Microlinux Enterprise Desktop 14.2 Guide d'Installation

Ce document vous guide pas à pas dans l'installation du poste de travail Microlinux Enterprise Desktop 14.2. Il part du principe que vous savez déjà installer et configurer Slackware Linux. Lisez-le attentivement. Les paragraphes sur lesquels vous décidez de faire l'impasse reviendront vous mordre les fesses.

Si vous n'êtes pas trop familiarisé avec Slackware, allez faire un tour sur le *Projet de Documentation Slackware* à l'adresse http://docs.slackware.com. La plupart des pages de ce site sont disponibles en français. Lisez les deux articles suivants dans la section **Débuter avec Slackware** :

- **Guide d'installation de Slackware**
- **Configurer votre nouveau système Slackware**

MLED s'installe par-dessus une base Slackware judicieusement choisie et n'est pas fourni sur un ISO d'installation à part. Tout ce qu'il vous faut, c'est un support d'installation Slackware classique (DVD, CD-Rom, clé USB) et une connexion Internet.

Installer le système de base Slackware Linux

Démarrez sur le support d'installation de Slackware : DVD, premier CD-Rom ou clé USB. Sélectionnez votre disposition clavier, connectez-vous en tant que root, partitionnez le disque dur et démarrez l'installateur.

```
root@slackware:/# setup
```

- **PACKAGE SERIES SELECTION** : décochez les groupes de paquets **E**, **KDE** et **KDEI** ;
- **SELECT PROMPTING MODE** : **full** ou **terse** ;
- **USE UTF-8 TEXT CONSOLE** : **Yes** ;
- **CONFIRM STARTUP SERVICES TO RUN** : confirmez les services par défaut ;
- **SELECT DEFAULT WINDOW MANAGER FOR X** : **xinitrc.wmaker** ou **xinitrc.fluxbox**. Ce choix n'est que temporaire. Je me sers habituellement des gestionnaires de fenêtres légers comme WindowMaker ou Fluxbox pour peaufiner la configuration du serveur graphique.

Terminez l'installation du système Slackware de base, quittez l'installateur et redémarrez. Ne définissez pas encore le ou les utilisateurs. Nous attendrons que les profils MLED soient installés.

Télécharger les scripts Microlinux

Microlinux fournit une collection de scripts pour accélérer le processus d'installation. Téléchargez cette arborescence de fichiers vers votre répertoire `/root` en utilisant la commande suivante :

```
# cd
# git clone https://github.com/kikinovak/microlinux
```

Configurer slackpkg

Dans la configuration par défaut, le gestionnaire de paquets `slackpkg` ne fonctionne qu'avec les dépôts officiels de Slackware. Nous devons télécharger et installer le plugin `slackpkg+` de Matteo Rossini pour activer l'utilisation de dépôts de paquets tiers. Microlinux vous évite la corvée en fournissant un paquet `slackpkg+` préconfiguré pour l'utilisation des dépôts MLED.

En ligne de commande, utilisez le navigateur Links pour accéder aux dépôts distants :

```
# links http://www.microlinux.fr/microlinux/
```

Sur un système 32-bits, naviguez vers le répertoire `desktop-14.2-32bit/slackware/ap`. Si vous utilisez un système 64-bits, allez dans `desktop-14.2-64bit/slackware64/ap/`. Téléchargez le paquet `slackpkg+` depuis le dépôt (en utilisant la touche `[D]` comme *Download* dans le navigateur Links) et installez-le :

```
# installpkg slackpkg+-1.7.0-noarch-1_microlinux.txz
```

Éditez `/etc/slackpkg/mirrors` et sélectionnez un miroir Slackware en fonction de votre pays, par exemple :

```
# /etc/slackpkg/mirrors
...
# GERMANY (DE)
ftp://ftp.fu-berlin.de/unix/linux/slackware/slackware-14.2/
...
```

Sur un système 64-bit :

```
# /etc/slackpkg/mirrors
...
# GERMANY (DE)
ftp://ftp.fu-berlin.de/unix/linux/slackware/slackware64-14.2/
...
```

Assurez-vous de ne choisir qu'un seul miroir pour Slackware stable.[1]

Récupérez les clés GPG :

```
# slackpkg update gpg
```

1 Si vous utilisez MLED en France, optez pour le miroir ftp.fu-berlin.de ou mirror.switch.ch. Le miroir OVH est inutilisable comme à peu près tout ce qui vient de chez OVH.

Mettez à jour les informations sur les paquets disponibles :

```
# slackpkg update
```

Élaguer l'installation de base

Les sous-répertoires `desktop-14.2-32bit/tools/` et `desktop-14.2-64bit/tools/` fournissent chacun un script `trim.sh` qui se charge de deux choses :

- installer les paquets de base requis ;
- supprimer les paquets de base superflus.

