Optimizing AIX 7 operating system performance, Part 2: Monitoring logical volumes and analyzing the results

Martin Brown
Ken Milberg
October 19, 2010

Discover how to use appropriate disk placement prior to creating your logical volumes to improve disk performance. These investigations are based on AIX 7 beta and updating information from the original AIX 5L version of this article. Part 2 of this series focuses on monitoring your logical volumes and the commands and utilities (iostat, lvmstat, lsvg, lsvg, and lsvg) used to analyze results.

View more content in this series

About this series

This three-part series (see Related topics) on the AIX® disk and I/O subsystem focuses on the challenges of optimizing disk I/O performance. While disk tuning is arguably less exciting than CPU or memory tuning, it is a crucial component in optimizing server performance. In fact, partly because disk I/O is your weakest subsystem link, there is more you can do to improve disk I/O performance than any other subsystem.

Introduction

Unlike the tuning of other subsystems, tuning disk I/O should actually be started during the architectural phase of building your systems. While there are virtual memory equivalents of I/O tuning parameters (ioo and lvmo), the best way to increase disk I/O performance is by properly configuring your systems and not tuning parameters. Unlike virtual memory tuning, it is much more complex to change the way you structure your logical volumes after they have been created and are running, so you usually get only one chance to do this right. In this article, we will discuss ways that you can configure your logical volumes and where to actually place them with respect to the physical disk. We'll also address the tools used to monitor your logical volumes. Most of these tools are not meant to be used for long-term trending and are specific AIX tools that provide information on how the logical volumes are configured and if they have been optimized for your environment.
There are few changes to the main toolset and tunable parameters available in AIX 7, but it is worth re-examining the functionality to ensure that you are getting the best information and performance out of your system.

Part 1 (see Related topics) of this series introduced iostat, but it did not address using the tool outside of viewing asynchronous I/O servers. Part 2 uses iostat to monitor your disks and shows you what it can do to help quickly determine your I/O bottleneck. While iostat is a generic UNIX® utility that was not developed specifically for AIX, it is very useful for quickly determining what is going on in your system. The more specific AIX logical volume commands help drill down deeper into your logical volumes to help you really analyze what your problems are, if any. It's important that you clearly understand what you're looking for before using these tools. This article describes the tools and also shows you how to analyze their output, which helps in analyzing your disk I/O subsystem.

**Logical volume and disk placement overview**

This section defines the Logical Volume Manager (LVM) and introduces some of its features. Let's drill down into logical volume concepts, examine how they relate to improving disk I/O utilization, and talk about logical volume placement as it relates to the physical disk, by defining and discussing both intra-policy and inter-policy disk practices.

Conceptually, the logical volume layer sits between the application and physical layers. In the context of disk I/O, the application layers are the file system or raw logical volumes. The physical layer consists of the actual disk. LVM is an AIX disk management system that maps the data between logical and physical storage. This allows data to reside on multiple physical platters and to be managed and analyzed using specialized LVM commands. LVM actually controls all the physical disk resources on your system and helps provide a logical view of your storage subsystem. Understanding that it sits between the application layer and the physical layer should help you understand why it is arguably the most important of all the layers. Even your physical volumes themselves are part of the logical layer, as the physical layer only encompasses the actual disks, device drivers, and any arrays that you might have already configured. Figure 1 illustrates the concepts and shows how tightly integrated the logical I/O components relate to the physical disk and its application layer.
Let's now quickly introduce the elements that are part of LVM, from the bottom up. Each of the drives is named as a physical volume. Multiple physical volumes make up a volume group. Within the volume groups, logical volumes are defined. The LVM enables the data to be on multiple physical drives, though they might be configured to be on a single volume group. These logical volumes can be either one or multiple logical partitions. Each of the logical partitions has a physical partition that correlates to it. Here is where you can have multiple copies of the physical portions for purposes such as disk mirroring.

Let's take a quick look at how logical volume creation correlates with physical volumes. Figure 2 illustrates the actual storage position on the physical disk platter.

As a general rule, data that is written toward its center has faster seek times than data written on the outer edge. This has to do with the density of data. Because it is more dense as it moves toward its center, there is actually less movement of the head. The inner edge usually has the
slowest seek times. As a best practice, the more intensive I/O applications should be brought closer to the center of the physical volumes. Note that there are exceptions to this. Disks hold more data per track on the edges of the disk, not on the center. That being said, logical volumes being accessed sequentially should actually be placed on the edge for better performance. The same holds true for logical volumes that have Mirror Write Consistency Check (MWCC) turned on. This is because the MWCC sector is on the edge of the disk and not at the center of it, which relates to the intra-disk policy of logical volumes.

