Sun Microsystems

Preparing the Sun Fire X4500 for Lustre Installation

Oct 2006

VERSION CONTROL

Business owner:
Address: Sun Microsystems
Telephone:
Fax:
E-mail:

VERSION 1.0

Version	*Date*	*Written/Modified By*
1.0	02nd Nov 2006	Syuuchii Iihara, Lawrence McIntosh, Michael Berg, Deepak Jeevan Kumar

Table of Contents

1 Introduction

This document describes the following

- Sun Fire X4500 internals and Linux device names
- Installation procedure for RHEL4U4 on Sun Fire X4500 using N1SM
- Re-installation procedure for RHEL4U4 on Sun Fire X4500 using N1SM
- Preparation of the Sun Fire X4500 for Lustre installation

2 Sun Fire X4500 internals and Linux device names

The technical specifications of the Sun Fire X4500 can be found at http://www.sun.com/x4500

The salient points to note regarding the 48 disks are:

- 48 SATA disks spread across 6 controllers
- 8 disks per controller
- only 2 disks are bootable (they are on controller 3)

The following diagram describes the Linux device names that are assigned by default to the disks. It is important to note that in case some disks are missing, there is a possibility that this assignment could change.

SYSTEM FRONT

	sdy	sdz	sdaa	sdab	CTLR-3	
	sdac	sdad	sdae	sdaf		
	sdq	sdr	sds	sdt	CTLR-2	
	sdu	sdv	sdw	sdx		
	sdao	sdap	sdaq	sdar	CTLR-5	PS-1
	sdas	sdat	sdau	sdav		
	sdag	sdah	sdai	sdaj	CTLR-4	
	sdak	sdal	sdam	sdan		
	sdi	sdj	sdk	skl	CTLR-1	PS-0
	sdm	sdn	sdo	sdp		
	sda	sdb	sdc	sdd	CTLR-0	
	sde	sdf	sdg	sdh		

SYSTEM REAR

The bootable disks are highlighted in bold face (sdy and sdac).

Under Solaris, the device names will look like in the diagram below:

SYSTEM FRONT

c5	**c5t0d0**	c5t1d0	c5t2d0	c5t3d0	CTLR-3	
	c5t4d0	c5t5d0	c5t6d0	c5t7d0		
c4	c4t0d0	c4t1d0	c4t2d0	c4t3d0	CTLR-2	
	c4t4d0	c4t5d0	c4t6d0	c4t7d0		
c7	c7t0d0	c7t1d0	c7t2d0	c7t3d0	CTLR-5	PS-1
	c7t4d0	c7t5d0	c7t6d0	c7t7d0		
c6	c6t0d0	c6t1d0	c6t2d0	c6t3d0	CTLR-4	
	c6t4d0	c6t5d0	c6t5d0	c6t7d0		
c1	c1t0d0	c1t1d0	c1t2d0	c1t3d0	CTLR-1	PS-0
	c1t4d0	c1t5d0	c1t6d0	c1t7d0		
c0	c0t0d0	c0t1d0	c0t2d0	c0t3d0	CTLR-0	
	c0t4d0	c0t5d0	c0t6d0	c0t7d0		

SYSTEM REAR

The following diagram illustrates the SCSI host ids assigned to the disks when is OS is being installed:

SYSTEM FRONT											SYSTEM REAR
	26	sdy	27	sdz	28	sdaa	29	sdab	CTLR-3		
	30	sdac	31	sdad	31	sdae	33	sdaf			
	18	sdq	19	sdr	20	sds	21	sdt	CTLR-2		
	22	sdu	23	sdv	24	sdw	25	sdx			
	42	sdao	43	sdap	44	sdaq	45	sdar	CTLR-5	PS-1	
	46	sdas	47	sdat	48	sdau	49	sdav			
	34	sdag	35	sdah	36	sdai	37	sdaj	CTLR-4		
	38	sdak	39	sdal	40	sdam	41	sdan			
	10	sdi	11	sdj	12	sdk	13	skl	CTLR-1	PS-0	
	14	sdm	15	sdn	16	sdo	17	sdp			
	2	sda	3	sdb	4	sdc	5	sdd	CTLR-0		
	6	sde	7	sdf	8	sdg	9	sdh			