Élaguez votre système Slackware de base :

```
# cd microlinux/desktop-14.2-32bit/tools/
# ./trim.sh
```

Ou alors :

```
# cd microlinux/desktop-14.2-64bit/tools/
# ./trim.sh
```

Notez en passant que les scripts `trim.sh` et les choix de paquets dans les arborescences respectives `32bit` et `64bit` sont identiques.

Installez manuellement deux paquets qui figurent dans le dépôt `extra/` officiel :

```
# slackpkg install mplayerplug-in recordmydesktop
```

Remarque sur les paquets non inclus

MLED ne repose pas sur la sélection complète de paquets Slackware. Voici tout ce qui n'est pas installé :

- tout ce qui est relatif à KDE, puisque MLED est basé sur Xfce ;
- une panoplie de polices exotiques, qui encombrent le sélecteur de polices de LibreOffice ;
- quelques applications X11 comme Seamonkey, Blueman, Gnuchess, etc.

Bien évidemment, si cette sélection ne vous convient pas, vous pouvez toujours installer ces paquets (Seamonkey, Blueman, polices exotiques, etc.) avec un simple `slackpkg install <paquet>`.

Mettre à jour le système de base

À présent, lancez la mise à jour du système de base Slackware :

```
# slackpkg upgrade-all
```

Certains paquets Slackware comme `audacious-plugins`, `MPlayer` ou `tumbler` sont « mis à jour » c'est-à-dire remplacés par un paquet MLED. Ne vous inquiétez pas, c'est tout à fait normal. Ces

paquets ont été recompilés pour offrir plus de fonctionnalités.

Installer les paquets MLED

Installez le jeu complet de paquets MLED comme ceci :

```
# slackpkg install microlinux-desktop
```

Maintenant que nous sommes arrivés au terme de l'installation initiale, il reste encore plusieurs choses à faire avant de pouvoir utiliser notre nouveau système. Les explications qui suivent concernent une série de détails cruciaux dans la configuration. Là encore, n'hésitez pas à faire un tour sur le *Projet de Documentation Slackware* à l'adresse http://docs.slackware.com pour vous renseigner de manière plus approfondie.

Vérifier les variables d'environnement

Dans une installation par défaut, les variables `LANG` et `LC_COLLATE` sont déjà définies comme `fr_FR.UTF8`, étant donné que MLED est surtout utilisé en France. Si vous souhaitez changer ceci, il faudra éditer le fichier `/etc/profile.d/lang.sh` :

```
# /etc/profile.d/lang.sh
...
export LANG=fr_FR.utf8
export LC_COLLATE=fr_FR.utf8
```

Nettoyer le menu des applications

Les sous-répertoires `desktop-14.2-32bit/tools/` et `desktop-14.2-64bit/tools/` fournissent chacun l'utilitaire `cleanmenu.sh`, un petit script Bash qui se charge de nettoyer les entrées du menu des applications de façon à les rendre lisibles pour Madame Michu. Lancez ce script :

```
# cd microlinux/desktop-14.2-32bit/tools/
# ./cleanmenu.sh
```

Ou alors :

```
# cd microlinux/desktop-14.2-64bit/tools/
# ./cleanmenu.sh
```

Deux remarques :

- Là encore, les scripts figurant dans les arborescences respectives `32bit` et `64bit` sont identiques.
- Le script `cleanmenu.sh` remplace toute une série de fichiers `*.desktop` dans `/usr/share/applications/` et des endroits similaires du système par des entrées de menu « maison ». Pour l'instant, je fournis ces fichiers en anglais, en français et en allemand. Évitez de lancer le script si vous utilisez une autre langue.

Basculer vers le noyau GENERIC

Lors du premier redémarrage, vous avez peut-être jeté un oeil distrait sur les messages affichés à l'écran, et vous avez été vaguement inquiété par quelque chose qui ressemble à ceci :

```
(sda3): error: couldn't mount because of unsupported optional features
```

Jetons un oeil sur l'unique stance actuellement configurée dans le chargeur de démarrage LILO, vers la fin du fichier `/etc/lilo.conf` :

```
# /etc/lilo.conf
...
image = /boot/vmlinuz
  root = /dev/sda3
  label = Linux
  read-only
```

Regardons de près tout ce qu'il y a comme noyaux dans le répertoire `/boot` :

```
# ls -l /boot/vmlinuz*
lrwxrwxrwx 1 root root ... /boot/vmlinuz -> vmlinuz-huge-smp-4.4.14-smp
-rw-r--r-- 1 root root ... /boot/vmlinuz-generic- 4.4.14
-rw-r--r-- 1 root root ... /boot/vmlinuz-generic-smp-4.4.14-smp
-rw-r--r-- 1 root root ... /boot/vmlinuz-huge- 4.4.14
-rw-r--r-- 1 root root ... /boot/vmlinuz-huge-smp-4.4.14-smp
```

Nous utilisons actuellement le noyau `vmlinuz-huge-smp-4.4.14-smp`, la cible du lien symbolique `/boot/vmlinuz`. Il s'agit d'un noyau « prêt-à-porter » avec toutes les options compilées « en dur ».

