# Linux From Scratch pour Raspberry Pi

**Table of Contents**

Cet article déroule les étapes nécessaires à la construction d'un système LFS spécifique à Pi.

Ce guide a été testé et vérifié sur une carte SD SanDisk de 32 Go avec Raspbian Buster (2020-05-27) comme système hôte, mais la plupart des distributions Linux devraient pouvoir fonctionner comme plate-forme hôte pour faire un build LFS si les outils requis peuvent être ajouté. La taille minimale de la carte SD est probablement de 4 Go.

Le temps de construction total est compris entre 60 heures sur RPi 1 et 8 heures sur RPi 4.

## Image de base de la carte SD LFS

Le site https://intestinate.com/pilfs/ propose des images précompilées de la toolchain LFS, on peut utiliser une de ce images comme base pour un projet ou comme environnement de construction dans lequel construire une distribution en suivant le guide PiLFS.

Pour se connecter en tant que root utiliser le mot de passe **pilfs**

Pour se connecter à un point d'accès Wi-Fi renseigner le mot de passe dans le fichier `wpa_supplicant`:

```
wpa_passphrase your_SSID your_password > /etc/sysconfig/wpa_supplicant-wlan0.conf
```

DHCP est actif par défaut, pour utiliser une adresse statique, modifier `/etc/sysconfig/static.eth0`

Quelques packages supplémentaires ont été ajoutés pour plus de commodité, par ex. sshd, wget, tmux, git et la mise à jour rpi de Hexxeh

Il y en a une liste dans `/usr/src/installed_tarballs.txt`

## Construction de la Toolchain avec les sources

Le site https://intestinate.com/pilfs/ propose un ensemble de scriptes permettant de construire un toolchain LFS

<codedoc> <codedoc toggle wget-list: liste des paquets>
https://download.savannah.gnu.org/releases/acl/acl-2.3.1.tar.xz
https://download.savannah.gnu.org/releases/attr/attr-2.5.1.tar.gz
https://ftp.gnu.org/gnu/autoconf/autoconf-2.71.tar.xz
https://ftp.gnu.org/gnu/automake/automake-1.16.4.tar.xz
https://ftp.gnu.org/gnu/bash/bash-5.1.8.tar.gz
https://github.com/gavinhoward/bc/releases/download/5.0.0/bc-5.0.0.tar.xz
https://ftp.gnu.org/gnu/binutils/binutils-2.37.tar.xz https://ftp.gnu.org/gnu/bison/bison-3.7.6.tar.xz
https://www.sourceware.org/pub/bzip2/bzip2-1.0.8.tar.gz
https://github.com/libcheck/check/releases/download/0.15.2/check-0.15.2.tar.gz
https://ftp.gnu.org/gnu/coreutils/coreutils-8.32.tar.xz
https://ftp.gnu.org/gnu/dejagnu/dejagnu-1.6.3.tar.gz https://ftp.gnu.org/gnu/diffutils/diffutils-3.8.tar.xz
https://downloads.sourceforge.net/project/e2fsprogs/e2fsprogs/v1.46.4/e2fsprogs-1.46.4.tar.gz
https://sourceware.org/ftp/elfutils/0.185/elfutils-0.185.tar.bz2
https://dev.gentoo.org/~blueness/eudev/eudev-3.2.10.tar.gz
https://prdownloads.sourceforge.net/expat/expat-2.4.1.tar.xz
https://prdownloads.sourceforge.net/expect/expect5.45.4.tar.gz
https://astron.com/pub/file/file-5.40.tar.gz https://ftp.gnu.org/gnu/findutils/findutils-4.8.0.tar.xz
https://github.com/westes/flex/releases/download/v2.6.4/flex-2.6.4.tar.gz
https://ftp.gnu.org/gnu/gawk/gawk-5.1.0.tar.xz
https://ftp.gnu.org/gnu/gcc/gcc-11.2.0/gcc-11.2.0.tar.xz
https://ftp.gnu.org/gnu/gdbm/gdbm-1.20.tar.gz https://ftp.gnu.org/gnu/gettext/gettext-0.21.tar.xz
https://ftp.gnu.org/gnu/glibc/glibc-2.34.tar.xz https://ftp.gnu.org/gnu/gmp/gmp-6.2.1.tar.xz
https://ftp.gnu.org/gnu/gperf/gperf-3.1.tar.gz https://ftp.gnu.org/gnu/grep/grep-3.7.tar.xz
https://ftp.gnu.org/gnu/groff/groff-1.22.4.tar.gz https://ftp.gnu.org/gnu/gzip/gzip-1.10.tar.xz
https://github.com/Mic92/iana-etc/releases/download/20210611/iana-etc-20210611.tar.gz
https://ftp.gnu.org/gnu/inetutils/inetutils-2.1.tar.xz
https://launchpad.net/intltool/trunk/0.51.0/+download/intltool-0.51.0.tar.gz
https://www.kernel.org/pub/linux/utils/net/iproute2/iproute2-5.13.0.tar.xz
https://www.kernel.org/pub/linux/utils/kbd/kbd-2.4.0.tar.xz
https://www.kernel.org/pub/linux/utils/kernel/kmod/kmod-29.tar.xz
https://www.greenwoodsoftware.com/less/less-590.tar.gz
https://www.linuxfromscratch.org/lfs/downloads/development/lfs-bootscripts-20210608.tar.xz
https://www.kernel.org/pub/linux/libs/security/linux-privs/libcap2/libcap-2.53.tar.xz
https://github.com/libffi/libffi/releases/download/v3.4.2/libffi-3.4.2.tar.gz

https://download.savannah.gnu.org/releases/libpipeline/libpipeline-1.5.3.tar.gz
https://ftp.gnu.org/gnu/libtool/libtool-2.4.6.tar.xz https://ftp.gnu.org/gnu/m4/m4-1.4.19.tar.xz
https://ftp.gnu.org/gnu/make/make-4.3.tar.gz
https://download.savannah.gnu.org/releases/man-db/man-db-2.9.4.tar.xz
https://www.kernel.org/pub/linux/docs/man-pages/man-pages-5.13.tar.xz
https://github.com/mesonbuild/meson/releases/download/0.59.1/meson-0.59.1.tar.gz
https://ftp.gnu.org/gnu/mpc/mpc-1.2.1.tar.gz https://www.mpfr.org/mpfr-4.1.0/mpfr-4.1.0.tar.xz
https://ftp.gnu.org/gnu/ncurses/ncurses-6.2.tar.gz
https://github.com/ninja-build/ninja/archive/v1.10.2/ninja-1.10.2.tar.gz
https://www.openssl.org/source/openssl-1.1.1l.tar.gz https://ftp.gnu.org/gnu/patch/patch-2.7.6.tar.xz
https://www.cpan.org/src/5.0/perl-5.34.0.tar.xz
https://pkg-config.freedesktop.org/releases/pkg-config-0.29.2.tar.gz
https://sourceforge.net/projects/procps-ng/files/Production/procps-ng-3.3.17.tar.xz
https://sourceforge.net/projects/psmisc/files/psmisc/psmisc-23.4.tar.xz
https://www.python.org/ftp/python/3.9.6/Python-3.9.6.tar.xz
https://www.python.org/ftp/python/doc/3.9.6/python-3.9.6-docs-html.tar.bz2
https://ftp.gnu.org/gnu/readline/readline-8.1.tar.gz https://ftp.gnu.org/gnu/sed/sed-4.8.tar.xz
https://github.com/shadow-maint/shadow/releases/download/v4.9/shadow-4.9.tar.xz
https://www.infodrom.org/projects/sysklogd/download/sysklogd-1.5.1.tar.gz
https://download.savannah.gnu.org/releases/sysvinit/sysvinit-2.99.tar.xz
https://ftp.gnu.org/gnu/tar/tar-1.34.tar.xz https://downloads.sourceforge.net/tcl/tcl8.6.11-src.tar.gz
https://downloads.sourceforge.net/tcl/tcl8.6.11-html.tar.gz
https://ftp.gnu.org/gnu/texinfo/texinfo-6.8.tar.xz
https://www.iana.org/time-zones/repository/releases/tzdata2021a.tar.gz
https://anduin.linuxfromscratch.org/LFS/udev-lfs-20171102.tar.xz
https://www.kernel.org/pub/linux/utils/util-linux/v2.37/util-linux-2.37.2.tar.xz
https://anduin.linuxfromscratch.org/LFS/vim-8.2.3337.tar.gz
https://cpan.metacpan.org/authors/id/T/TO/TODDR/XML-Parser-2.46.tar.gz
https://tukaani.org/xz/xz-5.2.5.tar.xz https://zlib.net/zlib-1.2.11.tar.xz
https://github.com/facebook/zstd/releases/download/v1.5.0/zstd-1.5.0.tar.gz
https://www.linuxfromscratch.org/patches/lfs/development/binutils-2.37-upstream_fix-1.patch
https://www.linuxfromscratch.org/patches/lfs/development/bzip2-1.0.8-install_docs-1.patch
https://www.linuxfromscratch.org/patches/lfs/development/coreutils-8.32-i18n-1.patch
https://www.linuxfromscratch.org/patches/lfs/development/file-5.40-upstream_fixes-1.patch
https://www.linuxfromscratch.org/patches/lfs/development/glibc-2.34-fhs-1.patch
https://www.linuxfromscratch.org/patches/lfs/development/kbd-2.4.0-backspace-1.patch
https://www.linuxfromscratch.org/patches/lfs/development/perl-5.34.0-upstream_fixes-1.patch
https://www.linuxfromscratch.org/patches/lfs/development/sysvinit-2.99-consolidated-1.patch
https://intestinate.com/pilfs/patches/gcc-9.1.0-rpi1-cpu-default.patch
https://intestinate.com/pilfs/patches/gcc-9.1.0-rpi2-cpu-default.patch
https://intestinate.com/pilfs/patches/gcc-9.1.0-rpi3-cpu-default.patch
https://intestinate.com/pilfs/patches/gcc-9.1.0-rpi4-cpu-default.patch
https://intestinate.com/pilfs/patches/coreutils-8.32-aarch64-fix.patch
https://intestinate.com/pilfs/patches/expect5.45-aarch64-fix.patch
https://intestinate.com/pilfs/scripts/ch5-build.sh https://intestinate.com/pilfs/scripts/ch7-build.sh
https://intestinate.com/pilfs/scripts/pilfs-bootscripts-20190902.tar.xz
https://github.com/raspberrypi/linux/archive/rpi-5.10.y.tar.gz
https://github.com/raspberrypi/firmware/archive/master.tar.gz
https://github.com/raspberrypi/rpi-eeprom/archive/refs/tags/v2021.04.29-138a1.tar.gz </codedoc>

<codedoc toggle ch5-build.sh: Script LFS chapters 5.2 - Binutils to 6.18 - GCC - Pass 2>

# !/bin/bash

# # PiLFS Build Script for LFS Version r11.0-27

# Builds chapters 5.2 - Binutils to 6.18 - GCC - Pass 2

# https://intestinate.com/pilfs

# # Optional parameteres below:

RPI*MODEL=4 # Which Raspberry Pi model are you building for - this selects the right GCC CPU patch. # Put 64 to build for aarch64. PARALLEL*JOBS=4 # Number of parallel make jobs, 1 for RPi1 and 4 for RPi2 and up recommended.

