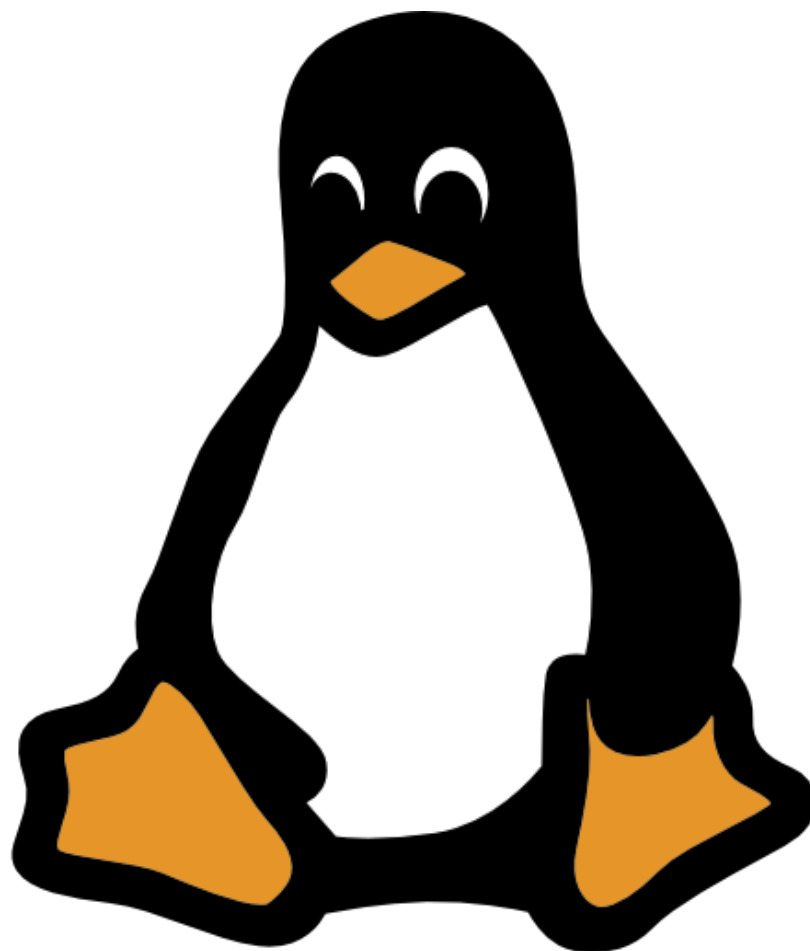


FT_Linux

42 Project - The first project in the KFS series. Build your own linux distribution from scratch. I provided a tutorial to build your own LFS from scratch.

[View the Project on GitHub](#) vvaucoui/FT_Linux



FT_LINUX

Cr  er sa distribution linux from scratch !

