

Linux

Documentations liées à Linux, applications linux, serveurs...

- [Change color prompt](#)
- [Redimensionner disque lvm d'une VM](#)
- [SELINUX](#)
 - [Selinux manage port](#)
- [MSMTP](#)
- [DHCPD](#)
 - [Debian - Installation DHCP Serveur](#)
- [Backuppc](#)
 - [BackupPC Configuration](#)
- [Iptables](#)
 - [Memo Configuration](#)
- [SSH](#)
 - [Permissions for .ssh folder](#)
 - [Debian: Configuration serveur SFTP + chroot](#)
- [Desktop](#)
 - [Mojave Theme pour gnome](#)
 - [Chromium - color emojis](#)
- [Memo commandes](#)
- [Memo VIM](#)
- [nftables memo](#)
- [Wireguard - Mise en place](#)
- [Création VLANS - Debian](#)
- [Création service systemd](#)

- [Bash Memo](#)

Change color prompt

Procédure pour personnaliser le prompt Bash

- \u : Display the current username.
- \h : Display the hostname
- \W : Print the base of current working directory.
- @ : Display current time in 12-hour am/pm format

```
$ export PS1="[\u@\h \W \@]\$"  
[linodadmin@centos-01 ~ 01:50 PM]$
```

Pour le rendre Permanent copier-coller PS1 Value dans le .bashrc à la fin du fichier

Exemple (config srv)

```
## $ export PS1="[[\033[01;33m]\u@\h[\033[00m] ~ [\033[01;30m]\w[\033[00m]]  
$"
```

Docs:

→ [How to Change Bash Shell Prompt Colorful and Attractive in Linux](#)

→ [Change command line color in centos](#)

Redimensionner disque lvm d'une VM

Sur une VM, si on redimensionne le disque virtuel et qu'on veut agrandir une partition LVM sur le disque

On demande au système de scanner les disques, pour qu'il prenne en compte l'agrandissement du disque dur:

```
rescan-scsi-bus.sh
```

Il faut d'abord agrandir la partition sur laquelle le PV est créé (ici sda2) :

```
cfdisk /dev/sda2
```

Dans cfdisk, il doit afficher l'espace libre en faire, ajouté sur le disque.

Ensuite sélectionner /dev/sda2 et redimensionner la partition
Enfin, écrire les données et fermer cfdisk.

Via **fdisk -l**, on doit voir la partition sda2 agrandie.

Ensuite on agrandi le volume physique:

```
pverysize /dev/sda2
```

Ensuite on agrandit le volume logique, en ajoutant 10Giga
Pour voir le nom du LV, taper la commande **lvs**

```
lvresize -L +10G -r /dev/cl/root
```

Ou si on veut tout allouer l'espace du PV à notre LV:

```
lvresize -l +100%FREE /dev/cl/root
```

Il reste à redimensionner le système de fichier, pour que tout soit pris en compte:

```
resize2fs /chemin/disque/lvm # Dans le cas d'un système ext4
```

Agrandir le disque en recréant la table de partition

Il se peut qu'on ne puisse pas agrandir directement la partition contenant les données LVM: la partition étendue étant avant la LVM, et cfdisk ne peut pas agrandir le disque.

Dans ce cas, il faut utiliser fdisk, on va supprimer notre partition LVM et la recréer. L'idée étant de ne pas supprimer les données, mais de juste modifier la table de partition, il faudra que nos partitions commencent sur le même secteur et finissent sur un secteur plus loin. La partition sera la même e mais agrandie, donc nos données seront toujours présentes.

! Bien faire une backup au préalable, c'est toujours délicat de toucher aux tables de partitions

Configuration fdisk

Lancer fdisk sur le disque à modifier (ici /dev/sda):

```
fdisk /dev/sda
```

Détail des commandes fdisk (les détail de sont en commentaire):

```
Bienvenue dans fdisk (util-linux 2.38.1).
```

```
Les modifications resteront en mémoire jusqu'à écriture.
```

```
Soyez prudent avant d'utiliser la commande d'écriture.
```

```
Le disque est actuellement utilisé – le repartitionner est  
probablement une mauvaise idée.
```

```
Il est recommandé de démonter tous les systèmes de fichiers et désactiver (avec  
swapoff) toutes les partitions d'échange de ce disque.
```

```
Commande (m pour l'aide) : p # afficher les partitions et copier les valeurs dans un bloc-note
```

```
Disque /dev/sda : 40 GiB, 42949672960 octets, 83886080 secteurs
```

```
Modèle de disque : Virtual Disk
```

```
Unités : secteur de 1 × 512 = 512 octets
```

```
Taille de secteur (logique / physique) : 512 octets / 4096 octets
```

```
taille d'E/S (minimale / optimale) : 4096 octets / 4096 octets
```

```
Type d'étiquette de disque : dos
```

```
Identifiant de disque : 0x248fe134
```

Périphérique	Amorçage	Début	Fin	Secteurs	Taille	Id	Type
/dev/sda1	*	2048	1953791	1951744	953M	83	Linux

```
/dev/sda2          1953792  2148351   194560    95M 83 Linux
/dev/sda3          2150398 41209855 39059458  18,6G  5 Étendue
/dev/sda5          2150400 41209855 39059456  18,6G 8e LVM Linux
```

La partition 3 ne commence pas sur une frontière de cylindre physique.

