Installer Ubuntu à partir d'un système Unix/Linux dans un chroot

Table of Contents

Cette section explique comment installer Ubuntu à partir d’un système Unix ou Linux existant, sans utiliser le programme d’installation piloté par menu.

Une fois le nouveau système Ubuntu configuré, on peut y migrer les données utilisateur existantes (le cas échéant) et continuer ainsi. Il s’agit donc d’une installation Ubuntu à «zéro temps d’arrêt». C'est également un moyen de gérer du matériel qui ne fonctionne pas correctement avec différents supports de démarrage ou d'installation.

Préparation de l'environnement

Préparation du support

Avec les outils de partitionnement habituels, partitionner le disque dur de destination, en créant au moins un système de fichiers et un échange. Il faut environ 506 Mo d'espace disponible pour une installation sur console uniquement.

Ensuite, créer des systèmes de fichiers sur les partitions. Par exemple, pour créer un système de fichiers ext3 sur la partition `/dev/sda6`:

```
mke2fs -j /dev/sda6
```

Pour créer un système de fichiers ext2 à la place, omettre -j.

Initialiser et activer le swap (remplacer le numéro de partition par la partition de swap Ubuntu prévue):

```
mkswap /dev/sda5
sync
swapon /dev/sda5
```

Repmarque: Au lieu d'utiliser une partition swap dédiée, on peut omettre la configuration de la partition swap ici et utiliser simplement un fichier swap.

Monter une partition en tant que `/mnt/ubuntu` (le point d'installation, qui sera le système de fichiers racine (/) sur le nouveau système). Le nom du point de montage est strictement arbitraire, il est référencé plus tard.

```
mkdir /mnt/ubuntu
mount /dev/sda6 /mnt/ubuntu
```

Pour monter des parties du système de fichiers (par exemple, /usr) sur des partitions distinctes, il faudra créer et monter ces répertoires manuellement avant de passer à l'étape suivante.

Installer debootstrap

L'utilitaire utilisé par le programme d'installation Ubuntu et reconnu comme le moyen officiel d'installer un système de base Ubuntu est **debootstrap**. Il utilise **wget** et **ar**, mais ne dépend sinon que de `/bin/sh` et des outils de base Unix/Linux. Installer **wget** et **ar** s'ils ne sont pas déjà sur lee système actuel, puis télécharger et installer **debootstrap**. Si ces étapes sont exécutées sous Ubuntu, on peut le faire simplement en installant **debootstrap**.

Ou, on peut utiliser la procédure suivante pour l'installer manuellement. Créer un dossier de travail

pour extraire le .deb dans:

```
mkdir work
cd work
```

Le binaire debootstrap se trouve dans l'archive Ubuntu (sélectionner le fichier approprié pour l'architecture). Télécharger le fichier **debootstrap.deb** à partir du pool, copier le package dans le dossier de travail et extraire les fichiers de celui-ci. Il fuat utiliser des privilèges root pour installer les fichiers.

```
ar -x debootstrap_0.X.X_all.deb
cd /
zcat /full-path-to-work/work/data.tar.gz | tar xv
```

Lancer debootstrap

debootstrap peut télécharger les fichiers nécessaires directement à partir de l'archive lorsqu'on l'exécute. On peut remplacer ports.ubuntu.com/ubuntu-ports par n'importe quel miroir d'archive Ubuntu dans l'exemple de commande ci-dessous, de préférence un miroir proche de votre réseau. Les miroirs sont répertoriés à l'adresse http://wiki.ubuntu.com/Archive.

Si on a un CD bionique Ubuntu monté sur /cdrom, vous pouvez remplacer l'URL du fichier par l'URL http: file:/cdrom/ubuntu/

Dans la commande debootstrap, remplacer **ARCH** par l'un des éléments suivants: **amd64**, **arm64**, **armhf**, **i386**, **powerpc**, **ppc64el** ou **s390x**.

```
/usr/sbin/debootstrap --arch ARCH bionic /mnt/ubuntu
```

Configurer le système de base

On a maintenant un vrai système Ubuntu, bien que maigre, sur disque. se chrooté dedans:

```
LANG=C.UTF-8 chroot /mnt/ubuntu /bin/bash
```

Configuration du terminal

Après avoir chrooté, il faudra peut-être définir la définition du terminal pour qu'elle soit compatible avec le système de base Ubuntu, par exemple:

```
```

```
export TERM=xterm-color
```

En fonction de la valeur de **TERM**, il faudra peut-être installer le package **ncurses-term** pour obtenir de l'aide.