# End of optional parameters

set -o nounset set -o errexit

function prebuild*sanity*check {

```
  if [[ $(whoami) != "lfs" ]] ; then
      echo "Not running as user lfs, you should be!"
      exit 1
  fi
```

```
  if ! [[ -v LFS ]] ; then
      echo "You forgot to set your LFS environment variable!"
      exit 1
  fi
```

```
  if ! [[ -v LFS_TGT ]] || [[ $LFS_TGT != "armv6l-lfs-linux-gnueabihf" &&
$LFS_TGT != "armv7l-lfs-linux-gnueabihf" && $LFS_TGT != "aarch64-lfs-linux-
gnu" ]] ; then
      echo "Your LFS_TGT variable should be set to armv6l-lfs-linux-
gnueabihf for RPi1, armv7l-lfs-linux-gnueabihf for RPi2 - 4 or aarch64-lfs-
linux-gnu for aarch64"
```

```
      exit 1
  fi
```

```
  if ! [[ -d $LFS ]] ; then
      echo "Your LFS directory doesn't exist!"
      exit 1
  fi
```

```
  if ! [[ -d $LFS/sources ]] ; then
      echo "Can't find your sources directory!"
      exit 1
  fi
```

```
  if [[ $(stat -c %U $LFS/sources) != "lfs" ]] ; then
      echo "The sources directory should be owned by user lfs!"
      exit 1
  fi
```

```
  if ! [[ -d $LFS/tools ]] ; then
      echo "Can't find your tools directory!"
      exit 1
  fi
```

```
  if [[ $(stat -c %U $LFS/tools) != "lfs" ]] ; then
      echo "The tools directory should be owned by user lfs!"
      exit 1
  fi
```

```
  if [[ "$RPI_MODEL" == "64" && $(uname -m) != "aarch64" ]] ; then
      echo "You need to build your aarch64 LFS on an aarch64 host system!"
      exit 1
  fi
```

}

function check*tarballs { LIST*OF_TARBALLS=“ binutils-2.37.tar.xz gcc-11.2.0.tar.xz gcc-9.1.0-rpi1-cpu-default.patch gcc-9.1.0-rpi2-cpu-default.patch gcc-9.1.0-rpi3-cpu-default.patch gcc-9.1.0-rpi4-cpu-default.patch mpfr-4.1.0.tar.xz gmp-6.2.1.tar.xz mpc-1.2.1.tar.gz rpi-5.10.y.tar.gz glibc-2.34.tar.xz glibc-2.34-fhs-1.patch m4-1.4.19.tar.xz ncurses-6.2.tar.gz bash-5.1.8.tar.gz coreutils-8.32.tar.xz coreutils-8.32-aarch64-fix.patch diffutils-3.8.tar.xz file-5.40.tar.gz findutils-4.8.0.tar.xz gawk-5.1.0.tar.xz grep-3.7.tar.xz gzip-1.10.tar.xz make-4.3.tar.gz patch-2.7.6.tar.xz sed-4.8.tar.xz tar-1.34.tar.xz xz-5.2.5.tar.xz ”

for tarball in $LIST*OF*TARBALLS ; do

```
  if ! [[ -f $LFS/sources/$tarball ]] ; then
      echo "Can't find $LFS/sources/$tarball!"
      exit 1
  fi
```

done }

function timer {

```
  if [[ $# -eq 0 ]]; then
      echo $(date '+%s')
  else
      local stime=$1
      etime=$(date '+%s')
      if [[ -z "$stime" ]]; then stime=$etime; fi
      dt=$((etime - stime))
      ds=$((dt % 60))
      dm=$(((dt / 60) % 60))
      dh=$((dt / 3600))
      printf '%02d:%02d:%02d' $dh $dm $ds
  fi
```

}

prebuild*sanity*check check_tarballs

if grep 'Swap:' | tr -d ' ' | cut -d ':' -f2) == "000" ; then

```
  echo -e "\nYou are almost certainly going to want to add some swap space before building!"
  echo -e "(See https://intestinate.com/pilfs/beyond.html#addswap for instructions)"
  echo -e "Continue without swap?"
  select yn in "Yes" "No"; do
      case $yn in
          Yes) break;;
          No) exit;;
      esac
  done
```

fi

echo -e “\nThis is your last chance to quit before we start building… continue?” echo “(Note that if anything goes wrong during the build, the script will abort mission)” select yn in “Yes” “No”; do

```
  case $yn in
      Yes) break;;
      No) exit;;
  esac
```

done

total*time=$(timer) sbu*time=$(timer)

echo “# 5.2. Binutils-2.37 - Pass 1” cd $LFS/sources tar -Jxf binutils-2.37.tar.xz cd binutils-2.37 mkdir -v build cd build ../configure –prefix=$LFS/tools \

1. -with-sysroot=$LFS \

2. -target=$LFS*TGT \ –disable-nls \ –disable-werror make -j $PARALLEL*JOBS

make -j 1 install cd $LFS/sources rm -rf binutils-2.37

echo -e “\n=============================” printf 'Your SBU time is: %s\n' $(timer $sbu_time) echo -e “=============================\n”

echo “# 5.3. gcc-11.2.0 - Pass 1” tar -Jxf gcc-11.2.0.tar.xz cd gcc-11.2.0 if "$RPI_MODEL" == "64" ; then

```
  sed -e '/mabi.lp64=/s/lib64/lib/' -i.orig gcc/config/aarch64/t-aarch64-linux
```

else

```
  patch -Np1 -i ../gcc-9.1.0-rpi$RPI_MODEL-cpu-default.patch
```

fi tar -Jxf ../mpfr-4.1.0.tar.xz mv -v mpfr-4.1.0 mpfr tar -Jxf ../gmp-6.2.1.tar.xz mv -v gmp-6.2.1 gmp tar -zxf ../mpc-1.2.1.tar.gz mv -v mpc-1.2.1 mpc mkdir -v build cd build ../configure \

1. -target=$LFS*TGT \ –prefix=$LFS/tools \ –with-glibc-version=2.11 \ –with-sysroot=$LFS \ –with-newlib \ –without-headers \ –enable-initfini-array \ –disable-nls \ –disable-shared \ –disable-multilib \ –disable-decimal-float \ –disable-threads \ –disable-libatomic \ –disable-libgomp \ –disable-libquadmath \ –disable-libssp \ –disable-libvtv \ –disable-libstdcxx \ –enable-languages=c,c++ make -j 1 make install cd .. cat gcc/limitx.h gcc/glimits.h gcc/limity.h > `dirname $($LFS_TGT-gcc -print-libgcc-file-name)`/install-tools/include/limits.h cd $LFS/sources rm -rf gcc-11.2.0 echo “# 5.4. Raspberry Pi Linux API Headers” tar -zxf rpi-5.10.y.tar.gz cd linux-rpi-5.10.y make mrproper make headers find usr/include -name '.*' -delete rm usr/include/Makefile cp -rv usr/include $LFS/usr cd $LFS/sources rm -rf linux-rpi-5.10.y echo “# 5.5. Glibc-2.34” tar -Jxf glibc-2.34.tar.xz cd glibc-2.34 patch -Np1 -i ../glibc-2.34-fhs-1.patch mkdir -v build cd build echo “rootsbindir=/usr/sbin” > configparms ../configure \ –prefix=/usr \ –host=$LFS*TGT \
   1. -build=$(../scripts/config.guess) \
   2. -enable-kernel=3.2 \
   3. -with-headers=$LFS/usr/include \

```
  libc_cv_slibdir=/usr/lib
```

make -j $PARALLEL*JOBS make DESTDIR=$LFS install sed '/RTLDLIST=/s@/usr@@g' -i $LFS/usr/bin/ldd $LFS/tools/libexec/gcc/$LFS*TGT/11.2.0/install-tools/mkheaders

# Compatibility symlink for non ld-linux-armhf awareness

ln -sv ld-2.34.so $LFS/tools/lib/ld-linux.so.3 cd $LFS/sources rm -rf glibc-2.34

echo “# 5.6. Libstdc++ from GCC-11.2.0 - Pass 1” tar -Jxf gcc-11.2.0.tar.xz cd gcc-11.2.0 mkdir -v build cd build ../libstdc++-v3/configure \

1. -host=$LFS*TGT \ –build=$(../config.guess) \ –prefix=/usr \ –disable-multilib \ –disable-nls \ –disable-libstdcxx-pch \ –with-gxx-include-dir=/tools/$LFS*TGT/include/c++/11.2.0

make -j $PARALLEL_JOBS make DESTDIR=$LFS install cd $LFS/sources rm -rf gcc-11.2.0

echo “# 6.2. M4-1.4.19” tar -Jxf m4-1.4.19.tar.xz cd m4-1.4.19 ./configure –prefix=/usr \

1. -host=$LFS*TGT \ –build=$(build-aux/config.guess) make -j $PARALLEL*JOBS

make DESTDIR=$LFS install cd $LFS/sources rm -rf m4-1.4.19

echo “# 6.3. Ncurses-6.2” tar -zxf ncurses-6.2.tar.gz cd ncurses-6.2 sed -i s/mawk *configure mkdir build pushd build ../configure make -j $PARALLELJOBS -C include make -j $PARALLELJOBS -C progs tic popd ./configure –prefix=/usr \ –host=$LFSTGT \ –build=$(./config.guess) \ –mandir=/usr/share/man \ –with-manpage-format=normal \ –with-shared \ –without-debug \ –without-ada \ –without-normal \ –enable-widec make -j $PARALLELJOBS make DESTDIR=$LFS TIC_PATH=$(pwd)/build/progs/tic install echo “INPUT(-lncursesw)” > $LFS/usr/lib/libncurses.so cd $LFS/sources rm -rf ncurses-6.2 echo “# 6.4. Bash-5.1.8” tar -zxf bash-5.1.8.tar.gz cd bash-5.1.8 ./configure –prefix=/usr \ –build=$(support/config.guess) \ –host=$LFSTGT \ –without-bash-malloc make -j $PARALLELJOBS make DESTDIR=$LFS install ln -sv bash $LFS/bin/sh cd $LFS/sources rm -rf bash-5.1.8 echo “# 6.5. Coreutils-8.32” tar -Jxf coreutils-8.32.tar.xz cd coreutils-8.32 if "$RPI_MODEL" == "64" ; then patch -Np1 -i ../coreutils-8.32-aarch64-fix.patch fi ./configure –prefix=/usr \ –host=$LFSTGT \ –build=$(build-aux/config.guess) \ –enable-install-program=hostname \ –enable-no-install-program=kill,uptime make -j $PARALLELJOBS make DESTDIR=$LFS install mv -v $LFS/usr/bin/chroot $LFS/usr/sbin mkdir -pv $LFS/usr/share/man/man8 mv -v $LFS/usr/share/man/man1/chroot.1 $LFS/usr/share/man/man8/chroot.8 sed -i 's/“1”/“8”/' $LFS/usr/share/man/man8/chroot.8 cd $LFS/sources rm -rf coreutils-8.32 echo “# 6.6. Diffutils-3.8” tar -Jxf diffutils-3.8.tar.xz cd diffutils-3.8 ./configure –prefix=/usr –host=$LFSTGT make -j $PARALLELJOBS make DESTDIR=$LFS install cd $LFS/sources rm -rf diffutils-3.8 echo “# 6.7. File-5.40” tar -zxf file-5.40.tar.gz cd file-5.40 mkdir build pushd build ../configure –disable-bzlib \ –disable-libseccomp \ –disable-xzlib \ –disable-zlib make -j $PARALLELJOBS popd ./configure –prefix=/usr –host=$LFSTGT –build=$(./config.guess) make -j $PARALLELJOBS FILECOMPILE=$(pwd)/build/src/file make DESTDIR=$LFS install cd $LFS/sources rm -rf file-5.40 echo “# 6.8. Findutils-4.8.0” tar -Jxf findutils-4.8.0.tar.xz cd findutils-4.8.0 ./configure –prefix=/usr \ –localstatedir=/var/lib/locate \ –host=$LFSTGT \ –build=$(build-aux/config.guess) make -j $PARALLELJOBS make DESTDIR=$LFS install cd $LFS/sources rm -rf findutils-4.8.0 echo “# 6.9. Gawk-5.1.0” tar -Jxf gawk-5.1.0.tar.xz cd gawk-5.1.0 sed -i 's/extras'* Makefile.in ./configure –prefix=/usr \