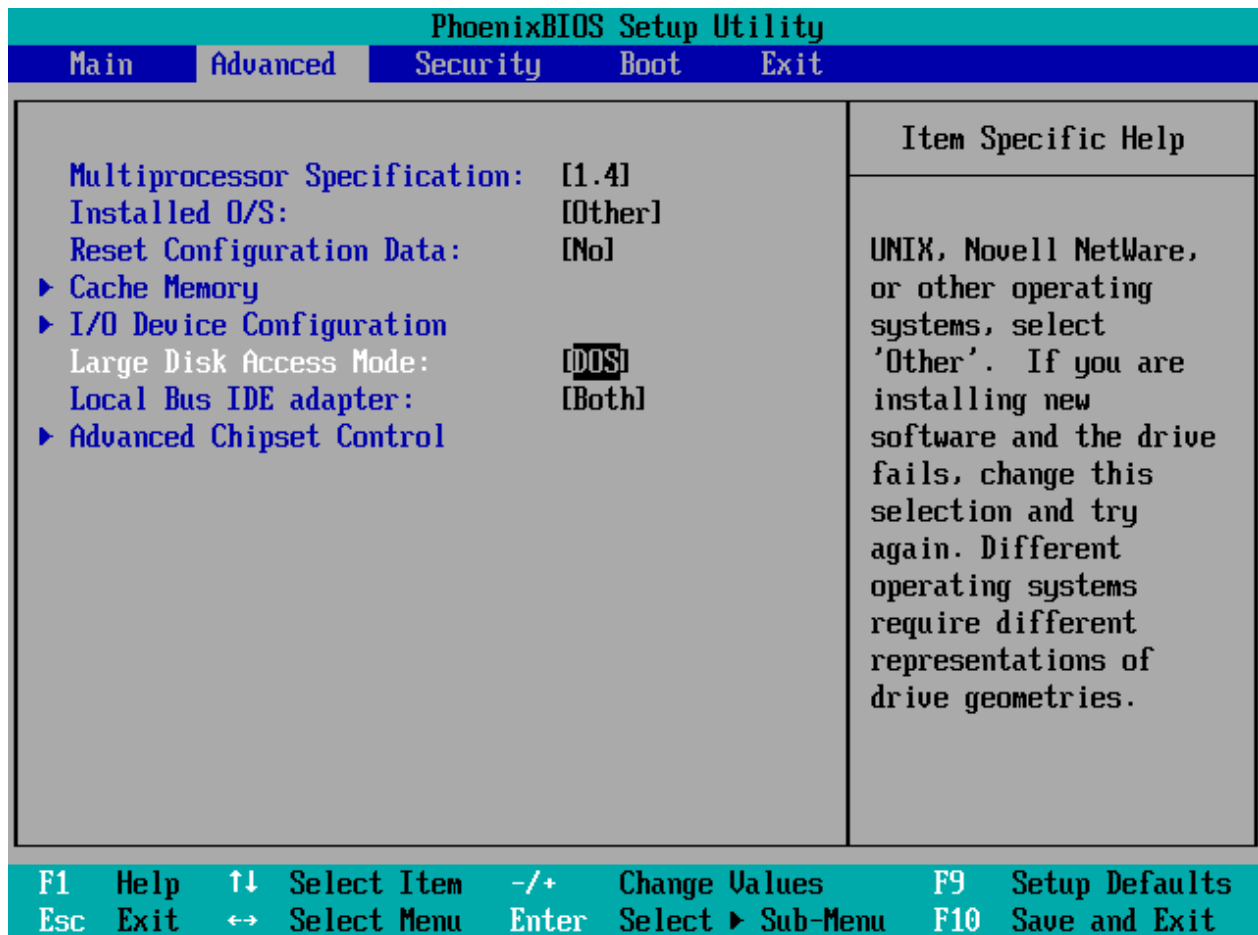
Tutoriel de r  f  rence: [LFS](#)

- LFS **11.1**
- Kernel Linux **5.16.9**

Cr  ation de la VM

- Installez un linux en tant que systeme host pour créer votre LFS (Ubuntu, debian, Arch, Fedora, ...)
- **[SDA]** Distribution LFS: Espace disque: 32GB vdi
- **[SDB]** Système Host. Espace disque: 16GB vdi

⚠ Si vous êtes sur VMWare, pensez à créer un disque virtuel IDE. De plus, il faut impérativement changer dans le bios l'option 'Large Disk Access Mode' paramétré par défaut sur 'DOS' pour 'Other' !



Prérequis

SSH:

```
sudo su
apt-get update
apt-get upgrade -y
apt-get install -y openssh-server
exec <&-
ip address | grep inet
```

- Ajoutez une règle port forwarding sur virtual box
 - Name: SSH
 - Protocol: TCP
 - Host IP: 127.0.0.1
 - Host Port: 2222
 - IP Guest: Empty
 - Port Guest: 22

Paquets à installer:

```
sudo apt-get update -y
sudo apt-get upgrade -y
sudo apt-get dist-upgrade -y
sudo apt-get autoremove --purge -y </dev/null
sudo apt-get autoclean -y </dev/null
sudo apt-get clean -y </dev/null
sudo rm -rf /bin/sh
sudo ln -s /usr/bin/bash /bin/sh
sudo apt-get install apt-file automake build-essential git libclo
```

Important ! Passez directement en mode root

```
sudo su
```

Partitions

Deux solutions s'offrent à vous. L'utilisation de l'outil 'Gparted' ou la commande 'fdisk'. Choisissez celle qui vous convient le mieux.

⚠ Attention: Si vous utilisez Gparted, vous devez utiliser le mode 'MBR' pour que les partitions soient correctement créées. Par défaut, Gparted utilise le mode 'GPT'. Si vous restez en mode 'GPT', il sera impossible d'installer grub.