Si des avertissements comme `bash: warning: setlocale: LC_ALL: impossible de modifier les paramètres régionaux (en_US.UTF-8)`,se produisent, les fichiers de localisation requis doivent être générés:
`sudo locale-gen en_US.UTF-8`

Configurer Apt

Debootstrap aura créé un fichier `/etc/apt/sources.list` très basique qui permettra d'installer des paquets supplémentaires. Toutefois, il est suggéré d'ajouter des sources supplémentaires, par exemple pour les packages source et les mises à jour de sécurité:

```
deb-src http://archive.ubuntu.com/ubuntu bionic main

deb http://security.ubuntu.com/ubuntu bionic-security main
deb-src http://security.ubuntu.com/ubuntu bionic-security main
```

Exécuter `apt update` après avoir modifié la liste des sources.

Installer des paquets supplémentaires

Maintenant, il est nécessaire d'installer certains paquets supplémentaires, tels que **makedev** (nécessaire pour la section suivante): apt install makedev

Ou si on utilise l'architecture **s390x** le paquet obligatoire **s390-tools**: `apt install s390-tools`

Préparer le système de fichier

Créer des fichiers de périphérique

À ce stade, `/dev/` ne contient que des fichiers de périphérique très élémentaires. Pour les prochaines étapes de l'installation, des fichiers de périphérique supplémentaires peuvent être nécessaires. Il existe différentes façons de procéder. La méthode à utiliser dépend du système hôte utilisé pour l'installation, de l'intention d'utiliser un noyau modulaire ou non, et de l'intention d'utiliser fichiers de périphérique Dynamic (par exemple, avec udev).) ou des fichiers de périphérique statiques pour le nouveau système.

Quelques unes des options disponibles sont:

- créer un ensemble par défaut de fichiers de périphériques statiques à l'aide de (après chrooter)

```
mount none /proc -t proc
cd /dev
MAKEDEV générique
# ou selon l'architecture spécifique:
MAKEDEV std
cd ..
```

- créer manuellement uniquement des fichiers de périphérique spécifiques à l'aide de MAKEDEV

```
# Sur s390x, les périphériques DASD doivent être créés comme suit:
cd /dev
MAKEDEV dasd
cd ..
```

- faire un `mount --bind` à partir du système hôte au-dessus de /dev dans le système cible, par exemple:

```
mount --bind dev / dev
```

Les scripts `postinst` de certains paquetages peuvent essayer de créer des fichiers de périphérique. Cette option ne doit donc être utilisée qu'avec précaution.

Monter proc et sys

On peut monter les systèmes de fichiers **proc** et **sys** plusieurs fois et à des emplacements arbitraires, bien que `/proc` et `/sys` soient respectivement usuels.

```
mount -t proc proc /proc
mount -t sysfs sysfs /sys
```

La commande `ls /proc` doit maintenant afficher un répertoire non vide. Si cela échoue, il faudra peut-être monter **proc** de l'extérieur du chroot:

```
mount -t proc proc /mnt/ubuntu/proc
```

Monter les partitions

Il faut créer /etc/fstab.

```
vi /etc/fstab
```

Voici un exemple de contenu que l'on peut modifier en fonction des besoins :

```
# /etc/fstab: informations sur le système de fichiers statique.
# fabriqué à la main
# options de type de point de montage du système de fichiers dump pass
#
/dev/dasdb1/ ext4 default 0 1

/swapfile none swap sw 0 0
```

Utiliser `mount -a` pour monter tous les systèmes de fichiers spécifiés dans le nouveau `/etc/fstab` ou, pour monter des systèmes de fichiers individuellement, utiliser:

```
mount /path # exemple: mount/usr
```

Les systèmes Ubuntu actuels ont des points de montage pour les supports amovibles sous `/media`, mais conservent les liens symboliques de compatibilité dans /. Créer ses liens au besoin, par exemple:

```
cd /media
mkdir cdrom0
ln -s cdrom0 cdrom
cd /
ln -s media /cdrom
```

Même si **lsdasd** répertorie déjà les périphériques **DASD**:

```
lsdasd

  ID de bus Nom d'état Type de périphérique BlkSz Size Blocks
  ================================================
============================
  0.0.1601 actif dasda 94: 0 ECKD 4096 7043MB 1803060
  0.0.260a actif dasdb 94: 4 ECKD 4096 7043MB 1803060
```

... ainsi que d'autres périphériques **CCW** tels que **DASD**, **FCP** ou **QETH**, ceux-ci ne peuvent pas encore être utilisés complètement et de manière persistante.

```
lszdev --online | head -n 1 && lszdev --online | grep dasd-eckd

  TYPE ID SUR NOMS PERS.
  dasd-eckd 0.0.0123 oui non dasda
  dasd-eckd 0.0.1234 oui non dasdb
```