1. -host=$LFS*TGT \ –build=$(./config.guess) make -j $PARALLEL*JOBS

make DESTDIR=$LFS install cd $LFS/sources rm -rf gawk-5.1.0

echo “# 6.10. Grep-3.7” tar -Jxf grep-3.7.tar.xz cd grep-3.7 ./configure –prefix=/usr \

1. -host=$LFS*TGT make -j $PARALLEL*JOBS

make DESTDIR=$LFS install cd $LFS/sources rm -rf grep-3.7

echo “# 6.11. Gzip-1.10” tar -Jxf gzip-1.10.tar.xz cd gzip-1.10 ./configure –prefix=/usr –host=$LFS*TGT make -j $PARALLEL*JOBS make DESTDIR=$LFS install cd $LFS/sources rm -rf gzip-1.10

echo “# 6.12. Make-4.3” tar -zxf make-4.3.tar.gz cd make-4.3 ./configure –prefix=/usr \

1. -without-guile \
2. -host=$LFS*TGT \ –build=$(build-aux/config.guess) make -j $PARALLEL*JOBS

make DESTDIR=$LFS install cd $LFS/sources rm -rf make-4.3

echo “# 6.13. Patch-2.7.6” tar -Jxf patch-2.7.6.tar.xz cd patch-2.7.6 ./configure –prefix=/usr \

1. -host=$LFS*TGT \ –build=$(build-aux/config.guess) make -j $PARALLEL*JOBS

make DESTDIR=$LFS install cd $LFS/sources rm -rf patch-2.7.6

echo “# 6.14. Sed-4.8” tar -Jxf sed-4.8.tar.xz cd sed-4.8 ./configure –prefix=/usr \

1. -host=$LFS*TGT make -j $PARALLEL*JOBS

make DESTDIR=$LFS install cd $LFS/sources rm -rf sed-4.8

echo “# 6.15. Tar-1.34” tar -Jxf tar-1.34.tar.xz cd tar-1.34 ./configure –prefix=/usr \

1. -host=$LFS*TGT \ –build=$(build-aux/config.guess) make -j $PARALLEL*JOBS

make DESTDIR=$LFS install cd $LFS/sources rm -rf tar-1.34

echo “# 6.16. Xz-5.2.5” tar -Jxf xz-5.2.5.tar.xz cd xz-5.2.5 ./configure –prefix=/usr \

1. -host=$LFS*TGT \ –build=$(build-aux/config.guess) \ –disable-static \ –docdir=/usr/share/doc/xz-5.2.5 make -j $PARALLEL*JOBS

make DESTDIR=$LFS install cd $LFS/sources rm -rf xz-5.2.5

echo “# 6.17. Binutils-2.37 - Pass 2” tar -Jxf binutils-2.37.tar.xz cd binutils-2.37 mkdir -v build cd build ../configure \

1. -prefix=/usr \
2. -build=$(../config.guess) \
3. -host=$LFS*TGT \ –disable-nls \ –enable-shared \ –disable-werror \ –enable-64-bit-bfd make -j $PARALLEL*JOBS

make -j 1 DESTDIR=$LFS install install -vm755 libctf/.libs/libctf.so.0.0.0 $LFS/usr/lib cd $LFS/sources rm -rf binutils-2.37

echo “# 6.18. gcc-11.2.0 - Pass 2” tar -Jxf gcc-11.2.0.tar.xz cd gcc-11.2.0 if "$RPI_MODEL" == "64" ; then

```
  sed -e '/mabi.lp64=/s/lib64/lib/' -i.orig gcc/config/aarch64/t-aarch64-linux
```

else

```
  patch -Np1 -i ../gcc-9.1.0-rpi$RPI_MODEL-cpu-default.patch
```

fi tar -Jxf ../mpfr-4.1.0.tar.xz mv -v mpfr-4.1.0 mpfr tar -Jxf ../gmp-6.2.1.tar.xz mv -v gmp-6.2.1 gmp tar -zxf ../mpc-1.2.1.tar.gz mv -v mpc-1.2.1 mpc mkdir -v build cd build mkdir -pv $LFS*TGT/libgcc ln -s*

*../../../libgcc/gthr-posix.h $LFS*TGT/libgcc/gthr-default.h ../configure \

1. -build=$(../config.guess) \
2. -host=$LFS*TGT \ –prefix=/usr \ CC*FOR*TARGET=$LFS*TGT-gcc \
3. -with-build-sysroot=$LFS \
4. -enable-initfini-array \
5. -disable-nls \
6. -disable-multilib \
7. -disable-decimal-float \
8. -disable-libatomic \
9. -disable-libgomp \
10. -disable-libquadmath \
11. -disable-libssp \
12. -disable-libvtv \
13. -disable-libstdcxx \
14. -enable-languages=c,c++

make -j 1 make DESTDIR=$LFS install ln -sv gcc $LFS/usr/bin/cc cd $LFS/sources rm -rf gcc-11.2.0

echo -e "——————————————————-" echo -e "\nYou made it! This is the end of chapter 6!" printf 'Total script time: %s\n' $(timer $total_time) echo -e "Now continue reading from \"7. Entering Chroot and Building Additional Temporary Tools\"" </codedoc>

<codedoc toggle ch7-build.sh: Script LFS chapters 7.7 Libstdc++ from GCC - Pass 2 to 8.75 - Sysvinit>

# !/bin/bash

# # PiLFS Build Script for LFS Version r11.0-27

# Builds chapters 7.7 Libstdc++ from GCC - Pass 2 to 8.75 - Sysvinit

# https://intestinate.com/pilfs

# # Optional parameteres below:

RPI*MODEL=4 # Which Raspberry Pi model are you building for - this selects the right GCC CPU patch. # Put 64 to build for aarch64. PARALLEL*JOBS=4 # Number of parallel make jobs, 1 for RPi1 and 4 for RPi2 and up recommended. LOCAL*TIMEZONE=Europe/London # Use this timezone from*

*/usr/share/zoneinfo/ to set /etc/localtime. See “8.8.2 Configuring Glibc”. GROFF*PAPER*SIZE=A4 # Use this default paper size for Groff. See “8.57 Groff”. INSTALL*OPTIONAL*DOCS=1 # Install optional documentation when given a choice? INSTALL*ALL*LOCALES=0 # Install all glibc locales? By default only en*US.ISO-8859-1 and en_US.UTF-8 are installed.

# End of optional parameters

set -o nounset set -o errexit

function prebuild*sanity*check {

```
  if [[ $(whoami) != "root" ]] ; then
      echo "You should be running as root for chapter 7!"
      exit 1
  fi
```

```
  if ! [[ -d /sources ]] ; then
      echo "Can't find your sources directory! Did you forget to chroot?"
      exit 1
  fi
```

```
  if ! [[ -d /tools ]] ; then
      echo "Can't find your tools directory! Did you forget to chroot?"
      exit 1
  fi
```

}

function check*tarballs { LIST*OF*TARBALLS=“ gcc-11.2.0.tar.xz gcc-9.1.0-rpi1-cpu-default.patch gcc-9.1.0-rpi2-cpu-default.patch gcc-9.1.0-rpi3-cpu-default.patch gcc-9.1.0-rpi4-cpu-default.patch bison-3.7.6.tar.xz gettext-0.21.tar.xz perl-5.34.0.tar.xz perl-5.34.0-upstream*fixes-1.patch Python-3.9.6.tar.xz python-3.9.6-docs-html.tar.bz2 texinfo-6.8.tar.xz util-linux-2.37.2.tar.xz man-pages-5.13.tar.xz tcl8.6.11-src.tar.gz tcl8.6.11-html.tar.gz expect5.45.4.tar.gz expect5.45-aarch64-fix.patch dejagnu-1.6.3.tar.gz iana-etc-20210611.tar.gz glibc-2.34.tar.xz glibc-2.34-fhs-1.patch tzdata2021a.tar.gz zlib-1.2.11.tar.xz bzip2-1.0.8.tar.gz bzip2-1.0.8-install*docs-1.patch xz-5.2.5.tar.xz zstd-1.5.0.tar.gz file-5.40.tar.gz file-5.40-upstream*fixes-1.patch readline-8.1.tar.gz m4-1.4.19.tar.xz bc-5.0.0.tar.xz flex-2.6.4.tar.gz binutils-2.37.tar.xz binutils-2.37-upstream_fix-1.patch gmp-6.2.1.tar.xz mpfr-4.1.0.tar.xz mpc-1.2.1.tar.gz attr-2.5.1.tar.gz acl-2.3.1.tar.xz libcap-2.53.tar.xz shadow-4.9.tar.xz pkg-config-0.29.2.tar.gz ncurses-6.2.tar.gz sed-4.8.tar.xz psmisc-23.4.tar.xz grep-3.7.tar.xz bash-5.1.8.tar.gz libtool-2.4.6.tar.xz gdbm-1.20.tar.gz gperf-3.1.tar.gz expat-2.4.1.tar.xz inetutils-2.1.tar.xz XML-Parser-2.46.tar.gz intltool-0.51.0.tar.gz autoconf-2.71.tar.xz automake-1.16.4.tar.xz kmod-29.tar.xz elfutils-0.185.tar.bz2 libffi-3.4.2.tar.gz openssl-1.1.1l.tar.gz ninja-1.10.2.tar.gz meson-0.59.1.tar.gz coreutils-8.32.tar.xz coreutils-8.32-i18n-1.patch coreutils-8.32-aarch64-fix.patch check-0.15.2.tar.gz diffutils-3.8.tar.xz gawk-5.1.0.tar.xz findutils-4.8.0.tar.xz groff-1.22.4.tar.gz less-590.tar.gz gzip-1.10.tar.xz iproute2-5.13.0.tar.xz kbd-2.4.0.tar.xz kbd-2.4.0-backspace-1.patch libpipeline-1.5.3.tar.gz make-4.3.tar.gz patch-2.7.6.tar.xz man-db-2.9.4.tar.xz tar-1.34.tar.xz vim-8.2.3337.tar.gz eudev-3.2.10.tar.gz udev-lfs-20171102.tar.xz procps-ng-3.3.17.tar.xz e2fsprogs-1.46.4.tar.gz sysklogd-1.5.1.tar.gz sysvinit-2.99.tar.xz sysvinit-2.99-

consolidated-1.patch master.tar.gz v2021.04.29-138a1.tar.gz ”

for tarball in $LIST*OF*TARBALLS ; do

```
  if ! [[ -f /sources/$tarball ]] ; then
      echo "Can't find /sources/$tarball!"
      exit 1
  fi
```

done }

function timer {

```
  if [[ $# -eq 0 ]]; then
      echo $(date '+%s')
  else
      local stime=$1
      etime=$(date '+%s')
      if [[ -z "$stime" ]]; then stime=$etime; fi
      dt=$((etime - stime))
      ds=$((dt % 60))
      dm=$(((dt / 60) % 60))
      dh=$((dt / 3600))
      printf '%02d:%02d:%02d' $dh $dm $ds
  fi
```

