Software RAID in Linux Workstations

Mini-White Paper

Introduction

This document provides an overview of software RAID solutions for HP workstations running the Linux operating system. It assumes a basic understanding of computer hardware, filesystems, and the Linux operating system. It covers the various RAID levels, the differences between hardware RAID and software RAID, steps to configure software RAID on Linux workstations, and information regarding disk failure, recovery, and other information pertaining to running software RAID on a Linux system. Please note that while BIOS messages may refer to hardware RAID, hardware RAID is not supported by HP Linux workstations.

RAID Basics

RAID Overview

Standing for Redundant Array of Independent (or Inexpensive) Disks, RAID is a technology that allows a computer system to increase capacity, performance, and/or reliability of storage solutions using multiple physical hard disks. There are several kinds of RAID; each RAID configuration is assigned to its own RAID "level." There are four basic RAID levels: 0, 1, 5, and 10. Over time, a number of other RAID levels have been defined as well; RAID levels 1-5 were those defined by the original RAID paper. These levels have been implemented with dedicated RAID disk controllers, *hardware RAID*, and with kernel-level code controlling RAID operations, *software RAID*.

RAID was originally developed as a means to provide a degree of redundancy for disk data. It first became popular in servers with SCSI drives where data reliability was key. Today, RAID can be found across a number of different systems, including workstations, and a variety of drive types, including IDE, SATA, and SAS, as well as SCSI.

Software RAID in Linux

The Linux kernel, as of revision 2.4, offers integrated software RAID without the need for additional hardware disk controllers or kernel patches. All that is required is multiple hard disks and a small amount of setup. Unlike most hardware RAID solutions, software RAID can be used with IDE disks as well as SATA, SAS, and SCSI.

Software RAID does not come entirely without cost: in order to manage the disks, break up data as necessary, and manage parity data, the CPU must take on some extra loading. It has been found that heavily disk-intensive workloads result in roughly double the CPU overhead (for example, from 15% to 30%) when software RAID is in use. For most applications, this overhead is easily handled by excess headroom in the processors, but for some applications where disk performance and CPU performance are very well balanced and already near-bottleneck, this additional CPU overhead may become burdensome and hardware RAID may look attractive. Hardware RAID also offers advantages due to its large hardware cache, and the capability for better scheduling of operations in parallel; however, software RAID offers much more flexibility in terms of disk and disk controller setup. Additionally, hardware RAID requires that a failed RAID controller must be replaced with an identical model in order to avoid data loss, whereas software RAID imposes no such requirements.

Some software RAID schemes offer data protection through mirroring (copying the data to multiple disks in case one fails) or parity data (checksums that allow error detection and limited rebuilding of data in case of a failure), but it should be noted that all software RAID solutions on workstations will require the shutdown of the system so that the failed drive can be replaced before redundancy can be restored. Currently only external hardware RAID allows online, invisible failure recovery through a hot-swap, however, the replacement for failed drives in software RAID requires only a minimum amount of work. Hardware RAID is not supported by HP Linux workstations.

Performance and Bottlenecks

Disk I/O bandwidth is typically limited by the system's bus speeds, the disk controller, and the disks themselves. Each generation of disks tends to get faster and newer rotation speeds premiere from time to time. The balance of these hardware limiters, as affected by the software configuration, determines where the real bottleneck is in the system.

Several RAID levels offer improved performance relative to a standalone disk. If your disk throughput is throttled by a single disk controller, there is probably little you can do with RAID to improve the performance without adding another controller. On the other hand, if the raw disk performance is the bottleneck, a tuned software RAID solution can dramatically improve the throughput. In fact, the slower the disk is relative to the rest of the system, the better RAID performance will scale, since the slowest piece of the performance pipeline is being directly addressed by moving to RAID.

RAID Levels

RAID-Linear: Concatenating Disks

RAID-linear can only generously be considered to be in the RAID family. It is generally used as a method of making two smaller disks look like a larger one by "appending" one to the other. For instance, two 40GB disks configured with RAID-linear look like one 80GB disk. RAID-linear offers no practical performance or reliability gains over standard separate disks. Spare disks are not supported in RAID-linear; if one disk fails, the RAID array fails as well. *RAID-linear is not supported by HP Linux workstation.*

RAID-0: Striping

RAID-0, or striping, means that the data is spread out in small chunks over multiple disks so that a read or write operation can fully utilize all the disk and disk controller bandwidth available by having each disk and/or disk controller share the load of any particular operation. The drives in the array are treated as one large address space and successive data blocks are typically written to different drives in the array. RAID-0 arrays are comprised of a minimum of two disk drives. In order to optimize read and write speeds, partitions in RAID-0 arrays should be located on different physical hard disks. RAID-0 was not included in the original RAID proposal as it does not technically meet the redundancy requirement, nor does it offer any error protection. *Software RAID-0 is supported by HP Linux workstations.*

Figure 1. Efficiency of Software RAID-0

Read Performance	Write Performance	Space Efficiency	Reliability
Excellent	**Excellent**	**~100%**	**Decreased**
Read performance is excellent, scaling towards controller maximum speed as more disks are added. The use of multiple controllers will aid in scaling.	Write performance is excellent as well, typically scaling even better than reads.	Space efficiency is quite good; there is very little overhead.	RAID-0 offers no protection against disk crashes; in fact, it is somewhat worse than non-RAID configurations because a single disk crash is guaranteed to affect the data spread out over the array.

RAID-1: Mirroring

RAID-1, or mirroring, means that the data is copied or mirrored across multiple physical disks. The data from a single physical disk or partition, once put into a RAID-1 array, will be copied transparently to the other disks in the array. This provides increased reliability for data integrity; should one disk fail, the data still remains on the other(s).