Si vous reprenez de zéro avec un LFS partiellement effectué, lancez la commande suivante:

```
wipefs -a /dev/sda
```


- fdisk:

```
fdisk /dev/sda
```

```
g
n default default +1M
t 4
n default default +200M
t 2 1
n default default +4G
t 2 19
n default default default
w
```

- gparted:
 - **[SDA1]**: Partition Bios Boot [grub] -> 1MB
 - **[SDA2]**: Partition Boot -> 200MB
 - **[SDA3]**: Partition SWAP -> 4000MB
 - **[SDA4]**: Partition Root -> reste

SWAP: (1 / 8 eme de la taille de la partition root)



Partition	File System	Mount Point	Size	Used	Unused	Flags
/dev/sda1	grub2 core.img		1.00 MiB	—	—	bios_grub
/dev/sda2	ext2		200.00 MiB	38.45 MiB	161.55 MiB	boot, esp
/dev/sda3	linux-swap		4.00 GiB	0.00 B	4.00 GiB	swap
/dev/sda4	ext4	/mnt/lfs	27.80 GiB	5.81 GiB	21.99 GiB	

```
sudo mkfs -v -t ext2 /dev/sda2
sudo mkfs -v -t ext4 /dev/sda4
sudo mkswap /dev/sda3
```

- Verifications:

```
lsblk -o NAME,UUID,FSTYPE,MOUNTPOINT,SIZE /dev/sda
```

- *Resultat:*

NAME	UUID	FSTYPE	MOUNTPOINT	S
sda				
sda1				
sda2	5e75b741-c5e3-4ac9-88a8-9ccc05856e0c	ext2		1
sda3	520b9f4b-aadd-42df-aa43-cca4987df169	swap		
sda4	ea882b4-466e-46c5-8c19-75993770dc8f	ext4		27

Préparations

```
cd
export LFS=/mnt/lfs
mkdir -v $LFS
mount -v -t ext4 /dev/sda4 $LFS
```

```
swapoff /dev/sda3
mkswap /dev/sda3
swapon /dev/sda3
```

- Avant de récupérer les paquets, assurez-vous que le dossier \$LFS/ soit bien vide ! Pour en être sûr, lancez cette commande > 'rm -rvf /mnt/lfs/*'

```
mkdir -v $LFS/sources
chmod -v a+wt $LFS/sources
```

```
curl https://raw.githubusercontent.com/vvaucoul/FT_Linux/main/wget
curl https://raw.githubusercontent.com/vvaucoul/FT_Linux/main/md5sums
```

```
wget --input-file=wget-list --continue --directory-prefix=$LFS/sources
cp md5sums $LFS/sources/md5sums
```

```
pushd $LFS/sources
md5sum -c md5sums
popd
```

Préparations suite...

```

mkdir -v $LFS/tools
ln -sv $LFS/tools /

mkdir -pv $LFS/{etc,var} $LFS/usr/{bin,lib,sbin}

for i in bin lib sbin; do
    ln -sv usr/$i $LFS/$i
done

case $(uname -m) in
    x86_64) mkdir -pv $LFS/lib64 ;;
esac

groupadd lfs
useradd -s /bin/bash -g lfs -m -k /dev/null lfs
printf 'toor\ntoor\n' | passwd lfs

chown -v lfs $LFS/tools
chown -v lfs $LFS/sources

chown -v lfs $LFS/{usr{,/},lib,var,etc,bin,sbin,tools}
case $(uname -m) in
    x86_64) chown -v lfs $LFS/lib64 ;;
esac

```

- Edition du mode LFS

```

su - lfs

cat > ~/.bash_profile << "EOF"
exec env -i HOME=$HOME TERM=$TERM PS1='\u:\w\$ ' /bin/bash
EOF

cat > ~/.bashrc << "EOF"
set +h
umask 022
LFS=/mnt/lfs
LC_ALL=POSIX
LFS_TGT=$(uname -m)-lfs-linux-gnu

```

```
PATH=/usr/bin
if [ ! -L /bin ]; then PATH=/bin:$PATH; fi
PATH=$LFS/tools/bin:$PATH
CONFIG_SITE=$LFS/usr/share/config.site
export LFS LC_ALL LFS_TGT PATH CONFIG_SITE
EOF

source ~/.bash_profile
export MAKEFLAGS='-j4'

exec <&-
```