}

prebuild*sanity*check check_tarballs

if wc -l) == 1 ; then

```
  echo -e "\nYou are almost certainly going to want to add some swap space
before building!"
  echo -e "(See https://intestinate.com/pilfs/beyond.html#addswap for
instructions)"
  echo -e "Continue without swap?"
  select yn in "Yes" "No"; do
      case $yn in
          Yes) break;;
          No) exit;;
      esac
  done
```

fi

echo -e “\nThis is your last chance to quit before we start building… continue?” echo “(Note that if anything goes wrong during the build, the script will abort mission)” select yn in “Yes” “No”; do

```
  case $yn in
      Yes) break;;
      No) exit;;
  esac
```

done

total_time=$(timer)

echo “# 7.7. Libstdc++ from GCC-11.2.0, Pass 2” tar -Jxf gcc-11.2.0.tar.xz cd gcc-11.2.0 ln -s gthr-posix.h libgcc/gthr-default.h mkdir -v build cd build if "$RPI_MODEL" == "64" ; then

```
  ../libstdc++-v3/configure            \
      CXXFLAGS="-g -O2 -D_GNU_SOURCE"  \
      --prefix=/usr                    \
      --disable-multilib               \
      --disable-nls                    \
      --host=$(uname -m)-lfs-linux-gnu \
      --disable-libstdcxx-pch
```

else

```
  ../libstdc++-v3/configure            \
      CXXFLAGS="-g -O2 -D_GNU_SOURCE"  \
      --prefix=/usr                    \
      --disable-multilib               \
      --disable-nls                    \
      --host=$(uname -m)-lfs-linux-gnueabihf \
      --disable-libstdcxx-pch
```

fi make -j $PARALLEL_JOBS make install cd /sources rm -rf gcc-11.2.0

echo “# 7.8. Gettext-0.21” tar -Jxf gettext-0.21.tar.xz cd gettext-0.21 ./configure –disable-shared make -j $PARALLEL_JOBS cp -v gettext-tools/src/{msgfmt,msgmerge,xgettext} /usr/bin cd /sources rm -rf gettext-0.21

echo “# 7.9. Bison-3.7.6” tar -Jxf bison-3.7.6.tar.xz cd bison-3.7.6 ./configure –prefix=/usr \

1. -docdir=/usr/share/doc/bison-3.7.6

make -j $PARALLEL_JOBS make install cd /sources rm -rf bison-3.7.6

echo “# 7.10. Perl-5.34.0” tar -Jxf perl-5.34.0.tar.xz cd perl-5.34.0 sh Configure -des \

1. Dprefix=/usr \
2. Dvendorprefix=/usr \
3. Dprivlib=/usr/lib/perl5/5.34/core*perl \ -Darchlib=/usr/lib/perl5/5.34/core*perl \
4. Dsitelib=/usr/lib/perl5/5.34/site*perl \ -Dsitearch=/usr/lib/perl5/5.34/site*perl \
5. Dvendorlib=/usr/lib/perl5/5.34/vendor*perl \ -Dvendorarch=/usr/lib/perl5/5.34/vendor*perl

make -j $PARALLEL_JOBS make install cd /sources rm -rf perl-5.34.0

echo “# 7.11. Python-3.9.6” tar -Jxf Python-3.9.6.tar.xz cd Python-3.9.6 ./configure –prefix=/usr \