Tip:
Although RAID-1, as well as other RAID levels such as RAID-5, does provide some data protection, this is only protection against common hardware problems such as failure of a single disk. RAID should not be viewed as a substitute for regular data back-ups.

Since it is possible to read from two (or more) disks at once, there is a possible performance increase using RAID-1. Typically performance is sacrificed for recovery of data. Because data is mirrored, only half of the physical space is utilized, and data must be replicated to multiple disks, marginally increasing write times. *Software RAID-1 is supported by HP Linux workstations.*

Figure 2. Efficiency of Software RAID-1

Read Performance	Write Performance	Space Efficiency	Reliability
Slight Improvement	**Somewhat Slower**	**~50%**	**Excellent**
Theoretically, RAID-1 configuration could deliver higher read performance, but there have not been significant gains seen using software RAID-1.	Writes will be fractionally slower than non-RAID configurations since data must be replicated to several disks.	Only 50% space efficiency for two-disk RAID-1, since the data is fully replicated on each disk.	Reliability is very good; if one disk crashes, the other(s) still have a full copy of the data.

RAID-2: Error Checking and Correction

RAID-2 adds error checking and correcting (ECC) checksums to RAID-1. ECC stands for either "Error Correcting Code" or "Error Checking and Correcting." It is a code in which each data signal conforms to specific rules of construction so that departures from this construction in the received signal can generally be automatically detected and corrected. RAID-2 is rarely seen these days because most hard-disk controllers already do ECC, and this scheme offers few advantages over other RAID configurations. *Software RAID-2 is not supported by HP Linux workstations.*

RAID-3: Byte-Level Striping with Parity Disk

Like RAID-0, RAID-3 does striping, but at a very small granularity. It also adds a parity disk which helps in error detection and recovery. Parity in regards to hard disks refers to use of a parity bit, which counts whether the number of 1 bits in some preceding data was even or odd; if a single bit is changed in transmission, the parity bit will change, thus providing a redundancy check for data transmission. Parity bits are a very simple example of ECCs, or Error Correcting Codes. The details of ECC and parity are beyond the scope of this paper. The small granularity of RAID-3 leads to gains only when record sizes are very small and drive spindles are carefully synchronized. RAID-5 is generally preferred, and RAID-3 is very seldom used these days. *Software RAID-3 is not supported by HP Linux workstations.*

RAID-4: Block-Level Striping with Parity Disk

RAID-4 attempts to add error checking and recovery to RAID-3 by doing block-level striping, as in RAID-0, with the addition of a single parity disk. At least 3 disks are required for a RAID-4 array. Since all operations access the parity disk, that disk can become a bottleneck. RAID-4 can work in some operations, but RAID-5 is generally preferred where a compromise between speed and reliability is sought. *Software RAID-4 is not supported by HP Linux workstations.*

RAID-5: Block-Level Striping with Distributed Parity

In order to add error checking and recovery to RAID-0 and to eliminate the parity disk bottleneck of RAID-4, RAID-5 is implemented as a combination of data striping and parity, where data and parity blocks are successively written across the drives of the array. RAID-5 is used to ensure data integrity; should a single disk fail, it is possible to recover the data on the lost disk from the parity data on the others. RAID-5 requires a minimum of 3 disks, and the effective disk space availability of *n* disks in a RAID-5 array is *n-1* disks. *Software RAID-5 is supported by HP Linux workstations.*

Figure 3. Efficiency of Software RAID-5

Read Performance	Write Performance	Space Efficiency	Reliability
Poor to Moderate	**Moderate**	**66-75% (3 and 4-disk)**	**Good**
For both small and large blocks, RAID-5 performs poorly because parity data is interspersed with real data. The performance of RAID-5 can be dramatically improved by filesystem tuning.	While RAID-5 eliminates the parity-disk bottleneck of RAID-4, block-write performance is still slower than raw disks.	RAID-5's efficiency is about the same as RAID-4, where one disk is used to hold parity data, except the parity data is spread rather than concentrated on one drive.	Parity data is retained for all data. RAID-5 arrays can tolerate the failure of one disk.

Additional RAID Levels

More RAID levels can be created from the basic RAID levels by means of nesting. This is accomplished by creating a second RAID array of other RAID disks, instead of creating a RAID of physical disks. Nested RAID levels include RAID-0+1, a mirrored array of striped disks, RAID-10 (RAID-1+0), a striped array of mirrored disks, RAID-100 (RAID-10+0), which is a striped array of RAID-10 arrays, RAID-5+0, RAID-0+5, and other combinations. In nested RAID solutions, it is preferable to have a RAID-0 level on top to provide better performance as well as reducing the number of disks that need to be regenerated when a disk fails. For example, RAID-10 is usually preferred over RAID-0+1. Nested RAID levels are not supported by the Red Hat Enterprise Linux installer, and must be configured manually. *Software RAID-10 is the only additional RAID level supported by HP Linux workstations.*

Please note that while the Linux kernel allows for RAID-linear, -0, -1, -4, -5, and nested RAID levels, HP only provides support for RAID-0, 1, 5, and 10.

RAID Configuration Strategies

Configuration of RAID workstations must take several factors into account in order to deliver optimal performance, capacity, and redundancy. Note that the cost to implement RAID solutions increases as desired performance, capacity, and redundancy increases. Users should consider the following when designing an ideal RAID workstation:

Performance

Multiple factors must be taken into account, including both random and sequential read and write speeds, latency times, and bus types; CPU and RAM configurations play a heavy part in the performance of a software RAID solution as well. Such technical details are beyond the scope of this document.

Capacity versus Fault-Tolerance

With most implementations of RAID, capacity and fault-tolerance are mutually exclusive.

Cost

Obviously, as the number of physical disks required for an installation increases, the cost also increases.

