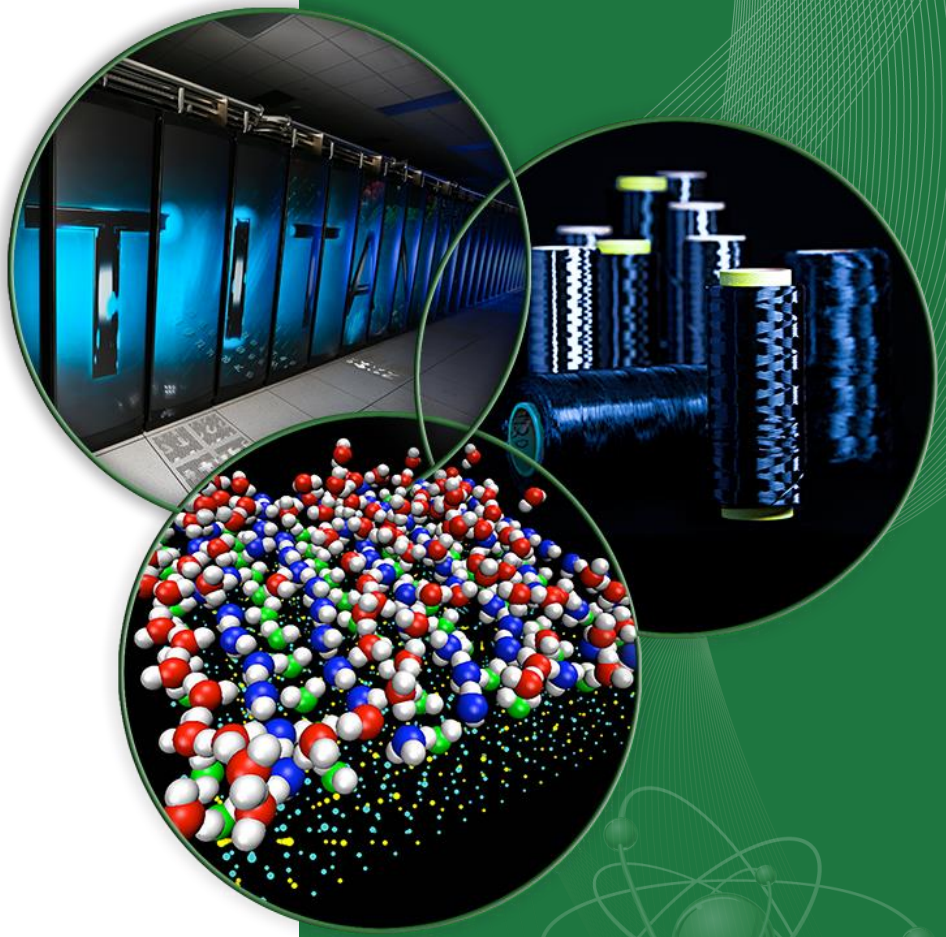
Oak Ridge National Laboratory
Computing and Computational Sciences Directorate

Creating a Lustre Test System from Source with Virtual Machines



Jeffrey Rossiter
Rick Mohr
Sarp Oral
Michael Brim
Jason Hill
Joel Reed
Neena Imam