Ici, DASD 1234 est celui utilisé pour **debootstrap**. Il faut activer cette **DASD** de manière persistante

à l'aide de l'outil habituel **chzdev**:

```
chzdev -e 1234

  ECKD DASD 0.0.1234 configuré

lszdev --en ligne 1234

  TYPE ID SUR NOMS PERS.
  dasd-eckd 0.0.1234 oui oui dasdb
```

Répéter les mêmes étapes pour d'autres périphériques **CCW**, tels que **FCP**, **QETH** ou d'autres périphériques **DASD**, autant que nécessaire.

Réglage du fuseau horaire

Le réglage de la troisième ligne du fichier `/etc/adjtime` sur "UTC" ou "LOCAL" détermine si le système interprétera l'horloge matérielle comme étant définie sur l'heure locale UTC respective.

```
vi /etc/adjtime
```

```
0,0 0 0,0
0
UTC
```

La commande suivante permet de choisir le fuseau horaire.

```
dpkg-reconfigure tzdata
```

Configurer le réseau

Pour configurer la mise en réseau, éditer `/etc/network/interfaces`, `/etc/resolv.conf`, `/etc/hostname` et `/etc/hosts`.

```
cat /etc/network/interfaces << "EOF"
################################################## ####################
# / etc / network / interfaces - fichier de configuration pour ifup (8),
ifdown (8)
# Voir la page de manuel interfaces (5) pour plus d'informations sur les
options disponibles.
# disponible.
################################################## ####################
```

```
# Interface de bouclage.
#
auto lo
iface lo inet loopback

# Pour utiliser dhcp:
#
# auto eth0
# iface eth0 inet dhcp

# Exemple de configuration IP statique: (la diffusion et la passerelle sont
facultatifs)
#
# auto eth0
# iface eth0 inet statique
# adresse 192.168.0.42
# réseau 192.168.0.0
# masque de réseau 255.255.255.0
# diffusion 192.168.0.255
# passerelle 192.168.0.1
EOF
```

Entrer le serveur de noms et vos directives de recherche dans `/etc/resolv.conf`:

```
cat /etc/resolv.conf<< "EOF"
search hqdom.local
nameserver 10.1.1.36
nameserver 192.168.9.100
EOF
```

Entrer le nom d'hôte du système (2 à 63 caractères):

```
echo UbuntuHostName> /etc/hostname
```

Et un `/etc/hosts` de base avec support IPv6:

```
cat /etc/hosts << "EOF"
127.0.0.1 localhost
127.0.1.1 UbuntuHostName

# Les lignes suivantes sont souhaitables pour les hôtes compatibles IPv6
:: 1 ip6-localhost ip6-loopback
fe00 :: 0 ip6-localnet
ff00 :: 0 ip6-mcastprefix
ff02 :: 1 ip6-allnodes
ff02 :: 2 ip6-allrouters
ff02 :: 3 ip6-allhosts
EOF
```

Lorsqu'on a plusieurs cartes réseau, il faut organiser les noms des modules de pilotes dans le fichier `/etc/modules` dans l'ordre souhaité. Ensuite, au démarrage, chaque carte sera associée au nom d'interface (eth0, eth1, etc.) attendu.

Configurer les paramètres régionaux et le clavier

Pour configurer les paramètres régionaux de manière à utiliser une langue autre que l'anglais, il fautinstaller les modules de langue appropriés et les configurer. Actuellement, l'utilisation des paramètres régionaux UTF-8 est recommandée.

```
apt installer language-pack-fr language-pack-gnome-fr
```

Puis configurer votre clavier (si nécessaire):

```
apt install console-setup
dpkg-reconfigure keyboard-configuration
```

Le clavier ne peut pas être configuré dans le chroot, mais sera configuré pour le prochain redémarrage.

Installer un noyau

Pour démarrer ce système, Il est nécessaire d'avoir un noyau Linux et un chargeur de démarrage. Identifier les noyaux pré-emballés disponibles avec:

```
apt-cache search linux-image
```

Ensuite, installer le paquet de noyau en utilisant son nom.

```
apt install linux-image-arch-etc
```

(On peut également installer `linux-image-generic`.)

Configurer le chargeur d'amorçage

Pour que le système Ubuntu soit amorçable, configurer le chargeur de démarrage pour charger le noyau installé avec la nouvelle partition racine. **debootstrap** n'installe pas de chargeur de démarrage, utiliser apt dans votre chroot Ubuntu pour le faire.