Commande (m pour l'aide) : d # Suppression partition

Numéro de partition (1-3,5, 5 par défaut) : 3 # la n°3 (/dev/sda3)

La partition 3 a été supprimée.

Commande (m pour l'aide) : n # Créer un nouvelle partition

Type de partition

p primaire (2 primaire, 0 étendue, 2 libre)

e étendue (conteneur pour partitions logiques)

Sélectionnez (p par défaut) : e # Créer un nouvelle partition de type étendue

Numéro de partition (3,4, 3 par défaut) : 3

Premier secteur (2148352-83886079, 2148352 par défaut) : 2150398 # La nouvelle partition commence exactement au même endroit que celle qu'on vient de supprimer

Dernier secteur, +/-secteurs ou +/-taille{K,M,G,T,P} (2150398-83886079, 83886079 par défaut) :
on laisse par défaut, ce qui correspond à la fin du disque.

Commande (m pour l'aide) : p # On réaffiche la table de partition

Disque /dev/sda : 40 GiB, 42949672960 octets, 83886080 secteurs

Modèle de disque : Virtual Disk

Unités : secteur de 1 × 512 = 512 octets

Taille de secteur (logique / physique) : 512 octets / 4096 octets

taille d'E/S (minimale / optimale) : 4096 octets / 4096 octets

Type d'étiquette de disque : dos

Identifiant de disque : 0x248fe134

Périphérique	Amorçage	Début	Fin	Secteurs	Taille	Id	Type
/dev/sda1	*	2048	1953791	1951744	953M	83	Linux
/dev/sda2		1953792	2148351	194560	95M	83	Linux
/dev/sda3		2150398	83886079	81735682	39G	5	Étendue

La partition 3 ne commence pas sur une frontière de cylindre physique.

Commande (m pour l'aide) : n # on créé une nouvelle partition n°5

Tout l'espace des partitions primaires est utilisé.

Ajout de la partition logique 5

Premier secteur (2152446-83886079, 2152448 par défaut) :

Dernier secteur, +/-secteurs ou +/-taille{K,M,G,T,P} (2152448-83886079, 83886079 par défaut) :

Une nouvelle partition 5 de type « Linux » et de taille 39 GiB a été créée.

La partition #5 contient une signature ext4.

Voulez-vous supprimer la signature ? [O]ui/[N]on : o # Suppression signature

La signature sera supprimée par une commande d'écriture.

Commande (m pour l'aide) : x # passage en mode "avancé" de fdisk

Commande pour spécialistes (m pour l'aide) : b # on souhaite déplacer le début des données dans une partition

Numéro de partition (1-3,5, 5 par défaut) : 5 # pour la n°5

Nouveau début de données (2150399-83886079, 2152448 par défaut) : 2150400 # Début qu'on repositionne exactement au même endroit que l'initiale.

Commande pour spécialistes (m pour l'aide) : r # il reste juste à modifier le type, retour en mode normal.

Commande (m pour l'aide) : t # modifier le type d'une partition

Numéro de partition (1-3,5, 5 par défaut) : 5 # la n°5

Code Hexa ou synonyme (taper L pour afficher tous les codes) :L # on liste les types existan

...

lvm - 8E

linuxex - 85

...

Code Hexa ou synonyme (taper L pour afficher tous les codes) :8E # on veut une partition LVM linux comme initialement.

Type de partition « Linux » modifié en « Linux LVM ».

Commande (m pour l'aide) : p # On affiche l'état de nos partitions

Disque /dev/sda : 40 GiB, 42949672960 octets, 83886080 secteurs

Modèle de disque : Virtual Disk

Unités : secteur de $1 \times 512 = 512$ octets

Taille de secteur (logique / physique) : 512 octets / 4096 octets

taille d'E/S (minimale / optimale) : 4096 octets / 4096 octets

Type d'étiquette de disque : dos

Identifiant de disque : 0x248fe134

Périphérique	Amorçage	Début	Fin	Secteurs	Taille	Id	Type
/dev/sda1	*	2048	1953791	1951744	953M	83	Linux
/dev/sda2		1953792	2148351	194560	95M	83	Linux
/dev/sda3		2150398	83886079	81735682	39G	5	Étendue
/dev/sda5		2150400	83886079	81735680	39G	8e	LVM Linux

Commande (m pour l'aide) : w # Tout est bon On écrit les modifications sur le disque

La table de partitions a été altérée.

Impossible de mettre à jour les informations du système à propos de la partition 3:

Périphérique ou ressource occupé

Le noyau continue à utiliser les anciennes partitions. La nouvelle table sera utilisée lors du prochain démarrage.

Synchronisation des disques.

Il reste à redémarrer le système pour que ça soit pris en compte

```
reboot
```

Prise en compte dans LVM

Enfin, comme précédemment il faudra agrandir la partition LVM avec pvs et lvresize

```
pvresize /dev/sda5
lvextend -l +100%FREE /chemin/disque/lvm
```

File system

Il va rester à redimensionner le système de fichier posé sur la partition pour que lui aussi occupe tout l'espace :

```
resize2fs /chemin/disque/lvm
```


Liens

→ [Vidéo linuxtricks](#)

→ [Augmenter la taille d'un disque LVM \(pour l'agrandissement avec fdisk\)](#)

SELINUX

Selinux manage port

Sur une machine avec SELinux d'installé et d'activé, on ne peut associer un port réseau à un service, que s'il est autorisé par SELinux.