ORNL is managed by UT-Battelle
for the US Department of Energy

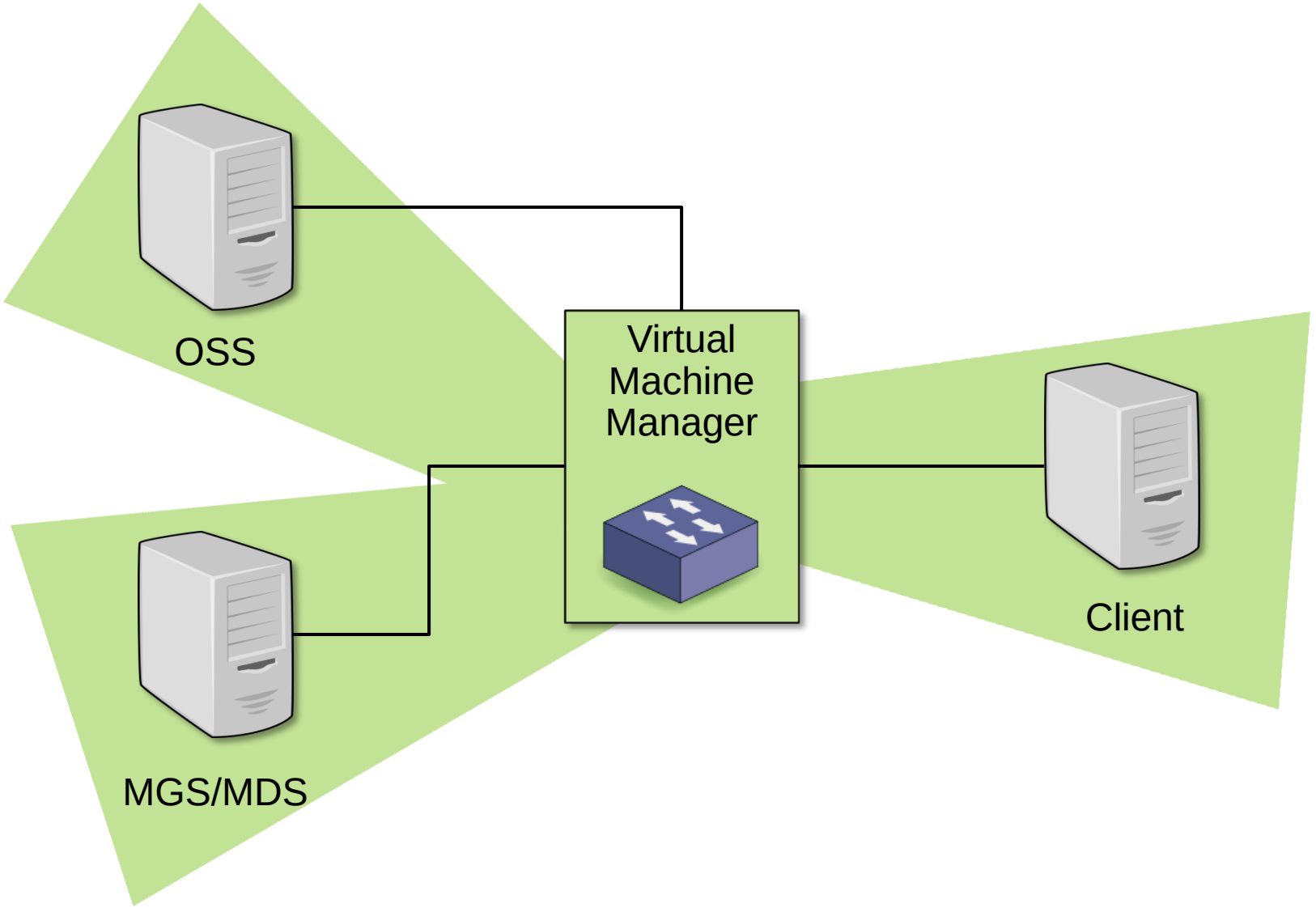
OAK RIDGE National Laboratory

Introduction

- We will be setting up a basic Lustre system using virtual machines
  - Lustre RPMs will be built from source
  - Linux kernel will be patched
  - Server and client machines will be configured

This process will be useful to system administrators or developers who want to experiment with or develop Lustre.

System Overview



Outline

- Setup a “template” VM instance
- Download Sources & build RPMs
- Install Lustre
- Create Lustre servers and client(s) from “template”
- Configure MGS/MDS
- Configure OSS/OSTs
- Start Using Lustre

Up Next

- **Setup a "template" VM instance**
- Download Sources & build RPMs
- Install Lustre
- Create Lustre servers and client(s) from "template"
- Configure MGS/MDS
- Configure OSS/OSTs
- Start Using Lustre

Setup a "template" VM Instance (1)

- Configure your instance with at least 2 storage devices

The MGS/MDS will use the devices as an MGT and an MDT. The OSS will use them as OSTs. The client will not use the devices.

Setup a "template" VM Instance (2)

- We use CentOS 6.5 for x86_64 architecture
  - Installed from the “minimal” ISO
    - http://www.centos.org/download/

Be aware that the Lustre build requires around 10 GB of disk space

Setup a "template" VM Instance (3)

- Groupinstall “Development Tools”
- Install the following tools / libraries

epel-release	quilt	libselinux-devel	python-docutils
xmlto	asciidoc	elfutils-libelf-devel	elfutils-devel
zlib-devel	rng-tools	binutils-devel	python-devel
sg3_utils	newt-devel	perl-ExtUtils-Embed	audit-libs-devel
lsof	hmaccalc		

Consider saving a snapshot so that you have a valid state to roll back to or to clone new instances from.