Le chargeur de démarrage **zipl** fait partie du package **s390-tools** installé précédemment. Consulter `man zipl.conf` pour des instructions sur la configuration du chargeur de démarrage. Créer une configuration `zipl.conf` à partir de rien ou copier et modifier celle existante.

Voici un exemple de base de `/etc/zipl.conf`:

```
[defaultboot]
defaultmenu=menu

:menu
target=/boot
1=Ubuntu
défaut=1

[Ubuntu]
target=/boot
image=/boot/vmlinuz
ramdisk=/boot/initrd.img
parameters=root=/dev/dasdb1
```

Étant donné que les noms de fichiers du noyau et initrd dans /boot sont versionnés, il est recommandé de les lier à des noms par défaut non versionnés, tels que:

```
ln -s /boot/vmlinuz-4.15.0-23-generic /boot/vmlinuz
ln -s /boot/initrd.img-4.15.0-23-generic /boot/initrd.img

ls -la / boot / vmlinuz * /boot/initrd.img*

  lrwxrwxrwx 1 racine 28 juin 20 07:31 /boot/initrd.img -> initrd.img-
<version> -generic
  -rw-r - r-- 1 racine racine 11245088 20 juin 07:14 /boot/initrd.img-
<version>-générique
  lrwxrwxrwx 1 racine racine 25 juin 20 07:31 / boot / vmlinuz -> vmlinuz-
<version> -generic
  -rw ------- 1 racine racine 4390912 23 mai 12:54 / boot / vmlinuz-
<version> -generic
```

Enfin, exécuter la commande `zipl` pour écrire la configuration sur le disque:

```
zipl
```

```
  Utilisation du fichier de configuration '/etc/zipl.conf'
  Construire bootmap dans '/ boot'
  Menu de construction 'menu'
  Ajout n ° 1: section IPL 'ubuntu' (par défaut)
  Préparation du périphérique d’amorçage: dasdb (<votre numéro de
périphérique dasd>).
  Terminé.
```

Touches finales

Accès à distance: Installation de SSH et configuration de l'accès

Pour pouvoir se connecter au système via le réseau, installer SSH et configurer l'accès.

```
apt install openssh-server
```

La connexion root avec mot de passe est désactivée par défaut. Il faut donc configurer l'accès en définissant un mot de passe root

```
passwd
```

et en réactivant la connexion root avec mot de passe dans `/etc/ssh/sshd_config`:

```
PermitRootLogin yes
```

L'accès peut également être configuré en ajoutant une clé ssh au compte root:

```
mkdir /root/.ssh
cat << EOF> /root/.ssh/authorized_keys
ssh-rsa ....
EOF
```

Enfin, l'accès peut être configuré en ajoutant un utilisateur non root et en lui définissant un mot de passe:

```
adduser joe
passwd joe
```

Complétion de l'installation

Comme mentionné précédemment, le système installé sera très basique. Pour rendre le système un peu plus mature, il existe une méthode simple pour installer tous les packages correspondants à une “tâche” particulière.

Tasksel est une application d'installation de logiciels faisant partie intégrante de l'installeur Debian. **Tasksel** regroupe les paquets à installer par tâches (ex. serveur LAMP, création audio, etc.), permettant ainsi à l'utilisateur d'installer très facilement l'ensemble des paquets nécessaires à une tâche particulière.

`apt install tasksel tasksel install standard` Bien sûr, on peut aussi simplement utiliser apt pour installer les paquets individuellement.

Nettoyage du système

Après l'installation, il y aura beaucoup de packages téléchargés dans `/var/cache/apt/archives/`. On peut libérer de l'espace disque en lançant:

```
apt clean
```

Utilisateurs admin

Utiliser la commande adduser pour créer un nouveau compte utilisateur:

```
adduser myusername
```

Répondre à l'invite pour entrer un nom complet et un mot de passe.

La configuration normale d'Ubuntu consiste à permettre à ce nouvel utilisateur d'administrer le système à l'aide de **sudo**. Pour le configurer, créer d'abord un groupe des administrateurs et y ajouter le nouvel utilisateur:

```
addgroup --system admin
adduser myusername admin
```

On peut maintenant utiliser la commande `visudo` pour ajouter ces lignes à la fin de `/etc/sudoers`, afin que tout utilisateur du groupe **admin** puisse administrer le système:

```
# Les membres du groupe admin peuvent obtenir les privilèges root
%admin ALL=(ALL) ALL
```

Lorsqu'on n'utilise pas cette configuration, il faut définir un mot de passe root: `passwd root`