- Edition du mode ROOT

```
sudo su

cat > ~/.bash_profile << "EOF"
exec env -i HOME=$HOME TERM=$TERM PS1='\u:\w\$ ' /bin/bash
EOF

cat > ~/.bashrc << "EOF"
set +h
umask 022
LFS=/mnt/lfs
LC_ALL=POSIX
LFS_TGT=$(uname -m)-lfs-linux-gnu
PATH=/usr/bin
if [ ! -L /bin ]; then PATH=/bin:$PATH; fi
PATH=$LFS/tools/bin:$PATH
CONFIG_SITE=$LFS/usr/share/config.site
export LFS LC_ALL LFS_TGT PATH CONFIG_SITE
EOF

source ~/.bash_profile
export MAKEFLAGS='-j4'

exec <&-
```

```
echo "dash dash/sh boolean false" | debconf-set-selections
```

```
DEBIAN_FRONTEND=noninteractive dpkg-reconfigure dash
apt-get install -y gawk
apt-get install -y bison
apt-get install -y build-essential
apt-get update && apt-get upgrade -y
curl http://www.linuxfromscratch.org/lfs/view/stable/chapter02/h
bash version-check.sh | grep not
```

Création du système temporaire

- Avant de lancer ces scripts, repassez sur le shell d'origine puis vérifiez que les variables d'environnement 'LFS' et 'LFS_TGT' existent et sont correctement mises dans le shell LFS & ROOT.
- Pensez à bien exécuter le shell lfs via l'utilisateur root (sudo su)

```
chmod 755 /etc/sudoers
echo "lfs      ALL=(ALL:ALL) ALL" >> /etc/sudoers
sudo su - lfs
export LFS=/mnt/lfs
export MAKEFLAGS='-j4'
cd $LFS/sources/
```

- Avant de lancer ces scripts, vous pouvez lancer le script [check-lfs-initialisation.sh](#) pour vérifier si toutes les variables ont bien été initialisées. Sinon, les paquets qui seront installés vont écraser ceux qui se trouvent sur votre distribution HOST.

Lancez les scripts:

⚠ *Attention: L'installation des paquets peut être très long. Pour connaître le temps d'installation de tous les paquets, utilisez le script [get-time.sh](#) en indiquant le temps de compilation du premier paquet, binutils... Pour obtenir le temps de compilations du paquet binutils, lancez le script [binutils-time-calculator.sh](#)*

- [install_softwares.sh](#)
- [install_softwares_02.sh](#)
- [install_softwares_03.sh](#)

Création des outils temporaires supplémentaires


```
# Revenir au shell root
exec <&-

chown -R root:root $LFS/{usr,lib,var,etc,bin,sbin,tools}
case $(uname -m) in
    x86_64) chown -R root:root $LFS/lib64 ;;
esac

export LFS=/mnt/lfs
mkdir -pv $LFS/{dev,proc,sys,run}

mknod -m 600 $LFS/dev/console c 5 1
mknod -m 666 $LFS/dev/null c 1 3
```

Entrer dans le mode 'CHROOT'

```
mount -v --bind /dev $LFS/dev

mount -v --bind /dev/pts $LFS/dev/pts
mount -vt proc proc $LFS/proc
mount -vt sysfs sysfs $LFS/sys
mount -vt tmpfs tmpfs $LFS/run

if [ -h $LFS/dev/shm ]; then
    mkdir -pv $LFS/$(readlink $LFS/dev/shm)
fi
```

```
sudo chroot "$LFS" /usr/bin/env -i \
    HOME=/root \
    TERM="$TERM" \
    PS1='(lfs chroot) \u:\w\$ ' \
    PATH=/usr/bin:/usr/sbin \
    /bin/bash --login
```