The following diagram illustrates the SCSI host ids assigned to the disks after the OS has been installed, that is, during normal system operation:

SYSTEM FRONT											SYSTEM REAR
	24	sdy	25	sdz	26	sdaa	27	sdab	CTLR-3		
	28	sdac	29	sdad	30	sdae	31	sdaf			
	16	sdq	17	sdr	18	sds	19	sdt	CTLR-2		
	20	sdu	21	sdv	22	sdw	23	sdx			
	40	sdao	41	sdap	42	sdaq	43	sdar	CTLR-5	PS-1	
	44	sdas	45	sdat	46	sdau	47	sdav			
	32	sdag	33	sdah	34	sdai	35	sdaj	CTLR-4		
	36	sdak	37	sdal	38	sdam	39	sdan			
	8	sdi	9	sdj	10	sdk	11	skl	CTLR-1	PS-0	
	12	sdm	13	sdn	14	sdo	15	sdp			
	0	sda	1	sdb	2	sdc	3	sdd	CTLR-0		
	4	sde	5	sdf	6	sdg	7	sdh			

3 Installation procedure for RHEL4U4 on Sun Fire X4500 using N1SM

The first step to set up a profile for installation of RHELU4 on the N1SM server. Assuming this has been done, you can read ahead.

There are two main problems that are encountered here:

- **PROBLEM 1:** For reasons unknown, N1SM is unable to install the OS on either of the two X4500 boot disks (/dev/sdy and /dev/sdac). It is able to install the OS only on /dev/sda. However, as /dev/sda is not bootable, when the server restarts after installation, the BIOS is unable to find a bootable OS
- **PROBLEM 2:** Most of the X4500s come with Solaris ZFS installed. This creates problems under Linux as the partition tables under individual disks are recognised as GNU parted partition tables. This creates problems subsequently during raid groups creation for Lustre.

3.1 PROBLEM 1: Inability of N1SM to install OS on X4500 boot disks

This problem is solved by adding a kick-start post installation script to the N1SM X4500 OS installation profile. This script accomplishes the following:

- Clears the partition table in /dev/sdy (SCSI host 26 during OS installation)
- Clears the partition table in /dev/sdac (SCSI host 30 during OS installation)
- Makes appropriate changes to GRUB device maps and /etc/fstab
- Creates mirrors for /boot, / and swap on /dev/sdy and /dev/sdac
- Copies files from /dev/sda to these newly created mirrors
- Clears the partition table on /dev/sda

```
# sda2sdy.sh
#!/bin/sh

BOOTDEV="/dev/md0"
ROOTDEV="/dev/md1"
SWAPDEV="/dev/md2"
MOUNTPT=/tmp/a
KERNEL=2.6.9-42.EL
LUSTERKERNEL=2.6.9-42.0.2.EL_lustre.1.4.7.1

for path in /sys/block/sd*; do
# clear the partition table of /dev/sdy
        ls -l $path/device | grep host26>/dev/null
        if [ $? -eq 0 ]; then
                hd0=`basename $path`
            dd if=/dev/zero of=/dev/${hd0} bs=512 count=20
      fi

# clear the partition table of /dev/sdac
        ls -l $path/device | grep host30>/dev/null
        if [ $? -eq 0 ]; then
                hd1=`basename $path`
            dev_num=`cat $path/dev | sed -e 's/:/ /g'`
            mknod /dev/${hd1} b ${dev_num}
```

```
            dd if=/dev/zero of=/dev/${hd1} bs=512 count=20
      fi
done

# add entries in to grub device map
cp /boot/grub/device.map /boot/grub/device.map.org
echo "(hd1)     /dev/${hd0}" >> /boot/grub/device.map
echo "(hd2)     /dev/${hd1}" >> /boot/grub/device.map

if [ -f /boot/grub/grub.conf ]; then
      cp /boot/grub/grub.conf /boot/grub/grub.conf.org
      cat /boot/grub/grub.conf.org  | sed -e 's/root=.[^ ]* /root=\/dev\/md1 /g'
-e 's/root (hd/#root (hd/g' > /boot/grub/grub.conf
fi

# add entries into /etc/fstab
if [ -f /etc/fstab ]; then
      cp /etc/fstab /etc/fstab.org
      cat /etc/fstab.org | sed -e 's/LABEL/#LABEL/g' > /etc/fstab
cat << EOF >> /etc/fstab
${BOOTDEV}              /boot                   ext3    defaults        1 2
${ROOTDEV}              /                       ext3    defaults        1 1
${SWAPDEV}              swap                    swap    defaults        0 0
EOF
fi

# copy the partition table of /dev/sda to /dev/sdy and /dev/sdac
sfdisk -d /dev/sda | sfdisk /dev/${hd0}
sfdisk -d /dev/sda | sfdisk /dev/${hd1}

# create mirrors with only partitions of /dev/sdy
echo "yes" | mdadm -C ${BOOTDEV} -l 1 -n 2 missing /dev/${hd0}1
echo "yes" | mdadm -C ${ROOTDEV} -l 1 -n 2 missing /dev/${hd0}2
echo "yes" | mdadm -C ${SWAPDEV} -l 1 -n 2 missing /dev/${hd0}3

# set the id of the partitions to Linux auto-detect raid
for i in 1 2 3; do
      sfdisk -c /dev/${hd0} $i fd
      sfdisk -c /dev/${hd1} $i fd
done

# make the filesystems on the mirrors
mkfs.ext3 ${BOOTDEV}
mkfs.ext3 ${ROOTDEV}
mkswap ${SWAPDEV}

if [ ! -d ${MOUNTPT} ]; then
        mkdir ${MOUNTPT}
fi
mount ${ROOTDEV} ${MOUNTPT}
```

```
mkdir ${MOUNTPT}/boot
mount ${BOOTDEV} ${MOUNTPT}/boot

# copy the files from /dev/sda to the mirrors
cd /; find . -xdev | cpio -pmd ${MOUNTPT}
cd /boot; find . -xdev | cpio -pmd ${MOUNTPT}/boot

# add the partitions of /dev/sdac to the mirrors
mdadm ${BOOTDEV} -a /dev/${hd1}1
mdadm ${ROOTDEV} -a /dev/${hd1}2
mdadm ${SWAPDEV} -a /dev/${hd1}3

# create new kernel boot images
if [ -f ${MOUNTPT}/boot/initrd-${KERNEL}.img ]; then
      mkinitrd    -f    --with=raid1    --with=sata_mv    ${MOUNTPT}/boot/initrd-
${KERNEL}.img ${KERNEL}
fi

if [ -f ${MOUNTPT}/boot/initrd-${KERNEL}smp.img ]; then
      mkinitrd    -f    --with=raid1    --with=sata_mv    ${MOUNTPT}/boot/initrd-
${KERNEL}smp.img ${KERNEL}smp
fi

if [ -f ${MOUNTPT}/boot/initrd-${LUSTERKERNEL}smp.img ]; then
      mkinitrd    -f    --with=raid1    --with=sata_mv    ${MOUNTPT}/boot/initrd-
${LUSTERKERNEL}smp.img ${LUSTERKERNEL}smp
fi

# make changes to grub
grub <<EOF
device (hd1) /dev/${hd0}
root (hd1,0)
setup (hd1)
device (hd2) /dev/${hd1}
root (hd2,0)
setup (hd2)
EOF

# clear the partition table of /dev/sda
dd if=/dev/zero of=/dev/sda bs=512 count=20
```

