

Configuration de strongswan ipsec avec un routeur cisco

configuration de base

I-configuration cisco

1-Gateway parameters (Live)

VPN gateway device : Cisco ISR 3845

VPN Gateway device IP (or public ip) :41.191.70.9

Encryption Domain/ Network/ Host (or subnetip) :10.177.64.58 and 10.177.64.131

Local/remote TCP Ports : HTTPS (443)

2 -Tunnel Properties

- Phase 1:

Exchange Mode:Main

Encryption Schema : ikev1

Authentication Method :PRESHARED (via sms ou appel)

Encryption : AES-256

Hash : SHA-1

Diffie-Hellman Group :5

Lifetime (Seconds) :36000

- Phase 2:

Encryption : AES-256

Hash : SHA-1

Perfect Forward Secrecy (or pfs): YES

Diffie-Hellman Group :5

Lifetime (Seconds) :24000

II- ubuntu 20 configuration

public ip : 160.154.66.19

subnet ip : 192.168.2.110 and 192.168.2.121

fin de la configuration de base

installation et configuration de strongswan

strongSwan is an open-source, cross-platform, full-featured and widely-used IPsec-based VPN (Virtual Private Network) implementation that runs on Linux, FreeBSD, OS X, Windows, Android, and iOS. It is primarily a keying daemon that supports the Internet Key Exchange protocols (IKEv1 and IKEv2) to establish security associations (SA) between two peers.

This article describes how to set up a site-to-site IPSec VPN gateways using strongSwan on Ubuntu and Debian servers. By site-to-site we mean each security gateway has a sub-net behind it. Besides, the peers will authenticate each other using a pre-shared key (PSK).

Step 1: Enabling Kernel Packet Forwarding

1. First, you need to configure the kernel to enable packet forwarding by adding the appropriate system variables in `/etc/sysctl.conf` configuration file on both security gateways.

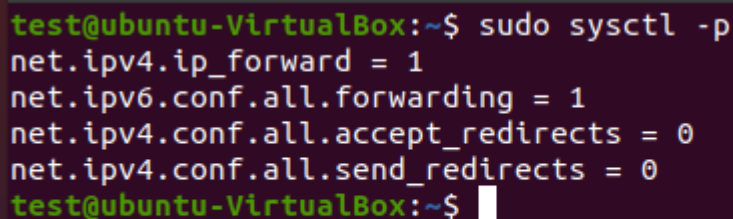
#- sudo nano /etc/sysctl.conf

Look for the following lines and uncomment them and set their values as shown (read comments in the file for more information).

```
net.ipv4.ip_forward = 1
net.ipv6.conf.all.forwarding = 1
net.ipv4.conf.all.accept_redirects = 0
net.ipv4.conf.all.send_redirects = 0
```

2. Next, load the new settings by running the following command.

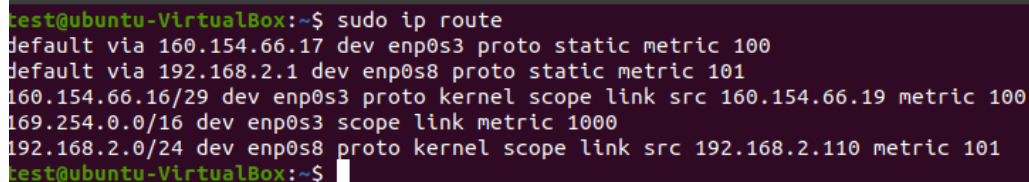
#- sudo sysctl -p



```
test@ubuntu-VirtualBox:~$ sudo sysctl -p
net.ipv4.ip_forward = 1
net.ipv6.conf.all.forwarding = 1
net.ipv4.conf.all.accept_redirects = 0
net.ipv4.conf.all.send_redirects = 0
test@ubuntu-VirtualBox:~$
```

3. If you have a UFW firewall service enabled, you need to add the following rules to the `/etc/ufw/before.rules` configuration file just before the filter rules in either security gate