1. -enable-shared \
2. -without-ensurepip

make -j $PARALLEL_JOBS make install cd /sources rm -rf Python-3.9.6

echo "# 7.12. Texinfo-6.8" tar -Jxf texinfo-6.8.tar.xz cd texinfo-6.8 sed -e 's/attributenonnull/nonnull/' -i gnulib/lib/malloc/dynarray-skeleton.c ./configure –prefix=/usr make -j $PARALLELJOBS make install cd /sources rm -rf texinfo-6.8 echo "# 7.13. Util-linux-2.37.2" tar -Jxf util-linux-2.37.2.tar.xz cd util-linux-2.37.2 mkdir -pv /var/lib/hwclock ./configure ADJTIMEPATH=/var/lib/hwclock/adjtime \ –libdir=/usr/lib \ –docdir=/usr/share/doc/util-linux-2.37.2 \ –disable-chfn-chsh \ –disable-login \ –disable-nologin \ –disable-su \ –disable-setpriv \ –disable-runuser \ –disable-pylibmount \ –disable-static \ –without-python \ runstatedir=/run make -j $PARALLELJOBS make install cd /sources rm -rf util-linux-2.37.2 echo "# 7.14. Cleaning up and Saving the Temporary System" rm -rf /usr/share/{info,man,doc}/* find /usr/{lib,libexec} -name *.la -delete rm -rf /tools # In order to keep our script running we will not exit the chroot environment here echo "# 8.3. Man-pages-5.13" tar -Jxf man-pages-5.13.tar.xz cd man-pages-5.13 make prefix=/usr install cd /sources rm -rf man-pages-5.13 echo "# 8.4. Iana-Etc-20210611" tar -zxf iana-etc-20210611.tar.gz cd iana-etc-20210611 cp services protocols /etc cd /sources rm -rf iana-etc-20210611 echo "# 8.5. Glibc-2.34" tar -Jxf glibc-2.34.tar.xz cd glibc-2.34 sed -e '/NOTIFYREMOVED)/s/)/ \&\& data.attr != NULL)/' -i sysdeps/unix/sysv/linux/mqnotify.c patch -Np1 -i ../glibc-2.34-fhs-1.patch mkdir -v build cd build echo "rootsbindir=/usr/sbin" > configparms ../configure –prefix=/usr \ –disable-werror \ –enable-kernel=3.2 \ –enable-stack-protector=strong \ –with-headers=/usr/include \ libccvslibdir=/usr/lib make -j $PARALLELJOBS touch /etc/ld.so.conf sed '/test-installation/s@$(PERL)@echo not running@' -i ../Makefile make install sed '/RTLDLIST=/s@/usr@@g' -i /usr/bin/ldd cp -v ../nscd/nscd.conf /etc/nscd.conf mkdir -pv /var/cache/nscd if $INSTALL_ALL_LOCALES = 1 ; then make localedata/install-locales else mkdir -pv /usr/lib/locale localedef -i enUS -f ISO-8859-1 enUS localedef -i enUS -f UTF-8 en_US.UTF-8 fi cat > /etc/nsswitch.conf « "EOF" # Begin /etc/nsswitch.conf passwd: files group: files shadow: files hosts: files dns networks: files protocols: files services: files ethers: files rpc: files # End /etc/nsswitch.conf EOF tar -zxf ../../tzdata2021a.tar.gz ZONEINFO=/usr/share/zoneinfo mkdir -pv $ZONEINFO/{posix,right} for tz in etcetera southamerica northamerica europe africa antarctica \ asia australasia backward; do zic -L /dev/null -d $ZONEINFO ${tz} zic -L /dev/null -d $ZONEINFO/posix ${tz} zic -L leapseconds -d $ZONEINFO/right ${tz} done cp -v zone.tab zone1970.tab iso3166.tab $ZONEINFO zic -d $ZONEINFO -p America/NewYork unset ZONEINFO if ! -f /usr/share/zoneinfo/$LOCAL_TIMEZONE ; then echo "Seems like your timezone won't work out. Defaulting to London. Either fix it yourself later or consider moving there :)" ln -sfv /usr/share/zoneinfo/Europe/London /etc/localtime else ln -sfv /usr/share/zoneinfo/$LOCALTIMEZONE /etc/localtime fi cat > /etc/ld.so.conf « "EOF" # Begin /etc/ld.so.conf /usr/local/lib EOF cat » /etc/ld.so.conf « "EOF" # Add an include directory include /etc/ld.so.conf.d/*.conf EOF mkdir -pv /etc/ld.so.conf.d # Compatibility symlink for non ld-linux-armhf awareness ln -sv ld-2.34.so /lib/ld-linux.so.3 cd /sources rm -rf glibc-2.34 echo "# 8.6. Zlib-1.2.11" tar -Jxf zlib-1.2.11.tar.xz cd zlib-1.2.11 ./configure –prefix=/usr make -j $PARALLEL_JOBS make install rm -fv /usr/lib/libz.a cd /sources rm -rf zlib-1.2.11 echo "# 8.7. Bzip2-1.0.8" tar -zxf bzip2-1.0.8.tar.gz cd bzip2-1.0.8 patch -Np1 -i ../bzip2-1.0.8-installdocs-1.patch sed -i 's@(ln -s -f )$(PREFIX)/bin/@\1@' Makefile sed -i "s@(PREFIX)/man@(PREFIX)/share/man@g" Makefile make -j $PARALLELJOBS -f Makefile-libbz2so make clean make -j $PARALLELJOBS make PREFIX=/usr install cp -av libbz2.so.* /usr/lib ln -sv libbz2.so.1.0.8 /usr/lib/libbz2.so cp -v bzip2-shared /usr/bin/bzip2 for i in /usr/bin/{bzcat,bunzip2}; do ln -sfv bzip2 $i done rm -fv /usr/lib/libbz2.a cd /sources rm -rf bzip2-1.0.8 echo "# 8.8. Xz-5.2.5" tar -Jxf xz-5.2.5.tar.xz cd xz-5.2.5 ./configure –prefix=/usr \ –disable-static \ –docdir=/usr/share/doc/xz-5.2.5 make -j $PARALLEL_JOBS make install cd /sources rm -rf xz-5.2.5 echo "# 8.9. Zstd-1.5.0" tar -zxf zstd-1.5.0.tar.gz cd zstd-1.5.0 make -j $PARALLEL_JOBS make prefix=/usr install rm -v /usr/lib/libzstd.a cd /sources rm -rf zstd-1.5.0 echo "# 8.10. File-5.40" tar -zxf file-5.40.tar.gz cd file-5.40 patch -Np1 -i ../file-5.40-upstreamfixes-1.patch ./configure –prefix=/usr make -j $PARALLELJOBS make install cd /sources rm -rf file-5.40 echo “# 8.11. Readline-8.1” tar -zxf readline-8.1.tar.gz cd readline-8.1 sed -i '/MV.old/d' Makefile.in sed -i '/{OLDSUFF}/c:' support/shlib-install ./configure –prefix=/usr \ –disable-static \ –with-curses \ –docdir=/usr/share/doc/readline-8.1 make -j $PARALLELJOBS SHLIBLIBS=“-lncursesw” make SHLIBLIBS=“-lncursesw” install if $INSTALL_OPTIONAL_DOCS = 1 ; then install -v -m644 doc/*.{ps,pdf,html,dvi} /usr/share/doc/readline-8.1 fi cd /sources rm -rf readline-8.1 echo “# 8.12. M4-1.4.19” tar -Jxf m4-1.4.19.tar.xz cd m4-1.4.19 ./configure –prefix=/usr make -j $PARALLELJOBS make install cd /sources rm -rf m4-1.4.19 echo “# 8.13. Bc-5.0.0” tar -Jxf bc-5.0.0.tar.xz cd bc-5.0.0 CC=gcc ./configure –prefix=/usr -G -O3 make -j $PARALLEL_JOBS make install cd /sources rm -rf bc-5.0.0 echo “# 8.14. Flex-2.6.4” tar -zxf flex-2.6.4.tar.gz cd flex-2.6.4 ./configure –prefix=/usr \ –docdir=/usr/share/doc/flex-2.6.4 \ –disable-static make -j $PARALLEL_JOBS make install ln -sv flex /usr/bin/lex cd /sources rm -rf flex-2.6.4 echo “# 8.15. Tcl-8.6.11” tar -zxf tcl8.6.11-src.tar.gz cd tcl8.6.11 tar -xf ../tcl8.6.11-html.tar.gz –strip-components=1 SRCDIR=$(pwd) cd unix ./configure –prefix=/usr \ –mandir=/usr/share/man make -j $PARALLELJOBS sed -e “s|$SRCDIR/unix|/usr/lib|” \ -e “s|$SRCDIR|/usr/include|” \ -i tclConfig.sh sed -e “s|$SRCDIR/unix/pkgs/tdbc1.1.2|/usr/lib/tdbc1.1.2|” \ -e “s|$SRCDIR/pkgs/tdbc1.1.2/generic|/usr/include|” \ -e “s|$SRCDIR/pkgs/tdbc1.1.2/library|/usr/lib/tcl8.6|” \ -e “s|$SRCDIR/pkgs/tdbc1.1.2|/usr/include|” \ -i pkgs/tdbc1.1.2/tdbcConfig.sh sed -e “s|$SRCDIR/unix/pkgs/itcl4.2.1|/usr/lib/itcl4.2.1|” \ -e “s|$SRCDIR/pkgs/itcl4.2.1/generic|/usr/include|” \ -e “s|$SRCDIR/pkgs/itcl4.2.1|/usr/include|” \ -i pkgs/itcl4.2.1/itclConfig.sh unset SRCDIR make install chmod -v u+w /usr/lib/libtcl8.6.so make install-private-headers ln -sfv tclsh8.6 /usr/bin/tclsh mv /usr/share/man/man3/{Thread,TclThread}.3 cd /sources rm -rf tcl8.6.11 echo “# 8.16. Expect-5.45.4” tar -zxf expect5.45.4.tar.gz cd expect5.45.4 if "$RPI_MODEL" == "64" ; then patch -Np1 -i ../expect5.45-aarch64-fix.patch fi ./configure –prefix=/usr \ –with-tcl=/usr/lib \ –enable-shared \ –mandir=/usr/share/man \ –with-tclinclude=/usr/include make -j $PARALLEL_JOBS make install ln -svf expect5.45.4/libexpect5.45.4.so /usr/lib cd /sources rm -rf expect5.45.4 echo “# 8.17. DejaGNU-1.6.3” tar -zxf dejagnu-1.6.3.tar.gz cd dejagnu-1.6.3 mkdir -v build cd build ../configure –prefix=/usr makeinfo –html –no-split -o doc/dejagnu.html ../doc/dejagnu.texi makeinfo –plaintext -o doc/dejagnu.txt ../doc/dejagnu.texi make install if $INSTALL_OPTIONAL_DOCS = 1 ; then install -v -dm755 /usr/share/doc/dejagnu-1.6.3 install -v -m644 doc/dejagnu.{html,txt} /usr/share/doc/dejagnu-1.6.3 fi cd /sources rm -rf dejagnu-1.6.3 echo “# 8.18. Binutils-2.37” tar -Jxf binutils-2.37.tar.xz cd binutils-2.37 patch -Np1 -i ../binutils-2.37-upstreamfix-1.patch sed -i '63d' etc/texi2pod.pl find -name *.1 -delete mkdir -v build cd build ../configure –prefix=/usr \ –enable-gold \ –enable-ld=default \ –enable-plugins \ –enable-shared \ –disable-werror \ –enable-64-bit-bfd \ –with-system-zlib make -j $PARALLELJOBS tooldir=/usr make -j 1 tooldir=/usr install rm -fv /usr/lib/lib{bfd,ctf,ctf-nobfd,opcodes}.a cd /sources rm -rf binutils-2.37 echo “# 8.19. GMP-6.2.1” tar -Jxf gmp-6.2.1.tar.xz cd gmp-6.2.1 ./configure –prefix=/usr \ –enable-cxx \ –disable-static \ –docdir=/usr/share/doc/gmp-6.2.1 make -j $PARALLELJOBS make install if $INSTALL_OPTIONAL_DOCS = 1 ; then make html make install-html fi cd /sources rm -rf gmp-6.2.1 echo “# 8.20. MPFR-4.1.0” tar -Jxf mpfr-4.1.0.tar.xz cd mpfr-4.1.0 ./configure –prefix=/usr \ –disable-static \ –enable-thread-safe \ –docdir=/usr/share/doc/mpfr-4.1.0 make -j $PARALLELJOBS make install if $INSTALL_OPTIONAL_DOCS = 1 ; then make html make install-html fi cd /sources rm -rf mpfr-4.1.0 echo “# 8.21. MPC-1.2.1” tar -zxf mpc-1.2.1.tar.gz cd mpc-1.2.1 ./configure –prefix=/usr \ –disable-static \ –docdir=/usr/share/doc/mpc-1.2.1 make -j $PARALLELJOBS make install if $INSTALL_OPTIONAL_DOCS = 1 ; then make html make install-html fi cd /sources rm -rf mpc-1.2.1 echo “# 8.22. Attr-2.5.1” tar -zxf attr-2.5.1.tar.gz cd attr-2.5.1 ./configure –prefix=/usr \ –disable-static \ –sysconfdir=/etc \ –docdir=/usr/share/doc/attr-2.5.1 make -j $PARALLELJOBS make install cd /sources rm -rf attr-2.5.1 echo “# 8.23. Acl-2.3.1” tar -Jxf acl-2.3.1.tar.xz cd acl-2.3.1 ./configure –prefix=/usr \ –disable-static \ –docdir=/usr/share/doc/acl-2.3.1 make -j $PARALLEL_JOBS make install cd /sources rm -rf acl-2.3.1 echo “# 8.24. Libcap-2.53” tar -Jxf libcap-2.53.tar.xz cd libcap-2.53 sed -i '/install -m.STA/d' libcap/Makefile make -j $PARALLEL_JOBS prefix=/usr lib=lib make prefix=/usr lib=lib install

```
chmod -v 755 /usr/lib/lib{cap,psx}.so.2.53 cd /sources rm -rf libcap-2.53 echo “# 8.25. Shadow-4.9”
tar -Jxf shadow-4.9.tar.xz cd shadow-4.9 sed -i 's/groups$(EXEEXT) ' src/Makefile.in find man -name
Makefile.in -exec sed -i 's/groups.1 / /' {} \; find man -name Makefile.in -exec sed -i 's/getspnam.3 / /'
{} \; find man -name Makefile.in -exec sed -i 's/passwd.5 / /' {} \; sed -e 's:#ENCRYPTMETHOD
DES:ENCRYPTMETHOD SHA512:' \ -e 's:/var/spool/mail:/var/mail:' \ -e
'/PATH=/{s@/sbin:@@;s@/bin:@@}' \ -i etc/login.defs sed -e “224s/rounds/minrounds/” -i
libmisc/salt.c touch /usr/bin/passwd ./configure –sysconfdir=/etc \ –with-group-name-max-length=32
make -j $PARALLELJOBS make exec_prefix=/usr install make -C man install-man mkdir -p /etc/default
useradd -D –gid 999 pwconv grpconv sed -i 's/yes/no/' /etc/default/useradd # passwd root # Root
password will be set at the end of the script to prevent a stop here cd /sources rm -rf shadow-4.9 echo
“# 8.26. GCC-11.2.0” tar -Jxf gcc-11.2.0.tar.xz cd gcc-11.2.0 if "$RPI_MODEL" == "64" ; then sed -e
'/mabi.lp64=/s/lib64/lib/' -i.orig gcc/config/aarch64/t-aarch64-linux else patch -Np1 -i ../gcc-9.1.0-
rpi$RPIMODEL-cpu-default.patch fi sed -e '/static.SIGSTKSZ/d' \ -e 's/return kAltStackSize/return
SIGSTKSZ * 4/' \ -i libsanitizer/sanitizercommon/sanitizerposixlibcdep.cpp mkdir -v build cd build
../configure –prefix=/usr \ LD=ld \ –enable-languages=c,c++ \ –disable-multilib \ –disable-bootstrap \
–with-system-zlib make -j 1 make install rm -rf /usr/lib/gcc/$(gcc -dumpmachine)/11.2.0/include-
fixed/bits/ chown -v -R root:root /usr/lib/gcc/linux-gnu/11.2.0/include{,-fixed} ln -svr /usr/bin/cpp /lib ln
-sfv ../../libexec/gcc/$(gcc -dumpmachine)/11.2.0/libltoplugin.so /usr/lib/bfd-plugins/ mkdir -pv
/usr/share/gdb/auto-load/usr/lib mv -v /usr/lib/gdb.py /usr/share/gdb/auto-load/usr/lib cd /sources rm -
rf gcc-11.2.0 echo “# 8.27. Pkg-config-0.29.2” tar -zxf pkg-config-0.29.2.tar.gz cd pkg-config-0.29.2
./configure –prefix=/usr \ –with-internal-glib \ –disable-host-tool \ –docdir=/usr/share/doc/pkg-
config-0.29.2 make -j $PARALLELJOBS make install cd /sources rm -rf pkg-config-0.29.2 echo “# 8.28.
Ncurses-6.2” tar -zxf ncurses-6.2.tar.gz cd ncurses-6.2 ./configure –prefix=/usr \
–mandir=/usr/share/man \ –with-shared \ –without-debug \ –without-normal \ –enable-pc-files \
–enable-widec make -j $PARALLELJOBS make install for lib in ncurses form panel menu ; do rm -vf
/usr/lib/lib${lib}.so echo “INPUT(-l${lib}w)” > /usr/lib/lib${lib}.so ln -sfv ${lib}w.pc
/usr/lib/pkgconfig/${lib}.pc done rm -vf /usr/lib/libcursesw.so echo “INPUT(-lncursesw)” >
/usr/lib/libcursesw.so ln -sfv libncurses.so /usr/lib/libcurses.so rm -fv /usr/lib/libncurses++w.a if
$INSTALL_OPTIONAL_DOCS = 1 ; then mkdir -v /usr/share/doc/ncurses-6.2 cp -v -R doc/*
/usr/share/doc/ncurses-6.2 fi cd /sources rm -rf ncurses-6.2 echo “# 8.29. Sed-4.8” tar -Jxf
sed-4.8.tar.xz cd sed-4.8 ./configure –prefix=/usr make -j $PARALLELJOBS make install if
$INSTALL_OPTIONAL_DOCS = 1 ; then make html install -d -m755 /usr/share/doc/sed-4.8 install -m644
doc/sed.html /usr/share/doc/sed-4.8 fi cd /sources rm -rf sed-4.8 echo “# 8.30. Psmisc-23.4” tar -Jxf
psmisc-23.4.tar.xz cd psmisc-23.4 ./configure –prefix=/usr make -j $PARALLEL_JOBS make install cd
/sources rm -rf psmisc-23.4 echo “# 8.31. Gettext-0.21” tar -Jxf gettext-0.21.tar.xz cd gettext-0.21
./configure –prefix=/usr \ –disable-static \ –docdir=/usr/share/doc/gettext-0.21 make -j $PARALLELJOBS
make install chmod -v 0755 /usr/lib/preloadablelibintl.so cd /sources rm -rf gettext-0.21 echo “# 8.32.
Bison-3.7.6” tar -Jxf bison-3.7.6.tar.xz cd bison-3.7.6 ./configure –prefix=/usr
–docdir=/usr/share/doc/bison-3.7.6 make -j $PARALLEL_JOBS make install cd /sources rm -rf
bison-3.7.6 echo “# 8.33. Grep-3.7” tar -Jxf grep-3.7.tar.xz cd grep-3.7 ./configure –prefix=/usr make -
j $PARALLEL_JOBS make install cd /sources rm -rf grep-3.7 echo “# 8.34. Bash-5.1.8” tar -zxf
bash-5.1.8.tar.gz cd bash-5.1.8 ./configure –prefix=/usr \ –docdir=/usr/share/doc/bash-5.1.8 \
–without-bash-malloc \ –with-installed-readline make -j $PARALLEL_JOBS make install # exec /bin/bash
–login +h # Don't know of a good way to keep running the script after entering bash here. cd /sources
rm -rf bash-5.1.8 echo “# 8.35. Libtool-2.4.6” tar -Jxf libtool-2.4.6.tar.xz cd libtool-2.4.6 ./configure
–prefix=/usr make -j $PARALLEL_JOBS make install rm -fv /usr/lib/libltdl.a cd /sources rm -rf
libtool-2.4.6 echo “# 8.36. GDBM-1.20” tar -zxf gdbm-1.20.tar.gz cd gdbm-1.20 ./configure
–prefix=/usr \ –disable-static \ –enable-libgdbm-compat make -j $PARALLEL_JOBS make install cd
/sources rm -rf gdbm-1.20 echo “# 8.37. Gperf-3.1” tar -zxf gperf-3.1.tar.gz cd gperf-3.1 ./configure
–prefix=/usr –docdir=/usr/share/doc/gperf-3.1 make -j $PARALLEL_JOBS make install cd /sources rm -rf
gperf-3.1 echo “# 8.38. Expat-2.4.1” tar -Jxf expat-2.4.1.tar.xz cd expat-2.4.1 ./configure –prefix=/usr
```