Let’s discuss another important concept referred to as the inter-disk policy of logical volumes. The inter-disk policy defines the number of disks on which the physical partitions of a logical volume actually resides. The general rule is that the minimum policy provides the greatest reliably and availability, and the maximum policy improves performance. Simply put, the more drives that data is spread on, the better the performance. Some other best practices include allocating intensive logical volumes to separate physical volumes, defining the logical volumes to the maximum size you need, and placing logical volumes that are frequently used close together. This is why it is so important to know your data prior to configuring your systems so that you can create policies that make sense from the start.

You can define your policies when creating the logical volumes themselves using the following command or smit fastpath: `# mklv` or `# smitty mklv`.

### Monitoring logical volumes and analyzing results

This section provides instructions on how to monitor your logical volumes and analyze the results. Various commands are introduced along with the purposes for which they are used, and we will examine the output.

A ticket has just been opened up with the service desk that relates to slow performance on some database server. You suspect that there might be an I/O issue, so you start with `iostat`. If you recall, this command was introduced in the first installment of the series (see Related topics), though only for the purposes of viewing asynchronous I/O servers. Now, let’s look at `iostat` in more detail. `iostat`, the equivalent of using `vmstat` for virtual memory, is arguably the most effective way to get a first glance of what is happening with your I/O subsystem.

#### Listing 1. Using iostat

```bash
# iostat 1 System configuration: lcpu=4 disk=4 tty: tin tout avg-cpu: % user % sys % idle % iowait 0.0 392.0 5.2 5.5 88.3 1.1 Disks: % tm_act Kbps tps Kb_read Kb_wrtn hdisk1 0.5 19.5 1.4 53437739 21482563 hdisk0 0.7 29.7 3.0 93086751 21482563 hdisk4 1.7 278.2 6.2 238584732 832883320 hdisk3 2.1 294.3 8.0 300653060 832883320
```

What are you seeing here and what does this all mean?

- **% tm_act**: Reports back the percentage of time that the physical disk was active or the total time of disk requests.
- **Kbps**: Reports back the amount of data transferred to the drive in kilobytes.
- **tps**: Reports back the number of transfers per second issued to the physical disk.
- **Kb_read**: Reports back the total data (kilobytes) from your measured interval that is read from the physical volumes.
• **Kb_wrtn**: Reports back the amount of data (kilobytes) from your measured interval that is written to the physical volumes.

You need to watch % tm_act very carefully, because when its utilization exceeds roughly 60 to 70 percent, it usually is indicative that processes are starting to wait for I/O. This might be your first clue of impending I/O problems. Moving data to less busy drives can obviously help ease this burden. Generally speaking, the more drives that your data hits, the better. Just like anything else, too much of a good thing can also be bad, as you have to make sure you don't have too many drives hitting any one adapter. One way to determine if an adapter is saturated is to sum the Kbps amounts for all disks attached to one adapter. The total should be below the disk adapter throughput rating, usually less than 70 percent.

Using the `-a` flag (see **Listing 2**) helps you drill down further to examine adapter utilization.

**Listing 2. Using iostat with the `-a` flag**

```
# iostat -a Adapter: Kbps tps Kb_read Kb_wrtn scsi0 0.0 0.0 0 0 Paths/Disk: % tm_act Kbps tps Kb_read Kb_wrtn hdisk1_Path0 37.0 89.0 0.0 0 0 hdisk0_Path0 67.0 47.0 0.0 0 0 hdisk4_Path0 0.0 0.0 0.0 0 0 adapter: Kbps tps Kb_read Kb_wrtn ide0 0.0 0.0 0 0 Paths/Disk: % tm_act Kbps tps Kb_read Kb_wrtn cd0 0.0 0.0 0.0 0 0
```

Clearly, there are no bottlenecks here. Using the `-d` flag allows you to drill down to one specific disk (see **Listing 3**).

**Listing 3. Using iostat with the `-d` flag**

```
# iostat -d hdisk1 1 System configuration: lcpu=4 disk=5 Disks: % tm_act Kbps tps Kb_read Kb_wrtn hdisk1 0.5 19.4 1.4 53437743 21490480 hdisk1 5.0 78.0 23.6 3633 3564 hdisk1 0.0 0.0 0.0 0 0 hdisk1 0.0 0.0 0.0 0 0 hdisk1 0.0 0.0 0.0 0 0
```

Let's look at some specific AIX LVM commands. You examined disk placement earlier and the importance of architecting your systems correctly from the beginning. Unfortunately, you don't always have that option. As system administrators, you sometimes inherit systems that must be fixed. Let's look at the layout of the logical volumes on disks to determine if you need to change definitions or re-arrange your data.

Let's look first at a volume group and find the logical volumes that are a part of it. `lsvg` is the command that provides volume group information (see **Listing 4**).