Setup a "template" VM Instance (4)

- Disable SELinux
  - Set *SELINUX=disabled* in */etc/sysconfig/selinux*

SELinux should be disabled because the Lustre software does not currently support SELinux.

Setup a "template" VM Instance (4)

- Disable Linux firewall

```
chkconfig --levels 345 iptables off
chkconfig --levels 345 ip6tables off
```

This is a quick and dirty approach. Lustre uses port 988 so if you want to keep the firewall up you can open port 988.

Up Next

- Setup a “template” VM instance
- **Download Sources & build RPMs**
- Install Lustre
- Create Lustre servers and client(s) from “template”
- Configure MGS/MDS
- Configure OSS/OSTs
- Start Using Lustre

Download Sources & Build RPMs (3)

- Create & switch to a user that will build RPMs

```
useradd -m build
su build
cd ~/
```

Download Sources & Build RPMs (4)

- Download & prepare Lustre source

```
git clone http//git.hpdd.intel.com/fs/lustre-release.git
--branch b2_6
cd lustre-release
sh ./autogen.sh
```

We are using the 2.6 branch of Lustre because it is the latest version compatible with CentOS 6.5 according to the Lustre Support Matrix.

Download Sources & Build RPMs (5)

- Create directories for the build process

```
cd
mkdir –p kernel/rpmbuild/{BUILD,RPMS,SOURCES,SPECS,SRPMS}
```

- Let rpmbuild know where to build

```
echo '%_topdir %(echo $HOME)/kernel/rpmbuild' >
~/.rpmmacros
```

Download Sources & Build RPMs (6)

- Install Linux kernel source RPM

```
cd kernel
rpm -ivh
http://ftp.redhat.com/pub/redhat/linux/enterprise/6Server/
en/os/SRPMS/kernel-2.6.32-431.20.3.el6.src.rpm 2>&1 | grep
-v mockb
```

We use this version of the kernel because the Lustre 2.6 branch lists it in /lustre/kernel_patches/which_patch.

Download Sources & Build RPMs (7)

- Prepare the kernel source RPM for building

```
su root
rngd -r /dev/urandom
exit
cd ~/kernel/rpmbuild
rpmbuild -bp --target=`uname -m` ./SPECS/kernel.spec
```

Using rngd is not required but will speed up the rpmbuild process by keeping the kernel's entropy pool full for PGP key generation.

Download Sources & Build RPMs (8)

- Edit the kernel Makefile
  - edit
    - `~/kernel/rpmbuild/BUILD/kernel-2.6.32-431.20.3.el6/linux-2.6.32-431.20.3.el6.x86_64/Makefile`
  - set
    - `EXTRAVERSION = .431.20.3.el6_lustre`

Setting EXTRAVERSION will differentiate the Lustre-patched kernel from other kernels that may be present on your system.

Download Sources & Build RPMs (9)

This slide and the next two deal with patching the Linux kernel. In our case we need to patch the kernel because we are using ldiskfs. If you use ZFS instead you don't need to patch the Linux kernel.

- Copy the Lustre version of the kernel config file

```
cd ~/kernel/rpmbuild/BUILD/kernel-2.6.32-
431.20.3.el6/linux-2.6.32-431.20.3.el6.x86_64/
cp ~/lustre-
release/lustre/kernel_patches/kernel_configs/kernel-
2.6.32-2.6-rhel6-x86_64.config ./.config
```

Download Sources & Build RPMs (10)

- Link Lustre kernel patches into the kernel build directories

```
ln -s ~/lustre-release/lustre/kernel_patches/series/2.6-rhel6.series series
ln -s ~/lustre-release/lustre/kernel_patches/patches patches
```

Download Sources & Build RPMs (11)

- Apply the Lustre kernel patches

```
quilt push -av
```

Download Sources & Build RPMs (12)

- Build the kernel RPM

```
cd ~/kernel/rpmbuild/BUILD/kernel-2.6.32-
431.20.3.el6/linux-2.6.32-431.20.3.el6.x86_64/
make oldconfig || make menuconfig
make include/asm
make include/linux/version.h
make SUBDIRS=scripts
make include/linux/utsrelease.h
make rpm
```

Download Sources & Build RPMs (13)

- Build the Lustre RPMs

```
cd ~/lustre-release/
./configure --with-
linux=/home/build/kernel/rpmbuild/BUILD/kernel-
2.6.32.431.20.3.el6_lustre/
make rpms
```

If you are using ZFS then you will have to add --with-spl=/path/to/spl/source and --with-zfs=/path/to/zfs/source options to the configure command.