La bonne pratique, recommandée par Patrick Volkerding, consiste à utiliser le noyau « générique » avec un initrd. Si vous ne savez pas ce que c'est qu'un initrd, imaginez une sorte de besace virtuelle contenant les modules nécessaires pour le démarrage du système. On y trouve notamment le support des systèmes de fichiers comme ext3, ext4, etc.

Dans un premier temps, nous devons savoir quels sont les modules à inclure dans notre initrd. Notre système fournit un utilitaire assez pratique pour cela. Rendez-vous dans le répertoire `/usr/share/mkinitrd` et lancez le script `mkinitrd_command_generator.sh` :

```
# cd /usr/share/mkinitrd
# ./mkinitrd_command_generator.sh
```

Le script nous affiche une commande avec toute une série de paramètres, comme ceci :

```
mkinitrd -c -k 4.4.14-smp -f ext4 -r /dev/sda3 -m mbcache:jbd2:ext4
  -u -o /boot/initrd.gz
```

Retenons les arguments qui suivent l'option `-m`, c'est-à-dire `mbcache:jbd2:ext4`. Ce sont là les modules dont notre machine a besoin pour démarrer, séparés par des « `:` ». Notons-les dans un coin de notre tête ou sur un bout de papier, et éditons un fichier `/etc/mkinitrd.conf` avec notre éditeur préféré. Le répertoire `/etc` contient déjà un modèle (*sample*) que nous allons adapter à nos besoins :

```
# cd /etc
```

```
# cp mkinitrd.conf.sample mkinitrd.conf
```

Décommentons l'ensemble des options des fichiers et éditons-les en fonction de notre configuration. Voici un exemple :

```
# /etc/mkinitrd.conf
SOURCE_TREE="/boot/initrd-tree"
CLEAR_TREE="1"
OUTPUT_IMAGE="/boot/initrd.gz"
KERNEL_VERSION="$(uname -r)"
KEYMAP="fr-latin1"
MODULE_LIST="mbcache:jbd2:ext4"
ROOTDEV="/dev/sda3"
ROOTFS="ext4"
RESUMEDEV="/dev/sda2"
RAID="0"
LVM="0"
UDEV="1"
MODCONF="0"
WAIT="1"
```

Vous aurez probablement deviné que `KEYMAP`, c'est la disposition du clavier dans la console. `MODULE_LIST` contient la liste des modules mentionnée plus haut, séparés par des « `:` ». Attention à ne pas inclure d'espace, ce qui peut entraîner des conséquences fatales. `ROOTDEV` désigne ici la partition principale de notre système (`/dev/sda3`), et `ROOTFS` le système de fichiers (`ext4`) utilisé sur celle-ci. `RESUMEDEV`, c'est notre partition d'échange ou swap (`/dev/sda2`). Une remarque au passage sur une source de confusion potentielle. `RAID="0"` signifie tout simplement que nous n'utilisons pas le RAID, et non pas que nous utilisons du RAID niveau 0. Enfin, si vous n'encryptez pas vos partitions, vous pouvez allègrement supprimer toutes les lignes `LUKS*`.

Créons maintenant notre initrd :

```
# mkinitrd -F
OK: /lib/modules/4.4.14/kernel/fs/mbcache.ko added.
OK: /lib/modules/4.4.14/kernel/fs/jbd2/jbd2.ko added.
OK: /lib/modules/4.4.14/kernel/fs/ext4/ext4.ko added.
26887 blocs
/boot/initrd.gz created.
Be sure to run lilo again if you use it.
```

Remarque : il se peut que certains modules apparaissent « en double » Cela n'a pas de conséquence sur le bon fonctionnement du système.