**Listing 4. Using lsvg**

```
# lsvg -l data2vg rootvg:LV NAME TYPE LPs PPs PVs LV STATE MOUNT POINT hd5 boot 1 1 1 closed/syncd N/A hd6 paging 24 24 1 open/syncd N/A hd8 jfs2log 1 1 1 open/syncd N/A hd4 jfs2 7 7 1 open/syncd / hd2 jfs2 76 76 1 open/syncd /usr hd9var jfs2 12 12 1 open/syncd /var hd3 jfs2 4 4 1 open/syncd /tmp hd1 jfs2 1 1 1 open/syncd /home hd10opt jfs2 12 12 1 open/syncd /opt hd11admin jfs2 4 4 1 open/syncd /admin livedump jfs2 8 8 1 open/syncd /var/adm/ras/livedump
```

Now, let's use `lslv`, which provides for specific data on logical volumes (see **Listing 5**).

**Listing 5. Using lslv**

```
# ls1v -1 data2vg rootvg:LV NAME TYPE LPs PPs PVs LV STATE MOUNT POINT hd5 boot 1 1 1 closed/syncd N/A hd6 paging 24 24 1 open/syncd N/A hd8 jfs2log 1 1 1 open/syncd N/A hd4 jfs2 7 7 1 open/syncd / hd2 jfs2 76 76 1 open/syncd /usr hd9var jfs2 12 12 1 open/syncd /var hd3 jfs2 4 4 1 open/syncd /tmp hd1 jfs2 1 1 1 open/syncd /home hd10opt jfs2 12 12 1 open/syncd /opt hd11admin jfs2 4 4 1 open/syncd /admin livedump jfs2 8 8 1 open/syncd /var/adm/ras/livedump
```
Listing 5. Using lslv

```bash
# lslv data2lv

```

This view provides a detailed description of your logical volume attributes. What do you have here? The intra-policy is at the center, which is normally the best policy to have for I/O-intensive logical volumes. As you recall from an earlier discussion, there are exceptions to this rule. Unfortunately, you've just hit one of them. Because Mirror Write Consistency (MWC) is on, the volume would have been better served if it were placed on the edge.

Let's look at its inter-policy. The inter-policy is minimum, which is usually the best policy to have if availability is more important than performance. Further, there are double the number of physical partitions than logical partitions, which signify that you are mirroring your systems. In this case, you were told that raw performance was the most important objective, so the logical volume was not configured in such a way as to the reality of how the volume is being utilized. Further, if you are mirroring your system and using an external storage array, this would even be worse, as you're already providing mirroring at the hardware layer, which is actually more effective than using AIX mirroring.

Let's drill down even further in Listing 6.

Listing 6. lslv with the -l flag

```bash
lslv -l hd4
```

The -l flag of lslv lists all the physical volumes associated with the logical volumes and distribution for each logical volume. You can then determine that 100 percent of the physical partitions on the disk are allocated to this logical volume. The distribution sections show the actual number of physical partitions within each physical volume. From here, you can detail its intra-disk policy. The order of these fields are as follows:

- Edge
- Middle
- Center
- Inner-middle
- Inner-edge

The reports show that most of the data is in the middle and some at the center.

Let's keep going and find out which logical volumes are associated with the one physical volume. This is done with the lspv command (see Listing 7).
Now you can actually identify which of the logical volumes on this disk are geared up for maximum performance.

You can drill down even further to get more specific (see Listing 8).

Listing 8. lspv with the -p flag

```
# lspv -p hdisk0
# PP RANGE STATE REGION LV NAME TYPE MOUNT POINT
1-1 used outer edge hd5 boot N/A
2-128 free outer middle hd6 paging N/A
145-152 used outer middle livedump jfs2 /var/adm/ras/livedump
153-160 free outer middle hd6 paging N/A
161-256 used center hd8 jfs2log
257-257 used center hd1 jfs2 /admin
FREE CENTER
512-639 free inner edge
```

This view tells you what is free on the physical volume, what has been used, and which partitions are used where. This is a nice view.

One of the best tools to look at LVM usage is with lvmstat (see Listing 9).

Listing 9. Using lvmstat

```
# lvmstat -v rootvg
lvmstat: Statistics collection is not enabled for this logical device. Use -e option to enable.
```

As you can see by the output here, it is not enabled (by default), so you need to enable it prior to running the tool using # lvmstat -v data2vg -e. The command shown in Listing 10 takes a snapshot of LVM information every second for 10 intervals.