Par exemple, par défaut SELinux n'autorise que le port 22 à être utilisé par le serveur OpenSSH. Si jamais ce port est modifié (par exemple 2222), il ne sera plus possible de faire fonctionner SSHD.

Cette sécurité assure qu'un service n'utilise que le port autorisé par SELinux. Donc si jamais une personne malveillante ou un programme, venait à changer le port associé à SSH, HTTP... il ne pourrait pas le faire.

Pour ajouter un port associé à un service enregistré dans SELinux, par exemple SSH, il faut utiliser la commande suivante:

```
semanage port -a -t ssh_port_t -p tcp 2222
```

Cette modification est persistente après un redémarrage du système.

Ainsi, SSH pourra utiliser le port 2222, bien évidemment il faut penser à autoriser ce port dans le pare-feu.

Pour en savoir plus et aller plus loin:

→ [Use SELinux Port Labeling To Allow Services To Use Non-Standard Ports](#)

MSMTP

MSMTP est un client d'envoi de mail SMTP, simple à configurer et à installer Compatible avec sendmail, donc très utile pour l'envoi de mail en ligne de commande ou pour des scripts

Installation

```
sudo apt install msmtp msmtp-mta
```

Configuration

Soit on configure les parametres d'envoi SMTP (serveur, login, port...) dans /etc/msmtprc (accessible a tous les monde), sont on créer le fichier **.msmtp** dans le home de chaque Utilisateur

```
Vi ~/.msmtprc
```

```
account default
# Serveur SMTP
host smtp.monfai.fr
# Adresse mail expéditeur
from prenom.nom@monfai.fr
# Si pas de MDP
auth off
# SI MDP SMTP:
#auth on
#Nom Utilisateur
user prenom.nom
#si votre fai vous identifie plus clairement si vous reprecisez le domaine, alors plutôt cela
:
#user prenom.nom@fai.fr
password monmotdepasse
```

→ Ex config avec connexion SSL + MDP:

```
account default
host smtp.nomsrv.fr
```

```
port 465
auth login
tls on
tls_starttls off
tls_certcheck on
tls_trust_file /etc/ssl/certs/ca-certificates.crt
logfile ~/.msmtp.log
from lpognant@inforoutes.fr
user username
password cobian
```

→ Pour que seul l'utilisateur accède au fichier, et seul lui voie le mot de passe SMTP:

```
chmod 600 ~/.msmtpc`
```

→ Tester en ligne de commande l'envoi d'un mail:

```
printf "Subject:DeQuoiOnParle\nLeCorpsDuMessage" | msmtp johnny@liday.ch
```

Rediriger sendmail vers msmtp via un lien symbolique:

```
sudo ln -s /usr/bin/msmtp /usr/lib/sendmail
```

Liens

[Doc Ubuntu avec des exmples de configurations](#)

[Doc Officielle de msmtp](#)

DHCPD

Docs Serveur DHCP sous linux

DHCPD

Debian - Installation DHCP Serveur

Installation et configuration serveur DHCP sous debian

Installation

Installation package:

```
sudo apt-get install isc-dhcp-server
```

Configuration

Dans `/etc/dhcp/dhcpd.conf`

```
option domain-name "spices.org"; # Suffixe Dans
option domain-name-servers 213.191.92.86, 213.191.74.18; # Serveur DNS à distribuer

# Configuration de Base

subnet 192.168.1.0 netmask 255.255.255.0 { # Rxx et netmask
    range 192.168.1.1 192.168.1.10; # Plage DHCP
    option routers pepper.spices.org; # gateway
}
```

Appliquer la conf:

```
/etc/init.d/isc-dhcp-server restart # version nouvelle
/etc/init.d/dhcp3-server restart # version ancienne
```

Démarrer le serveur DHCP

```
sudo service isc-dhcp-server stop
sudo service isc-dhcp-server start
sudo ifdown eth0
sudo ifup eth0
```

Exemple d'un serveur DHCP simple

```
option domain-name "mydebian";
→ Utilisation du serveur DNS public de Google (ou bien utilisez l'adresse du serveur DNS
fournie par votre fournisseur d'accès):
option domain-name-servers 8.8.8.8, 8.8.4.4;
→ Configuration de votre sous-réseau (subnet) souhaité :
subnet 192.168.1.0 netmask 255.255.255.0 {
    range 192.168.1.101 192.168.1.254;
    option subnet-mask 255.255.255.0;
    option broadcast-address 192.168.1.255;
    option routers 192.168.1.100;
    option domain-name-servers home;
}
default-lease-time 600;
max-lease-time 7200;
→ Indique que nous voulons être le seul serveur DHCP de ce réseau :
authoritative;
```

Administration

→ Logs DHCP serveur:

```
more /var/log/syslog | grep dhcp
```

→DHCP Leases

```
more /var/lib/dhcpd/dhcpd.leases
```

Liens

→ [Doc WIKI Debian](#)

Backuppc

BackupPC Configuration

Docs Expliquant en detail la mise en place d'un serveur BackupPC et la configuration de la svg d'un poste/serveur sur ce dernier:

→ [Sauvegardes à distance avec BackupPC](#)

→ [Serveur de sauvegarde automatique : BackupPC](#)

Installation

```
sudo aptitude install backuppc libfile-rsyncp-perl
```

Copier config apache backuppc:

```
cp /etc/backuppc/apache.conf /etc/apache2/sites-available/backuppc.conf
```

Appliquer la configuration apache:

```
a2ensite backuppc  
systemctl restart apache2
```