Il ne nous reste plus qu'à ajouter une stance au chargeur de démarrage LILO pour utiliser le noyau « générique » avec notre initrd nouvellement créé. Éditez `/etc/lilo.conf`, regardez la stance vers la fin du fichier pour la syntaxe, et ajoutez une deuxième stance juste après, comme ceci :

```
# /etc/lilo.conf
...
image = /boot/vmlinuz
  root = /dev/sda3
  label = LinuxHuge
  read-only
image = /boot/vmlinuz-generic-smp-4.4.14-smp
  initrd = /boot/initrd.gz
  root = /dev/sda3
  label = LinuxGeneric
```

```
  read-only
```

Sur un système 64-bit (ou un système non-SMP), la stance ressemblera à ceci :

```
image = /boot/vmlinuz-generic- 4.4.14
  initrd = /boot/initrd.gz
  root = /dev/sda3
  label = Generic
  read-only
```

Prenons en compte la nouvelle configuration de LILO :

```
# lilo
Added LinuxHuge  *
Added LinuxGeneric  +
```

LILO nous affiche « `Added` » pour chacune des stances définies. À partir de là, nous pouvons déjà démarrer sur le nouveau noyau, en prenant soin de choisir « `LinuxGeneric` » dans l'écran de sélection de LILO. Si tout se passe bien, les erreurs concernant d'éventuelles fonctionnalités non supportées au démarrage auront cédé la place au message suivant :

```
/boot/initrd.gz: Loading kernel modules from initrd image:
```

Il ne nous reste plus qu'à finaliser la configuration de LILO. Retournez dans `/etc/lilo.conf` et supprimez allègrement la stance initiale. Renommez éventuellement la stance pointant vers le noyau `GENERIC`, comme ceci :

```
image = /boot/vmlinuz-generic-smp-4.4.14-smp
  initrd = /boot/initrd.gz
  root = /dev/sda3
  label = Linux
  read-only
```

N'oubliez pas de prendre en compte les modifications :

```
# lilo
Added Linux *
```

Installer les Additions Invité VirtualBox

Si vous testez MLED avec VirtualBox, vous devrez installer les Additions Invité (*Guest Additions*) pour pouvoir configurer correctement l'affichage graphique. Dans le menu de votre machine virtuelle, repérez **Périphériques > Installer les Additions Invité**. Ensuite :

```
# mount /dev/cdrom /mnt/cdrom
# cd /mnt/cdrom
# ./VBoxLinuxAdditions.run
```

À partir de là, il suffit de redémarrer.

Ajouter un ou plusieurs utilisateurs

Pour l'instant, notre installation ne comporte que le seul compte root. Pour travailler au quotidien, il nous faut créer au moins un utilisateur « commun mortel ». Le script `adduser` permet de faire ceci très simplement :

```
# adduser kikinovak
Login name for new user: kikinovak
```

Sur un système Slackware, chaque utilisateur nouvellement créé fait automatiquement partie du groupe `users`. Or, l'accès à certains périphériques dépend de l'appartenance à une série de groupes. Lors de la sélection des groupes, appuyez sur la touche **[FlècheHaut]** pour présélectionner une panoplie cohérente de groupes et confirmez par **[Entrée]** :

```
Initial group [ users ]:
Additional UNIX groups:
...
Press ENTER to continue without adding any additional groups
Or press the UP arrow key to add/select/edit additional groups
: audio cdrom floppy plugdev video power netdev lp scanner
```

Vérifiez avec la commande `groups` :

```
# groups kikinovak
kikinovak : users lp floppy audio video cdrom plugdev power netdev scanner
```

X11 pour les impatients

Le serveur graphique X11 ne requiert plus aucun fichier de configuration `/etc/X11/xorg.conf`, et la configuration de X11 n'a plus rien à voir avec la galère que ça a pu être il y a quelques années. Les trois marques de cartes vidéo les plus communes sont Intel, AMD/ATi et NVidia. Jetons un coup d'oeil rapide sur la configuration de ces cartes.

Identifiez votre carte graphique. Voici un exemple :

```
$ /sbin/lspci | grep -i vga
00:02.0 VGA compatible controller: Intel Corporation
82G33/G31 Express Integrated Graphics Controller (rev 0a)
```

Les cartes Intel sont assez répandues. Elles fonctionnent avec le module de noyau `i915` en mode KMS (*Kernel Mode Setting*). Jusqu'à Slackware 14.1 inclus, il fallait ajouter ce module explicitement à l'initrd pour que la carte vidéo fonctionne correctement. Ce n'est plus nécessaire depuis Slackware 14.2.

Si vous disposez d'une carte AMD/ATi, les choses se passeront de manière similaire :

```
$ /sbin/lspci | grep -i vga
01:05.0 VGA compatible controller: AMD/ATI
[Advanced Micro Devices, Inc.] RS780C [Radeon 3100]
```

Ce genre de carte peut être configurée soit avec le driver libre `radeon`, soit avec le driver propriétaire `fglrx` fourni par AMD. Le driver `radeon` fonctionne plutôt bien.

