Build Lustre MASTER with Ldiskfs & ZFS on Ubuntu 14.04 /16.04 from Git

This walk-through is targeting developers who want to explore the bleeding edge of Lustre or build on CPUs/kernel combinations not currently supported by the automated builders.

If you are evaluating Lustre for production, you should choose a stable Lustre Release.

Note that as of the writing of this page, a pre-compiled Lustre Release is only available for Ubuntu as a Lustre Client, not Server.

Purpose

This page describes all the necessary steps to:

- Download, patch and build the Linux kernel in the appropriate version for Lustre.
- (Optional) Download and build the ZFS modules, so that Lustre can link against them.
- Create the Lustre Client and Server DEB packages
- Install the packages to get a Lustre Client and/or Server

Prerequisites

- A newly installed Ubuntu 16.04 or 18.04 machine connected to the internet.
- **NOTE:** The build instructions below assume that SE-Linux is disabled. If you enable it, this guide should only be used as a source of inspiration, not of authoritative information.

Overview

As mentioned in the outset, this guide is meant for both Ubuntu 14.04 and 16.04, since there are only small differences between them.

The main difference in this guide is the versions of ZFS that will be compiled against:

- Ubuntu 16.04: zfs-linux-0.6.5.6-0ubuntu8
- Ubuntu 18.04: zfs-dkms-0.7.5-1ubuntu15

Other than that, the following package versions will be used in this guide for both Ubuntu releases:

- Linux kernel 4.15.0-32

1.) Installing the build dependency packages

To compile Lustre for use as a server, you need to be able to compile the Linux kernel and Lustre.
You also need to compile and enable at least one backing file-system – either ZFS or LDISKFS (or both).

The fastest way to install the needed tools is by using the debian build-dependency feature:

```
sudo apt-get build-dep linux-image-$(uname -r)

sudo apt-get install build-essential debhelper devscripts fakeroot kernel-wedge libudev-dev
pciutils-dev

sudo apt-get install texinfo xmlto libelf-dev python-dev liblzma-dev libaudit-dev dh-systemd
libyaml-dev
```

```
sudo apt-get install module-assistant libreadline-dev dpatch libsnmp-dev quilt
```

To enable ZFS server support for Ubuntu 14.04., you need to add the Ubuntu-ZFS PPA.

```
sudo add-apt-repository ppa:zfs-native/stable
sudo apt-get update
```

Following that, you need to run the following on both Ubuntu 16.04. as well as 18.04:

```
sudo apt-get build-dep ubuntu-zfs spl-dpkg zfs-dpkg
```

To enable LDISKFS server support, you need to add the following packages:

```
sudo apt-get install uuid-dev libblkid-dev dietlibc-dev
```

NOTE: You only have to install these on your compile node if you are building Ubuntu server support. If you are only building Lustre for client nodes, and the final servers do not need any of these packages.

The following instructions assume that you run in a directory that has at least 20 GB free disk space and is writable by your user.
It will be assumed, that this path is available as ${BUILDPATH}, such as `/usr/src/lustre.`

2.) Building the Linux Kernel

Before you can build Lustre server suppor, you will need to have a compiled kernel source directory available, for your intended kernel version. If you are building only a client, then you do **NOT** need to build a kernel, only install the kernel headers and config file.

Depending on which backing filesystem you intend to use, you must then do the following:

- **If you intend to use LDISKFS**, you need to use a specific kernel version to generate the LDISKFS modules, and optionally patch it with performance and functionality improvements.
- **If you are *only* using ZFS**, you do not need to apply any changes to the kernel and can use any version you want, but you will still need to compile it.

The following two sections will outline each scenario. **Important:** You only have to follow ***one*** of these subsections, not both.

In either case, you will have to download the correct kernel for your distribution. You can choose between using the GIT or HTTP protocol for each.

- Ubuntu 16.04
  - `git clone git://kernel.ubuntu.com/ubuntu/ubuntu-xenial.git ubuntu-kernel`
  - `git clone http://kernel.ubuntu.com/git-repos/ubuntu/ubuntu-xenial.git/ ubuntu-kernel`
- Ubuntu 18.04
  - `git clone git://kernel.ubuntu.com/ubuntu/ubuntu-bionic.git ubuntu-kernel`
  - `git clone http://kernel.ubuntu.com/git-repos/ubuntu/ubuntu-bionic.git/ ubuntu-kernel`

2.1.) (ZFS, optionally LDISKFS) Building an Unmodified Kernel

If you're only using ZFS (or prefer to use an unmodified kernel on the server with LDISKFS, with some reduced functionality and performance), it is not necessary to patch the kernel or produce any Debian packages for it. You only need to compile the kernel locally to allow building modules against it.

As such, first figure out the version of the kernel on your target servers by executing:

```
[bash-4.3]$ uname -r
4.15.0-32-generic
```

You can also target any other kernel version, but in the following we will assume you target "4.15.0-35"

The first thing to do is to go into the kernel source directory and check out that version:

```
[bash-4.3]$ cd ${BUILDPATH}/ubuntu-kernel
```

```
[bash-4.3]$ git tag | grep 4.15.0-32
Ubuntu-lts-4.15.0-32.66_18.04.1

[bash-4.3]$ git checkout Ubuntu-lts-4.15.0-32.66_18.04.1
```

Do note that this tag will be different for Ubuntu 16.04 and 18.04. As you can see here, we're using Ubuntu 18.04.

