



Optimising Linux for Film Review and Colour Grading Stations.

Gavin Stewart

July 2004 (*Revision* : 1.4) © Xenon3D.

Abstract

This document describes a method to optimise a RedHat 9 based Linux distribution for use in real-time playback of 2K sized frames from disk.

Contents

1	Introduction	3
2	Hardware and Software Requirements	3
3	Hardware configuration	3
4	Operating System configuration	3
4.1	Update module-init-tools	3
4.2	Update mkinitrd	4
4.3	Install xfsprogs	5
4.4	Install linux-2.6.6	5
4.5	Configure software striping - RAID0	6
5	Programming tips for HighSpeed IO	8

1 Introduction

The system described in this document is derived from R&D work done by Xenon3D® on hardware provided by Xenon Systems®. The goal being to strike a balance between software compatibility and the quickest possible disk IO using easily available X86 based hardware components.

2 Hardware and Software Requirements

The hardware used here is considered current at the time of writing. Newer Intel® “Nocona” based systems have just become available. Initial tests show increased performance, however more stability and compatibility is required. As such the prescribed system is based on 533MHz FSB Xeons.

- Intel i7505 based mainboard
- Dual Xeon 2.4G CPU
- 8 x 15K RPM Ultra 320 SCSI drives
- 1 IDE system drive.
- Adaptec 39320 SCSI controller
- 1 Gig RAM

3 Hardware configuration

- Split drives across both SCSI channels. (4 each).
- Hostraid should be turned off.

4 Operating System configuration

- Install RedHat 9 onto IDE system drive.
- Update module-init-tools (see instructions below.)
- Update mkinitrd (see instructions below.)
- Install xfsprogs (see instructions below.)
- Install linux-2.6.6 kernel (see instructions below.)
- Configure software striping - RAID0 (see instructions below.)

4.1 Update module-init-tools

The 2.6.x kernels require a different set of module tools than those that come with RedHat 9. Locate the package `module-init-tools-0.9.15-pre4.tar.bz2` from:

`ftp://ftp.kernel.org/pub/linux/utils/kernel/module-init-tools/`

- Untar file.
- Read the provided README.
- The recommended method is to use 1a) as shown in the README.:

```
# ./configure --prefix=/
# make moveold #(if installing for the first time).
# make
# make install
```

- Ensure that a new `modprobe.conf` is generated from the old `modules.conf` by following step 3) of the README:

```
# ./generate-modprobe.conf /etc/modprobe.conf
```

- Add the line:
`alias scsi_hostadapter aic79xx`
to `/etc/modprobe.conf` if it does not appear already.
- Modify RedHat's `/etc/rc.d/rc.sysinit` to always use kernel modules ¹. Edit `/etc/rc.d/rc.sysinit` and comment out the `if` on line 360, and the `fi` on line 361, while leaving the line `USEMODULES=y` as follows:

```
#if ! LC_ALL=C grep -iq nomodules /proc/cmdline 2>/dev/null && [ -f /proc/ksyms ]; then
    USEMODULES=y
#fi
```

This setup will also allow backward compatibility when using a 2.4.x kernel.

4.2 Update mkinitrd

2.6.x kernels also require an updated `mkinitrd` tool to create the initial boot ramdisk used to load boot-time drivers. Locate the package `mkinitrd-3.5.22-1.src.rpm` from:

<http://download.fedora.redhat.com/pub/fedora/linux/core/2/i386/os/SRPMS/>

- Build a binary RPM from the source RPM.

```
# rpmbuild --rebuild mkinitrd-3.5.22-1.src.rpm
```

Should you have problems using `rpmbuild` (say if you have already booted a 2.6.x kernel). The error may appear as:

```
# rpmbuild --rebuild mkinitrd-3.5.22-1.src.rpm
Installing mkinitrd-3.5.22-1.src.rpm
rpmdb: unable to join the environment
error: db4 error(11) from dbenv->open: Resource temporarily unavailable
error: cannot open Packages index using db3 - Resource temporarily unavailable (11)
error: cannot open Packages database in /var/lib/rpm
```

You may will need to do the following first:

```
# export LD_ASSUME_KERNEL=2.4.1
```

¹The `sysinit` script fails to turn on module use as the file `/proc/ksyms` does not exist in 2.6.x kernels. This will result in modules not being auto-loaded as required. Manual insertion of modules will still function normally.