\ –disable-static \ –docdir=/usr/share/doc/expat-2.4.1 make -j $PARALLELJOBS make install if $INSTALL_OPTIONAL_DOCS = 1 ; then install -v -m644 doc/*.{html,png,css} /usr/share/doc/expat-2.4.1 fi cd /sources rm -rf expat-2.4.1 echo “# 8.39. Inetutils-2.1” tar -Jxf inetutils-2.1.tar.xz cd inetutils-2.1 ./configure –prefix=/usr \ –bindir=/usr/bin \ –localstatedir=/var \ –disable-logger \ –disable-whois \ –disable-rcp \ –disable-rexec \ –disable-rlogin \ –disable-rsh \ –disable-servers make -j $PARALLELJOBS make install mv -v /usr/{,s}bin/ifconfig cd /sources rm -rf inetutils-2.1 echo “# 8.40. Less-590” tar -zxf less-590.tar.gz cd less-590 ./configure –prefix=/usr –sysconfdir=/etc make -j $PARALLEL_JOBS make install cd /sources rm -rf less-590 echo “# 8.41. Perl-5.34.0” tar -Jxf perl-5.34.0.tar.xz cd perl-5.34.0 patch -Np1 -i ../perl-5.34.0-upstreamfixes-1.patch export BUILDZLIB=False export BUILDBZIP2=0 sh Configure -des \ -Dprefix=/usr \ -Dvendorprefix=/usr \ -Dprivlib=/usr/lib/perl5/5.34/coreperl \ -Darchlib=/usr/lib/perl5/5.34/coreperl \ -Dsitelib=/usr/lib/perl5/5.34/siteperl \ -Dsitearch=/usr/lib/perl5/5.34/siteperl \ -Dvendorlib=/usr/lib/perl5/5.34/vendorperl \ -Dvendorarch=/usr/lib/perl5/5.34/vendorperl \ -Dman1dir=/usr/share/man/man1 \ -Dman3dir=/usr/share/man/man3 \ -Dpager=“/usr/bin/less -isR” \ -Duseshrplib \ -Dusethreads make -j $PARALLELJOBS make install unset BUILDZLIB BUILDBZIP2 cd /sources rm -rf perl-5.34.0 echo “# 8.42. XML::Parser-2.46” tar -zxf XML-Parser-2.46.tar.gz cd XML-Parser-2.46 perl Makefile.PL make -j $PARALLEL_JOBS make install cd /sources rm -rf XML-Parser-2.46 echo “# 8.43. Intltool-0.51.0” tar -zxf intltool-0.51.0.tar.gz cd intltool-0.51.0 sed -i 's:\\\${:\\\$\\{:' intltool-update.in ./configure –prefix=/usr make -j $PARALLELJOBS make install if $INSTALL_OPTIONAL_DOCS = 1 ; then install -v -Dm644 doc/I18N-HOWTO /usr/share/doc/intltool-0.51.0/I18N-HOWTO fi cd /sources rm -rf intltool-0.51.0 echo “# 8.44. Autoconf-2.71” tar -Jxf autoconf-2.71.tar.xz cd autoconf-2.71 ./configure –prefix=/usr make -j $PARALLELJOBS make install cd /sources rm -rf autoconf-2.71 echo “# 8.45. Automake-1.16.4” tar -Jxf automake-1.16.4.tar.xz cd automake-1.16.4 ./configure –prefix=/usr –docdir=/usr/share/doc/automake-1.16.4 make -j $PARALLEL_JOBS make install cd /sources rm -rf automake-1.16.4 echo “# 8.46. kmod-29” tar -Jxf kmod-29.tar.xz cd kmod-29 ./configure –prefix=/usr \ –sysconfdir=/etc \ –with-xz \ –with-zstd \ –with-zlib make -j $PARALLEL_JOBS make install for target in depmod insmod lsmod modinfo modprobe rmmod; do ln -sfv ../bin/kmod /sbin/$target done ln -sfv kmod /bin/lsmod cd /sources rm -rf kmod-29 echo “8.47. Libelf from Elfutils-0.185” tar -jxf elfutils-0.185.tar.bz2 cd elfutils-0.185 ./configure –prefix=/usr \ –disable-debuginfod \ –enable-libdebuginfod=dummy make -j $PARALLEL_JOBS make -C libelf install install -vm644 config/libelf.pc /usr/lib/pkgconfig rm /usr/lib/libelf.a cd /sources rm -rf elfutils-0.185 echo “# 8.48. libffi-3.4.2” tar -zxf libffi-3.4.2.tar.gz cd libffi-3.4.2 ./configure –prefix=/usr \ –disable-static \ –with-gcc-arch=native \ –disable-exec-static-tramp make -j $PARALLEL_JOBS make install cd /sources rm -rf libffi-3.4.2 echo “# 8.49. OpenSSL-1.1.1l” tar -zxf openssl-1.1.1l.tar.gz cd openssl-1.1.1l ./config –prefix=/usr \ –openssldir=/etc/ssl \ –libdir=lib \ shared \ zlib-dynamic make -j $PARALLELJOBS sed -i '/INSTALLLIBS/s/libcrypto.a libssl.a' Makefile make MANSUFFIX=ssl install if $INSTALL_OPTIONAL_DOCS = 1 ; then mv -v /usr/share/doc/openssl /usr/share/doc/openssl-1.1.1l cp -vfr doc/* /usr/share/doc/openssl-1.1.1l fi cd /sources rm -rf openssl-1.1.1l echo “# 8.50. Python-3.9.6” tar -Jxf Python-3.9.6.tar.xz cd Python-3.9.6 ./configure –prefix=/usr \ –enable-shared \ –with-system-expat \ –with-system-ffi \ –with-ensurepip=yes \ –enable-optimizations make -j $PARALLELJOBS make install if $INSTALL_OPTIONAL_DOCS = 1 ; then install -v -dm755 /usr/share/doc/python-3.9.6/html tar –strip-components=1 –no-same-owner –no-same-permissions -C /usr/share/doc/python-3.9.6/html -jxf ../python-3.9.6-docs-html.tar.bz2 fi cd /sources rm -rf Python-3.9.6 echo “# 8.51. Ninja-1.10.2” tar -zxf ninja-1.10.2.tar.gz cd ninja-1.10.2 sed -i '/int Guess/a \ int j = 0;\ char* jobs = getenv( “NINJAJOBS” );\ if ( jobs != NULL ) j = atoi( jobs );\ if ( j > 0 ) return j;\ ' src/ninja.cc python3 configure.py –bootstrap install -vm755 ninja /usr/bin/ install -vDm644 misc/bash-completion /usr/share/bash-completion/completions/ninja install -vDm644 misc/zsh-completion /usr/share/zsh/site-functions/ninja cd /sources rm -rf ninja-1.10.2 echo “# 8.52. Meson-0.59.1” tar -zxf meson-0.59.1.tar.gz cd meson-0.59.1 python3 setup.py build python3 setup.py install –root=dest cp -rv dest/* / install -vDm644 data/shell-completions/bash/meson /usr/share/bash-completion/completions/meson install -