The next thing you need is the kernel configuration for that version. The easiest way to get it, is by grabbing it from a server running that version.
Assuming the current compile server already runs that kernel (which is not necessary, but certainly makes things easier):

```
cp /boot/config-4.15.0-32-generic .config
```

An alternative source of this config file is the Debian package for that kernel, which you can download over the Internet.
The file you seek is in the `linux-image-*.deb` file – where the value of "*" depends on the kernel version and whether you use the lowlatency kernel or not.
For example:

- https://packages.ubuntu.com/de/bionic-updates/linux-image-4.15.0-32-generic

Once you have the config file in place, you need to make sure that the same version number as used by Ubuntu is embedded in the kernel:

```
touch .scmversion

sed -i 's/^VERSION = .*/VERSION = 4/' Makefile
sed -i 's/^PATCHLEVEL = .*/PATCHLEVEL = 4/' Makefile
sed -i 's/^SUBLEVEL = .*/SUBLEVEL = 0/' Makefile
sed -i 's/^EXTRAVERSION =.*/EXTRAVERSION = -35-generic/' Makefile
```

Finally, you can compile the kernel source so that it is usable by Lustre:

```
make -j $(nproc)
```

Note: "`nproc`" returns the number of CPUs. If you don't have that tool or wish to use a different parallelism, just enter a number of parallel processes there.

3.) (Optional) Building ZFS packages

If you wish to compile Lustre so that it can use ZFS as a backing storage, the current compile server does **not** need to have the ZFS modules installed.
All that you need to do is, it to have the ZFS source code compiled, so that Lustre can link against its modules.

While it should be possible to compile against a subtly different ZFS version than what you will use later on the target system, the safest way is to use the same version.

First, create a directory that will hold both SPL and ZFS.

```
cd ${BUILDPATH}
mkdir zfs
cd zfs
```

Then, download the correct ZFS source code for your target Ubuntu version:

Ubuntu 16.04:

```
wget http://archive.ubuntu.com/ubuntu/pool/main/z/zfs-linux/zfs-linux_0.7.9.orig.tar.gz
wget http://archive.ubuntu.com/ubuntu/pool/main/z/zfs-linux/zfs-linux_0.7.9-3ubuntu6.debian.tar.xz

wget http://archive.ubuntu.com/ubuntu/pool/universe/s/spl-linux/spl-linux_0.7.9.orig.tar.gz
wget http://archive.ubuntu.com/ubuntu/pool/universe/s/spl-linux/spl-linux_0.7.9-3ubuntu2.debian.
tar.xz
```

Ubuntu 16.04:

```
wget http://archive.ubuntu.com/ubuntu/pool/main/z/zfs-linux/zfs-linux_0.7.12.orig.tar.gz
wget http://archive.ubuntu.com/ubuntu/pool/main/z/zfs-linux/zfs-linux_0.7.12-1ubuntu3.debian.tar.
xz

wget http://archive.ubuntu.com/ubuntu/pool/universe/s/spl-linux/spl-linux_0.7.12.orig.tar.gz
wget http://archive.ubuntu.com/ubuntu/pool/universe/s/spl-linux/spl-linux_0.7.12-1ubuntu1.debian.
tar.xz
```

The rest of the guide assumes you chose the Ubuntu 18.04. version. Adjust the file- and pathnames accordingly, if you are targeting 16.04.

Extract, patch and compile SPL (adjust the `ZFSVERSION` parameter accordingly):

```
ZFSVERSION=0.7.12

tar -xf spl-linux_${ZFSVERSION}.orig.tar.bz2
cd spl-linux-${ZFSVERSION}
tar -xf ../spl-linux_${ZFSVERSION}-1ubuntu1.debian.tar.gz
for f in debian/patches/*.patch; do patch -Np1 -i $f; done

./autogen.sh &&
./configure --with-linux=${BUILDPATH}/ubuntu-kernel &&
make -j $(nproc)
```

Then, do the same for ZFS:

```
cd ${BUILDPATH}/zfs

tar -xf zfs-linux_${ZFSVERSION}.orig.tar.bz2
cd zfs-linux-${ZFSVERSION}
tar -xf ../zfs-linux_${ZFSVERSION}-1~trusty.debian.tar.gz
for f in debian/patches/*.patch; do patch -Np1 -i $f; done

./autogen.sh &&
./configure --with-linux=${BUILDPATH}/ubuntu-kernel --with-spl=${BUILDPATH}/zfs/spl-
linux-${ZFSVERSION} &&
make -j $(nproc)
```

4.) (Optional) Building modified e2fsprogs for LDISKFS support

If you are building an LDISKFS server, you need to compile and install a modified version of the e2fsprogs package, which provides the virtual `ldiskfsprogs` package. This is **NOT** needed if you only intend to build a client, or use ZFS on the server.

