The LFS system will be built by using an already installed Linux distribution (such as Debian, OpenMandriva, Fedora, or openSUSE). This existing Linux system (the host) will be used as a starting point to provide necessary programs, including a compiler, linker, and shell, to build the new system. Select the “development” option during the distribution installation to be able to access these tools.

STEP 1: Preparing the Host system

* Creating a New Partition

LFS system requires a root partition of 10 GB . The *swap* partition for an LFS system can be the same as the one used by the host system. The Grub Bios partition must be on the drive that the BIOS uses to boot the system. This is not necessarily the same drive where the LFS root partition is located. Disks on a system may use different partition table types. The requirement for this partition depends only on the partition table type of the boot disk.

* Creating a File System on the Partition

LFS can use any file system recognized by the Linux kernel, but the most common types are ext3 and ext4. LFS assumes that the root file system (/) is of type ext4.

* Setting The $LFS Variable

The environmental variable LFS should be set to the name of the directory where LFS system is to be build. Choose a directory location and set the variable with the command export LFS =(directory path). Whenever $LFS is used , the shell will replace it with the directory path.

* Mounting the New Partition

Now, since the file system has been created, the partition needs to be made accessible. In order to do this, the partition needs to be mounted at a chosen mount point.

STEP 2: Create a working directory to store the downloaded packages

The working directory will be considered as $LFS/sources. The packages to be downloaded are:

* + Bash-2.05a
  + Binutils-2.12
  + Bison-1.875
  + Bzip2-1.0.2
  + Coreutils-5.0
  + Findutils-4.1.2
  + Grep-2.5.1a
  + Gzip-1.3.12
  + Make-4.0
  + Python 3.4
  + Perl-5.8.8
  + Texinfo-4

.http://www.linuxfromscratch.org/lfs/view/stable/chapter03/packages.html/

Donwload the above packages and the packages from the link provided above.

STEP 3: Create $LFS/tools directory

All the programs which will be compiled in constructing temporary system will be installed in this directory

STEP 4: Adding LFS User

Create a new user called *lfs* as a member of a new group (also named *lfs*) and use this user during the installation process.

groupadd lfs

To give LFS a password use passwd lfs. To give lfs the access rights to tools and source directory , use chown –v lfs (directory path). To start the login shell , use su – lfs.

STEP 5: Constructing a Temporary System

All the packages are compiled. Binutilis is the first package that should be compiled. Binutilis and Glibc requires a separate build directory.

STEP 6 : Preparing a Virtual Kernel File System

Various file systems exported by the kernel are used to communicate to and from the kernel itself. These file systems are virtual in that no disk space is used for them. The content of the file systems resides in memory.

1. Creating Initial Device Nodes: When the kernel boots the system, it requires the presence of a few device nodes, in particular the console and null devices. The device nodes must be created on the hard disk so that they are available when Linux is started
2. Mounting and Populating /dev: populate the /dev directory with devices to mount a virtual filesystem (such as *tmpfs*) on the /dev directory and allow the devices to be created dynamically on that virtual filesystem as they are detected or accessed.

STEP 7: Enter into chroot environment

chroot “$LFS” /tools/bin/env –i \

Home=/root

i option will clear all the environment variables and Home variable is been set

STEP 8: Creating essential files and symlinks

Create a number of symbolic links which will be replaced by real files

STEP 9: Install All Packages

STEP 10: Configure the system

STEP 11: Make LFS System Bootable

Install Linux -4.20.12 for linux kernel. Install the GRUB files into /boot/grub and set up the boot track.