Configuration

Serveur

- Changer le MDP de l'utilisateur **backuppc**, pour l'accès à l'interface web d'admin:

```
htpasswd /etc/backuppc/htpasswd backuppc
```

- Générer clé SSH sur le serveur avec l'utilisateur **backuppc**:
 - `sudo -i -u backuppc`
 - `ssh-keygen -t ed2551`

Ajouter une machine

- Via l'interface Web:

[backuppc_add_machine.png](#)

Client

Autoriser l'utilisateur **backup** à se connecter en ssh, en ajoutant le shell dans le fichier **passwd**:

```
backup:x:34:34:backup:/var/backups:/bin/sh
```

Il faut lui créer un dossier `.ssh` dans `/home/backup`, afin d'ajouter la clé ssh publique du serveur backuppc:

```
sudo vi /home/backup/.ssh/authorized_keys
```

Dans ce fichier, il faut copier-coller le contenu du fichier `id_rsa.pub`, dans `/home/backuppc.ssh`, du serveur backuppc.

Autoriser l'utilisateur backup à utiliser rsync, sans mdp avec sudo:

- Ajouter la ligne suivante dans le fichier `/etc/sudoers` ou en l'ouvrant avec `visudo`

```
backup ALL=NOPASSWD: /usr/bin/rsync
```

- Commande pour envoyer un mail de test avec BackupPC:

```
su -s /bin/sh backuppc -c '/usr/share/backuppc/bin/BackupPC_sendEmail -u user@domain.org'
```

Iptables

Memo Configuration

Memo commandes iptables

Accepter tout provenant d'une IP:

```
iptables -A INPUT -s 185.149.218.50 -j ACCEPT
iptables -A OUTPUT -d 185.149.218.50 -j ACCEPT
```

Accepter ICMP:

```
iptables -A OUTPUT -p icmp -m conntrack --ctstate NEW,ESTABLISHED,RELATED -j ACCEPT -> Tout
accepter
iptables -A INPUT -m conntrack --ctstate ESTABLISHED -j ACCEPT -> Juste ICMP déjà établit par
la machine
```

Refuser tout sauf ce qui est déjà autorisé (dernière règle) entrer:

```
iptables -P INPUT DROP
```

Accepter tout venant du réseau local:

```
iptables -A INPUT -s 192.168.0.0/24 -j ACCEPT
iptables -A OUTPUT -d 192.168.0.0/24 -j ACCEPT
```

Lister règles actives dans iptable:

```
iptables -L
```

Avec numéro de ligne (pratique quand on veut modifier ou supprimer une règle)

```
iptables -L --line-numbers
```

Supprimer une règle:

```
iptables -D INPUT/OUTPUT x (x= num ligne) ex: iptables -D INPUT 5
```

Ajouter une règle à la Xènieme position:

```
iptables -I INPUT/OUTPUT x règleaAppliquer ACCEPT -> ex iptables -I INPUT 3 -s 77.146.24.106 -p  
tcp -i etho0 --dport 80 -j ACCEPT
```

Remplacer une règle:

```
iptables -R INPUT/OUTPUT x règleaAppliquer ACCEPT -> ex iptables -R INPUT 3 -s 77.146.24.106 -p  
tcp -i etho0 --dport 80 -j ACCEPT
```

Sauvegarder configuration:

```
iptables-save -c  
netfilter-persistent save -> Pour prise en compte au redémarrage (installer iptables-  
persistent avant, apt-get install iptables-persistent)
```

ou

```
iptables-save -c >> /etc/iptables/rules.v4 (Pareil qui netfilter-persistent save)  
netfilter-persistent reload (Permet de tester si la config est bien sauvegardée)
```

Liens Utiles:

- ✓ [Doc Ubuntu](#)
- ✓ [Developpez Forum - Iptables](#)
- ✓ [ubuntu Forum](#)

SSH

SSH

Permissions for .ssh folder

Memo droits à appliquer aux différents fichiers dans le dossier .ssh

```
chmod 700 ~/.ssh  
chmod 644 ~/.ssh/authorized_keys  
chmod 644 ~/.ssh/known_hosts  
chmod 644 ~/.ssh/config  
chmod 600 ~/.ssh/id_rsa  
chmod 644 ~/.ssh/id_rsa.pub
```

Links

→ [Page github](#)

SSH

Debian: Configuration serveur SFTP + chroot

Explication pour mettre en place un serveur SFTP avec chroot. Le but étant de permettre à un utilisateur de se connecter à une machine via SFTP (transfert de fichier via SSH) et qu'il ait uniquement accès à son dossier personnel qu'il lui est dédié.

Cela étant très utile dans le cas d'un serveur Web, où l'on veut donner uniquement accès au dossier web à un utilisateur, sans qu'il puisse accéder au reste du système.

Dans cette documentation, la configuration est pour debian, mais elle fonctionne sur d'autres distributions.

