

<http://publications.levinux.org>

Historique des versions
Mention légale
Copyright : ;
Résumé
Construire un laboratoire LTSP dans un vserver... afin de mieux gérer!

## Table of Contents

1. Ajouter un localhost .....	1
2. Débloquer le ssh -CX sur le vserver .....	2
3. Permettre d'accéder à chacun des vservers et à la machine physique via ssh .....	2
4. Serveurs DHCP et NFS .....	2
5. Activer localdev pour ltsp (disquette, cdrom et usb) dans un vserver .....	3
6. ldap .....	3
7. Copier un vserver .....	3
7.1. Récupérer un vserver de terminaux .....	4
8. Utiliser les périphériques de la machine hôte (le serveur physique) dans un vserver .....	5
9. Configuration de cups dans un vserver .....	6
10. Monter nfs dans un vserver et modifier le /tmp .....	7

## 1. Ajouter un localhost

- Réduire le netmask de **lo** sur la machine physique 127.0.0.1/8 à 127.0.0.1/16
- Créer un fichier (distributions de la famille debian... pour les familles redhat et quelques autres, le fichier existe déjà) /etc/modprobe.conf et y ajouter l'instruction suivante **options dummy numdummies=5** (on met le chiffre qu'on veut... ici, on en crée 5)
- On lance le module dummy<sup>1</sup>

```
modprobe dummy
```

- On déclare une interface dummy0 dans `/etc/vservers/<vserver>interface/<id>` puis on met dummy0 dans **dev** et 127.1.0.1/24 dans **ip**
- On relance le vserver
- On modifie le fichier `/etc/hosts` du vserver **localhost 127.1.0.1**
- Référence: [mirabellug](#)<sup>1</sup>

## 2. Débloquer le ssh -CX sur le vserver

- Ajouter dans `/etc/ssh/sshd_config` du vserver l'instruction suivante **X11UseLocalHost**<sup>2</sup> **no**
- Lancer `xhost +` sur le terminal qui appelle le vserver
- On s'assure que `/etc/hosts` contient une ligne définissant notre vserver (son adresse ip et son nom!)
- Référence: [LinuxÉdu-Québec](#)<sup>3</sup>

## 3. Permettre d'accéder à chacun des vservers et à la machine physique via ssh

- On ajoute dans le fichier `/etc/ssh/sshd_config` de chacune des machines (la machine physique et chacun des vservers) l'adresse IP sur laquelle le daemon sshd va écouter **ListenAddress**<sup>4</sup> **n.n.n.n**
- On relance les sshd
- Référence [deb.riseup.net](#)<sup>5</sup>

## 4. Serveurs DHCP et NFS

- Pour DHCP, il faut ajouter les options `CAP_NET_RAW` et `CAP_NET_BROADCAST` dans `/etc/vservers/<vserver>bcapabilities`
- Pour nfs, il y a seulement `nfs-user-server` qui fonctionne; on oublie `nfs-kernel-server`... Mais `nfs-user` offre de bonnes performances pour LTSP
- Référence: on consultera la [FAQ de J. Gélinas](#)<sup>6</sup> et celle de [Sladen](#)<sup>7</sup>

## 5. Activer localdev pour Itsp (disquette, cdrom et usb) dans un vserver

- Il faut ajouter l'option suivante dans `/etc/vservers/<vserver>bcapabilities CAP_SYS_ADMIN` afin de permettre au vserver de monter (smbmount) les fichiers partagés sur les terminaux.

## 6. ldap

- La recette à suivre pour ldap sur breezy ou debian est [ici](#)<sup>8</sup>. Il faut cependant ajouter un certain nombre de choses.
  - Il manque une commande dans le tutoriel: après avoir créé le fichier `Myuser.ldif`, il faut ajouter

```
ldapadd -x -D "cn=admin,dc=example,dc=org" -W -f Myuser.ldif
```

- Tout fonctionne à merveille quand les répertoires `/home` sont existants et qu'on ne passe pas par `gdm` (ou `kdm` ou `xdm`), en particulier sur LTSP; autrement, il faut quelques ajouts:
  - Le syslog indique que `gdm` se plaint de deux répertoires manquant: on les crée

```
mkdir /var/run/nscd -p
mkdir /var/db && mkdir /var/db/nscd
```