3.2 PROBLEM 2: Clearing the partition tables of remaning disks

Sun Fire X4500 usually come installed with Solaris and ZFS. This can cause problems during raid groups creation for Lustre. The following script is run to

- clear the partition tables in individual disks
- create partition tables for the disks on which will host the journalling file systems for the data raid groups

```
# cleanup.sh
#!/bin/sh
```

```
DEVS="b c d e f g h i j k l m n o p q r s t u v w x z aa ab ad ae af ag ah ai
aj ak al am an ao ap aq ar as at au av"

for i in $DEVS;
do
      echo /dev/sd${i}
      dd if=/dev/zero of=/dev/sd${i} bs=512 count=20
      echo 'w' | fdisk /dev/sd${i}
done

JDEVS="as ak m e"
for i in $JDEVS;
do
      echo /dev/sd${i}
      sfdisk -uM /dev/sd${i} << EOF
,512,L
,512,L
,512,L
EOF
dd if=/dev/zero of=/dev/sd${i}1 bs=512 count=1
dd if=/dev/zero of=/dev/sd${i}2 bs=512 count=1
dd if=/dev/zero of=/dev/sd${i}3 bs=512 count=1
done
```

4 Re-installation procedure for RHEL4U4 on Sun Fire X4500 using N1SM

The same procedure can be followed. Initially problems were faced, as the old mirror existed on the boot disks as the partition table of /dev/sdac was not being cleared in the kickstart script. This has been fixed in the script shown in section 3.1.

5 Preparation of the Sun Fire X4500 for Lustre installation

5.1 Upgrading of Linux Kernel for Lustre and Infiniband

An 'update' called cfs_rh4u4as_x64 is created in N1SM with the script 'update_cfs_rh.sh'. This script accomplishes the following:

- Unpack the rpms from file /tmp/rh4u4as_x64.tar into /tmp/update/rh4u4asx64 and /tmp/updates/e2fsprogs
- Forcefully updates the e2fsprogs rpms
- Installs new kernel and Lustre modules
  - kernel-smp-2.6.9-42.0.2.EL_lustre.1.4.7.1.x86_64.rpm
  - kernel-smp-devel-2.6.9-42.0.2.EL_lustre.1.4.7.1.x86_64.rpm
  - lustre-1.4.7.1-2.6.9_42.0.2.EL_lustre.1.4.7.1smp.x86_64.rpm
  - lustre-modules-1.4.7.1-2.6.9_42.0.2.EL_lustre.1.4.7.1smp.x86_64.rpm
- Re-boots the server
- Installs the Voltaire IB rpm
  - ibhost-biz-3.5.5_21-1.k2.6.9_42.0.2.EL_lustre.1.4.7.1smp.x86_64.rpm

```
# update_cfs_rh.sh
#!/bin/sh
#
#update script to install addition rpms for Lustre and Voltaire on the RHEL4 U4
servers

# putting selinux --disabled in your ks.cfg file would work too
echo "SELINUX=disabled" > /etc/selinux/config
chmod 644 /etc/selinux/config
chown root:root /etc/selinux/config
setenforce 0

mkdir /tmp/update
cd /tmp/update
echo "untar source"
tar -xvf /tmp/rh4u4as_x64.tar

cd /tmp/update/rh4u4as_x64

for i in `ls /tmp/update/e2fsprogs/`
do
rpm -U /tmp/update/e2fsprogs/$i --force >> /tmp/update.log
done

#rm -rf /tmp/update


echo "#!/bin/sh
cd /tmp/update/rh4u4as_x64
rpm -i ibhost-biz-3.5.5_21-1.k2.6.9_42.0.2.EL_lustre.1.4.7.1smp.x86_64.rpm
rm -rf /etc/rc.d/rc2.d/S99rh4u4as_x64
```

```
rm -rf /etc/rc.d/rc3.d/S99rh4u4as_x64
rm -rf /etc/rc.d/rc4.d/S99rh4u4as_x64
rm -rf /etc/rc.d/rc5.d/S99rh4u4as_x64
rm -rf /etc/rc.d/rc6.d/S99rh4u4as_x64
/sbin/shutdown -r now
" > /tmp/update/rh4u4as_x64/rh4u4as_x64

chmod 755 /tmp/update/rh4u4as_x64/rh4u4as_x64
ln -s /tmp/update/rh4u4as_x64/rh4u4as_x64 /etc/rc.d/rc2.d/S99rh4u4as_x64
ln -s /tmp/update/rh4u4as_x64/rh4u4as_x64 /etc/rc.d/rc3.d/S99rh4u4as_x64
ln -s /tmp/update/rh4u4as_x64/rh4u4as_x64 /etc/rc.d/rc4.d/S99rh4u4as_x64
ln -s /tmp/update/rh4u4as_x64/rh4u4as_x64 /etc/rc.d/rc5.d/S99rh4u4as_x64
ln -s /tmp/update/rh4u4as_x64/rh4u4as_x64 /etc/rc.d/rc6.d/S99rh4u4as_x64

cd /tmp/update/rh4u4as_x64

rpm -i kernel-smp-2.6.9-42.0.2.EL_lustre.1.4.7.1.x86_64.rpm
rpm -i kernel-smp-devel-2.6.9-42.0.2.EL_lustre.1.4.7.1.x86_64.rpm
rpm -i lustre-1.4.7.1-2.6.9_42.0.2.EL_lustre.1.4.7.1smp.x86_64.rpm
rpm -i lustre-modules-1.4.7.1-2.6.9_42.0.2.EL_lustre.1.4.7.1smp.x86_64.rpm

shutdown -r now

exit
```