- Install the compiled binary RPM.
(note we are using “-nodeps” as the package has a dependency on lvm2, which we aren’t using in this example.)

```
# rpm -Uhv --nodeps /usr/src/redhat/RPMS/i386/mkinitrd-3.5.22-1.i386.rpm
```

4.3 Install xfsprogs

Although the Linux kernel contains support for SGI’s XFS filesystem, the RedHat 9 distribution does not.

Locate the package `xfsprogs-2.6.13.src.tar.gz` from:

`ftp://oss.sgi.com/projects/xfs/cmd_tars/`

- Untar file.
- Run configure.

```
# ./configure --prefix=/
```

Should configure exit with the following error, you will need to install `e2fsprogs-devel-1.32-6.i386.rpm` first:

```
FATAL ERROR: could not find a valid UUID header.
Install the Universally Unique Identifiers development package.
```

- Make and install.

```
# make
# make install
```

4.4 Install linux-2.6.6

Locate the package `linux-2.6.6.tar.bz2` from:

`ftp://ftp.kernel.org/pub/linux/kernel/v2.6/`

Locate the kernel configuration file `config-2.6.6-Xenon3D-2K-0.3` from:

`http://www.xenon3d.com.au/RND/`

- Untar file in `/usr/src`
- Copy `config-2.6.6-Xenon3D-2K-0.3` to `/usr/src/linux-2.6.6/.config`
- From `/usr/src/linux-2.6.6/` run `make oldconfig`
OR
- `make menuconfig`
(If customisations are required.)
- Build and install kernel.

```
# make bzImage
...
# make modules
...
# make modules_install
...
# cp arch/i386/boot/bzImage /boot/vmlinuz-2.6.6
# mkinitrd /boot/initrd-2.6.6.img 2.6.6
```

- Ensure the kernel and initial ramdisk image is properly configured in `/etc/lilo` or `/etc/grub.conf` as required. Remember to execute `lilo` after editing. Nothing further is required if using `grub`.
- Reboot system and select new kernel.

4.5 Configure software striping - RAID0

- Check that all 8 SCSI drives have been detected:

```
# cat /proc/scsi/scsi
Attached devices:
Host: scsi0 Channel: 00 Id: 00 Lun: 00
  Vendor: MAXTOR   Model: ATLAS15K_73SCA   Rev: DT60
  Type:   Direct-Access           ANSI SCSI revision: 03
Host: scsi0 Channel: 00 Id: 01 Lun: 00
  Vendor: MAXTOR   Model: ATLAS15K_73SCA   Rev: DT60
  Type:   Direct-Access           ANSI SCSI revision: 03
Host: scsi0 Channel: 00 Id: 02 Lun: 00
  Vendor: MAXTOR   Model: ATLAS15K_73SCA   Rev: DT60
  Type:   Direct-Access           ANSI SCSI revision: 03
Host: scsi0 Channel: 00 Id: 03 Lun: 00
  Vendor: MAXTOR   Model: ATLAS15K_73SCA   Rev: DT60
  Type:   Direct-Access           ANSI SCSI revision: 03
Host: scsi0 Channel: 00 Id: 06 Lun: 00
  Vendor: SUPER    Model: GEM318           Rev: 0
  Type:   Processor              ANSI SCSI revision: 02
Host: scsi1 Channel: 00 Id: 00 Lun: 00
  Vendor: MAXTOR   Model: ATLAS15K_73SCA   Rev: DT60
  Type:   Direct-Access           ANSI SCSI revision: 03
Host: scsi1 Channel: 00 Id: 01 Lun: 00
  Vendor: MAXTOR   Model: ATLAS15K_73SCA   Rev: DT60
  Type:   Direct-Access           ANSI SCSI revision: 03
Host: scsi1 Channel: 00 Id: 02 Lun: 00
  Vendor: MAXTOR   Model: ATLAS15K_73SCA   Rev: DT60
  Type:   Direct-Access           ANSI SCSI revision: 03
Host: scsi1 Channel: 00 Id: 03 Lun: 00
  Vendor: MAXTOR   Model: ATLAS15K_73SCA   Rev: DT60
  Type:   Direct-Access           ANSI SCSI revision: 03
Host: scsi1 Channel: 00 Id: 06 Lun: 00
  Vendor: SUPER    Model: GEM318           Rev: 0
  Type:   Processor              ANSI SCSI revision: 02
```