vDm644 data/shell-completions/zsh/meson /usr/share/zsh/site-functions/meson cd /sources rm -rf meson-0.59.1 echo “# 8.53. Coreutils-8.32” tar -Jxf coreutils-8.32.tar.xz cd coreutils-8.32 patch -Np1 -i ../coreutils-8.32-i18n-1.patch if "$RPI_MODEL" == "64" ; then patch -Np1 -i ../coreutils-8.32-aarch64-fix.patch fi autoreconf -fiv FORCEUNSAFECONFIGURE=1 ./configure \ –prefix=/usr \ –enable-no-install-program=kill,uptime make -j $PARALLEL_JOBS make install mv -v /usr/bin/chroot /usr/sbin mv -v /usr/share/man/man1/chroot.1 /usr/share/man/man8/chroot.8 sed -i 's/“1”/“8”/' /usr/share/man/man8/chroot.8 cd /sources rm -rf coreutils-8.32 echo “# 8.54. Check-0.15.2” tar -zxf check-0.15.2.tar.gz cd check-0.15.2 ./configure –prefix=/usr –disable-static make -j $PARALLEL_JOBS make docdir=/usr/share/doc/check-0.15.2 install cd /sources rm -rf check-0.15.2 echo “# 8.55. Diffutils-3.8” tar -Jxf diffutils-3.8.tar.xz cd diffutils-3.8 ./configure –prefix=/usr make -j $PARALLEL_JOBS make install cd /sources rm -rf diffutils-3.8 echo “# 8.56. Gawk-5.1.0” tar -Jxf gawk-5.1.0.tar.xz cd gawk-5.1.0 sed -i 's/extras' Makefile.in ./configure –prefix=/usr make -j $PARALLELJOBS make install if $INSTALL_OPTIONAL_DOCS = 1 ; then mkdir -v /usr/share/doc/gawk-5.1.0 cp -v doc/{awkforai.txt,*.{eps,pdf,jpg}} /usr/share/doc/gawk-5.1.0 fi cd /sources rm -rf gawk-5.1.0 echo “# 8.57. Findutils-4.8.0” tar -Jxf findutils-4.8.0.tar.xz cd findutils-4.8.0 ./configure –prefix=/usr –localstatedir=/var/lib/locate make -j $PARALLELJOBS make install cd /sources rm -rf findutils-4.8.0 echo “# 8.58. Groff-1.22.4” tar -zxf groff-1.22.4.tar.gz cd groff-1.22.4 PAGE=$GROFFPAPERSIZE ./configure –prefix=/usr make -j 1 make install cd /sources rm -rf groff-1.22.4 # 8.59. GRUB-2.04 # We don't use GRUB on ARM echo “# 8.60. Gzip-1.10” tar -Jxf gzip-1.10.tar.xz cd gzip-1.10 ./configure –prefix=/usr make -j $PARALLEL_JOBS make install cd /sources rm -rf gzip-1.10 echo “# 8.61. IPRoute2-5.13.0” tar -Jxf iproute2-5.13.0.tar.xz cd iproute2-5.13.0 sed -i /ARPD/d Makefile rm -fv man/man8/arpd.8 sed -i 's/mipt.o' tc/Makefile make -j $PARALLELJOBS make SBINDIR=/usr/sbin install if $INSTALL_OPTIONAL_DOCS = 1 ; then mkdir -v /usr/share/doc/iproute2-5.13.0 cp -v COPYING README* /usr/share/doc/iproute2-5.13.0 fi cd /sources rm -rf iproute2-5.13.0 echo “# 8.62. Kbd-2.4.0” tar -Jxf kbd-2.4.0.tar.xz cd kbd-2.4.0 patch -Np1 -i ../kbd-2.4.0-backspace-1.patch sed -i 's/(RESIZECONSPROGS=)yes/\1no/g' configure sed -i 's/resizecons.8 ' docs/man/man8/Makefile.in ./configure –prefix=/usr –disable-vlock make -j $PARALLELJOBS make install if $INSTALL_OPTIONAL_DOCS = 1 ; then mkdir -v /usr/share/doc/kbd-2.4.0 cp -R -v docs/doc/* /usr/share/doc/kbd-2.4.0 fi cd /sources rm -rf kbd-2.4.0 echo “# 8.63. Libpipeline-1.5.3” tar -zxf libpipeline-1.5.3.tar.gz cd libpipeline-1.5.3 ./configure –prefix=/usr make -j $PARALLEL_JOBS make install cd /sources rm -rf libpipeline-1.5.3 echo “# 8.64. Make-4.3” tar -zxf make-4.3.tar.gz cd make-4.3 ./configure –prefix=/usr make -j $PARALLEL_JOBS make install cd /sources rm -rf make-4.3 echo “# 8.65. Patch-2.7.6” tar -Jxf patch-2.7.6.tar.xz cd patch-2.7.6 ./configure –prefix=/usr make -j $PARALLEL_JOBS make install cd /sources rm -rf patch-2.7.6 echo “# 8.66. Tar-1.34” tar -Jxf tar-1.34.tar.xz cd tar-1.34 FORCEUNSAFECONFIGURE=1 \ ./configure –prefix=/usr make -j $PARALLELJOBS make install if $INSTALL_OPTIONAL_DOCS = 1 ; then make -C doc install-html docdir=/usr/share/doc/tar-1.34 fi cd /sources rm -rf tar-1.34 echo “# 8.67. Texinfo-6.8” tar -Jxf texinfo-6.8.tar.xz cd texinfo-6.8 ./configure –prefix=/usr sed -e 's/attributenonnull/nonnull/' -i gnulib/lib/malloc/dynarray-skeleton.c make -j $PARALLEL_JOBS make install cd /sources rm -rf texinfo-6.8 echo “# 8.68. Vim-8.2.3337” tar -zxf vim-8.2.3337.tar.gz cd vim-8.2.3337 echo '#define SYSVIMRCFILE “/etc/vimrc”' » src/feature.h ./configure –prefix=/usr make -j $PARALLEL_JOBS make install ln -sv vim /usr/bin/vi for L in /usr/share/man/{,*/}man1/vim.1; do ln -sv vim.1 $(dirname $L)/vi.1 done ln -sv ../vim/vim82/doc /usr/share/doc/vim-8.2.3337 cat > /etc/vimrc « “EOF” “ Begin /etc/vimrc ” Ensure defaults are set before customizing settings, not after source $VIMRUNTIME/defaults.vim let skipdefaultsvim=1 set nocompatible set backspace=2 set mouse= syntax on if (&term == “xterm”) || (&term == “putty”) set background=dark endif “ End /etc/vimrc EOF cd /sources rm -rf vim-8.2.3337 echo ”# 8.69. Eudev-3.2.10“ tar -zxf eudev-3.2.10.tar.gz cd eudev-3.2.10 ./configure –prefix=/usr \ –bindir=/usr/sbin \ –sysconfdir=/etc \ –enable-manpages \ –disable-static make -j $PARALLEL_JOBS mkdir -pv /lib/udev/rules.d mkdir -pv /etc/udev/rules.d make install tar -Jxf ../udev-lfs-20171102.tar.xz make -f udev-lfs-20171102/Makefile.lfs install udevadm hwdb –update cd /sources rm -rf eudev-3.2.10 echo ”# 8.70. Man-DB-2.9.4“ tar -Jxf man-db-2.9.4.tar.xz cd man-db-2.9.4 ./configure –prefix=/usr \ –docdir=/usr/share/doc/man-db-2.9.4 \ –sysconfdir=/etc \ –disable-setuid \ –enable-cache-owner=bin \ –with-browser=/usr/bin/lynx \ –with-vgrind=/usr/bin/vgrind \ –with-grap=/usr/bin/grap \ –with-systemdtmpfilesdir= \ –with-systemdsystemunitdir= make -j $PARALLEL_JOBS make install cd /sources rm -rf man-db-2.9.4 echo ”# 8.71. Procps-ng-3.3.17“ tar -Jxf procps-ng-3.3.17.tar.xz cd procps-3.3.17 ./configure –prefix=/usr \ –docdir=/usr/share/doc/procps-ng-3.3.17 \ –disable-static \ –disable-kill make -j $PARALLEL_JOBS make install cd /sources rm -rf procps-3.3.17 echo ”# 8.72. Util-linux-2.37.2“ tar -Jxf util-linux-2.37.2.tar.xz cd util-linux-2.37.2 ./configure ADJTIMEPATH=/var/lib/hwclock/adjtime \ –libdir=/usr/lib \ –docdir=/usr/share/doc/util-linux-2.37.2 \ –disable-chfn-chsh \ –disable-login \ –disable-nologin \ –disable-su \ –disable-setpriv \ –disable-runuser \ –disable-pylibmount \ –disable-static \ –without-python \ –without-systemd \ –without-systemdsystemunitdir \ runstatedir=/run make -j $PARALLELJOBS make install cd /sources rm -rf util-linux-2.37.2 echo ”# 8.73. E2fsprogs-1.46.4“ tar -zxf e2fsprogs-1.46.4.tar.gz cd e2fsprogs-1.46.4 mkdir -v build cd build ../configure –prefix=/usr \ –sysconfdir=/etc \ –enable-elf-shlibs \ –disable-libblkid \ –disable-libuuid \ –disable-uuidd \ –disable-fsck make -j $PARALLELJOBS make install rm -fv /usr/lib/{libcomerr,libe2p,libext2fs,libss}.a if $INSTALL_OPTIONAL_DOCS = 1 ; then gunzip -v /usr/share/info/libext2fs.info.gz install-info –dir-file=/usr/share/info/dir /usr/share/info/libext2fs.info makeinfo -o doc/comerr.info ../lib/et/comerr.texinfo install -v -m644 doc/comerr.info /usr/share/info install-info –dir-file=/usr/share/info/dir /usr/share/info/comerr.info fi cd /sources rm -rf e2fsprogs-1.46.4 echo ”# 8.74. Sysklogd-1.5.1“ tar -zxf sysklogd-1.5.1.tar.gz cd sysklogd-1.5.1 sed -i '/Error loading kernel symbols/{n;n;d}' ksymmod.c sed -i 's/union wait/int/' syslogd.c make -j $PARALLELJOBS make BINDIR=/sbin install cat > /etc/syslog.conf « “EOF” # Begin /etc/syslog.conf auth,authpriv.* -/var/log/auth.log .;auth,authpriv.none -/var/log/sys.log daemon.* -/var/log/daemon.log kern.* -/var/log/kern.log mail.* -/var/log/mail.log user.* -/var/log/user.log *.emerg * # End /etc/syslog.conf EOF cd /sources rm -rf sysklogd-1.5.1 echo ”# 8.75. Sysvinit-2.99“ tar -Jxf sysvinit-2.99.tar.xz cd sysvinit-2.99 patch -Np1 -i ../sysvinit-2.99-consolidated-1.patch make -j $PARALLEL_JOBS make install cd /sources rm -rf sysvinit-2.99 echo -e ”——————————————————————–“ echo -e “\nYou made it! Now there are just a few things left to take care of…” printf 'Total script time: %s\n' $(timer $total_time) echo -e “\nYou have not set a root password yet. Go ahead, I'll wait here.\n” passwd root echo -e “\nNow about the firmware…” echo “You probably want to copy the supplied Broadcom libraries to /opt/vc?” select yn in “Yes” “No”; do case $yn in Yes) tar -zxf master.tar.gz cp -rv /sources/firmware-master/hardfp/opt/vc /opt echo ”/opt/vc/lib“ » /etc/ld.so.conf.d/broadcom.conf ldconfig if | "$RPI_MODEL" == "64" ; then tar -zxf v2021.04.29-138a1.tar.gz cd rpi-eeprom-2021.04.29-138a1 cp -v rpi-eeprom-update-default /etc/default/rpi-eeprom-update cp -v rpi-eeprom-config rpi-eeprom-update /opt/vc/bin mkdir -pv /lib/firmware/raspberrypi cp -rv firmware /lib/firmware/raspberrypi/bootloader cd /sources rm -rf rpi-eeprom-2021.04.29-138a1 fi break ;; No) break;; esac done echo -e “\nIf you're not going to compile your own kernel you probably want to copy the kernel modules from the firmware package to /lib/modules?” select yn in “Yes” “No”; do case $yn in Yes) cp -rv /sources/firmware-master/modules /lib; break;; No) break;; esac done echo -e “\nLast question, if you want I can mount the boot partition and overwrite the kernel and bootloader with the one you downloaded?” select yn in “Yes” “No”; do case $yn in Yes) mount /dev/mmcblk0p1 /boot && cp -rv /sources/firmware-master/boot / && umount /boot; break;; No) break;; esac done echo -e “\nThere, all done! Now continue reading from \”8.76. About Debugging Symbols\” to make your system bootable.“ echo “And don't forget to check out https://intestinate.com/pilfs/beyond.html when you're done with your build!” </codedoc> </codedoc> ### Préparation de la distribution [2.2. Configuration système hôte requise] Il est préférable d'avoir la quantité maximale de RAM disponible pour la construction. Utiliser soit raspi-config pour réduire la quantité de mémoire allouée au GPU (graphique) à 16, ou ajouter gpu_mem=16 à /boot/config.txt et redémarrer. Si on utilise un Pi avec 256 Mo de RAM, on aura besoin d'un espace de swap. Sur Raspbian, définir un mot de passe root avec `sudo passwd root` et se connecter en tant que root. Ajouter les paquets prérequis : `apt update apt install bison gawk m4 texinfo` ### Ajouter une partition LFS [2.4. Création d'une *nouvelle partition] Utiliser **fdisk** ou **cfdisk** pour travailler sur la table de partition de la carte SD : `cfdisk /dev/mmcblk0` * **mmcblk0p1** est la partition FAT32 qui est montée sous /boot et contient le chargeur de démarrage et le noyau Linux. * **mmcblk0p2** contient le système hôte. Sélectionner l'espace libre et créer une nouvelle partition principale avec l'espace restant. Il faut ensuite écrire les modifications sur le disque avant de quitter. Il faudra peut-être redémarrer à ce stade pour que Linux puisse voir la nouvelle partition.*