5.2 thumper_lvm update

An 'update' called 'thumper_lvm' is created in N1SM with the script 'update_cfs_rh.sh'. This script accomplishes the following:

- Untars the scripts from /tmp/scripts.tar.gz to /opt/lvm_setup
- Assigns Solaris like disk names to the Linux disk names through symbolic links
- Create 6 Linux Raid groups (mk-raid_oss.sh)
- Create Journal raid groups (mk-journal_dev.sh)
- Create /etc/mdadm.conf from a template

```
#!/bin/sh

#
# uncompress files
#
gunzip /tmp/scripts.tar.gz

mkdir /opt/lvm_setup
cd /opt/lvm_setup
tar -xvf /tmp/scripts.tar
```

```
# device maps, these run first then udevstart (for solaris like device names)
cp /opt/lvm_setup/scripts/99-dsk.rules /etc/udev/rules.d/99-dsk.rules
cp /opt/lvm_setup/scripts/minus-1.sh /etc/udev/scripts/minus-1.sh
/sbin/udevstart

# Create raid groups
/opt/lvm_setup/scripts/mk-raid_oss.sh

# Create journal file systems
/opt/lvm_setup/scripts/mk-journal_dev.sh

# Create /etc/mdadm.conf from a template, stop and start the raid groups
cp /opt/lvm_setup/scripts/mdadm.conf /etc
mdadm -S /dev/md1{1,2,3,5,6,7}
mdadm -S /dev/md2{1,2,3,5,6,7}
mdadm -A --scan

# stops Open IB module to make sure it does not cause interference with
Voltaire IB module
config --level 0123456 openibd off

mdadm --stop --scan

sleep 10
shutdown -r now
```

5.2.1 Assignment of Solaris like device names to the Linux disk device names

This is accomplished as shown in the script above. After udevstart is run, Solaris like device names will be available under /dev/dsk. The figure below shows the relationship between Solaris device names and Linux device names