RAID Performance Considerations

With the many different RAID levels available, as well as the many more combinations provided by nesting RAID levels, it can often be difficult to decide upon the most effective RAID configuration for your needs. Since high performance speeds and data protection are almost always mutually exclusive in RAID implementations, there will be a tradeoff to consider between these two factors. The following are examples of different RAID configurations that will be more efficient in different situations.

- If a filesystem had an external backup method, such as automated nightly tape backups, but needed to have quick read and write times, a RAID-0 configuration would make optimal use of the disk and disk controller bandwidth, but not provide any data protection. RAID-0 is most useful when performance is a key issue, but data protection is not as big a factor.
- If it is necessary to have backup drives available at all time so there is no loss of data, even in case of a disk failure, a RAID-1 setup using several physical hard disks would provide a good degree of redundancy so that no critical data is lost. This would be most valuable in a system that depends highly upon data reliability and less upon higher disk speeds.
- If a compromise between speed and redundancy is required, RAID-5 is fairly easy to implement, and provides a good balance between disk speeds and having parity data available to rebuild the RAID array in case of failure. RAID-5 is moderately efficient in both these areas, with neither being at a significant disadvantage, and is simpler to set up than a nested RAID configuration.

The optimal RAID configuration for a particular system will depend on the specific needs of the system; for example, how often will reads or writes have to be performed, versus how important is the ability to survive in case of multiple disk failures? You may want to experiment with different configurations to determine the RAID setup that works best for your system.

Configuring Red Hat Enterprise Linux with Software RAID

In order to configure software RAID on a workstation running Linux, you need to be using Linux revision 2.4 or later.

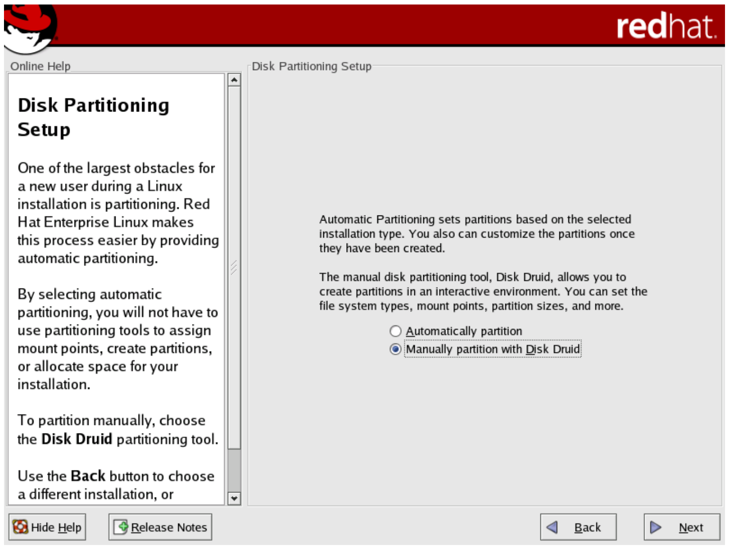
Tip:
It is possible to configure software RAID with late 2.2.x or 2.0.x Linux kernels if you have a matching RAID patch and version .90 of the `raidtools` installed as well. The patch and the `raidtools` can both be downloaded from http://people.redhat.com/mingo.

While it is possible to manually configure software RAID after a Linux system has been installed and configured, it is HP's recommendation that software RAID should be configured at installation time. The Red Hat Enterprise Linux installation utility includes a configuration tool to set up software RAID partitions. Please note that while the Linux kernel itself allows for RAID-linear, -0, -1, -4, -5, and nested RAID levels, the Anaconda installer only allows for RAID-0, -1, and -5.

The steps to configure software RAID using RHEL media are outlined below:

1. Power your HP workstation; boot to your RHEL installation media.
2. Continue through the installation until the "Disk Partitioning Setup" screen.
3. Select "Manually partition with Disk Druid."

Figure 4. Disk Partitioning Setup



Online Help

Disk Partitioning Setup

One of the largest obstacles for a new user during a Linux installation is partitioning. Red Hat Enterprise Linux makes this process easier by providing automatic partitioning.

By selecting automatic partitioning, you will not have to use partitioning tools to assign mount points, create partitions, or allocate space for your installation.

To partition manually, choose the **Disk Druid** partitioning tool.

Use the **Back** button to choose a different installation, or

Disk Partitioning Setup

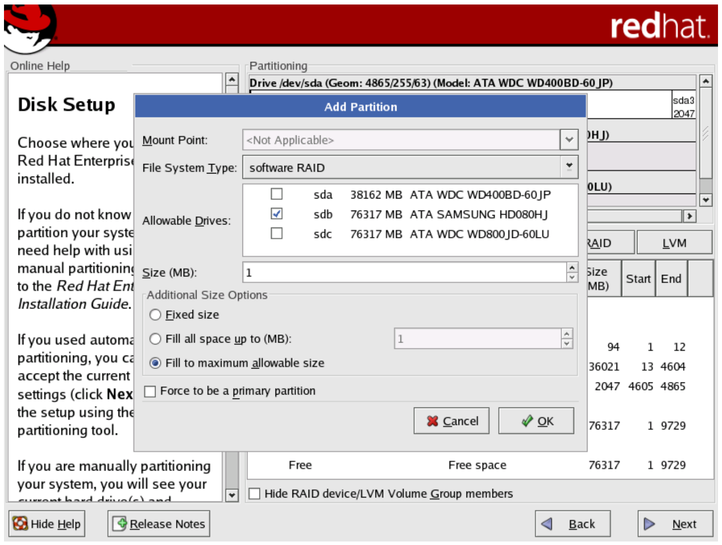
Automatic Partitioning sets partitions based on the selected installation type. You also can customize the partitions once they have been created.

The manual disk partitioning tool, Disk Druid, allows you to create partitions in an interactive environment. You can set the file system types, mount points, partition sizes, and more.