Listing 10. lvmstat with the -v flag

```
# lvmstat -v rootvg 1 10
Logical Volume iocnt Kb_read Kb_wrtn Kbps hd8 54 0 216 0.00 hd9var 15 0 64 0.00 hd2 11 0 44 0.00 hd4 5 0 20 0.00 livedump 0 0 0 0.00 hd1 jfs2log
Logical Volume iocnt Kb_read Kb_wrtn Kbps hd4 11 0 44 44.00 hd8 8 0 32 32.00 livedump 0 0 0 0.00 hd1 jfs2log
Logical Volume iocnt Kb_read Kb_wrtn Kbps hd4 11 0 44 44.00 hd8 8 0 32 32.00 hd9var 8 0 36 36.00 hd3 6 0 24
24.00 hd2 2 0 8 8.00
```

What is particularly useful about this view is that it only shows the logical volumes where there has been activity. This can make it very convenient when monitoring specific applications and correlating that with the specific logical volume usage.

This view shows the most utilized logical volumes on your system since you started the data collection tool. This is very helpful when drilling down to the logical volume layer when tuning your systems.
What are you looking at here?

- **% iocnt**: Reports back the number of read and write requests.
- **Kb_read**: Reports back the total data (kilobytes) from your measured interval that is read.
- **Kb_wrtn**: Reports back the amount of data (kilobytes) from your measured interval that is written.
- **Kbps**: Reports back the amount of data transferred in kilobytes.

Look at the man pages for all the commands discussed before you start to add them to your repertoire.

**Tuning with lvm0**

This section goes over using a specific logical volume tuning command. The `lvm0` is used to set and display your pbuf tuning parameters. It is also used to display blocked I/O statistics.

`lvm0` allows you to change the pbuf, or pinned memory buffers, used for each volume group, and therefore shows and allows control over the memory used to cache volume group data.

Let's display your `lvm0` tunables for the data2vg volume group (see Listing 11).

**Listing 11. Displaying lvm0 tunables**

```
# lvm0 -v data2vg -a vgname = data2vg pv_pbuf_count = 1024 total_vg_pbubs = 1024 mag_vg_pbuf_count = 8192
perv_blocked_io_count = 7455 global_pbuf_count = 1024 global_blocked_io_count = 7455
```

What are the tunables here?

- **pv_pbuf_count**: Reports back the number of pbufs added when a physical volume is added to the volume group.
- **max_vg_pbuf_count**: Reports back the max amount of pbufs that can be allocated for a volume group.
- **global_pbuf_count**: Reports back the number of pbufs that are added when a physical volume is added to any volume group.

Let's increase the pbuf count for this volume group:

```
# lvm0 -v redvg -o pv_pbuf_count=2048
```

Quite honestly, we usually stay away from `lvm0` and use `ioo`. We are more accustomed to tuning the global parameters. It's important to note that if you increase the pbuf value too much, you can actually see a degradation in performance.

**Conclusion**

This article focused on logical volumes and how they relate to the disk I/O subsystem. It defined logical volumes at a high level and illustrated how it relates to the application and physical layers. It
also defined and discussed some best practices for inter-disk and intra-disk polices as they relate to creating and maintaining logical volumes. You looked at ways to monitor I/O usage for your logical volumes, and you analyzed the data that was captured from the commands that were used to help determine what your problems were. Finally, you actually tuned your logical volumes by determining and increasing the amount of pbufs used in a specific volume group. Part 3 of this series will focus on the application layer as you move on to file systems, using various commands to monitor and tune your file systems and disk I/O subsystems.
Related topics

- **Database Performance Tuning on AIX**: An IBM Redbook designed to help system designers, system administrators, and database administrators design, size, implement, maintain, monitor, and tune a Relational Database Management System (RDBMS) for optimal performance on AIX.
- "**Processor affinity on AIX**" (developerWorks, November 2006): Using process affinity settings to bind or unbind threads can help you find the root cause of troublesome hang or deadlock problems. Read this article to learn how to use processor affinity to restrict a process and run it only on a specified central processing unit (CPU).
- **The AIX 7.1 Information Center** is your source for technical information about the AIX operating system.
- **The IBM AIX Version 6.1 Differences Guide** can be a useful resource for understanding changes in AIX 6.1.
- **Operating System and Device Management**: Provides users and system administrators with complete information that can affect your selection of options when performing such tasks as backing up and restoring the system, managing physical and logical storage, and sizing appropriate paging space.
- **AIX and UNIX**: The AIX and UNIX developerWorks zone provides a wealth of information relating to all aspects of AIX systems administration and expanding your UNIX skills.
- Search the AIX and UNIX library by topic:
  - System administration
  - Application development
  - Performance
  - Porting
  - Security
  - Tips
  - Tools and utilities
  - Java™ technology
  - Linux®
  - Open source
- **IBM trial software**: Build your next development project with software for download directly from developerWorks.
- **Future Tech**: Visit Future Tech’s site to learn more about their latest offerings.

© Copyright IBM Corporation 2010
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)