SYSTEM FRONT											SYSTEM REAR
	c5	c5t0	sdy	c5t1	sdz	c5t2	sdaa	c5t3	sdab	CTLR-3	
		c5t4	sdac	c5t5	sdad	c5t6	sdae	c5t7	sdaf		
	c4	c4t0	sdq	c4t1	sdr	c4t2	sds	c4t3	sdt	CTLR-2	
		c4t4	sdu	c4t5	sdv	c4t6	sdw	c4t7	sdx		
	c7	c7t0	sdao	c7t1	sdap	c7t2	sdaq	c7t3	sdar	CTLR-5	PS-1
		c7t4	sdas	c7t5	sdat	c7t6	sdau	c7t7	sdav		
	c6	c6t0	sdag	c6t1	sdah	c6t2	sdai	c6t3	sdaj	CTLR-4	
		c6t4	sdak	c6t5	sdal	c6t5	sdam	c6t7	sdan		
	c1	c1t0	sdi	c1t1	sdj	c1t2	sdk	c1t3	skl	CTLR-1	PS-0
		c1t4	sdm	c1t5	sdn	c1t6	sdo	c1t7	sdp		
	c0	c0t0	sda	c0t1	sdb	c0t2	sdc	c0t3	sdd	CTLR-0	
		c0t4	sde	c0t5	sdf	c0t6	sdg	c0t7	sdh		

5.2.2 Creation of raid groups

This is accomplished by the script mk-raid_oss.sh. 6 data raid groups are created and 6 corresponding journalling raid groups are created. The data raid groups consist of 4 data disks, 1 parity disk and 1 spare disk each. The journalling

raid groups are mirrors.

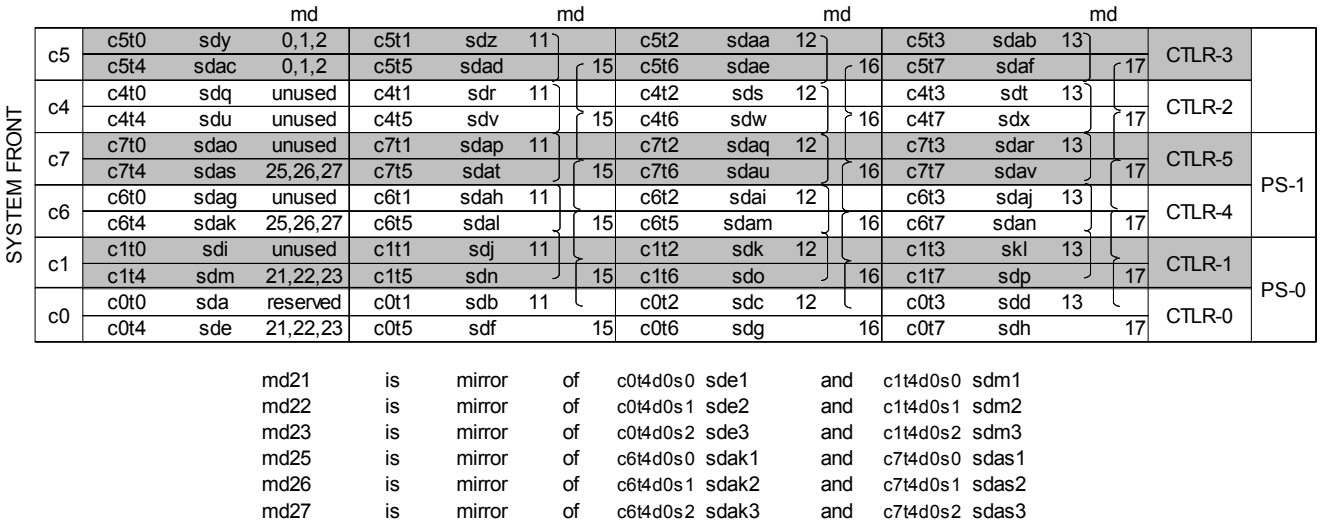
```
# mk-raid_oss.sh
#!/bin/sh

for i in 0 1 6 7; do
sfdisk -uM /dev/dsk/c${i}t4d0 << EOF
,512,L
,512,L
,512,L
EOF
done

# creation of the journalling raid groups (mirrors)
echo "yes" | mdadm -C /dev/md21 -l 1 -n 2 /dev/dsk/c{0,1}t4d0s0
echo "yes" | mdadm -C /dev/md22 -l 1 -n 2 /dev/dsk/c{0,1}t4d0s1
echo "yes" | mdadm -C /dev/md23 -l 1 -n 2 /dev/dsk/c{0,1}t4d0s2
echo "yes" | mdadm -C /dev/md25 -l 1 -n 2 /dev/dsk/c{6,7}t4d0s0
echo "yes" | mdadm -C /dev/md26 -l 1 -n 2 /dev/dsk/c{6,7}t4d0s1
echo "yes" | mdadm -C /dev/md27 -l 1 -n 2 /dev/dsk/c{6,7}t4d0s2

# creation of the data raid groups (raid 5 of 4 data disks, 1 parity disk and 1
spare disk
echo "yes" | mdadm -C /dev/md11 -l 5 -n 5 -x 1 /dev/dsk/c{0,1,4,5,6,7}t1d0
echo "yes" | mdadm -C /dev/md12 -l 5 -n 5 -x 1 /dev/dsk/c{0,1,4,5,6,7}t2d0
echo "yes" | mdadm -C /dev/md13 -l 5 -n 5 -x 1 /dev/dsk/c{0,1,4,5,6,7}t3d0
echo "yes" | mdadm -C /dev/md15 -l 5 -n 5 -x 1 /dev/dsk/c{0,1,4,5,6,7}t5d0
echo "yes" | mdadm -C /dev/md16 -l 5 -n 5 -x 1 /dev/dsk/c{0,1,4,5,6,7}t6d0
echo "yes" | mdadm -C /dev/md17 -l 5 -n 5 -x 1 /dev/dsk/c{0,1,4,5,6,7}t7d0
```