To compile the needed package, execute the following steps:

```
cd ${BUILDPATH}

git clone git://git.whamcloud.com/tools/e2fsprogs.git
cd e2fsprogs
git checkout master-lustre
```

If you can't use the GIT protocol (e.g. if you're behind a corporate proxy), download a snapshot from:

- https://git.whamcloud.com/tools/e2fsprogs.git/
- Direct TGZ download: https://git.whamcloud.com/?p=tools/e2fsprogs.git;a=snapshot;h=**master-lustre**;sf=tgz

Then, generate the Ubuntu Debian packages:

```
wget -P ../ http://archive.ubuntu.com/ubuntu/pool/main/e/e2fsprogs/e2fsprogs_1.44.4-2.debian.tar.
xz
tar --exclude "debian/changelog" -xf ../e2fsprogs_1.44.4-2.debian.tar.xz

sudo apt-get install libfuse-dev libattr1-dev libblkid-dev uuid-dev

./configure
```

```
dpkg-buildpackage -b -us -uc
```

This will create several "`.deb`" files in `${BUILDPATH}`.
Save them somewhere sane, so that you can install the modified e2fsprogs on the target servers later.

Note 1: The downloading of the Ubuntu Debian install files is necessary, as the ones form the GIT repo are not Ubuntu specific.
Please make sure to use the same version numbers for the git checkout as well as the Ubuntu download.

Note 2: The '`sed`' call works around the build scripts not properly identifying a build output artifact.
This has no effect on running the `e2fsprogs`, but might cause issues if you use something that compiles against the `e2fsprogs`, as you will miss that particular header file.

5.) Building Lustre Client & Server Packages

5.1) Downloading Lustre Sources

The first step is to download the most recent Lustre sources from the HPDD Git repo:

```
cd ${BUILDPATH}

git clone git://git.whamcloud.com/fs/lustre-release.git
cd lustre-release
git checkout b2_10
```

This checks out the latest 2.10 code. If you need a newer version, adjust the checked-out branch accordingly.

All releases up to (and including) 2.10 do not come with Ubuntu Server support enabled. Later versions should have the needed changes already.

Unless GIT reports a merge-conflict, you should now have a Ubuntu-Server enabled Lustre source code base.
Versions ***later* than 2.10** most likely have this change integrated already and do not need having that change applied manually.

Now, you need to compile Lustre twice – once to generate the packages for Lustre Clients and once again to generate the packages for Lustre Servers.
They differ in the kernel modules and tools that will be installed by them.

Do note: When installing on servers, you do **not** need to install the Client packages. The Server packages include and supersede those.

5.2) Building Lustre Client Packages

Building the Client packages has the least demands and only needs the kernel headers being available.
You don't even need to have a 4.15.0-series kernel present as for the server build. The clients can run under ***any*** 2.6.32 or later kernel, as long as the kernel headers installed match the running kernel version.

Unless the build system for those has already been made compatible with DKMS (Dynamic Kernel Module Support), it will be necessary to generate a suitable package for each kernel version your clients run under.

To build the client packages, execute the following:

```
cd ${BUILDPATH}/lustre-release
git reset --hard && git clean -dfx

sh autogen.sh &&
./configure --disable-server --with-linux=${BUILDPATH}/ubuntu-kernel &&

make debs -j $(nproc)
```

Unless there was a compilation error, you should find a set of Debian packages in the "lustre-release/debs" subdirectory.
Copy these to a sane location, so that you can use them for installing Lustre Clients.

Also remember what was said earlier: *Unless DKMS has been enabled, you need to compile a separate set of modules for each target kernel version!*

5.3) Building Lustre Server Packages

When building the Server packages, you need to make sure to have all the requirements above compiled and ready, depending on which file systems you wish to support.

Of special note is, that if you want LDISKFS support, you must use a 4.15.0-series Linux kernel. Other kernel versions might need significant alterations to the EXT4-to-LDISKFS patches. If you wish to ***only use ZFS***, you do not need to care about that. Similar to the Client packages, the source code will then be compatible with ***any*** Linux kernel version >= 2.6

Now, compiling the Lustre server packages is quite straightforward:

```
cd ${BUILDPATH}/lustre-release
git reset --hard && git clean -dfx

sh autogen.sh &&
./configure --with-linux=${BUILDPATH}/ubuntu-kernel \
  --enable-server --enable-modules \
  --with-zfs="${BUILDPATH}/zfs/zfs-linux-${ZFSVERSION}" --with-spl="${BUILDPATH}/zfs/spl-linux-${ZFSVERSION}" \
  --enable-ldiskfs &&

make debs -j $(nproc)
```

If you **do not need LDISKFS**, simply drop the "`--enable-ldiskfs`" flag.
If you **do not need ZFS**, simply drop the "`--with-zfs`" and "`--with-spl`" flags. (Note: `ZFSVERSION` was set in section (3))

Do note that at least ***one*** of the two needs to be present, or the configuration will fail. After all, a server can't work without at least one backing file system.

If the compilation succeeded, you should find the server modules again under "`lustre-release/debs`".
Copy these to a sane location, so that you can use them for installing Lustre Servers.