Après avoir effectuer le chroot vous devriez avoir un prompt nommé: '(lfs chroot) I have no name!:/#'. Si ce n'est pas le cas ou que vous n'arrivez pas à accéder à la commande chroot, c'est que quelque chose ne s'est pas bien passé lors de l'installation.

```
mkdir -pv /{boot,home,mnt,opt,svr}

mkdir -pv /etc/{opt,sysconfig}
mkdir -pv /lib/firmware
mkdir -pv /media/{floppy,cdrom}
mkdir -pv /usr/{,local/}{include,src}
mkdir -pv /usr/local/{bin,lib,sbin}
mkdir -pv /usr/{,local/}share/{color,dict,doc,info,locale,man}
mkdir -pv /usr/{,local/}share/{misc,terminfo,zoneinfo}
mkdir -pv /usr/{,local/}share/man/man{1..8}
mkdir -pv /var/{cache,local,log,mail,opt,spool}
mkdir -pv /var/lib/{color,misc,locate}

ln -sfv /run /var/run
ln -sfv /run/lock /var/lock

install -dv -m 0750 /root
install -dv -m 1777 /tmp /var/tmp
```

```
ln -sv /proc/self/mounts /etc/mtab

cat > /etc/hosts << EOF
127.0.0.1 localhost $(hostname)
::1      localhost
EOF

cat > /etc/passwd << "EOF"
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/dev/null:/usr/bin/false
daemon:x:6:6:Daemon User:/dev/null:/usr/bin/false
messagebus:x:18:18:D-Bus Message Daemon User:/run/dbus:/usr/bin/
uidd:x:80:80:UUID Generation Daemon User:/dev/null:/usr/bin/fal
nobody:x:99:99:Unprivileged User:/dev/null:/usr/bin/false
EOF

cat > /etc/group << "EOF"
root:x:0:
bin:x:1:daemon
sys:x:2:
```

```
kmem:x:3:
tape:x:4:
tty:x:5:
daemon:x:6:
floppy:x:7:
disk:x:8:
lp:x:9:
dialout:x:10:
audio:x:11:
video:x:12:
utmp:x:13:
usb:x:14:
cdrom:x:15:
adm:x:16:
messagebus:x:18:
input:x:24:
mail:x:34:
kvm:x:61:
uudd:x:80:
wheel:x:97:
nogroup:x:99:
users:x:999:
EOF

echo "tester:x:101:101:./home/tester:/bin/bash" >> /etc/passwd
echo "tester:x:101:" >> /etc/group
install -o tester -d /home/tester

exec /usr/bin/bash --login

touch /var/log/{btmp,lastlog,faillog,wtmp}
chgrp -v utmp /var/log/lastlog
chmod -v 664 /var/log/lastlog
chmod -v 600 /var/log/btmp
```

Lancez ensuite le script:

- [install_softwares_04.sh](#)

Nettoyage & sauvegarde du systeme temporaire

```
rm -rf /usr/share/{info,man,doc}/*  
find /usr/{lib,libexec} -name \*.la -delete  
rm -rf /tools
```

- Sauvegarde (Optionnel)

Temps de sauvegarde ~10 minutes

```
exit  
umount $LFS/dev/pts  
umount $LFS/{sys,proc,run,dev}  
  
cd $LFS  
tar -cJpf $HOME/lfs-temp-tools-11.1.tar.xz .
```

- Restauration du systeme (Optionnel)

```
cd $LFS  
rm -rf ./*  
tar -xpf $HOME/lfs-temp-tools-11.1.tar.xz  
mount -v -t ext4 /dev/sda4 $LFS  
  
mount -v --bind /dev $LFS/dev  
  
mount -v --bind /dev/pts $LFS/dev/pts  
mount -vt proc proc $LFS/proc  
mount -vt sysfs sysfs $LFS/sys  
mount -vt tmpfs tmpfs $LFS/run
```

Construction du système LFS

```
grep -l -e 'libfoo.*deleted' /proc/*/maps |  
tr -cd 0-9\\n | xargs -r ps u
```

Avant de lancer le prochain script, pensez à faire une backup de votre VM ! Il se peut que le système host soit corrompu lors de l'installation.

Lancez ensuite le script:

- [install_software_05.sh](#)

Une fois fini, via le nouveau bash installé, rendez vous dans le dossier /sources et exécutez le script suivant:

- [install_software_06.sh](#)

Ce script met beaucoup de temps pour se finir. Pour savoir où vous en êtes dans le processus, "GCC", référez-vous au document : [GCC-GNU-TEST-RESULTS](#) (archive gzip'd text file)

- Installation des paquets additionnels (Optionnel)

L'installation de ces paquets est optionnel.

```
# Ouvrez un nouveau terminal ou repassez au shell par défaut
sudo su
cd
curl https://raw.githubusercontent.com/vvaucoul/FT_Linux/main/wget
wget --input-file=wget-additional-list --continue --directory-pr

# Repassez en Chroot
sudo chroot "$LFS" /usr/bin/env -i \
    HOME=/root TERM="$TERM" \
    PS1='(lfs chroot) \u:\w\$ ' \
    PATH=/bin:/usr/bin:/sbin:/usr/sbin \
    /bin/bash --login
cd /sources

# Puis lancez le script install-additional-software.sh
```

- [install-additional-software.sh](#)