#-sudo ip route pour voir les routes au niveau de mes adresses



```
test@ubuntu-VirtualBox:~$ sudo ip route
default via 160.154.66.17 dev enp0s3 proto static metric 100
default via 192.168.2.1 dev enp0s8 proto static metric 101
160.154.66.16/29 dev enp0s3 proto kernel scope link src 160.154.66.19 metric 100
169.254.0.0/16 dev enp0s3 scope link metric 1000
192.168.2.0/24 dev enp0s8 proto kernel scope link src 192.168.2.110 metric 101
test@ubuntu-VirtualBox:~$
```

nb : ici on va utiliser notre carte qui se trouve au niveau du dev (c'est notre passerelle de l'ip public de ubuntu que nous avons communiquer)

Donc on va utiliser la carte **enp0s3**

#- **sudo nano /etc/ufw/before.rules**

avant la ligne ligne ***règle** ajouter:

```
*nat
-A POSTROUTING -s 10.177.64.131/32 -o enp0s3 -m policy --pol ipsec --dir out -j ACCEPT

-A POSTROUTING -s 10.177.64.131/32 -o enp0s3 -j MASQUERADE

COMMIT

*mangle
-A FORWARD --match policy --pol ipsec --dir in -s 10.177.64.131/32 -o enp0s3 -p tcp -m tcp --tcp-flags SYN,RST SYN -m tcpmss --mss 1361:1536 -j TCPMSS --set-mss 1360

COMMIT
```

```
*nat
-A POSTROUTING -s 10.177.64.131/32 -o enp0s3 -m policy --pol ipsec --dir out -j ACCEPT
-A POSTROUTING -s 10.177.64.131/32 -o enp0s3 -j MASQUERADE
COMMIT

*mangle
-A FORWARD --match policy --pol ipsec --dir in -s 10.177.64.131/32 -o enp0s3 -p tcp -m tcp --tcp-flags SYN,RST SYN -m tcpmss --mss 1361:1536 -j TCPMSS --set-mss 1360
COMMIT

#*nat
#-A POSTROUTING -s 192.168.2.110 -d 10.177.64.131 -m policy --pol ipsec --dir out -j ACCEPT
#-A POSTROUTING -s 192.168.2.110 -d 10.177.64.131 -j MASQUERADE
#COMMIT

#*mangle
#-A FORWARD --match policy --pol ipsec --dir in -s 192.168.2.110 -d 10.177.64.131 -p tcp -m tcp --tcp-flags SYN,RST SYN -m tcpmss --mss 1361:1536 -j TCPMSS --set-mss 1360
#COMMIT
# Don't delete these required lines, otherwise there will be errors
*filter
:ufw-before-input - [0:0]
:ufw-before-output - [0:0]
:ufw-before-forward - [0:0]
:ufw-not-local - [0:0]
# End required lines
```

nb : n'oubliez pas de remplacer enps03 par le nom de votre carte reseau

Après la ligne des *règles , ajouter cette ligne :

```
-A ufw-before-forward --match policy --pol ipsec --dir in --proto esp -s 10.177.64.131/32 -j ACCEPT
-A ufw-before-forward --match policy --pol ipsec --dir out --proto esp -d 10.177.64.131/32 -j ACCEPT
```

```
*filter
:ufw-before-input - [0:0]
:ufw-before-output - [0:0]
:ufw-before-forward - [0:0]
:ufw-not-local - [0:0]
# End required lines
#ajouter par moi
-A ufw-before-forward --match policy --pol ipsec --dir in --proto esp -s 10.177.64.131/32 -j ACCEPT
-A ufw-before-forward --match policy --pol ipsec --dir out --proto esp -d 10.177.64.131/32 -j ACCEPT
```

4. Once firewall rules have been added, then apply the new changes by restarting **UFW** as shown.

```
# - sudo ufw disable
# - sudo ufw enable
```

Step 2: Installing strongSwan in Debian and Ubuntu

5. Update your package cache on both security gateways and install the **strongswan** package using the [APT package manager](#).

```
# - sudo apt update
# - sudo apt install strongswan
```