○ Automatically partition
● Manually partition with Disk Druid

4. Select "New" to create a new partition.
5. From the File System Type menu, choose "Software RAID."
6. Select one physical disk to create the partition on.
7. Choose the size of the partition.
8. Click OK.

Figure 5. Disk Setup



Online Help

Disk Setup

Choose where you Red Hat Enterprise installed.

If you do not know partition your syste need help with usi manual partitioning to the *Red Hat Ent Installation Guide.*

If you used automa partitioning, you ca accept the current settings (click **Nex** the setup using the partitioning tool.

If you are manually partitioning your system, you will see your

Partitioning

Drive /dev/sda (Geom: 4865/255/63) (Model: ATA WDC WD400BD-60 JP)

Add Partition

Mount Point: <Not Applicable>

File System Type: software RAID

Allowable Drives:

☐	sda	38162 MB	ATA WDC WD400BD-60JP
✓	sdb	76317 MB	ATA SAMSUNG HD080HJ
☐	sdc	76317 MB	ATA WDC WD800JD-60LU

Size (MB): 1

Additional Size Options
○ Fixed size
○ Fill all space up to (MB): 1
● Fill to maximum allowable size

☐ Force to be a primary partition

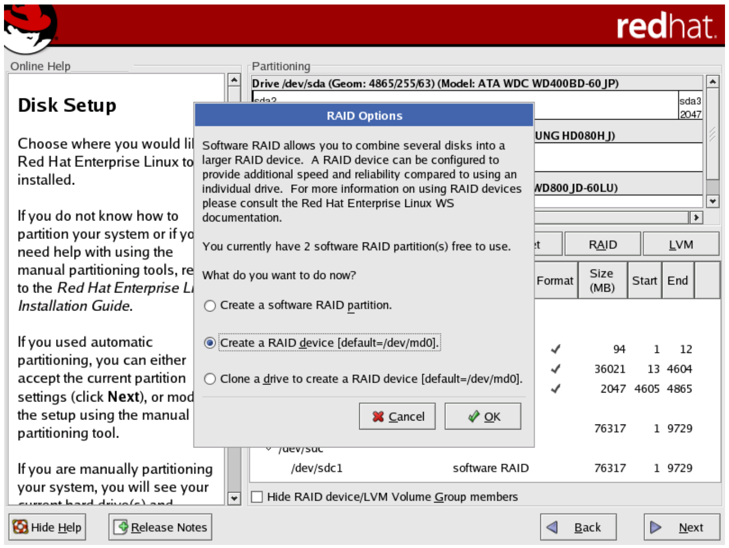
Size (MB)	Start	End
94	1	12
36021	13	4604
2047	4605	4865
76317	1	9729

Free Free space 76317 1 9729

☐ Hide RAID device/LVM Volume Group members

9. Repeat steps 4-8 until all necessary software RAID partitions are created.
10. From the main partitioning menu, choose "RAID."
11. From the "RAID Options" screen, choose "Create a RAID Device."

Figure 6. Creating a RAID Device



Disk Setup

Choose where you would li
Red Hat Enterprise Linux to
installed.

If you do not know how to
partition your system or if yo
need help with using the
manual partitioning tools, re
to the *Red Hat Enterprise Li*
Installation Guide.

If you used automatic
partitioning, you can either
accept the current partition
settings (click **Next**), or mod
the setup using the manual
partitioning tool.

If you are manually partitioning
your system, you will see your

RAID Options

Software RAID allows you to combine several disks into a larger RAID device. A RAID device can be configured to provide additional speed and reliability compared to using an individual drive. For more information on using RAID devices please consult the Red Hat Enterprise Linux WS documentation.

You currently have 2 software RAID partition(s) free to use.

What do you want to do now?

- ○ Create a software RAID partition.
- ● Create a RAID device [default=/dev/md0].
- ○ Clone a drive to create a RAID device [default=/dev/md0].

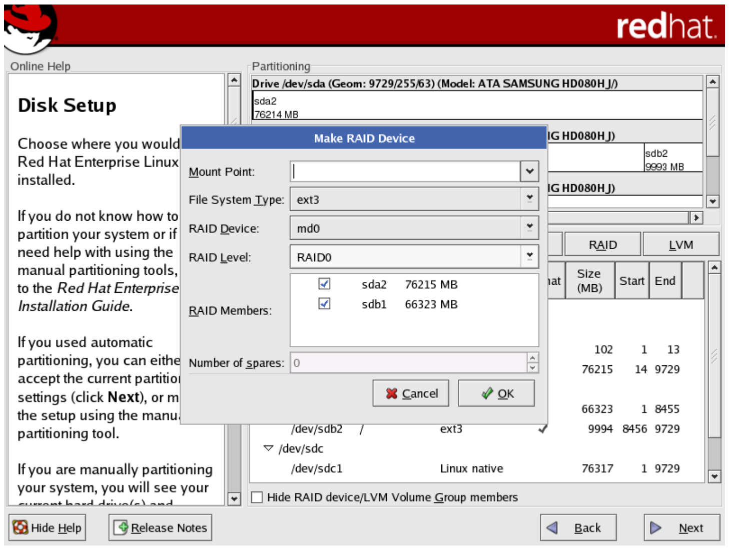
```
                               ✓     94     1    12
                               ✓  36021    13  4604
                               ✓   2047  4605  4865

                                  76317     1  9729
  /dev/sdc
    /dev/sdc1    software RAID    76317     1  9729
```

☐ Hide RAID device/LVM Volume Group members

12. Choose a mount point, file system type, device, and RAID level for this partition.

Figure 7. Creating a RAID Device



Disk Setup

Choose where you would
Red Hat Enterprise Linux
installed.

If you do not know how to
partition your system or if
need help with using the
manual partitioning tools,
to the *Red Hat Enterprise*
Installation Guide.