Si vous souhaitez ne pas déboguer les logiciels systèmes, vous pouvez libérer 2GO en lançant le script suivant: cleanup_software.sh

```
rm -rf /tmp/*
find /usr/lib /usr/libexec -name \*.la -delete
```

```
find /usr -depth -name $(uname -m)-lfs-linux-gnu\* | xargs rm -r  
userdel -r tester
```

Configuration du système

```
tar xvf lfs-bootscripts-20210608.tar.xz  
cd lfs-bootscripts-20210608  
make install  
cd ..  
rm -rf lfs-bootscripts-20210608
```

```
bash /usr/lib/udev/init-net-rules.sh
```

```
cd /etc/sysconfig/  
cat > ifconfig.eth0 << "EOF"  
ONBOOT=yes  
IFACE=eth0  
SERVICE=ipv4-static  
IP=192.168.1.2  
GATEWAY=192.168.1.1  
PREFIX=24  
BROADCAST=192.168.1.255  
EOF  
  
cat > /etc/resolv.conf << "EOF"  
# Begin /etc/resolv.conf  
  
nameserver 1.1.1.1  
nameserver 1.0.0.1  
  
# End /etc/resolv.conf  
EOF  
  
HOST_NAME="vvaucoul"  
echo $HOST_NAME > /etc/hostname  
  
# set up /etc/hosts  
printf "\n"  
# Begin /etc/hosts\n\
```

```
\n\
127.0.0.1 localhost\n\
$NETW_IP $HOST_NAME\n\
::1      localhost ip6-localhost ip6-loopback\n\
ff02::1  ip6-allnodes\n\
ff02::2  ip6-allrouters\n\
\n\
# End /etc/hosts\n\
"> /etc/hosts

# configuring sysvinit
cat > /etc/inittab << "EOF"
# Début de /etc/inittab

id:3:initdefault:

si::sysinit:/etc/rc.d/init.d/rc S

l0:0:wait:/etc/rc.d/init.d/rc 0
l1:S1:wait:/etc/rc.d/init.d/rc 1
l2:2:wait:/etc/rc.d/init.d/rc 2
l3:3:wait:/etc/rc.d/init.d/rc 3
l4:4:wait:/etc/rc.d/init.d/rc 4
l5:5:wait:/etc/rc.d/init.d/rc 5
l6:6:wait:/etc/rc.d/init.d/rc 6

ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now

su:S016:once:/sbin/sulogin

1:2345:respawn:/sbin/agetty --noclear tty1 9600
2:2345:respawn:/sbin/agetty tty2 9600
3:2345:respawn:/sbin/agetty tty3 9600
4:2345:respawn:/sbin/agetty tty4 9600
5:2345:respawn:/sbin/agetty tty5 9600
6:2345:respawn:/sbin/agetty tty6 9600

# Fin de /etc/inittab
EOF
```

```
cat > /etc/sysconfig/clock << "EOF"
# Début de /etc/sysconfig/clock

UTC=1

# Mettez ici les options que vous pourriez avoir besoin de donner
# comme le type de l'horloge matérielle de la machine pour les A
CLOCKPARAMS=

# Fin de /etc/sysconfig/clock
EOF

cat > /etc/sysconfig/console << "EOF"
# Début de /etc/sysconfig/console

UNICODE="1"
KEYMAP="de-latin1"
KEYMAP_CORRECTIONS="euro2"
LEGACY_CHARSET="iso-8859-15"
FONT="LatArCyrHeb-16 -m 8859-15"

# Fin de /etc/sysconfig/console
EOF

# set locale
CHOSEN_CHAR_MAP="en_US.iso88591"
LC_ALL=$CHOSEN_CHAR_MAP locale charmap
LC_ALL=$CHOSEN_CHAR_MAP locale language
LC_ALL=$CHOSEN_CHAR_MAP locale charmap
LC_ALL=$CHOSEN_CHAR_MAP locale int_curr_symbol
LC_ALL=$CHOSEN_CHAR_MAP locale int_prefix

# create /etc/profile
printf "\n\
# Begin /etc/profile\n\
\n\
export LANG=$CHOSEN_CHAR_MAP\n\
\n\
# End /etc/profile\n\
" > /etc/profile
```



```
cat > /etc/inputrc << "EOF"
# Début de /etc/inputrc
# Modifié par Chris Lynn <roryo@roryo.dynup.net>

# Permettre à l'invite de commande d'aller à la ligne
set horizontal-scroll-mode Off

# Activer l'entrée sur 8 bits
set meta-flag On
set input-meta On

# Ne pas supprimer le 8ème bit
set convert-meta Off

# Conserver le 8ème bit à l'affichage
set output-meta On

# « none », « visible » ou « audible »
set bell-style none

# Toutes les indications qui suivent font correspondre la séquer
# d'échappement contenue dans le 1er argument à la fonction
# spécifique de readline
"\e0d": backward-word