The raid groups are shown in the figure below:



md21	is	mirror	of	c0t4d0s0	sde1	and	c1t4d0s0 sdm1
md22	is	mirror	of	c0t4d0s1	sde2	and	c1t4d0s1 sdm2
md23	is	mirror	of	c0t4d0s2	sde3	and	c1t4d0s2 sdm3
md25	is	mirror	of	c6t4d0s0	sdak1	and	c7t4d0s0 sdas1
md26	is	mirror	of	c6t4d0s1	sdak2	and	c7t4d0s1 sdas2
md27	is	mirror	of	c6t4d0s2	sdak3	and	c7t4d0s2 sdas3

The following points can be noted

- md11 consists of all disks on target 1 (t1): c5t1d0, c4t1d0, c7t1d0, c6t1d0, c1t1d0 and c0t1d0

- md12 consists of all disks on target 2 (t2): c5t2d0, c4t2d0, c7t2d0, c6t2d0, c1t2d0 and c0t2d0
- md13 consists of all disks on target 3 (t3): c5t3d0, c4t3d0, c7t3d0, c6t3d0, c1t3d0 and c0t3d0
- md15 consists of all disks on target 5 (t5): c5t5d0, c4t5d0, c7t5d0, c6t5d0, c1t5d0 and c0t5d0
- md16 consists of all disks on target 6 (t6): c5t6d0, c4t6d0, c7t6d0, c6t6d0, c1t6d0 and c0t6d0
- md17 consists of all disks on target 7 (t7): c5t7d0, c4t7d0, c7t7d0, c6t7d0, c1t7d0 and c0t7d0
- md21 is the journalling filesystem for md11
- md22 is the journalling filesystem for md12
- md23 is the journalling filesystem for md13
- md25 is the journalling filesystem for md15
- md26 is the journalling filesystem for md16
- md27 is the journalling filesystem for md17
- Disks c4t0d0, c4t4d0, c7t0d0, c6t0d0 and c1t0d0 are unused in the current configuration
- Disk c0t0d0 (sda) has not been used, as the OS will be installed here during OS re-installations
- md0, md1 and md2 were created acroos /dev/sdy and /dev/sdac in the kickstart script sda2sdy.sh and respectively contain /boot, / and swap partitions