If you used automatic
partitioning, you can eithe
accept the current partitio
settings (click **Next**), or m
the setup using the manu
partitioning tool.

If you are manually partitioning
your system, you will see your

Make RAID Device

Mount Point:	
File System Type:	ext3
RAID Device:	md0
RAID Level:	RAID0
RAID Members:	☑ sda2 76215 MB ☑ sdb1 66323 MB
Number of spares:	0

```
                                   102     1    13
                                 76215    14  9729

                                 66323     1  8455
    /dev/sdb2   /   ext3       ✓  9994  8456  9729
  /dev/sdc
    /dev/sdc1       Linux native 76317     1  9729
```

☐ Hide RAID device/LVM Volume Group members

13. Click OK.
14. Continue with setup as normal.

The Red Hat Enterprise Linux Anaconda installer only supports RAID-0, RAID-1, and RAID-5. Other levels or nested RAID must be configured manually after installation. If you are creating a nested RAID array, you can either configure the "bottom" RAID level (for example, the RAID-1 array in the case of nested RAID-10) in Anaconda and the "top" RAID level manually later, or configure both parts of the nested RAID level manually after installation.

HP only provides support for software RAID-0, -1, -5, and -10.

Tip:
The GRUB and LILO boot loaders only support booting to RAID-1, or no RAID at all. If you wish to use a RAID level other than RAID-1, you must put the /boot partition on a non-RAID part of the filesystem.

Manual Setup of Software RAID Data Partitions

While HP recommends using the above method to set up software RAID on your Linux workstations, it is possible to manually set up a software RAID partition on multiple disks after installation.

1. Create partitions for the RAID array on two or more disks. You can use `Disk Druid`, `fdisk`, `sfdisk`, or any number of other partitioning tools to do this. It helps if the partitions are of the same size and on the same type of disk. The type of the partitions will be “`Linux raid auto`,” or `0xfd`. A reboot may be required after writing the new partition table.

Tip:
If you are setting up a RAID-1 array, it is best to use the same partition number on each disk for elements in the array. When one disk fails, if it is not the last disk in the array, the other disk(s) will be renamed on restart. For example, if `sdb` fails but `sdc` is still functional, when your system is restarted, the old `sdc` is renamed `sdb`. If you use a consistent partition number, your configuration files will remain valid (e.g. `sdb1` and `sdc1` being RAID partitions), no matter how the devices are renamed.

2. These partitions will correspond to Linux block file devices, e.g. the second partition on the third SCSI disk will be `/dev/sdc2`; the first partition on the second IDE disk will be `/dev/hdb1`.
3. Create the `/etc/raidtab` file. Sample configuration files can usually be found in `/usr/share/doc/raidtools-*`.

Tip:
You can view the file named `/proc/mdstat` by running the command `cat /proc/mdstat`.

This file should tell you that you have the right personality (RAID mode) registered, as well as what RAID devices are currently active.

4. Use `mkraid` to create the initial RAID configuration on the partitions you previously specified in `/etc/raidtab` by entering the following:

   ```
   mkraid /dev/md0
   ```

   or if you are using `mdadm`, enter:

   ```
   mdadm ñcreate ñverbose /dev/md0 --level=raidlevel --raid-
   devices=n /dev/sda1 /dev/sdb2
   ```

   where /dev/md0 is the name of the RAID device, *raidlevel* is the RAID level specified as a number (e.g. 0, 1), *n* is the number of disks in the RAID array, and /dev/sda1, /dev/sdb2, etc are the names of the drives in the RAID array.

5. Create a filesystem on your RAID device as follows:

   ```
   mkfs ñt ext3 /dev/md0
   ```

6. Mount the filesystem like a normal block file device:

```
        mkdir /raid
        mount /dev/md0 /raid
```

where `/raid` is wherever the filesystem you created in step 5 will reside.

7. Edit your `/etc/fstab` file. You will need to comment out the lines containing the partitions that are now in the RAID array, such as `/dev/sda1` and `/dev/sdb2`, by adding a # at the beginning of the line. Add a line for the RAID array filesystem:

```
         /dev/md0     /raid     ext3     defaults       0 1
```

Manual Configuration Examples

Most of the work in manual configuration of a software RAID array comes from the creation of the appropriate `/etc/raidtab` file. The following are examples of `/etc/raidtab` files and the corresponding `mdadm` commands needed to set up each of the basic RAID configurations supported by HP.

RAID-0

`/etc/raidtab` file:

```
raiddev /dev/md0
      raid-level              0
      nr-raid-disks           2
      nr-spare-disks          0
      persistent-superblock   1
      chunk-size              4
      device                  /dev/hda1
      raid-disk               0
      device                  /dev/hdb1
      raid-disk               1
```

Command:

```
mdadm -Cv /dev/md0 --level=0 --raid-devices=2 /dev/hda1 /dev/hdb1
```

RAID-1

`/etc/raidtab` file:

```
raiddev /dev/md0
      raid-level              1
      nr-raid-disks           2
      nr-spare-disks          0
      persistent-superblock   1
      chunk-size              4
      device                  /dev/sda1
      raid-disk               0
      device                  /dev/sdb1
      raid-disk               1
```

Command:

```
mdadm -Cv /dev/md0 --level=1 --raid-devices=2 /dev/sda1 /dev/sdb1
```

RAID-5

/etc/raidtab file:
```
raiddev /dev/md0
      raid-level              5
      nr-raid-disks           4
      nr-spare-disks          0
      persistent-superblock   1
      chunk-size              4
      device                  /dev/hda1
      raid-disk               0
      device                  /dev/hdb1
      raid-disk               1
      device                  /dev/hdc1
      raid-disk               2
      device                  /dev/hdd1
      raid-disk               3
```