"\e0c": forward-word

# Pour la console linux
"\e[1~": beginning-of-line
"\e[4~": end-of-line
"\e[5~": beginning-of-history
"\e[6~": end-of-history
"\e[3~": delete-char
"\e[2~": quoted-insert

# pour xterm
"\e0H": beginning-of-line
"\e0F": end-of-line

# pour Konsole
```

```
"\e[H": beginning-of-line
```

```
"\e[F": end-of-line
```

```
# Fin de /etc/inputrc
```

```
EOF
```

```
cat > /etc/sysconfig/rc.site << "EOF"
```

```
# rc.site
```

```
# Optional parameters for boot scripts.
```

```
# Distro Information
```

```
# These values, if specified here, override the defaults
```

```
#DISTRO="Linux From Scratch" # The distro name
```

```
#DISTRO_CONTACT="lfs-dev@linuxfromscratch.org" # Bug report address
```

```
#DISTRO_MINI="LFS" # Short name used in filenames for distro core
```

```
# Define custom colors used in messages printed to the screen
```

```
# Please consult `man console_codes` for more information
```

```
# under the "ECMA-48 Set Graphics Rendition" section
```

```
#
```

```
# Warning: when switching from a 8bit to a 9bit font,
```

```
# the linux console will reinterpret the bold (1;) to
```

```
# the top 256 glyphs of the 9bit font. This does
```

```
# not affect framebuffer consoles
```

```
# These values, if specified here, override the defaults
```

```
#BRACKET="\033[1;34m" # Blue
```

```
#FAILURE="\033[1;31m" # Red
```

```
#INFO="\033[1;36m" # Cyan
```

```
#NORMAL="\033[0;39m" # Grey
```

```
#SUCCESS="\033[1;32m" # Green
```

```
#WARNING="\033[1;33m" # Yellow
```

```
# Use a colored prefix
```

```
# These values, if specified here, override the defaults
```

```
#BMPREFIX=" "
```

```
#SUCCESS_PREFIX="${SUCCESS} * ${NORMAL} "
```

```
#FAILURE_PREFIX="${FAILURE}*****${NORMAL} "
```

```
#WARNING_PREFIX="${WARNING} *** ${NORMAL} "
```

```
# Manually seet the right edge of message output (characters)
# Useful when resetting console font during boot to override
# automatic screen width detection
#COLUMNS=120

# Interactive startup
#IPROMPT="yes" # Whether to display the interactive boot prompt
#itime="3"      # The amount of time (in seconds) to display the p

# The total length of the distro welcome string, without escape
#wlen=$(echo "Welcome to ${DISTRO}" | wc -c )
#welcome_message="Welcome to ${INFO}${DISTRO}${NORMAL}"

# The total length of the interactive string, without escape coo
#ilen=$(echo "Press 'I' to enter interactive startup" | wc -c )
#i_message="Press '${FAILURE}I${NORMAL}' to enter interactive st

# Set scripts to skip the file system check on reboot
#FASTBOOT=yes

# Skip reading from the console
#HEADLESS=yes

# Write out fsck progress if yes
#VERBOSE_FSCK=no

# Speed up boot without waiting for settle in udev
#OMIT_UDEV_SETTLE=y

# Speed up boot without waiting for settle in udev_retry
#OMIT_UDEV_RETRY_SETTLE=yes

# Skip cleaning /tmp if yes
#SKIPTMPCLEAN=no

# For setclock
#UTC=1
#CLOCKPARAMS=
```

```

# For consolelog (Note that the default, 7=debug, is noisy)
#LOGLEVEL=7

# For network
#HOSTNAME=mylfs

# Delay between TERM and KILL signals at shutdown
#KILLDELAY=3

# Optional syslogd parameters
#SYSKLOGD_PARMS="-m 0"

# Console parameters
#UNICODE=1
#KEYMAP="de-latin1"
#KEYMAP_CORRECTIONS="euro2"
#FONT="lat0-16 -m 8859-15"
#LEGACY_CHARSET=
EOF

cat > /etc/shells << "EOF"
# Begin /etc/shells

/bin/sh
/bin/bash

# Fin de /etc/shells
EOF