Pré-requis

Il faut s'assurer que le serveur SSH est installé (la plupart du temps il est déjà installé):

```
apt install openssh-{client,server} openssh-sftp-server
```

Configuration et mise en place

Dans un premier temps, on va créer notre arborescence. Notre utilisateur s'appellera **luc** et son dossier home sera dans `/var/sftp/luc`

Création dossier `/var/sftp`, et affectations droits à root:

```
mkdir /var/sftp
chown root:root /var/sftp
chmod 755 /var/sftp
```

On crée le groupe sftpusers:

```
groupadd sftpusers
```

Création utilisateur sftp (luc), son dossier HOME dans `/var/sftp`, et sans shell "nologin":

```
useradd -g sftpusers -m -d /var/sftp/luc -s /sbin/nologin luc
```

Il faut lui définir un mot de passe:

```
passwd luc
```

On met les droits root sur son dossier home, sinon le chroot ne fonctionnera pas:

```
chown root:root /var/sftp/luc
```

Notre utilisateur ne pourra pas créer de fichiers et dossier dans son home, car les droits sont assignés à root. On va donc créer un sous dossier, où il aura tous les droits:

```
mkdir /var/sftp/luc/data  
chown -R luc:sftpusers /var/sftp/luc/data
```

Configuration service sshd

Nous allons créer une règles dans notre service SSH, afin que les utilisateurs du groupe **sftpusers** puissent seulement se connecter en sftp sur notre machine, et qu'ils aient uniquement accès à leur dossier, via chroot

Sur debian, on va créer un fichier `sftp_users.conf` dans `/etc/ssh/sshd_config.d/`:

```
Match group sftpusers  
ChrootDirectory /var/sftp/%u  
X11Forwarding no  
AllowTcpForwarding no  
Forcecommand internal-sftp
```

On redémarre le service, pour prise en compte:

```
systemctl restart ssh
```

Test connexion:

```
sftp luc@IP_MACHINE
```

Si c'est OK, après saisie du mot de passe:

```
Connected to localhost.  
sftp> ls  
data
```

On peut aussi utiliser un client graphique (winscp, filezilla, nautilus...) pour se connecter.

Liens

[It.fr - Installer et configurer un serveur SFTP](#)

[how-to-set-up-sftp-chroot-jail](#)

Desktop

Desktop

Mojave Theme pour gnome

Installer **Gnome TweakTool**, l'outil permettant de personnaliser Gnome et modifier theme avec le theme et icons de son choix

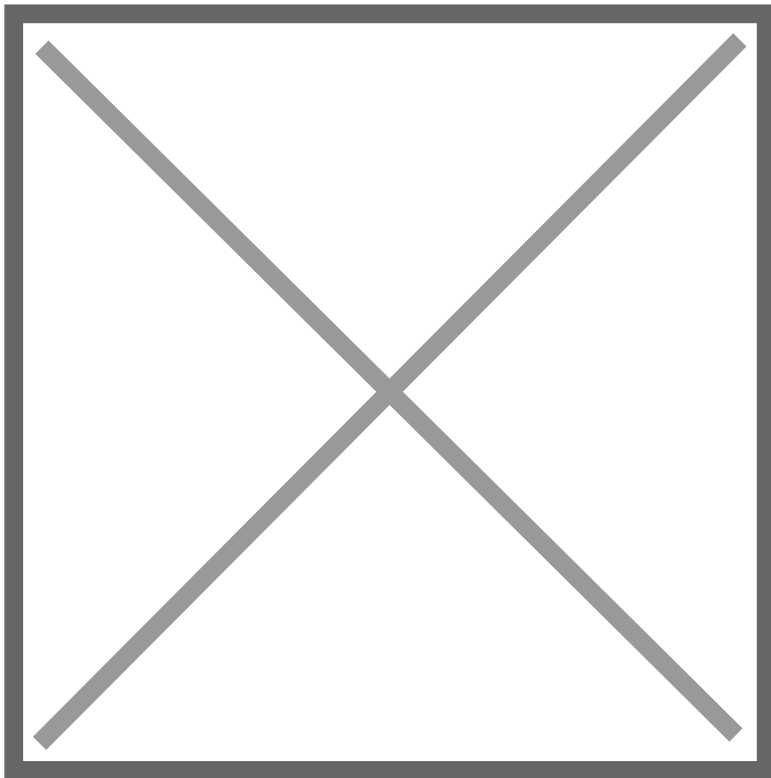
[MacOS Theme](#) → A mettre dans **home/.themes** (a créer si besoin)

MacOS Theme

[Curseur souris version mojave](#) → A mettre dans **home/.icons** (à créer si besoin)

drawing

[Icones MacOS](#) → A mettre dans **home/.icons** (à créer si besoin)



Pour avoir le dock version MacOS, installer plank → `apt-get install plank`

→ [MacOs icons pour plank](#) → A mettre dans **home/.local/share/plank/themes**

→ [25 fonds d'écrans pour MacOS Mojave](#)

→ [Vidéo tutoriel pour aide installation](#)

→ [Dash To Dock](#)

Pour mettre d'autres themes et icons, la procédure est le même, il suffit d'aller sur le site de [Gnome Look](#) pour DL des themes et icones

Chromium - color emojis

Pour avoir les emojis colorés sur les navigateurs chromium sur Linux

- Installer la police Google Noto Color Emoji:

```
sudo dnf install google-noto-emoji-color-fonts
```

- Créer le fichier **~/.config/fontconfig/fonts.conf** et metre le contenu suivant:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fontconfig SYSTEM "fonts.dtd">
<fontconfig>
  <alias>
    <family>serif</family>
    <prefer>
      <family>Noto Color Emoji</family>
    </prefer>
  </alias>
  <alias>
    <family>sans-serif</family>
    <prefer>
      <family>Noto Color Emoji</family>
    </prefer>
  </alias>
  <alias>
    <family>monospace</family>
    <prefer>
      <family>Noto Color Emoji</family>
    </prefer>
  </alias>
</fontconfig>
```

- Clear the font cache:

```
fc-cache -f
```

→ [Adding color emojis to Chrome on Fedora](#)

Memo commandes

Description	Commande	Exemple
Set DNS with systemd (cmd)	systemd-resolve --set-dns="DNS_IP" --interface=NET_INTERFACE	<i>systemd-resolve --set-dns="1.1.1.1" --interface=wlp2s0</i>
Changer Disposition clavier	stxkbmap langue	<code>stxkbmap fr</code> → Pour mettre en français

Memo VIM

memo commandes sur l'éditeur de texte vi

Remplacer texte:

Remplacer que la prochaine occurrence après le curseur:

```
:s/text_a_remplacer/texte_de_replacement/g
```

Remplacer tout les occurrences:

```
:%s/text_a_remplacer/texte_de_replacement/g
```

Remplacer les occurrences d'un intervalle de ligne (ex: 5 à 8):

```
:5,8s/foo/faa/
```

Espaces de travail

Split fenêtre

Splitter la fenêtre vim en 2 horizontalement:

```
split [nom_fichier]
```

Ou verticalement:

```
vsplit [nom_fichier]
```

Navigation espaces de travail

- Pour bouger d'une vue à une autre: **ctrl+w** & **touche directionnelle clavier** ou **h,j,k,l**
- Pour switcher la vue de place: **ctrl+w** & **H,J,K,L** (en majuscule)

Changer taille de la vue

- Pour que les vues soient redimensionnées de taille égale: **ctrl+w** & **"=**"
- Augmenter ou diminuer la hauteur de la vue: **ctrl+w** & **"+"** (augmenter) ou **"-"** (diminuer)
- Augmenter ou diminuer la largeur de la vue: **ctrl+w** & **">"** (augmenter) ou **"<"** (diminuer)
- Étendre la vue au max verticalement: **ctrl+w** & **"|"**
- Étendre la vue au max horizontalement: **ctrl+w** & **"_"**

Onglets

Pour ouvrir un fichier dans un nouvel onglet:

```
tabe [nom_fichier]
```

- Pour basculer d'un onglet à un autre: **ctrl+w** & "**gt**" ou "**gT**"
- Fermer tous les onglet sauf celui affiché: **:.tabonly**

nftables memo

Memo sur la gestion du pare-feu netfilter, via l'outil nftables.
Les commandes essentielles sont présentées, avec des exemples.

Gestion des tables

Création d'une table:

```
nft add table add inet example_filter
```

Ici notre table se nomme "example_filter".

- inet: table ipv' et ipv-
- ip: table ipv4
- ip6: table ipv6
- arp: table pour arp

Pour la liste des tables: [Wiki nftable](#)

Suppression d'une table :

```
nft delete table inet example_filter
```

Lister le contenu d'une table:

```
nft list inet example_filter
```

Gestion des chaines

Ensuite il va falloir ajouter des chaines à notre table et les lier à des "*hooks*". Ces derniers vont définir à quel type de hook de netfilter ils sont rattachés.

Ces chaines peuvent avoir n'importe quel nom, du moment qu'elles sont rattachées au bon hook (input, output...).

Ici on va créer une chaine "INPUT" et "OUPUT" pour que ça soit clair, mais on aurait pu les appeler autrement, du moment qu'ils bien rattachés aux hook:

```
nft add chain inet filter input '{type filter hook input priority 0;}'
```

```
nft add chain inet filter output '{type filter hook output priority 0;}'
```

Ici le "priority 0" fait référence à la priorité de la table, c'est à dire l'ordre de prise en compte des chaînes. Par défaut c'est 0, mais on aurait pu créer plusieurs types de chaînes, liées au hook 'input', mais avec des priorités différentes.

On a donc la table suivante, avec ce contenu (commande ***nft list table inet filter***):

```
table inet filter {  
    chain input {  
        type filter hook input priority filter; policy accept;  
    }  
  
    chain output {  
        type filter hook output priority filter; policy accept;  
    }  
}
```

Gestion des règles

Ajout de règles

Ajout d'une règle:

```
nft add rule filter input regle_a_appliquer
```

Par exemple pour ouvrir le port 80:

```
nft add rule filter input dport 80 accept
```

Et pour bloquer tout ce qui n'est pas explicitement écrit, on place cette règle à la fin, donc après avoir défini toutes nos règles:

```
nft add rule filter input drop
```

Pareil, si on veut bloquer tout le trafic sortant:

```
nft add rule filter output drop
```

Insertion de règles

Si on veut ajouter des règles, il va falloir les insérer. En effet, l'ajout du drop, aura pour conséquence la non prise en compte des règles ajoutée après.

il faut donc passer par une insertion de règles par rapport à celles déjà présentes.

Pour cela, il faut d'abord lister nos règles avec l'option -a:

```
root@debian:~# nft -a list table ip mon_filtreIPv4
table ip mon_filtreIPv4 { # handle 4
    chain input { # handle 1
        type filter hook input priority filter; policy accept;
        tcp dport 80 accept # handle 4
        drop # handle 5
    }

    chain output { # handle 3
        type filter hook output priority filter; policy accept;
        tcp sport 80 accept # handle 7
        drop # handle 8
    }
}
```

le "handle" correspond à l'identifiant de la règle.

Donc si on veut ajouter une règle autorisant l'accès au port ssh:

```
nft add rule inet filter input position 4 tcp dport 22 accept
nft add rule inet filter output position 7 tcp sport 22 accept
```

La règle sera donc ajoutée après position ciblée (ici 4 et 7).

Si on veut ajouter la règle avant la position ciblée, il faut juste faire un insert:

```
nft insert rule inet filter input position 4 tcp dport 22 accept
nft insert rule inet filter output position 7 tcp sport 22 accept
```

Suppression d'une règle

Pour supprimer une règle, il faudra utiliser le même principe que l'ajout, c'est à dire utiliser les handle (avec la commande **nft -a list**).

Par exemple pour supprimer la règle en position 7:

```
nft delete rule inet filter input handle 7
```

Sauvegarde des règles

La sauvegarde des règles de nftable est assez similaire à iptables.

On sauvegarde les règles existantes dans un fichier:

```
nft list table inet filter > nft_rules.rules
```

Les règles seront donc sauvegardées dans le fichier *nft_rules.rules*.

Pour restaurer les règles:

```
nft -f nft_rules.rules
```

Comme iptables, si on redémarre la machine, les règles ne sont pas persistantes.

Il faut soit faire un script qui se lance au démarrage et qui restaure nos règles avec "nft -f", soit ajouter la ligne suivante au fichier */etc/network/interface*:

```
allow-hotplug ens192
iface ens192 inet static
    address 192.168.5.2/24
    gateway 192.168.5.254
    pre-up nft -f /etc/nftables/nft.fw
```

Ici nos règles sont dans le fichier */etc/nftables/nft.fw*.

Cela ne marche que sur les systèmes sous debian.

Ressources:

[→ Cours nftables IT-Connect](#)

[→ Get Started with nftables](#)

[→ Wiki nftables](#)

Wireguard - Mise en place

Configuration serveur Wireguard

Documentation de mise en place et configuration du VPN wireguard

Installation

Sous debian 12, wireguard est déjà installé dans les packages de base. S'il n'est pas installé :

```
apt update && apt install wireguard
```

Configuration Serveur

Génération de fichier contenant la clé publique du serveur, ainsi que la clé privée :

```
root@wireguard:~# wg genkey |sudo tee /etc/wireguard/wg-srv-private.key | wg pubkey | sudo tee /etc/wireguard/wg-srv-public.key
```

Cela va générer 2 fichiers :

- **wg-srv-private.key** → Clé privée pour le serveur
- **wg-srv-public.key** → Clé publique, commune aux clients, à renseigner sur les configurations des clients

Ensuite, on va créer le fichier de configuration, contenant les paramètres du serveur, ainsi que la configuration des clients autorisés à se connecter dessus.

On va donc créer le fichier `/etc/wireguard/wg0.conf` :

```
root@wireguard:~# vi /etc/wireguard/wg0.conf
[Interface]

Address = X.X.X./24
SaveConfig = true
ListenPort = 51820
PrivateKey = CléPrivéeServeurWg
```

Détail des options :

- **Address** : Adresse IP du serveur, utilisée pour la communication entre les clients et le serveur
- **SaveConfig** : Pour conserver la config quand le VPN est actif, protégeant la config après arrêt du tunnel
- **ListenPort** : port d'écoute de Wireguard
- **PrivateKey** : clé privée du serveur, pour le chiffrement des échanges

Ensuite on démarre notre tunnel wireguard :

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

Après ça, on devrait voir apparaître notre tunnel, via la commande « ip -a » :

```
root@wireguard:~# ip a show dev wg0

13: wg0: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 1420 qdisc noqueue state UNKNOWN group default qlen 1000
    link/none
    inet 10.8.8.254/24 scope global wg0
        valid_lft forever preferred_lft forever
```

Et aussi avec la commande « wg show », pour voir les interfaces de wireguard actives :

```
root@wireguard:~# wg show

interface: wg0
  public key: snWqZfuc54sEVrWBghBkxi1Lk01X4W1t0sJtJD5S9n8=
  private key: (hidden)
  listening port: 41820
```

Pour que notre configuration soit persistante et active après redémarrage du serveur, on crée un service associé :

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

Pour que les paquets soient routés d'une interface à une autre, il faut activer l'IP Forwarding :

Dans le fichier **/etc/sysctl.conf**, décommenter la ligne suivante et modifier la valeur :

```
net.ipv4.ip_forward=1
```

Il faut aussi activer l'IP masquerade, pour mettre en place le NAT sur la machine.

Configuration du parefeu nftable :

```
table inet filter {

    chain input {
        type filter hook input priority filter; policy accept;
        ip saddr 10.8.8.0/24 accept
        udp dport 41820 accept
        ct state vmap { invalid : drop, established : accept, related : accept }
        drop
    }

    chain output {
        type filter hook output priority filter; policy accept;
        ct state invalid drop
    }

    chain prerouting {
        type nat hook prerouting priority filter; policy accept;
    }

    chain postrouting {
        type nat hook postrouting priority filter; policy accept;
        ip saddr 10.8.8.0/24 oif "ens256" snat ip to 192.168.1.1
        ip saddr 10.8.8.0/24 masquerade
    }
}
```

Ajout d'un client

Après avoir créer un couple de clé privée / publique sur le client, via wireguard sous windows, il faut ajouter le client sur le serveur.

Arret du service wgà:

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

Ajout du client dans le fichier /etc/wireguard.wg0.conf:

```
# Config VPN Exemple
[Peer]
```

```
PublicKey = 4eLau52rcERa1CTCbISUs3ysCnkpjqb1j0K2fhuKBTE=  
AllowedIPs = 10.8.8.1/32
```

On relance le service de wireguard:

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

On peut aussi utiliser la commande suivante sur le serveur, pour ajouter le client (évitant de redémarrer le service):

```
sudo wg set wg0 peer 4eLau52rcERa1CTCbISUs3ysCnkpjqb1j0K2fhuKBTE= allowed-ips 10.8.8.1/32
```

la configuration côté client, doit être la suivante:

```
[Interface]  
PrivateKey = client_priv_key  
Address = 10.8.8.1/24  
  
[Peer]  
PublicKey = server_pub_key  
AllowedIPs = 10.8.8.0/24  
Endpoint = ip_publique_serverur_wg:41820
```

Si tout est fonctionnel, notre client doit pouvoir pinguer le serveur.

Docs supplémentaires

[→ Digital Ocean: wireguard](#)

[→ IT-Connect: Mise en place wireguard debian 11](#)

Création VLANS - Debian

Commandes pour mettre en place des VLANS sous LINUX. Debian est utilisé mais les commandes marche sur la plupart des distributions linux.

Installation package

Installation du package pour activer la prise en charge des VLANS:

```
apt update  
apt install vlan
```

Activation du module:

```
echo 8021q >> /etc/modules  
modprobe 8021q
```

Maintenant que le module est activé, il faut passer à la mise en place de nos VLANS.

Création de l'interface virtuelle VLAN

On va créer un vlan 10, rattaché à l'interface eth0 dans notre exemple, à adapter en fonction des VLANS à créer et du nom des interfaces réseaux. L'interface eth0.10, correspond au nom de notre interface VLAN.

Ce nom permet de voir l'id du vlan, ainsi que l'interface rattachée.

```
# Création de l'interface  
ip link add link eth0 name eth0.10 type vlan id 10  
# Activation de l'interface  
ip link set dev eth0.10 up
```

Si on veut rattacher une IP à notre interface:

```
ip addr add 192.168.10.1/24 dev eth0.10
```

Ces commandes ne sont pas persistante, au prochain redémarrage, notre interface VLAN sera perdue.

Pour la rendre persistante, il faut éditer le fichier **/etc/network/interfaces**, et ajouter les lignes suivantes:

```
auto eth0.10  
address 192.168.10.1  
netmask 255.255.255.0  
vlan-raw-device eth0
```

La ligne "vlan-raw-device" est optionnelle si le nom de l'interface est du type ***nomEth.idVLAN***.

Source: => [***Vlan : sur Debian ou Ubuntu***](#)

Création service systemd

Sur les distributions utilisant systemd, voici comment créer un fichier service, par exemple pour lancer un script au démarrage.

En effet, si on veut qu'un script se lance au démarrage, l'option la plus adapté est de créer un service systemd, plutôt que de passer par rc.local, qui est obsolète.

Exemple de structure d'un fichier "service":

```
[Unit]
Description=
After=
ConditionPathExists=

[Service]
Type=
ExecStartPre=
ExecStart=
ExecStop=
ExecStopPost=
RemainAfterExit=
Restart=

[Install]
WantedBy=
```

Ce fichier se décompose en 3 sections:

- **Unit** → Descriptions service et les prérequis pour son fonctionnement
 - **Description**: Description du service
 - **After**: Après quel service il doit se lancer, par exemple après l'activation réseau (network.target) (facultatif)
 - **ConditionPathExists**: Pour tester si un répertoire ou fichier existe (facultatif)
- **Service** → Options pour le lancement du service
 - **Type**: Type de service (simple, oneshoot...)
 - **ExecStartPre**: commande ou script à lancer avant (facultatif)
 - **ExecStart**: Commande à lancer au démarrage du service
 - **ExecStop**: Commande à lancer lors de l'arrêt du service (facultatif)
 - **ExecStopPost**: Commande à lancer après l'arrêt du service (facultatif)

- **RemainAfterExit**: Si le service doit être considéré comme toujours actif après la fin de son exécution (*ExecStart*)
- **Restart**: Pour restart le service lors d'un plantage
- **Install** → Option pour le déclenchement

Pour le détail des options supplémentaire, la doc de redhat est assez fournie.

Exemple pour lancer un script au lancement du système:

```
[Unit]
Description="nom du service"
After=network.target

[Service]
ExecStart=/chemin/du/script
RemainAfterExit=yes

[Install]
WantedBy=multi-user.target
```

Il faut créer ce fichier dans `/etc/systemd/system`, et le nom doit être de type `"nom_du_service.service"`.

Ensuite pour activer ce service au lancement du système:

```
systemctl enable mon_service.service
```

Liens supplémentaires

Liens doc pour plus d'exemples et de précisions:

[linuxtricks - systemd : Créer des services, timers \(unités\)](#)

[Article nextinpact](#)

[Doc redhat](#)

Bash Memo

Raccourcis Clavier

Liste des raccourcis clavier de bash qui peuvent être utile et faire gagner du temps:

Suppression mot courant du curseur jusqu'à la fin	ctrl + d
Suppression mot courant du curseur jusqu'au début	ctrl + w
Déplacement début du mot courant	alt + b
Déplacement fin du mot courant	alt + f
Déplacement début ligne	ctrl + a
Déplacement fin ligne	ctrl + e
Suppression texte curseur => Fin ligne	ctrl + k
Suppression texte début ligne => Curseur	ctrl + u
Clean le terminal	ctrl + l