Consider saving a snapshot so that you have a valid state to roll back to or to clone new instances from.

Download Sources & Build RPMs (14)

- You should now have the following RPMs in ~build/kernel/rpmbuild/RPMS/x86_64

Lustre RPM	Description
lustre-2.6	Provides user space tools and files for Lustre
lustre-iokit	Provides a collection of Lustre benchmark tools
lustre-modules	Server and network drivers for the kernel.
lustre-osd-ldiskfs (or lustre-osd-zfs)	Provides an OSD API for using the ldiskfs (or zfs) file system on the Lustre servers
Lustre-tests	Provides binaries and scripts for Lustre testing framework

Up Next

- Setup a “template” VM instance
- Download Sources & build RPMs
- **Install Lustre**
- Create Lustre servers and client(s) from “template”
- Configure MGS/MDS
- Configure OSS/OSTs
- Start Using Lustre

Install Lustre (1)

- Install the kernel RPM

```
su root
rpm -ivh ~build/kernel/rpmbuild/RPMS/x86_64/kernel-
2.6.32.431.20.3.el6_lustre-1.x86_64.rpm
```

- Install the kernel

```
/sbin/new-kernel-pkg --package kernel --mkinitrd --dracut
--depmod --install 2.6.32.431.20.3.el6_lustre
reboot
```

Install Lustre (2)

- Download tools needed by Lustre from https://downloads.hpdd.intel.com/public/e2fsprogs/latest/el6/RPMS/x86_64/

RPM	Description
e2fsprogs-1.42.12.wc1-7.el6.x86_64.rpm	Provides utilities for working with ext2/ext3/ext4 file systems
e2fsprogs-libs-1.42.12.wc1-7.el6.x86_64.rpm	Provides e2fsprogs shared libraries
libcom_err-1.42.12.wc1-7.el6.x86_64.rpm	Provides an error description library for e2fsprogs
libss-1.42.12.wc1-7.el6.x86_64.rpm	Provides a command line interface parsing library for e2fsprogs

Install Lustre (3)

- Upgrade / install the tools

```
rpm -Uvh e2fsprogs-1.42.12.wc1-7.el6.x86_64.rpm e2fsprogs-
libs-1.42.12.wc1-7.el6.x86_64.rpm libcom_err-1.42.12.wc1-
7.el6.x86_64.rpm libss-1.42.12.wc1-7.el6.x86_64.rpm
```

Install Lustre (4)

- Install Lustre modules

```
cd ~build/kernel/rpmbuild/RPMS/x86_64
rpm -ivh lustre-modules-*
rpm -ivh lustre-osd-ldiskfs-*
rpm -ivh lustre-2.6*
rpm -ivh lustre-iokit-2.6*
rpm -ivh lustre-tests-*
```

Up Next

- Setup a “template” VM instance
- Download Sources & build RPMs
- Install Lustre
- **Create Lustre servers and client(s) from “template”**
- Configure MGS/MDS
- Configure OSS/OSTs
- Start Using Lustre

Create Lustre Servers and Client(s) (1)

- Create an entry in */etc/modprobe.d/lustre.conf*

```
options lnet networks=tcp # or networks=tcp(eth0)
```

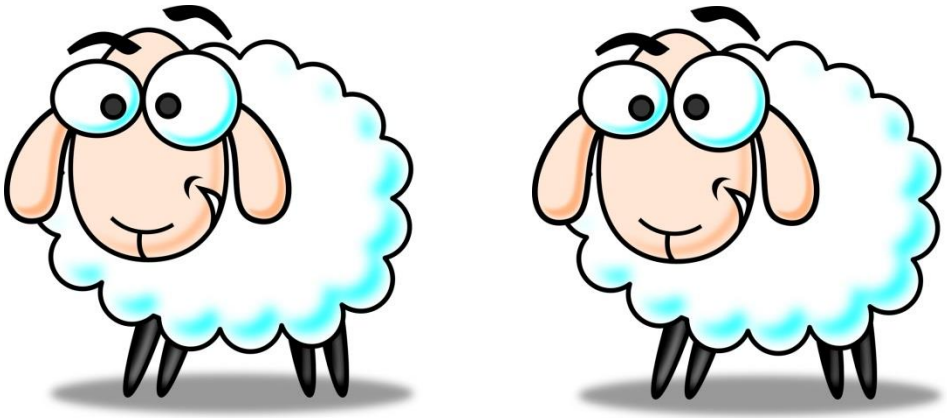
The *options lnet networks* entry specifies which interfaces should be mapped to which LNET subnets.