Command:
```
mdadm -Cv /dev/md0 --level=5 --raid-devices=4 /dev/hda1 /dev/hdb1
/dev/hdc1 /dev/hdd1
```

RAID-10

/etc/raidtab file:
```
raiddev /dev/md0
      raid-level              0
      nr-raid-disks           2
      nr-spare-disks          0
      chunk-size              4
      device                  /dev/sda1
      raid-disk               0
      device                  /dev/sdb1
      raid-disk               1

raiddev /dev/md1
      raid-level              0
      nr-raid-disks           2
      nr-spare-disks          0
      chunk-size              4
      device                  /dev/sdc1
      raid-disk               0
      device                  /dev/sdd1
      raid-disk               1

raiddev /dev/md2
      raid-level              1
      nr-raid-disks           2
      nr-spare-disks          0
      chunk-size              4
      device                  /dev/md0
      raid-disk               0
      device                  /dev/md1
      raid-disk               1
```

Commands:
```
mdadm ñCv /dev/md0 --level=0 --raid-devices=2 /dev/sda1 /dev/sdb1
mdadm ñCv /dev/md1 --level=0 --raid-devices=2 /dev/sdc1 /dev/sdd1
mdadm ñCv /dev/md2 --level=1 --raid-devices=2 /dev/md0 /dev/md1
```

In a RAID-10 configuration, you will need to add three lines to the `/etc/fstab` file, one for each of the RAID arrays. There does not need to be a mount point specified for `/dev/md0` or `/dev/md1`. If no mount point is specified, you will see error messages during startup, but the RAID-10 array will still initialize and mount correctly.

Please note that these configuration files are only meant as examples, and your `/etc/raidtab` file will differ based on your specific hard drive configuration.

Disk Failure and Recovery

Spare Disks and Disk Failure

Spare disks are disks that do not take part in the RAID configuration until one of the active disks fails. At that point, the failed device is marked as "bad" and reconstruction of the RAID array begins immediately on the first available spare disk.

Tip:
When reconstruction occurs, if multiple bad blocks have built up on the active disks over time, the reconstruction process can sometimes trigger the failure of one of the "good" disks, leading to failure of the entire array. However, performing regular filesystem checks (`fsck`) of the entire RAID filesystem should almost completely eliminate this risk.

Spare disks are not required for a RAID configuration, but they are highly recommended. While most RAID levels can handle the failure of one physical disk, the failure of another disk will cause the entire array to fail, thus it is recommended to start rebuilding the array as quickly as possible after a disk failure. When a disk fails, the crashed disk is marked as "faulty." Faulty disks still look and behave as members of the RAID array; they are simply treated as inactive parts of the filesystem.

When a disk fails, information regarding the failure will appear in the standard log and `stat` files. Looking in `/proc/mdstat` will show information regarding the drives in the RAID array. RAID role numbers show the role that the disks play in the RAID configuration; for an array with *n* disks, disks with RAID role numbers greater than or equal to *n* are designated spare disks. A failed disk will be marked with an "`F`" and will be replaced with the device with the lowest role number greater than *n* that is not failed.

Removing and replacing a failed disk can be done as follows:

1. Remove the failed disk from the RAID array by running the command:
   ```
   raidhotremove /dev/md0 /dev/sdc2
   ```
   where `/dev/sdc2` is the name of the failed drive. If you wish to use `mdadm` instead of `raidtools`, the command is:
   ```
   mdadm /dev/md0 -r /dev/sdc2
   ```
   Please note that `raidhotremove` can not be used to pull a disk out of a running array, and should only be used for removing failed disks.

2. After recovery ends, a new disk should be designated as `/dev/sdc2,` or whichever disk was the one that failed. This new disk now needs to be added to the array.
   ```
   raidhotadd /dev/md1 /dev/sdc2
   ```
   Using `mdadm`, the command is:
   ```
   mdadm /dev/md1 -a /dev/sdc2
   ```

Tip:
You can use mdadm as a daemon to monitor a RAID array, by running:

```
mdadm --monitor --mail=root@localhost --delay=1800
/dev/md0
```

This will poll the array in intervals of 1800 seconds, and critical events and failures will be emailed to the system administrator. There are many other monitoring systems available for Linux software RAID as well.

Multiple Disk Failure

In the case of a temporary failure of multiple disks, such as a disk controller failure or cable coming loose that affects multiple disks, the RAID superblocks will afterwards be out of sync and the RAID array can no longer be initialized. Using `mdadm`, you can run:

```
        mdadm --assemble ñforce
```

to try and recreate the array. If that method doesn't work, you can run:

```
        mkraid --force
```

to rewrite the RAID superblocks. In order for this to work, you will need to have a completely up-to-date `/etc/raidtab` file, otherwise, if the ordering of the disks is different than expected, data on all disks could be lost.

Additional Configuration Information

The Persistent Superblock

Previously, the `raidtools`, which are included with most major Linux distributions, would read your `/etc/raidtab` file, and then initialize the filesystem. This required that the filesystem on which `/etc/raidtab` resided was mounted, which was unfortunate if you wanted to boot on a RAID.

The persistent superblock solves these problems. When an array is initialized with the `persistent-superblock` option in the `/etc/raidtab` file, a special superblock is written to the beginning of all disks participating in the array. This allows the kernel to read the configuration of RAID devices directly from the disks involved, instead of reading from the `/etc/raidtab` configuration file that might not be available at all times. You should still maintain a consistent `/etc/raidtab` file, since you may need this file for later reconstruction of the array.

The persistent superblock is mandatory if you want auto-detection of your RAID devices upon system boot.

