, respectivement ici sous les fichiers privatekey et publickey :

```
# wg genkey > privatekey
# chmod 600 privatekey
# wg pubkey < privatekey > publickey
```

4) Création du fichier de configuration.

Après avoir récupéré le nom de votre carte réseau portant votre adresse IP publique grâce à la commande ifconfig (dans mon cas, il s'agit de l'interface enol), nous allons créer le fichier de configuration /etc/wireguard/wg0.conf et le compléter avec une première partie de la configuration du serveur :


```
[Interface]
Address =  10.200.200.1/24
ListenPort = 51820
PostUp = iptables -A FORWARD -i wg0 -j
ACCEPT; iptables -t nat -A POSTROUTING -o
eno1 -j MASQUERADE
PostDown = iptables -D FORWARD -i wg0 -j
ACCEPT; iptables -t nat -D POSTROUTING -o
eno1 -j MASQUERADE
PrivateKey = COPIER _ ICI _ LA _ CLE _
PRIVEE _ DU _ SERVEUR
SaveConfig = true
```

Détaillons quelques-unes des options :

- `Address` : cela permet de fixer l'adresse IP du serveur au sein du VPN. Ici, ce sera 10.200.200.1.
  Point important : le masque `/24` permet de dire à WireGuard que les adresses du réseau VPN vont de 10.200.200.0 à 10.200.200.255.
- `PostUp`/`Down` : ces lignes permettent de mettre en place des règles iptables de translation d'adresses à l'activation du VPN et de les inactiver à son arrêt. Cela permet de router les paquets réseau venant des clients vers Internet quand c'est nécessaire.

Note : veillez à bien valoriser la configuration ci-dessus avec :

- le nom de votre interface réseau (comme dit plus haut, dans mon cas `eno1`) ;
- et le contenu de votre fichier `privatekey` généré à l'étape précédente.

On modifie ensuite les droits du fichier afin d'en assurer la confidentialité (lecture possible uniquement par l'utilisateur *root*) :

```
chmod 600 /etc/wireguard/wg0.conf
```

Puis nous activons le VPN via les commandes `wg-quick up wg0` et `wg show` :

```
# sudo wg-quick up wg0
[#] ip link add wg0 type wireguard
[#] wg setconf wg0 /dev/fd/63
[#] ip -4 address add 10.200.200.1/24 dev
wg0
[#] ip link set mtu 1420 up dev wg0
[#] iptables -A FORWARD -i wg0 -j ACCEPT;
iptables -t nat -A POSTROUTING -o eno1 -j
MASQUERADE

# sudo wg show
interface: wg0
  public key:
wboyvPpONx4mSlJJXcF5k8SyY1f9r0C3mOXip53iuiY=
  private key: (hidden)
  listening port: 51820
```

3.3 Configuration du client VPN

Nous installons ensuite WireGuard sur l'ordinateur portable tournant sous Ubuntu 19.04.

Nous allons :

- installer les entêtes du noyau et DKMS ;
- installer le dépôt de WireGuard ;
- installer le logiciel ;
- activer le module (vous pouvez aussi relancer la machine) ;
- et vérifier le bon lancement du module.

```
# sudo apt-get install linux-headers-
$(uname -kernel-release) dkms
# sudo add-apt-repository ppa:wireguard/
wireguard
# sudo apt-get update
# sudo apt-get install wireguard
# modprobe wireguard
# lsmod | grep wireguard
wireguard              204800  0
ip6 _ udp _ tunnel         16384  1
wireguard
udp _ tunnel               16384  1 wireguard
```

Comme vous le comprenez, l'installation côté client va beaucoup ressembler à celle à laquelle nous avons procédé sur le serveur.

Générons ensuite les clés :

```
# cd /etc/wireguard
# wg genkey > privatekey
# chmod 600 privatekey
# wg pubkey < privatekey > publickey
```

Créons et remplissons le fichier de configuration /etc/wireguard/wg0.conf :

```
[Interface]
Address = 10.200.200.2
DNS = 9.9.9.9
PrivateKey = COPIER _ ICI _ LA _ CLE _
PRIVEE _ DU _ CLIENT

[Peer]
PublicKey = COPIER _ ICI _ LA _ CLE _
PUBLIQUE _ DU _ SERVEUR
AllowedIPs = 0.0.0.0/0
Endpoint = COPIER _ ICI _ L _ ADRESSE _ IP _
DU _ SERVEUR:51820
```

La clé privée du client est dans le fichier /etc/wireguard/privatekey sur l'ordinateur portable.

La clé publique du serveur est dans le fichier /etc/wireguard/publickey sur le serveur.

Un point important : dans la variable AllowedIPs, nous avons mis la valeur 0.0.0.0/0. Cela indique que tout le trafic sortant du client sera envoyé dans le tunnel VPN. Nous aurions pu mettre la valeur 10.200.200.0/24 qui indiquerait que seul le trafic à destination des machines ayant une adresse IP privée dans le VPN (serveur et clients) serait envoyé dans le tunnel.