Here we are saying that Lustre should use the LNET *tcp* subnet. Since we haven't specified which Ethernet interface to use, LNET will use the first available non-loopback interface.

The comment in the example specifies that eth0 should be part of the LNET tcp subnet. LNET also supports an InfiniBand subnet (o2ib).

Create Lustre Servers and Client(s) (2)

- Shut the machine down so that it can be cloned
- Clone “template” instance to create combined MGS/MDS
- Clone “template” instance to create OSS
- Clone “template” instance to create client(s)



Create Lustre Servers and Client(s) (3)

- Ensure that interface hardware addresses are unique/updated on each instance.

Your virtual machine manager should assign new MAC addresses for each clone. You may need to update the interface configuration files (e.g., ifcfg-eth0) to reflect the changes.

Create Lustre Servers and Client(s) (4)

- Ensure that each instance has a non-loopback entry for itself in */etc/hosts*

Lustre can only use non-loopback IP addresses.

Up Next

- Setup a “template” VM instance
- Download Sources & build RPMs
- Install Lustre
- Create Lustre servers and client(s) from “template”
- **Configure MGS/MDS**
- Configure OSS/OSTs
- Start Using Lustre

Configure MGS/MDS (1)

- Format the MGT for Lustre

```
[root@mgs_mds]$ mkfs.lustre --mgs /dev/sdb
```

Here we are using the default file system for Lustre targets, ldiskfs. This was for the sake of simplicity. Alternatively you could use multiple devices configured as ZFS zpools.

Configure MGS/MDS (2)

- Format the MDT for Lustre

```
[root@mgs_mds]$ mkfs.lustre --fsname=lustre \
--mgsnode=192.168.56.100@tcp --mdt --index=0 /dev/sdc
```

The IP addresses we use were assigned by the DHCP server of our virtual machine manager and were not chosen for any particular reason. You could use hostnames (e.g., --mgsnode=mgs@tcp) instead if you add them to your */etc/hosts* file.

Configure MGS/MDS (3)

- Create mount point for MGT and mount it

```
[root@mgs_mds]$ mkdir /mnt/mgt
[root@mgs_mds]$ mount -t lustre /dev/sdb /mnt/mgt
```

- Create mount point for MDT and mount it

```
[root@mgs_mds]$ mkdir /mnt/mdt
[root@mgs_mds]$ mount -t lustre /dev/sdc /mnt/mdt
```

Up Next

- Setup a "template" VM instance
- Download Sources & build RPMs
- Install Lustre
- Create Lustre servers and client(s) from "template"
- Configure MGS/MDS
- **Configure OSS/OSTs**
- Start Using Lustre

Configure OSS & OSTs (1)

- Format the OSTs for Lustre

```
[root@oss]$ mkfs.lustre --fsname=lustre --ost \
--mgsnode=192.168.56.100@tcp --index=0 /dev/sdb
[root@oss]$ mkfs.lustre --fsname=lustre --ost \
--mgsnode=192.168.56.100@tcp --index=1 /dev/sdc
```

Configure OSS & OSTs (2)

- Create mount points for OSTs and mount them

```
[root@oss]$ mkdir /mnt/ost0 /mnt/ost1
[root@oss]$ mount -t lustre /dev/sdb /mnt/ost0
[root@oss]$ mount -t lustre /dev/sdc /mnt/ost1
```

Up Next

- Setup a "template" VM instance
- Download Sources & build RPMs
- Install Lustre
- Create Lustre servers and client(s) from "template"
- Configure MGS/MDS/MDT
- Configure OSS/OSTs
- **Start Using Lustre**

Mount Lustre on Client(s) (1)

- Create mount point for Lustre and mount it

```
[root@client]$ mkdir /mnt/lustre
[root@client]$ mount -t lustre \
192.168.56.100@tcp:/lustre /mnt/lustre
```

- Create a test file to be sure the file system is working

```
[root@client]$ touch /mnt/lustre/testFile
[root@client]$ ls /mnt/lustre
```

Celebrate

- Congratulations! You should now have a working Lustre system running on virtual machines

Acknowledgements

This work was supported by the United States Department of Defense (DoD) and used resources of the DoD-HPC Program at Oak Ridge National Laboratory.