Chunk Sizes

The chunk size is defined as the smallest amount of data that can be written to a device. You can never write completely in parallel to a set of disks. If you had two disks and wanted to write a byte, you would have to write four bits on each disk, with every second bit going to disk 0 and the others to disk 1. Hardware doesn't support that, so chunk size is used instead. A write of 16kB with a chunk size of 4kB will cause the first and the third 4kB chunks to be written to the first disk, and the second and fourth chunks to be written to the second disk, in the RAID-0 case with two disks. Thus, for large writes, you may see lower overhead by having fairly large chunks, whereas arrays that are primarily holding small files may benefit more from a smaller chunk size.

Chunk sizes must be specified for all RAID levels, including linear mode. However, the chunk size does not make any difference for RAID-Linear. The argument to the `chunk-size` option in the `/etc/raidtab` file specifies the chunk size in kB, so “4” means “4kB.”

For optimal performance, you should experiment with the value, as well as with the block-size of the filesystem you put in the array.

Swap Space in a RAID Configuration

There is generally no need to use RAID for swap performance. The kernel itself can stripe the swap space over multiple physical disks, provided each stripe is given the same priority in the `/etc/fstab` file. For example, the `/etc/fstab` file for striping the swap space over three drives might look like the following:

```
/dev/sda2       swap          swap    defaults,pri=1   0 0
/dev/sdb2       swap          swap    defaults,pri=1   0 0
/dev/sdc2       swap          swap    defaults,pri=1   0 0
```

This will allow the machine to swap in parallel over multiple physical devices. The Linux kernel itself has this capability; there is no need for RAID implementation of swap space.

Boot Partitions in a Mirrored RAID Configuration

The mirroring for Linux in software RAID configuration is the mirroring of partitions, not physical drives. For this reason, the Master Boot Record (MBR) is not mirrored in a RAID-1 configuration, and therefore, the mirrored drive or drives are not inherently bootable. Currently there is no way to mirror the boot partition of a hard drive using utilities such as Disk Druid, so the mirrored drives must be configured manually in order to make them bootable by Linux in case the first drive fails.

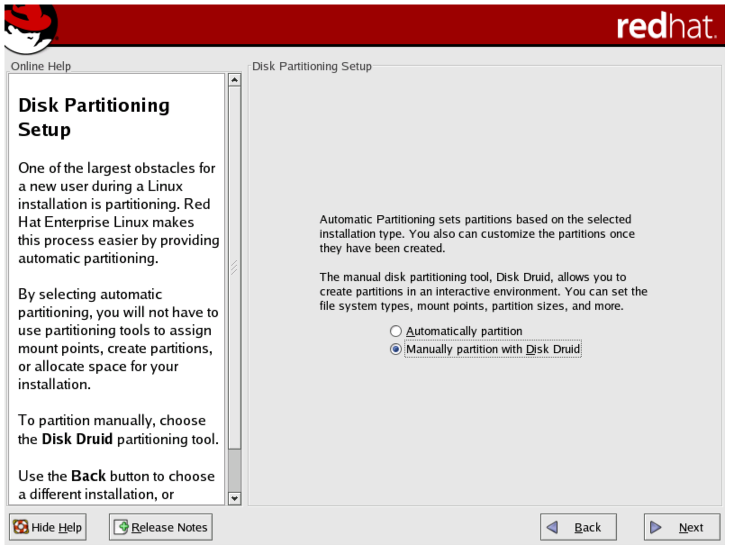
One possible configuration for a software RAID-1 array in Linux is to install the `/boot` partition on a separate physical hard disk from the rest of the filesystem. For example, `/dev/sda1` would be a hard disk containing the `/boot` partition, and `/dev/sdb1` and `/dev/sdc1` would be separate physical disks making up the `/` partition, configured in a RAID-1 array that is independent of the `/boot` partition. In this case, failure of one of the hard disks in the RAID array would not affect the disk with the `/boot` partition.

If placing the `/` and `/boot` partitions on separate physical drives as described above is not possible or desirable, more manual configuration is required to make sure that all hard drives in a RAID-1 array will be bootable in case of failure. In this case, you will need to create a bootable partition on each physical hard disk that will be part of the RAID-1 array. This is easiest when done during the original installation and configuration of the Linux workstation.

The steps to manually mirror your boot partition are outlined below:

1. Power your HP workstation; boot to your RHEL installation media.
2. Continue through the installation until the “Disk Partitioning Setup” screen.
3. Select “Manually partition with Disk Druid.”

Figure 8. Disk Partitioning Setup



Disk Partitioning Setup

One of the largest obstacles for a new user during a Linux installation is partitioning. Red Hat Enterprise Linux makes this process easier by providing automatic partitioning.

By selecting automatic partitioning, you will not have to use partitioning tools to assign mount points, create partitions, or allocate space for your installation.

To partition manually, choose the **Disk Druid** partitioning tool.

Use the **Back** button to choose a different installation, or

Automatic Partitioning sets partitions based on the selected installation type. You also can customize the partitions once they have been created.

The manual disk partitioning tool, Disk Druid, allows you to create partitions in an interactive environment. You can set the file system types, mount points, partition sizes, and more.

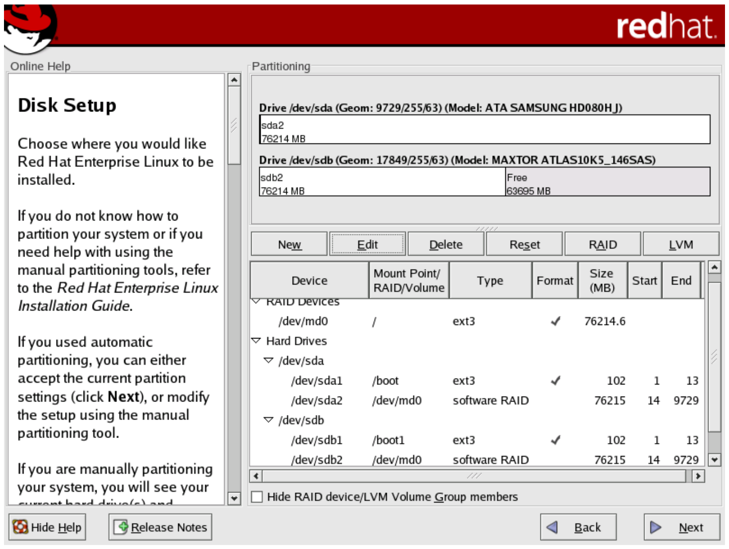
○ Automatically partition
[✓ Manually partition with Disk Druid]

4. On each disk that will be in the RAID-1 array, create a partition that will be bootable. One drive should have the actual `/boot` partition; each other drive should have a partition of the same size of format `ext3`.

