

# Apporter une IPv4 de datacenter sur un Linux via un VPN

L'objectif est de faire descendre une IP de datacenter sur un Linux avec un tunnel OpenVPN et du proxyARP.

Cette documentation existe aussi pour [pfSense](#).

Il vous sera nécessaire :

- un serveur OpenVPN linux avec:
  - Une IP fixe pour initier la session VPN
  - Une IP supplémentaire (nommé "IP Fail-Over" chez OVH par exemple)

Dans notre exemple, notre IP supplémentaire sera 172.32.0.1

## Configuration du serveur OpenVPN

La configuration d'OpenVPN est classique avec quelques exception, par exemple :

[proxyarp.conf](#)

```
mode server
tls-server
proto udp
port 1194
dev tap0
cipher AES-256-CBC
keepalive 10 30
persist-key
persist-tun
verb 3
status proxyarp_status.log
log-append /var/log/openvpn-proxyarp.log

ca /etc/openvpn/easy-rsa/keys/ca.crt
cert /etc/openvpn/easy-rsa/keys/server.crt
key /etc/openvpn/easy-rsa/keys/server.key
dh /etc/openvpn/easy-rsa/keys/dh4096.pem
tls-auth /etc/openvpn/easy-rsa/keys/ta.key 0
auth sha256
keysize 256
comp-lzo no

script-security 2
client-connect /etc/openvpn/proxy-arp-up.sh
```

```
client-disconnect /etc/openvpn/proxy-arp-down.sh
```

Vous noterez l'utilisation OBLIGATOIRE d'une interface TAP, l'absence de configuration réseau et l'ajout des trois lignes suivantes :

```
script-security 2
client-connect /etc/openvpn/proxyarp_up.sh
client-disconnect /etc/openvpn/proxyarp_down.sh
```

et d'ajouter dans le dossier `/etc/openvpn` les deux fichiers suivant (en les adaptant) :

[proxyarp\\_up.sh](#)

```
#!/bin/bash

echo '1' > /proc/sys/net/ipv4/conf/all/proxy_arp
ifconfig tap0 up
ip route add 172.32.0.1 dev tap0
```

et

[proxyarp\\_down.sh](#)

```
#!/bin/bash

ip route del 172.32.0.1 dev tap0
ifconfig tap0 down
```

et pour finir de les rendre exécutable :

```
# chmod +x proxyarp_up.sh proxyarp_down.sh
```

## Configuration du client OpenVPN Linux

On va créer un client OpenVPN sur le client Linux, si on suit l'exemple plus haut :

[proxyarp.conf](#)

```
tls-client
proto udp
proto udp6
port 1194
remote XXXXXX
dev tap0
cipher AES-256-CBC
keepalive 10 30
```

```
persist-key
persist-tun
verb 3
status proxyarp_status.log
log-append /var/log/openvpn-proxyarp.log

ca /etc/openvpn/keys/ca.crt
cert /etc/openvpn/keys/client.crt
key /etc/openvpn/keys/client.key
tls-crypt /etc/openvpn/keys/ta.key 0
auth sha512
keysize 256
comp-lzo no

script-security 2
up /etc/openvpn/proxyarp-up.sh
down /etc/openvpn/proxyarp-down.sh
```

Avec pour même spécificité : l'interface TAP et pas de configuration de réseau.

[/etc/openvpn/proxyarp-up.sh](#)

```
#!/bin/bash

# Configuration de l'interface
ip addr add 172.32.0.1/32 dev tap0

# Ajout des règles dans la table wan_vpn
ip rule add from 87.98.190.83 table wan_vpn
ip rule add fwmark 1 table wan_vpn

# Ajout la passerelle par défaut
ip route add 51.255.37.1 src 87.98.190.83 dev tap0 table wan_vpn
ip route add default via 51.255.37.1 src 87.98.190.83 dev tap0 table wan_vpn

# Ajout règles de firewall pour que le trafic entrant par le tunnel,
soit re-routé dans le tunnel
iptables -t mangle -A PREROUTING -i tap0 -j CONNMARK --set-xmark 0x1
iptables -t mangle -A PREROUTING -i eth0 -m connmark --mark 0x1 -j CONNMARK --restore-mark
```

[/etc/openvpn/proxyarp-down.sh](#)

```
#!/bin/bash

# Configuration de l'interface
ip addr del 172.32.0.1/32 dev tap0
```

```
# Suppression des règles dans la table wan_vpn
ip rule del from 87.98.190.83 table wan_vpn
ip rule del fwmark 1 table wan_vpn

# Suppression de la passerelle par défaut
ip route del 51.255.37.1 src 87.98.190.83 dev tap0 table wan_vpn
ip route del default via 51.255.37.1 src 87.98.190.83 dev tap0 table wan_vpn

# Suppression des règles de firewall pour que le trafic entrant par le tunnel, soit re-routé dans le tunnel
iptables -t mangle -D PREROUTING -i tap0 -j CONNMARK --set-xmark 0x1
iptables -t mangle -D PREROUTING -i eth0 -m connmark --mark 0x1 -j CONNMARK --restore-mark
```

Puis il faudra les rendre exécutable :

```
# chmod +x /etc/openvpn/proxyarp-up.sh /etc/openvpn/proxyarp-down.sh
```

et créer la table de routage **wan\_vpn**

```
# echo "1 wan_vpn" >> /etc/iproute2/rt_tables
```

Et maintenant, ça doit fonctionner !

## Informations optionnelle

Si vous souhaitez faire du NAT de port entrant, voici un exemple de règle :

```
# iptables -t nat -A PREROUTING -i tap0 -p tcp -m tcp --dport 25 -j DNAT --to-destination 192.168.1.10:25
```

Et si vous voulez faire du NAT sortant, il faut mettre les deux règles suivante :

```
# iptables -t mangle -A PREROUTING -s 192.168.1.10/32 -j MARK --set-xmark 0x1
```

```
# iptables -t nat -A POSTROUTING -s 192.168.1.10/32 -j SNAT --to-source 172.32.0.1
```

From:  
<https://wiki.virtit.fr/> - VirtIT

Permanent link:  
[https://wiki.virtit.fr/doku.php/kb:linux:generalites:apporter\\_une\\_ipv4\\_de\\_datacenter\\_sur\\_un\\_linux\\_via\\_un\\_vpn?rev=1570701116](https://wiki.virtit.fr/doku.php/kb:linux:generalites:apporter_une_ipv4_de_datacenter_sur_un_linux_via_un_vpn?rev=1570701116)

Last update: 2019/10/10 09:51