6. Once the installation is complete, the installer script will start the **strongswan** service and enable it to automatically start at system boot. You can check its status and whether it is enabled using the following command.

```
#~ sudo systemctl status strongswan.service
#~ sudo systemctl is-enabled strongswan.service
```

Step 3: Configuring Security Gateways

7. Next, you need to configure the security gateways using the `/etc/ipsec.conf` configuration file.

```
#~ sudo cp /etc/ipsec.conf /etc/ipsec.conf.orig
```

```
#~sudo nano /etc/ipsec.conf
```

```
config setup
    charondebug="all"
    uniqueids=yes
    strictcrpolicy=no

conn cerco-to-moov
    authby=secret
    left=%defaultroute
    leftid=160.154.66.19
    leftsubnet=192.168.2.110/32
    right=41.191.70.9
    rightid=41.191.70.9
    rightsubnet=10.177.64.131/32
    ike=aes256-sha1-modp1536!
    esp=aes256-sha1-modp1536
    keyingtries=%forever
    leftauth=psk
    rightauth=psk
    keyexchange=ikev1
    ikelifetime=36000s
    lifetime=24000s
```

```
dpddelay=60s
dpdtimeout=120s
dpdaction=restart
auto=add
type=tunnel
aggressive=no
```

```
conn second_address
    also=cerco-to-moov
    rightsubnet=10.177.64.58/32
```

```
config setup
    charondebug="all"
    uniqueids=yes
    strictcrpolicy=no

conn cerco-to-moov
    authby=secret
    left=%defaultroute
    leftid=160.154.66.19
    leftsubnet=192.168.2.110/32
    right=41.191.70.9
    rightid=41.191.70.9
    rightsubnet=10.177.64.131/32
    ike=aes256-sha1-modp1536!
    esp=aes256-sha1-modp1536
    keyingtries=%forever
    leftauth=psk
    rightauth=psk
    keyexchange=ikev1
    ikelifetime=36000s
    lifetime=24000s
    dpddelay=60s
    dpdtimeout=120s
    dpdaction=restart
    auto=add
    type=tunnel
    aggressive=no

conn second_address
    also=cerco-to-moov
    rightsubnet=10.177.64.58/32
```

Voici la signification de chaque paramètre de configuration :

- **config setup** - spécifie les informations de configuration générales pour IPSec qui s'appliquent à toutes les connexions.
- **charondebug** - définit combien de sortie de débogage Charon doit être enregistrée.
- **uniqueids** - spécifie si un identifiant de participant particulier doit rester unique.
- **conn prodgateway-to-devgateway** - définit le nom de la connexion.
- **type** - définit le type de connexion.
- **auto** - comment gérer la connexion lorsque IPSec est démarré ou redémarré.
- **keyexchange** - définit la version du protocole IKE à utiliser.
- **authby** - définit comment les pairs doivent s'authentifier.
- **left** - définit l'adresse IP de l'interface de réseau public du participant de gauche.
- **leftsubnet** - indique le sous-réseau privé derrière le participant de gauche.
- **right** - spécifie l'adresse IP de l'interface de réseau public du participant droit.
- **rightsubnet** - indique le sous-réseau privé derrière le participant de gauche.
- **ike** - définit une liste d'algorithmes de cryptage/authentification IKE/ISAKMP SA à utiliser. Vous pouvez ajouter une liste séparée par des virgules.
- **esp** - définit une liste d'algorithmes de cryptage/authentification ESP à utiliser pour la connexion. Vous pouvez ajouter une liste séparée par des virgules.
- **agressif** - indique s'il faut utiliser le mode agressif ou principal.
- **keyingtries** - indique le nombre de tentatives à effectuer pour négocier une connexion.
- **ikelifetime** - indique combien de temps le canal de saisie d'une connexion doit durer avant d'être renégocié.
- **durée de vie** - définit combien de temps une instance particulière d'une connexion doit durer, de la négociation réussie à l'expiration.
- **dpddelay** - spécifie l'intervalle de temps avec lequel les messages R_U_THERE/échanges d'INFORMATION sont envoyés à l'homologue.
- **dpdtimeout** - spécifie l'intervalle de temporisation, après lequel toutes les connexions à un pair sont supprimées en cas d'inactivité.
- **dpdaction** - définit comment utiliser le protocole Dead Peer Detection (DPD) pour gérer la connexion.

nb : pour plus d'info sur la commande ipsec : #~ [man ipsec.com](http://man.ipsec.com)

Étape 4 : Configuration de PSK pour l'authentification d'égal à égal

Ajoutez le **PSK** généré et qui vous a été envoyé dans le fichier **/etc/ipsec.secrets** sur les deux passerelles.