Tip:
It is recommended that you use the same partition number for each of the bootable partitions. For example, if your primary `/boot` partition resides on `/dev/sda1`, you should create a partition called `/boot1`, located on `/dev/sdb1`, and so on for each hard disk.

5. Create a bootable partition on each disk that will be in the RAID-1 array.

Figure 9. Disk Partitioning Setup



Disk Setup

Choose where you would like Red Hat Enterprise Linux to be installed.

If you do not know how to partition your system or if you need help with using the manual partitioning tools, refer to the *Red Hat Enterprise Linux Installation Guide*.

If you used automatic partitioning, you can either accept the current partition settings (click **Next**), or modify the setup using the manual partitioning tool.

If you are manually partitioning your system, you will see your

Drive /dev/sda (Geom: 9729/255/63) (Model: ATA SAMSUNG HD080HJ)

sda2 76214 MB

Drive /dev/sdb (Geom: 17849/255/63) (Model: MAXTOR ATLAS10K5_146SAS)

sdb2 76214 MB | Free 63695 MB

Device	Mount Point/ RAID/Volume	Type	Format	Size (MB)	Start	End
RAID Devices						
/dev/md0	/	ext3	✓	76214.6		
Hard Drives						
/dev/sda						
/dev/sda1	/boot	ext3	✓	102	1	13
/dev/sda2	/dev/md0	software RAID		76215	14	9729
/dev/sdb						
/dev/sdb1	/boot1	ext3	✓	102	1	13
/dev/sdb2	/dev/md0	software RAID		76215	14	9729

☐ Hide RAID device/LVM Volume Group members

6. Create a RAID device by selecting the “RAID” option in the main partitioning menu, then choosing “Create a RAID Device” from the “RAID Options” screen, and configuring as desired.
7. Continue with setup as normal.

At this point, finish with the initial installation and configuration of your workstation. You should now have a partition that will be made bootable on each hard disk in the RAID array. You can verify your current disk setup by running the `df` command in a terminal window.

You can now mirror the original `/boot` partition to the partitions you created on the other drives.

8. For each other bootable partition, copy the contents of the `/boot` partition to the new partition as follows:

   ```
   cp ña /boot/* /boot1/
   ```

   If your other bootable partition is named `/boot1`.
9. Repeat step 8 for each bootable partition until each hard drive in the RAID array has been configured as such.
10. The final step is to edit the `/etc/fstab` file to allow the system to boot without the original `/boot` partition. Comment out the line with the `/boot` partition by inserting a # at the beginning of that line in the file. The system will now be able to boot to any of the other drives in the RAID array if the first disk should fail.

Tip:
While you do not need to have `/boot` mounted for the system to boot normally, you will need to mount it if you are running a kernel update. After the update is complete, repeat these steps to mirror the new `/boot` partition to the bootable partitions on the other drives. After this is done, the `/boot` partition can safely be removed from the `/etc/fstab` file again.

Software RAID and LVM

Software RAID can be used with the Linux LVM, or Logical Volume Manager, to provide a greater degree of flexibility with regard to the setup of the filesystem. The LVM provides a level of abstraction of the physical disks in a filesystem, making it easier to manage. It works by grouping physical disks into a volume group, which can then be partitioned into logical volumes. These logical volumes behave much like ordinary disk block devices, except that logical volumes can be dynamically grown, shrunk, and/or moved without rebooting the system or entering into maintenance/standalone mode. In general, it adds a layer of abstraction between filesystem mount points such as / or /usr, and hard disk devices such as /dev/hda1 or /dev/sdb2.

The benefit of using LVM is the flexibility of being able to add or remove physical hard drives or move data between existing drives without disrupting the filesystem or users. LVM cannot be used to dynamically resize RAID devices, nor can physical drives be simply added and removed from a RAID array as they could while using LVM without RAID. There are still benefits to using software RAID with LVM, which can be implemented using the following steps:

1. Set up RAID partitions using `Disk Druid`, `fdisk`, or whatever partition manager you prefer.
2. Create the RAID array, either during installation using the Anaconda installer, or manually after installation. Any supported RAID level can be used with LVM, since neither software RAID nor LVM knows about the other.
3. Use LVM to create a physical volume on the RAID device. For example, if the RAID array is /dev/md0, then use the following command to create a physical volume:
   ```
   pvcreate /dev/md0
   ```
4. Create a volume group from the physical volume you just set up. For example, create the volume group using the command:
   ```
   vgcreate lvm-raid /dev/md0
   ```
   In this case, a volume group called `lvm-raid` is created from the device `/dev/md0`.
5. Logical volumes can now be created using the following command:
   ```
   lvcreate ñl 57235 lvm-raid ñn lvm0
   ```

This creates a logical volume called lvm0 of size 57235 from the lvm-raid volume group. As many partitions as necessary can be created in this way. The lvdisplay command can be used to check the status of logical volumes; the vgdisplay command provides status about the volume group as a whole. The available space can be seen using the vgdisplay command.

Setup of the filesystem can now continue as normal.

439278-001, October 2006