- Create partitions on SCSI drives as required. The whole drive may be used without partitions if desired.
- Configure `/etc/raidtab`. Example:

```

raiddev /dev/md0
    raid-level          0
    nr-raid-disks      8
persistent-superblock 1
chunk-size 128

    device              /dev/sda
    raid-disk           0
    device              /dev/sdb
    raid-disk           1
    device              /dev/sdc
    raid-disk           2
    device              /dev/sdd
    raid-disk           3
    device              /dev/sde
    raid-disk           4
    device              /dev/sdf
    raid-disk           5
    device              /dev/sdg
    raid-disk           6
    device              /dev/sdh
    raid-disk           7

```

- Make the new raid array, and add the XFS filesystem.

```

# mkraid -R /dev/md0
DESTROYING the contents of /dev/md0 in 5 seconds, Ctrl-C if unsure!
handling MD device /dev/md0
analyzing super-block
disk 0: /dev/sda, 71776260kB, raid superblock at 71776192kB
disk 1: /dev/sdb, 71776260kB, raid superblock at 71776192kB
disk 2: /dev/sdc, 71776260kB, raid superblock at 71776192kB
disk 3: /dev/sdd, 71776260kB, raid superblock at 71776192kB
disk 4: /dev/sde, 71776260kB, raid superblock at 71776192kB
disk 5: /dev/sdf, 71776260kB, raid superblock at 71776192kB
disk 6: /dev/sdg, 71776260kB, raid superblock at 71776192kB
disk 7: /dev/sdh, 71776260kB, raid superblock at 71776192kB

```

```

# mkfs.xfs -f /dev/md0
meta-data=/dev/md0          isize=256    agcount=32, agsize=4486016 blks
    =                       sectsz=512
data      =                 bsize=4096   blocks=143552256, imaxpct=25
    =                       sunit=32           swidth=256 blks, unwritten=1
naming    =version 2       bsize=4096
log       =internal log    bsize=4096   blocks=32768, version=1
    =                       sectsz=512    sunit=0 blks
realtime  =none           extsz=1048576 blocks=0, rtextents=0

```

```
# mount -t xfs /dev/md0 /raid
```

5 Programming tips for HighSpeed IO

SGI's XFS filesystem under Linux's 2.6.6 kernel has an efficient way to transfer data from disk to RAM via DMA. This manner of transfer is requested from the system by using the `O_DIRECT` flag when opening a file. Note that there are some alignment requirements that need to be met. See `open(2)` and `posix_memalign(3)`.

Example code snippets:

```
...
#include <stdio.h>
#define __USE_GNU 1
#include <fcntl.h>
#include <errno.h>
...
#define FILESIZE 12746752
#define ALIGN 4096
char *frame;
...
    ret = posix_memalign(&frame, ALIGN, FILESIZE);
    if (ret) {
        fprintf(stderr,"ALIGN ERR:%s\n",strerror(ret));
        exit(1);
    }
...
    fd=open(file_name, O_RDONLY|O_DIRECT|O_ASYNC);
    if (fd < 0) {
        perror("read_frames - error opening file");
        exit (1);
    }

    if (read(fd, frame, FILESIZE) < 0) {
        perror("read_frames - error reading file");
        exit (1);
    }

    close(fd);
...

```