- Il faut également ajouter au fichier `/etc/pam.d/common-session` les deux lignes suivantes

```
session optional                pam_ldap.so
session required                pam_mkhomedir.so skel=/etc/skel/ umask=0
```

- Enfin, si l'on souhaite offrir un service de courrier, on doit ajouter dans `/etc/skel` le répertoire **Maildir**, contenant les répertoires **cur new tmp**

## 7. Copier un vserver

- On oublie `cp` ou `scp`. On me dit que `tar --preserve` fonctionne.... je n'ai pas eu cette veine. Voici trois solutions qui fonctionnent
  - **vserver-copy [vserver] {hote:}[vservercopie]**
    - Cette commande ne fonctionne que si l'on crée un lien symbolique dans `/etc/vservers/[vserver]` vers `[vserver].conf`

- On ajuste ensuite dans `/etc/vservers` le répertoire `[vservercopie]`; on copie (ici on peut utiliser `cp`, `scp` ou `tar`) `/etc/vservers/[vserver]` dans `/etc/vservers/[vservercopie]` et on ajuste les variables (**interfaces/n n n... , name, uts/nodename et vdir**)
- **dump 0zf {path}[nomdufichier] [vserver]** est également très bien. On récupère avec **restore -fr {path}[nomdufichier]** en se mettant à la racine / de la machine de récupération.
- On peut également copier à distance avec la commande suivante

```
rsync [vserver] -aPvxze ssh --delete --stats --numeric-ids {host:}/vservers/[vserver]
```

- Référence : on consultera encore l'excellente FAQ de [Sladen](#)<sup>9</sup> pour `dump` et `rsync`

## 7.1. Récupérer un vserver de terminaux

- On veut récupérer un vserver de terminaux; quels sont les fichiers à adapter au nouvel environnement ? (tous les chemins sont donnés à partir d'une distribution de type `debian`)
  - pour la config du vserver (si la configuration du réseau est différente...)
    - `/etc/vserver/[vserver]/interfaces/[n]/ip` (et peut-être `dev`)
  - voir plus haut [\[un localhost](#)<sup>10</sup>] pour une interface "dummy", s'il y a lieu
  - pour `ldap`
    - `/etc/ldap/ldap.conf`
    - `/etc/ldap.secret`
    - `/etc/pam_ldap.conf`
    - `/etc/libnss-ldap.conf`
  - pour `ssh`
    - On ajuste le "listen" dans `/etc/ssh/sshd_config`, à la fois sur le vserver et sur la machine `host`
  - pour le `dhcp` (cela va de soi si le nouvel environnement propose un jeu d'adresses ip différent. Et ce, qu'il soit dans le vserver de terminaux ou à l'extérieur)
    - `/etc/dhcpd3/dhcpd.conf` sur la machine (virtuelle ou physique) qui héberge le `dhcp`

- On adapte le jeu d'adresses ip à distribuer à son environnement et on s'assure que "option root-path" pointe vers l'adresse du serveur ltsp.
- pour ltsp (même remarque que pour le dhcp)
  - /opt/ltsp/i386/lts.conf
  - On donne les bonnes adresses pour le serveur ltsp et le xdm
- pour cups
  - /etc/cups/cupsd.conf, si on est serveur
  - /etc/client.conf, si on est client
  - REMARQUE : on ne peut lancer cups à la fois sur la machine hôte et sur un vserver... à moins de trouver un "wrapper" qui fonctionne... je n'ai pas encore eu cette chance...
- pour gdm
  - On s'assure que le nombre de connexions simultanées est adapté à son nouvel environnement
- pour /tmp
  - On s'assure que l'espace est suffisant. Voir la rubrique [nfs dans un vserver](#) et modifier le /tmp<sup>11</sup> ci-dessous
- Si on change de nom...
  - Les fichiers /etc/vservers/[nouveaunom] ../name, ../uts/nodenane, le lien symbolique ../vdir à ajuster avec le nouveau nom dans [path\_des\_vservers]/vservers/[nouveaunom]