5.2.3 Creation of the journalling file systems

This is accomplished in the script mk-journal_dev.sh

```
# mk-journal_dev.sh
#!/bin/sh

JOURNAL_DEVS="md21 md22 md23 md25 md26 md27"

mke2fs -V 2>&1 | grep cfs > /dev/null
if [ $? -ne 0 ]; then
      echo "Please migrate to e2fsprogs-1.39.cfs1-1 before $0"
      exit 1
fi

for dev in ${JOURNAL_DEVS}; do
      mke2fs -O journal_dev -b 4096 /dev/${dev}
done
```

5.2.4 Creation of /etc/mdadm.conf from a template

The file /etc/mdadm.conf consists of the raid groups in a system and is checked during the OS bootup phase. The OS will start the raid groups as given in the file during the bootup phase. The contents of this file are:

```
#
# mdadm.conf
#

DEVICE /dev/dsk/c*t*d*

ARRAY /dev/md11 level=raid5 num-devices=5 devices=/dev/dsk/c[014567]t1d0
ARRAY /dev/md12 level=raid5 num-devices=5 devices=/dev/dsk/c[014567]t2d0
ARRAY /dev/md13 level=raid5 num-devices=5 devices=/dev/dsk/c[014567]t3d0
```

```
ARRAY /dev/md15 level=raid5 num-devices=5 devices=/dev/dsk/c[014567]t5d0
ARRAY /dev/md16 level=raid5 num-devices=5 devices=/dev/dsk/c[014567]t6d0
ARRAY /dev/md17 level=raid5 num-devices=5 devices=/dev/dsk/c[014567]t7d0

ARRAY /dev/md21 level=raid1 num-devices=2 devices=/dev/dsk/c[01]t4d0s0
ARRAY /dev/md22 level=raid1 num-devices=2 devices=/dev/dsk/c[01]t4d0s1
ARRAY /dev/md23 level=raid1 num-devices=2 devices=/dev/dsk/c[01]t4d0s2
ARRAY /dev/md25 level=raid1 num-devices=2 devices=/dev/dsk/c[67]t4d0s0
ARRAY /dev/md26 level=raid1 num-devices=2 devices=/dev/dsk/c[67]t4d0s1
ARRAY /dev/md27 level=raid1 num-devices=2 devices=/dev/dsk/c[67]t4d0s2
```

The following commands can be run to stop the raid groups:

```
# mdadm -S /dev/md1{1,2,3,5,6,7}
# mdadm -S /dev/md2{1,2,3,5,6,7}
```

The following command can be run to start the raid groups again:

```
# mdadm -A --scan
```

5.3 Synchronizing of the RAID groups

The RAID groups, once created are automatically synchronized. The data raid groups (md11, md12, md13, md15, md16 and md17) take up to 4-6 hours to resycnchronize. It is better to install Lustre only after this is done. However, during this period it is safe to reboot the server provided the raid groups are stopped using the following command:

```
# mdadm --stop --scan
```

5.4 Checking the status of the raid groups

The status of all raid groups in the system can be obtained by listing the contents of the file /proc/mdstat

```
# cat /proc/mdstat
```

The details of a particular raid group (let us say /dev/md11) can be obtained by running the following command:

```
# mdadm --query --detail --verbose /dev/md11
```

5.5 Stopping Linux raid groups before shutting down servers

It is prudent to stop the Linux raid groups using the following command before shutting down the servers

```
# mdadm --stop --scan
```