Pour les cartes NVidia, les choses se passent un peu différemment :

```
$ /sbin/lspci | grep -i vga
01:00.0 VGA compatible controller: NVIDIA Corporation GF119
[GeForce GT 520] (rev a1)
```

En théorie, les cartes NVidia peuvent être configurées à l'aide du driver libre `nouveau`. En pratique, j'ai trouvé que les performances de ces drivers laissaient à désirer. En contrepartie, les drivers propriétaires `nvidia` fournis par le constructeur NVidia fonctionnent très bien, et je vous conseille donc de les utiliser.

Le *Projet de Documentation Slackware* propose des tutos détaillés pour l'installation et la configuration des drivers propriétaires `fglrx` et `nvidia`, je ne vais donc pas réinventer la roue ici.

Une fois qu'on s'est assuré que les drivers sont installés proprement, on peut se connecter en tant qu'utilisateur normal et essayer de démarrer le serveur X :

```
$ startx
```

En fonction de votre choix lors de l'installation, c'est WindowMaker ou Fluxbox qui va démarrer. Si jamais vous souhaitez réafficher l'écran de sélection des gestionnaires de fenêtres, utilisez la commande suivante :

```
$ xwmconfig
```

Dans votre environnement graphique, ouvrez un terminal et testez la configuration vidéo :

```
$ glxinfo | head -n 3
name of display: :0
display: :0  screen: 0
direct rendering: Yes
```

Enfin, vous souhaiterez peut-être définir la disposition du clavier pour l'environnement graphique. Dans ce cas, on va partir d'un fragment de fichier de configuration :

```
# cd /etc/X11/xorg.conf.d
# cp /usr/share/X11/xorg.conf.d/90-keyboard-layout.conf .
```

Éditez ce bout de fichier en fonction de vos besoins. Pour un clavier français, il faudra indiquer ceci :

```
Option "XkbLayout" "fr"
```

Sur ma station de travail, le clavier suisse français par défaut est défini comme ceci :

```
Option "XkbLayout" "ch"
Option "XkbVariant" "fr"
```

Réinvoquez `startx` pour tester la configuration du clavier en mode graphique. Si tout s'est bien passé, nous pouvons désormais définir Xfce comme environnement graphique par défaut :

```
$ xwmconfig
```

Basculez vers le niveau d'exécution 4 en éditant `/etc/inittab` :

```
# /etc/inittab
...
# Default runlevel. (Do not set to 0 or 6)
id:4:initdefault:
...
```

Deux remarques importantes sur le niveau d'exécution 4 :

- Le paquet `lxdm` installe un fichier d'initialisation personnalisé `/etc/rc.d/rc.4`. Si vous voyez l'horrible gestionnaire de connexion XDM au lieu de LXDM, allez dans le répertoire `/etc/rc.d` et renommez le fichier `rc.4.new` en `rc.4`.
- Un utilisateur nouvellement créé doit explicitement définir une session Xfce dans le sélecteur de bureaux situé sur le panneau inférieur de LXDM avant la première connexion.

Applications supplémentaires

Une fois que votre système MLED est fonctionnel, vous pouvez chercher des applications supplémentaires à installer en utilisant `slackpkg` :

```
# slackpkg search microlinux-extras
# slackpkg install virtualbox
```

Maintenance de base

Voici quelques conseils pour la maintenance de votre installation MLED. Lisez les fichiers `ChangeLog.txt` à la racine de chaque dépôt de paquets pour vous tenir au courant des mises à jour et des ajouts. Si vous souhaitez en savoir plus sur l'état de votre système, à savoir qu'est-ce qui est installé, qu'est-ce qui ne l'est pas, quelles sont les mises à jour disponibles etc., vous pouvez faire tout cela en une poignée de commandes :

```
# slackpkg update
# slackpkg search microlinux-desktop
```

Ou encore :

```
# slackpkg search microlinux-extras
```

Enfin, les mises à jour des applications peuvent parfois écraser les entrées de menu personnalisées avec un fichier `*.desktop` par défaut. Dans ce cas, il suffit de réinvoquer le script `cleanmenu.sh` :

```
# cd
# cd microlinux
# git pull
# cd desktop-14.2-32bit/tools
# ./cleanmenu.sh
```

Ou bien :

```
# cd desktop-14.2-64bit/tools
# ./cleanmenu.sh
```

Régalez-vous bien avec votre poste de travail MLED flambant neuf !

– *Nicolas Kovacs <info@microlinux.fr>*