## 8. Utiliser les périphériques de la machine hôte (le serveur physique) dans un vserver

- On aimerait pouvoir gérer l'impression directement dans un vserver et on aimerait également graver des cd depuis le vserver.
- On doit pouvoir définir les **dev** manquant:
  - Ajouter CAP\_MKNOD dans le fichier /etc/vservers/<vserver>bcapabilities
  - Créer les dev nécessaires (cdrom, graveur CD et imprimante parallèle, par exemples)

```
mknod /dev/hdc b 22 0
mknod /dev/hdd b 22 64
mknod /dev/lp0 c 6 0
```

- Changer les groupes

```
chwon root:cdrom /dev/hdc
chwon root:cdrom /dev/hdd
chown root:lp /dev/lp0
```

- S'assurer que les utilisateurs à qui l'on donne des privilèges sur l'utilisation des périphériques (user1, user2, etc. et cupsys) aient accès aux ressources dans le fichier /etc/group. Par exemple, l'entrée cdrom devrait ressembler à cela

```
cdrom:x:24:user1,user2,userN
```

- On pourrait également ajouter scanner et autres périphériques....

## 9. Configuration de cups dans un vserver

\* Il m'a fallu pas mal de temps avant de comprendre qu'il ne FAUT PAS utiliser les options listen dans le fichier de config. Voici une configuration de base (très minimale) qui fonctionne parfaitement

```
LogLevel info
TempDir /var/spool/cups/tmp
# No 'Listen' directive !
Port 631
BrowseAddress @LOCAL
BrowseDeny All
BrowseAllow @LOCAL
BrowseOrder deny,allow
<Location />
  Order Deny,Allow
  Deny From All
  Allow From @LOCAL
  Allow From 192.168.0.0/16
</Location>
<Location /admin>
  AuthType Basic
  AuthClass System
  Order Deny,Allow
  Deny From All
  Allow From 192.168.0.0/24
```

```
# Or
# Allow From @LOCAL
</Location>
```

- Cette configuration permet également à des clients externes au vserver (dans le même sous-réseau) d'utiliser les ressources d'impression. On configure le script **client.conf** en conséquence
- Référence : j'ai trouvé la solution [ici](#)<sup>12</sup>

## 10. Monter nfs dans un vserver et modifier le /tmp

- Ouf! Cette opération-là m'a donné beaucoup de sueurs froides... Voici le scénario
  - Une machine host (anaconda) sur laquelle roule un vserver de terminaux (vs01) et une machine de type nas (marmotte) sur laquelle se trouve les home de même qu'un serveur nfs (pas de vservers).
  - On monte facilement marmotte/home sur anaconda/home; on peut également faire un "mount --bind" anaconda/home ...vs01/home
  - Si vs01 n'était pas contraint d'utiliser à son tour le portmap et nfs, ça roulerait sans problèmes (c'est d'ailleurs le cas avec /equinux/vs02 qui roule apache, mysql, postfix, imap, etc. Mais voilà : vs01 doit lancer portmap et nfs-user-server pour faire fonctionner les terminaux; il ne peut y avoir à la fois un portmap sur anaconda et sur anaconda/vs01. Que faire ?
  - La réponse paraît simple : ne pas monter marmotte/home sur anaconda/home, mais directement dans ..vs01/home (et faire un mount --bind depuis ..vs01/home vers anaconda/home, si nécessaire).
  - Ça ne monte pas sans options... (ou plutôt, ça monte, mais c'est hyper-hyper lent et pas utilisable avec les terminaux.!)
  - Après avoir fouillé le web, j'ai enfin trouvé une solution: on monte avec les options hard,intr,nolock. On peut même mettre le montage dans /etc/vservers/vs01/fstab. Voici ce que ça donne.

```
none    /proc          proc          defaults          0 0
none    /tmp           tmpfs         size=100m,mode=1777 0 0
none    /dev/pts       devpts        gid=5,mode=620    0 0
marmotte:/mnt/storage/home /home        nfs           hard,intr,nolock  0 0
```

- Référence: j'ai trouvé la solution [ici](#)<sup>13</sup>
- On remarquera que le /tmp est maintenant à 100 megs... Par défaut, il était seulement à 16... pas suffisant pour 25 terminaux. J'ai présumé (ça reste à vérifier) qu'il fallait compter de 3 à 4 megs par terminal, quand on utilise kde et openoffice.

**Table 1.**

test	test
test	

- 1 Voici une note de bas de page