#~sudo nano /etc/ipsec.secrets

structure :

ubuntu_public cisco_public : PSK "key"

```
# which knows the public part
160.154.66.19 41.191.70.9 : PSK "#M6CwBpeNACeRc0ci10Jr25@n2021&PsTvPSKhtkgc0"
```

Redémarrez le programme IPSec et vérifiez son état pour afficher les connexions.

#~ sudo ipsec restart

#~sudo ipsec up 'ajouter le nom de la connection configurer dans ipsec.conf'

```
test@ubuntu-VirtualBox:~$ sudo ipsec restart
Stopping strongSwan IPsec...
Starting strongSwan 5.6.2 IPsec [starter]...
test@ubuntu-VirtualBox:~$ sudo ipsec up cerco-to-moov
Initiating Main Mode IKE_SA cerco-to-moov[1] to 41.191.70.9
generating ID_PROT request 0 [ SA V V V V V ]
sending packet: from 160.154.66.19[500] to 41.191.70.9[500] (180 bytes)
received packet: from 41.191.70.9[500] to 160.154.66.19[500] (104 bytes)
parsed ID_PROT response 0 [ SA V ]
received NAT-T (RFC 3947) vendor ID
generating ID_PROT request 0 [ KE No NAT-D NAT-D ]
sending packet: from 160.154.66.19[500] to 41.191.70.9[500] (308 bytes)
received packet: from 41.191.70.9[500] to 160.154.66.19[500] (368 bytes)
parsed ID_PROT response 0 [ KE No V V V V NAT-D NAT-D ]
received Cisco Unity vendor ID
received DPD vendor ID
received unknown vendor ID: d4:89:42:0d:7d:be:19:b0:8f:3c:4e:7f:6f:25:f4:5c
received XAuth vendor ID
generating ID_PROT request 0 [ ID HASH N(INITIAL_CONTACT) ]
sending packet: from 160.154.66.19[500] to 41.191.70.9[500] (108 bytes)
received packet: from 41.191.70.9[500] to 160.154.66.19[500] (76 bytes)
parsed ID_PROT response 0 [ ID HASH ]
IKE_SA cerco-to-moov[1] established between 160.154.66.19[160.154.66.19]...41.191.70.9[41.191.70.9]
scheduling reauthentication in 35458s
maximum IKE_SA lifetime 35998s
generating QUICK_MODE request 1941457569 [ HASH SA No KE ID ID ]
sending packet: from 160.154.66.19[500] to 41.191.70.9[500] (380 bytes)
received packet: from 41.191.70.9[500] to 160.154.66.19[500] (92 bytes)
parsed INFORMATIONAL_V1 request 727385551 [ HASH N((24576)) ]
received (24576) notify
received packet: from 41.191.70.9[500] to 160.154.66.19[500] (396 bytes)
parsed QUICK_MODE response 1941457569 [ HASH SA No KE ID ID N((24576)) ]
CHILD_SA cerco-to-moov[1] established with SPIs cfa0f02f_i f948027e_o and TS 192.168.2.110/32 === 10.177.64.131/32
generating QUICK_MODE request 1941457569 [ HASH ]
sending packet: from 160.154.66.19[500] to 41.191.70.9[500] (60 bytes)
connection 'cerco-to-moov' established successfully
```