Corrigeons enfin les permissions du fichier de configuration :

```
# chmod 600 /etc/wireguard/wg0.conf
```

3.4 Finir de configurer le serveur VPN

La dernière étape de la configuration est de reporter les informations du client dans le fichier de configuration du serveur /etc/wireguard/wg0.conf. Nous ajoutons à la fin du fichier le paragraphe suivant qui décrit la configuration du client avec notamment sa clé publique :

```
[Peer]
PublicKey =  COPIER _ ICI _ LA _ CLE _
PUBLIQUE _ DU _ CLIENT
AllowedIPs = 10.200.200.2/32
```

Important : notez bien qu'il faut avoir stoppé le VPN avant de modifier le fichier si on ne veut pas voir sa modification écrasée par l'arrêt du VPN.

Si vous souhaitez ne pas générer de coupure du serveur VPN, vous pouvez utiliser la ligne de commandes suivante :

```
# sudo wg set wg0 peer COPIER _ ICI _ LA _
CLE _ PUBLIQUE _ DU _ CLIENT allowed-ips
ADRESSE _ IP _ DU _ CLIENT/32
```

3.5 Activation et désactivation de la connexion VPN

Il suffit de saisir la même commande tant du côté serveur que côté client :

```
# sudo wg-quick up wg0
```

Et de la même manière, vous pouvez arrêter la connexion VPN ainsi :

```
# sudo wg-quick down wg0
```

3.6 Test et performance de la connexion

Que ce soit côté serveur ou côté client, la commande wg show vous permet d'obtenir un point sur la configuration et des statistiques sur la connexion (dernier handshake et volume de données transférées). Vous avez ici la sortie de la commande lancée sur le client :

```
# sudo wg show
interface: wg0
  public key: CLE_PUBLIQUE_CLIENT
  private key: (hidden)
  listening port: 37857
  fwmark: 0xca6c

peer: CLE_PUBLIQUE_DU_SERVEUR
  endpoint: ADRESSE_IP_PUBLIQUE_DU_
SERVEUR:51820
  allowed ips: 0.0.0.0/0
  latest handshake: 16 seconds ago
  transfer: 171.46 MiB received, 7.33 MiB sent
```

Ce n'est pas très informatif et ce sont les seules informations auxquelles on a accès.

L'un des rares défauts de WireGuard est en effet le peu d'outils d'investigation dont on dispose en cas de souci de configuration initiale par exemple. Dans mon cas, j'avais fait une coquille dans l'adresse IP intra VPN du client et une autre dans iptables. Il a fallu lire et relire la configuration pour trouver les soucis en question.

Un simple ping du serveur (ping 10.200.200.1) depuis le client vous permettra de valider le bon fonctionnement ou non de votre connexion.

Une fois que nous avons vérifié que la connexion VPN était fonctionnelle, nous pouvons tester ensuite son niveau de performance en lançant 100 requêtes sur une URL sur Internet avec la commande ab.

```
# ab -n 100 https://www.brocas.org/
[...]
Benchmarking www.brocas.org (be
patient).....done

Document Length:        15762 bytes

Concurrency Level:      1
Time taken for tests:   18.499 seconds
Complete requests:      100
Failed requests:        0
Total transferred:      1602000 bytes
HTML transferred:       1576200 bytes
Requests per second:    5.41 [#/sec] (mean)
Time per request:       184.992 [ms] (mean)
Time per request:       184.992 [ms] (mean,
across all concurrent requests)
Transfer rate:          84.57 [Kbytes/sec]
received
```

```
Connection Times (ms)
              min  mean[+/-sd] median   max
[...]
Total:        147  185  49.1    177      619
```

Et voici les mêmes statistiques renvoyées par la commande si nous n'utilisons pas la connexion VPN :

```
# ab -n 100 https://www.brocas.org/
[...]
Benchmarking www.brocas.org (be patient).....done
[...]
Document Length:        15762 bytes

Concurrency Level:      1
Time taken for tests:   16.414 seconds
Complete requests:      100
Failed requests:        0
Total transferred:      1602000 bytes
HTML transferred:       1576200 bytes
Requests per second:    6.09 [#/sec] (mean)
Time per request:       164.141 [ms] (mean)
Time per request:       164.141 [ms] (mean, across
all concurrent requests)
Transfer rate:          95.31 [Kbytes/sec] received
[...]
Connection Times (ms)
              min  mean[+/-sd] median   max
[...]
Total:        128  164  23.9    159      235
```

Nous voyons que l'impact en termes de performances est des plus raisonnables.

Enfin, mon ressenti utilisateur à l'usage de la connexion VPN a été excellent :

- le téléchargement de gros fichiers type slides PDF ou vidéos de conférences est rapide ;
- la visualisation de vidéo en streaming est parfaitement fonctionnelle.

CONCLUSION

WireGuard est désormais configuré sur votre serveur et votre premier client. Vous n'avez plus qu'à utiliser ce VPN pour être en sécurité lors de vos déplacements tout en profitant de son excellent niveau de performances. Bonne continuation dans votre découverte de ce logiciel de sécurité réseau vraiment remarquable. ■