```

Rendre LFS Bootable

```

# create the fstab file
printf "\
# Begin /etc/fstab\n\
\n\
# file system  mount-point  type      options      dump  f
#
\n\
/dev/sda2      /boot          ext2      defaults     0      0

```

```

/dev/sda4      /          ext4      defaults          1      1
/dev/sda3      swap        swap      pri=1             0      0
proc           /proc       proc      nosuid,noexec,nodev 0      0
sysfs          /sys        sysfs     nosuid,noexec,nodev 0      0
devpts         /dev/pts    devpts    gid=5,mode=620     0      0
tmpfs          /run        tmpfs     defaults           0      0
devtmpfs       /dev        devtmpfs  mode=0755,nosuid   0      0
\n\
# End /etc/fstab\n
" > /etc/fstab
cat /etc/fstab

```

```

cd $LFS/sources
tar xvf linux-5.16.9.tar.xz
cd linux-5.16.9
make mrproper
make defconfig
make menuconfig
# Ajoutez votre login

```

Avant de compiler le kernel linux, pensez bien à vérifier que tout ce qui est listé dans le fichier [kernel-fonctionnalities](#) est bien activé et désactivé ! Sinon, il se peut que votre système ne boot pas.

```

make
make modules_install

# Revenir au Shell root
exec <&-
umount /boot
mount /dev/sda2 /boot
mount --bind /boot /mnt/lfs/boot

export LFS=/mnt/lfs
sudo chroot "$LFS" /usr/bin/env -i \
    HOME=/root TERM="$TERM" \
    PS1='(lfs chroot) \u:\w\$ ' \
    PATH=/bin:/usr/bin:/sbin:/usr/sbin \
    /bin/bash --login

```

```
cd /sources/linux-5.16.9

# For 64 Bits
cp -iv arch/x86_64/boot/bzImage /boot/vmlinuz-x64-5.16.9-vvaucoul

# For 32 Bits
cp -iv arch/x86/boot/bzImage /boot/vmlinuz-5.16.9-vvaucoul

cp -iv System.map /boot/System.map-5.16.9
cp -iv .config /boot/config-5.16.9
install -d /usr/share/doc/linux-5.16.9
cp -r Documentation/* /usr/share/doc/linux-5.16.9

cd ..
chown -R 0:0 linux-5.16.9

install -v -m755 -d /etc/modprobe.d
cat > /etc/modprobe.d/usb.conf << "EOF"
# Début de /etc/modprobe.d/usb.conf

install ohci_hcd /sbin/modprobe ehci_hcd ; /sbin/modprobe -i ohci_hcd
install uhci_hcd /sbin/modprobe ehci_hcd ; /sbin/modprobe -i uhci_hcd

# Fin de /etc/modprobe.d/usb.conf
EOF
```

Setup GRUB

```
grub-install /dev/sda

cat > /boot/grub/grub.cfg << "EOF"
# Début de /boot/grub/grub.cfg
set default=0
set timeout=5

insmod ext2
set root=(hd0,gpt2)

menuentry "GNU/Linux, Linux 5.16.9-vvaucoul" {
```

```
linux /vmlinuz-x64-5.16.9-vvaucoul root=/dev/sda4 ro  
}  
EOF
```

Pour une version 32Bits du kernel, enlevez le 'x64' de votre vmlinuz.

Finalisations

```
echo 11.1 > /etc/lfs-release  
cat > /etc/lsb-release << "EOF"  
DISTRIB_ID="Linux From Scratch"  
DISTRIB_RELEASE="11.1"  
DISTRIB_CODENAME="vvaucoul"  
DISTRIB_DESCRIPTION="Linux From Scratch"  
EOF  
  
cat > /etc/os-release << "EOF"  
NAME="Linux From Scratch"  
VERSION="11.1"  
ID=lfs  
PRETTY_NAME="Linux From Scratch 11.1"  
VERSION_CODENAME="vvaucoul"  
EOF  
  
# exit the chroot  
logout  
  
# unmount everything  
umount -v $LFS/dev/pts  
umount -v $LFS/dev  
umount -v $LFS/run  
umount -v $LFS/proc  
umount -v $LFS/sys  
umount -v $LFS  
  
# reboot  
shutdown -r now
```

This project is maintained by [vvaucoul](#)

Hosted on GitHub Pages — Theme by [orderedlist](#)