La connection à bien été **établie**

#~ sudo ipsec statusall (vérifie le statut de toute vos connections up)

```
test@ubuntu-VirtualBox:~$ sudo ipsec statusall
Status of IKE charon daemon (strongswan 5.6.2, Linux 5.4.0-91-generic, x86_64):
  uptime: 2 minutes, since Dec 03 12:37:54 2021
  malloc: sbrk 3276800, mmap 532400, used 1420256, free 1856544
  worker threads: 11 of 16 idle, 5/0/0/0 working, job queue: 0/0/0/0, scheduled: 3
  loaded plugins: charon test-vectors unbound ldap pkcs11 tpm aes rc2 sha2 sha1 md4 md5 mgf1 random nonce x509 revocation constraints acert pubkey pkcs1 pkcs7 pkcs8 pkcs12 pgp dnskey sshkey dnscert
  identity eap-sin eap-sin-pksc eap-aka eap-aka-3pp2 eap-slnaka-pseudonym eap-slnaka-reauth eap-md5 eap-gtc eap-mschapv2 eap-dynanac eap-radius eap-tls eap-tls eap-peap eap-tnc xauth-generic xauth-
  auth-pan xauth-noauth tnc tncs-20 tncs-11 tncs-dynanac dhcp whitelst lookip error-notify certexpire led radattr addrblock unity counters
Listening IP addresses:
  160.154.66.19
  192.168.2.110
  fd17:625c:f037:a802:8d74:1bdd:9c54:4151
  fd17:625c:f037:a802:27ee:efc2:cc0:bc42
Connections:
cerco-to-moov: Many...41.191.70.9 IKEv1, dpddelay=60s
cerco-to-moov: local: [160.154.66.19] uses pre-shared key authentication
cerco-to-moov: remote: [41.191.70.9] uses pre-shared key authentication
cerco-to-moov: child: 192.168.2.110/32 === 10.177.64.131/32 TUNNEL, dpdaction=restart
second_address: child: 192.168.2.110/32 === 10.177.64.58/32 TUNNEL, dpdaction=restart
Security Associations (1 up, 0 connecting):
cerco-to-moov[1]: ESTABLISHED 2 minutes ago, 160.154.66.19[160.154.66.19]...41.191.70.9[41.191.70.9]
cerco-to-moov[1]: IKEv1 SPIs: 227fb9e671ccad2d, 214ee5107dbf19b0_r, pre-shared key reauthentication in 9 hours
cerco-to-moov[1]: IKE proposal: AES_CBC_256/HMAC_SHA1_96/PRF_HMAC_SHA1/MOOP_1536
cerco-to-moov[1]: INSTALLED, TUNNEL, reqid 1, ESP SPIs: cfa0f02f_i f948027e_o
cerco-to-moov[1]: AES_CBC_256/HMAC_SHA1_96/MOOP_1536, 0 bytes_i, 0 bytes_o, rekeying in 6 hours
cerco-to-moov[1]: 192.168.2.110/32 === 10.177.64.131/32
```

Nb : dans SA_CHILD vous voyez que les deux adresse privée sont connectées

#~ sudo ipsec status

```
test@ubuntu-VirtualBox:~$ sudo ipsec status
Security Associations (1 up, 0 connecting):
cerco-to-moov[1]: ESTABLISHED 5 minutes ago, 160.154.66.19[160.154.66.19]...41.191.70.9[41.191.70.9]
cerco-to-moov[1]: INSTALLED, TUNNEL, reqid 1, ESP SPIs: cfa0f02f_i f948027e_o
cerco-to-moov[1]: 192.168.2.110/32 === 10.177.64.131/32
```

Vérifions que les sous-réseaux communiquent :

#~ telnet -b 192.168.2.110 10.177.64.131 443

-b permet de spécifier l'adresse source

```
test@ubuntu-VirtualBox:~$ telnet -b 192.168.2.110 10.177.64.131 443
Trying 10.177.64.131...
Connected to 10.177.64.131.
Escape character is '^]'.

```

liens utile : <https://www.tecmint.com/setup-ipsec-vpn-with-strongswan-on-debian-ubuntu/>
<https://docs.netgate.com/pfsense/en/latest/troubleshooting/ipsec.html>