Solid State Drives

A Caution! - There is a large amount of information on the topics of partitioning Large HDDs and SSDs and optimal operation - some of it contradictory. The following is a summary of information as I understand it - the lesson here is to do your homework as information about these topics is continually being updated.

It is important to be aware of the date any references were created, as earlier problems with SSDs in particular have been worked on and are less intrusive in later SSDs.

Large HDDs and Solid State Drives

***SSDs and large HDDs - partitioning gotchas*.**

Recently manufactured Hard Disk Drives (mechanical) store data in 4KB blocks rather than in 512B blocks that have been the normal size for many years.

With solid state disks the block size is typically 512kB, which is then subdivided into 4KB pages. Note that you can read and write to individual pages, so long as they are empty. However once a page has been written, it can’t be overwritten, it must be erased first before you can write to it again.

In summary, for typical SSDs you can read 4KB at a time, but you can only erase 512KB at a time. Reference 2) has a comprehensive discussion of these details.

Aligning partitions on large HDDs and SSDs can have a significant effect on their performance.

For large HDDs, you want to have the start of the first partition at a HDD disk block which is a multiple of 8 when counting from the start of the disk.

For SSDs the starting sector for the first SSD partition needs to be divisible by 512, giving an integer number. To cater for all variations of SSD page size and filesystem block size, the most frequently suggested starting sector is 2048 - which corresponds to 1MB.

In summary, you will need to use a disk partition utility which will allow you to specify a starting sector of the disk.

Solid State Drives

SSDs: Improving performance and reducing wear.

Typical standard MLC NAND flash in consumer level SSDs can only be erased around 10,000 times before it starts to give data storage errors. Reducing the number of writes is therefore beneficial.

Using fstab options for SSDs

Late model SSDs implement the ATA_TRIM command for sustained long term performance and wear leveling. TRIM support can be activated by using the ***discard*** option in the /etc/fstab file.

NB: You must use Linux kernel 2.6.33 or later and your SSD must support ATA_TRIM. Otherwise, attempting to set up TRIM using the discard option in /etc/fstab can result in data corruption.   At present, the ext4 and btrfs file systems support trim - use ext4.

You can also disable recording of last access times using noatime and nodiratime mount options in fstab, reducing the number of write operations.

An example fstab entry would be ..

```
/dev/sda     /    ext4  noatime,nodiratime,discard,errors=remount-ro    0    1
```

Solid State Drives

Check the I/Oscheduler

Check the I/O scheduler in use by starting a terminal and typing in the command

```
$ cat  /sys/block/sdX/queue/scheduler
```

the scheduler currently in use is enclosed in `[].`

eg/. on a Mint 14 system ..

```
$ cat /sys/block/sda/queue/scheduler
noop [deadline] cfq
```

Here, the deadline scheduler is active.

The default scheduler is often CFQ (Completely Fair Queing) which is tuned for HDDs (rotating platters). Other options are deadline and NOOP.

NOOP is basically a FIFO scheduler, while deadline ensures that reads are not unduly delayed by heavy writing. Thus the deadline scheduler appears more suited to SSDs.

Solid State Drives

To change the default I/O scheduler .. either

add the following line to the file **`/etc/rc.local`**

```
echo deadline >/sys/block/sdX/queue/scheduler
```

where X is your drive letter, eg/. **`a`** for **`/dev/sda`**

or

add/modify udev rules if **`/etc/rc.local`** is not used.
- see reference 7) for details

Solid State Drives

Redirect swap partition

If you have a mechanical HDD in your system, omit the swap partition on the SSD and put it on the HDD.

Redirect temporary and/or log files

Put temporary files (eg/. browser cache) on a ram disk or redirect to the HDD. See reference 4) for details.

Put **`/var`** on a partition on a HDD as this can be the target of many write operations in normal use. For example, log files and package updates are written to this directory tree.

Solid State Drives

References:

1) https://wiki.archlinux.org/index.php/Solid_State_Drives - SSDs on Linux

2) http://www.anandtech.com/show/2738 - Good background info

3) http://www.anandtech.com/show/2829 - Update of above

4) http://apcmag.com/how-to-maximise-ssd-performance-with-linux.htm - good practical advice.

5) http://techgage.com/article/enabling_and_testing_ssd_trim_support_under_linux/1/

6) http://techgage.com/article/enabling_and_testing_ssd_trim_support_under_linux/2/

7) http://wiki.archlinux.org/index.php/Solid_State_Drives