Il n'est pas nécessaire de créer une partition de swap dédiée ! Un fichier d'échange peut être ajouté à tout moment ultérieurement et est plus flexible tout en offrant les mêmes performances.

### Création d'un système de fichiers ext4 [2.5. Création d'un système de fichiers sur la partition] Utiliser la commande suivante pour créer un système de fichiers ext4 sur la nouvelle partition : `mkfs.ext4 -m 1 -L MyLFS /dev/mmcblk0p3` ### Paramétrage de la variable $LFS [2.7. Montage de la nouvelle partition] Pour construire sur une nouvelle partition, procéder comme suit : `export LFS=/mnt/lfs mkdir -pv $LFS mount -v -t ext4 /dev/mmcblk0p3 $LFS` Pour construire dans un répertoire faire simplement ceci : `export LFS=/lfs mkdir -pv $LFS` ### Récupérer tous les paquets [3.1. Introduction] Étant donné que le Pi a besoin d'un ensemble de packages et de correctifs légèrement différent, utiliser la wget-list suivante pour tout récupérer (y compris les scripts de construction, etc.) : `mkdir -v $LFS/sources chmod -v a+wt $LFS/sources cd $LFS/sources wget https://intestinate.com/pilfs/scripts/wget-list wget --input-file=wget-list --continue --directory-prefix=$LFS/sources` ### Réglage de l'environnement utilisateur lfs [4.4. Configuration de l'environnement]

Remplacer LFS=`/mnt/lfs` par LFS=`/lfs` si on ne construit pas sur une nouvelle partition.

La variable **LFS_TGT** doit être changée en **LFS_TGT=$(uname -m)-lfs-linux-gnueabihf**

### Build Chapitre 5 & 6 [5.2. Binutils - Passe 1]

Comme la construction est relativement longue il est recommandé de démarrer une session **tmux**[1] ou **screen**[2], parce que si on interromp la connexion ssh ou si le routeur a un problème, la construction s'arrêtera tout simplement.

Il faut d'abord éditer le script **ch5-build.sh** pour placer le modèle Pi en haut. Ensuite, commencer la construction comme ceci: `cd $LFS/sources chmod +x ch5-build.sh ./ch5-build.sh` Le script affichera le temps SBU après la fin de la première construction de binutils. ### Build Chapitre 7 [7.7. Libstdc++ de GCC - Passe 2] Avant d'entrer dans le chroot et de commencer la construction, éditer le script **ch7-build.sh** pour définir quelques paramètres facultatifs en haut. Entrer ensuite le chroot et exécuter le script : `cd/sources chmod +x ch7-build.sh ./ch7-build.sh` ###

Présentation des scripts d'amorçage PiLFS [9.6. Utilisation et configuration du script de démarrage System V] Le Pi n'a pas de moyen de garder le temps entre les redémarrages. Lorsqu'il démarre, il n'a aucune idée de l'heure qu'il est jusqu'à ce qu'il puisse récupérer l'heure correcte à partir d'un serveur NTP. Pour résoudre ce problème et quelques autres éléments spécifiques à Pi, quelques scripts de démarrage provenant de différentes sources ont été regroupés dans une archive tar pour une installation facile, ajouter ces quatre scripts : `make install-networkfix install-swapfix install-fake-hwclock install-switch-cpu-governor` ### Création du fstab [10.2. Création du fichier /etc/fstab] Tout d'abord, un fstab adapté à la nouvelle méthode de partition. Ici, on prend la partition de l'ancienne distribution et on la transforme en un espace dédié aux répertoires personnels de l'utilisateur (il faudra évidemment effacer l'ancien contenu après le premier démarrage). `/dev/mmcblk0p1 /boot vfat defaults 0 0 /dev/mmcblk0p2 /home ext4 defaults,noatime 0 1 /dev/mmcblk0p3 / ext4 defaults,noatime 0 2 #/swapfile 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` Si on opte pour la méthode de remaniement PC, fstab ressemblera plutôt à ceci : `/dev/mmcblk0p1 /boot vfat defaults 0 0 /dev/mmcblk0p2 / ext4 defaults,noatime 0 1 #/swapfile 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` ### Noyau, chargeur de démarrage et micrologiciel [10.3. Linux] Bien qu'on puisse potentiellement créer un noyau Linux, il est préférable de s'assurer d'abord que le système démarre correctement avec le noyau de la Fondation Raspberry Pi. Il n'y a pas de GRUB sur le Pi, et on compte plutôt sur le chargeur de démarrage de la Fondation Raspberry Pi pour nous démarrer. Si on a ajouté une nouvelle partition LFS, quitter le chroot à ce stade et éditer `/boot/cmdline.txt`. Il faut remplacer `root=/dev/mmcblk0p2` par `root=/dev/mmcblk0p3` afin que le chargeur de démarrage utilise la nouvelle partition LFS comme système de fichiers racine. Certains composants de la carte Raspberry Pi, comme les contrôleurs Wi-Fi et Bluetooth, nécessitent le chargement des blobs binaires du micrologiciel au démarrage. Ce sont des fichiers stockés dans `/lib/firmware` et le moyen le plus simple de les obtenir est simplement de les copier à partir de Raspbian ou de l'image de base PiLFS. ### Fin [11.3. Redémarrage du système] Alors voilà. le système est maintenant amorçable et on peut ajouter quelques éléments avant de l'utiliser (dhcpcd, wget, OpenSSH et ntp). ## Dépannage ### Réparation de la carte SD Maintenant, on va voir le processus de réparation de le carte SD sur un PC Linux. Monter la partition de données endommagée: `mount /dev/sdb2 /mnt` Ensuite, effacer tout sauf le répertoire /lfs, comme ceci : `cd /mnt shopt -s extglob rm -rf !(lfs)` Et maintenant on recopie l'ancien root : `mv /mnt/lfs/* /mnt` La commande mv conservera toutes les autorisations de fichier par défaut, mais si on souhaite plutôt copier lre système LFS, il faut ajouter l'indicateur de conservation des autorisations à cp, c'est-à-dire `cp -rp` ### Créer une image de carte SD adaptée à la distribution La partie délicate de la préparation d'une image de carte SD est qu'une copie d'image directe à l'aide de dd prend la même taille que l'ensemble de la carte SD. Voici comment préparer une image de carte SD à partir de zéro : Tout d'abord, il faut s'assurer d'avoir une copie complète de tous les fichiers qui vont aller sur la carte, y compris les éléments qui vont sur la partition FAT32 comme le chargeur de démarrage et le noyau. Ensuite, on doit recréer la table de partition à partir de zéro. Par exemple avec **Parted** pour l'édition de partition en raison de sa fonction d'alignement automatique. Effacer tout ce qui se trouve sur votre carte, puis créer une nouvelle table: N'éditez pas le mauvais disque… triple-vérifiez ! :) `parted /dev/sdb mkpart primary fat32 4 273` Cela crée la partition de démarrage FAT32, selon la disposition de Raspbian, d'une taille de 268 Mo. Ensuite, nous créer la partition de données : `mkpart primary ext4 277 2000` Ici, on définit une image qui pourrait s'adapter parfaitement sur une carte SD de 2 Go. Maintenant, si on affiche la table, on doit avoir quelque chose comme ceci : `Number Start End Size Type File system Flags 1 4194kB 273MB 268MB primary fat32 lba 2 277MB 2000MB 1723MB primary` `ext4` Maintenant qu'on a une nouvelle disposition de partition, formater les partitions, comme ceci : `mkdosfs -F 32 -n Pi-Boot -v /dev/sdb1 mkfs.ext4 -m 1 -L MyLFS /dev/sdb2` Copier ensuite toutes les binaires en place : `mount /dev/sdb1 /mnt cp -rvp bootfiles/* /mnt umount /mnt mount /dev/sdb2 /mnt cp -rvp lfsfiles/* /mnt umount /mnt` Utiliser dd pour faire une image de la carte mais en inclueant toute la partition ext4 : `dd if=/dev/sdb of=mylfs.img bs=1M count=2000`

[1] **tmux** est un multiplexeur de terminal. Il permet de basculer facilement entre plusieurs programmes dans un terminal, de les détacher pour continuer de fonctionner en arrière-plan et de les rattacher à un autre terminal.

[2] **screen** est un gestionnaire de fenêtres plein écran qui multiplexe un terminal physique entre plusieurs processus, généralement des shells interactifs. Lorsque **screen** est appelé, il crée une fenêtre unique, ensuite, à tout moment, on peut créer de nouvelles fenêtres avec d'autres programmes, tuer la fenêtre actuelle, afficher une liste des fenêtres actives, activer et désactiver la connexion de sortie, copier du texte entre fenêtres, afficher l'historique de défilement, basculer entre les fenêtres, etc. Toutes les fenêtres exécutent leurs programmes de manière totalement indépendante les unes des autres. Les programmes continuent de s'exécuter lorsque leur fenêtre n'est actuellement pas visible et même lorsque toute la session d'écran est détachée du terminal de l'utilisateur..