IBM Systems magazine

e-Newsletter Exclusive

Disk I/O and the Network

Increase performance with more tips for AIX 5.3, 6.1 and 7

October 2010 | by Jaqui Lynch

Editor's Note: This is the concluding article in a two-part series on AIX tuning. Part one covered paging, memory and I/O delays and this article focuses on disk I/O and the network.

There have been many technology levels (TLs) released and some of the recommendations may have changed leading up to the release of AIX 7. In this article, I'll share additional AIX tuning information looking at tunables in AIX 5.3, 6.1 and 7, as it relates to disk I/O and the network.

One key reminder: A fresh AIX 6 or 7 install will automatically install the new defaults for memory. If the system is migrated from AIX 5.3, then any tunables set in AIX 5.3 will be migrated across. Prior to performing a migration, it's suggested you make a note of all of the tunables that have been changed (take a copy of /etc/tunables/nextboot) and then reset the tunables to the defaults. After migration, check nextboot and make sure there's nothing in it. Now, go ahead and set the tunables that need to be changed for AIX 6 or 7.

Disk I/O

Many of the most common performance problems are related to I/O issues. In particular, data layout can affect performance more than any I/O tunable the administrator can set. Since changing these later is extremely painful, it's important to plan in advance to avoid these problems.

The trend in the industry right now is to provide fewer, larger hdisks to the server. For example, the server may be given one 500 GB hdisk that's spread across several disks in the disk subsystem, rather than being given 10, 50 GB or five 100 GB hdisks. However, I/O performance depends on bandwidth, not size. While that data may be spread across multiple disks in the back end, this doesn't help with queuing in the front end. At the server, the hdisk driver has an in-process and a wait queue. Once an I/O is built in the JFS2

buffer, it gets queued to the LUN (hdisk). Queue_depth for an hdisk (LUN) represents the number of in-flight I/Os that can be outstanding for an hdisk at any given time.

The in-process queue for the hdisk can contain up to queue-depth I/Os and the hdisk driver submits the I/Os to the adapter driver. Why is this important? If your data's striped by LVM across five hdisks then you can have more I/Os in process at the same time. With one big hdisk, you'll be queuing. Multipath I/O drivers such as subsystem device driver (SDD) won't submit more than queue_depth I/Os to an hdisk, which can affect performance. You either need to increase queue_depth or disable that limit. In SDD, use the "datapath qdepth disable" command.

Some vendors do a nice job of setting the queue_depth, but if you're using large logical-unit numbers from multiple disks in the back end, then you'll want to grow this. You can use the iostat -D or the sar -d commands to figure this out. Interactive nmon also has a -D option, which lets you monitor sqfull as well. If you're using sddpcm, then you can use "pcmpath query devstats" to monitor sqfull and "pcmpath query adaptstats" to monitor adapter queuing.

In particular, look at the avgsqsz, avgwqsz and sqfull fields to determine if you need to increase queue_depth. Don't increase queue_depth beyond the disk manufacturer's recommendations. lsattr -El hdisk? shows the current queue_depth setting. queue_depth is a disruptive change and requires a reboot.

For Fibre Channel, the adapter also has an in-process queue, which can hold up to num_cmd_elems of I/Os. The adapter submits the I/Os to the disk subsystem and it uses direct memory access (DMA) to perform the I/O. You may need to consider changing two settings on the adapter. By default num_cmd_elems is set to 200 and max_xfer_size is set to 0x100000. The latter equates to a DMA size of 16 MB. For a heavy I/O load, I increase the DMA to 0x200000 (128 MB) and I've set num_cmd_elems as high as 2,048, although I normally start at 1,024. This has to be done before the hdisks, etc., are assigned or you'll have to rmdev them all to set these values. lsattr -El fcs? shows the current settings. Before changing these, check with your disk vendor. The fcstat command can be used to monitor these. Look for entries like:

```
FC SCSI Adapter Driver Information
  No DMA Resource Count: 0
  No Adapter Elements Count: 2567
  No Command Resource Count: 34114051
```

In the above, it's clear that num_cmd_elems isn't high enough and that the DMA area also needs increasing. This is a disruptive change that requires a reboot.

When using VIO servers, max_xfer_size and num_cmd_elems should be set on the VIO servers and, if using N_Port ID Virtualization (NPIV), they'll also need to be set on the NPIV client LPARs. Don't set the values on the NPIV client LPAR higher than the VIO servers; I tried this and my LPAR wouldn't boot, which was probably lucky, as I am sure there would have been overruns.

Figure 1

Volume Groups and Filesystems

Once the Fibre-Channel cards and hdisks are correctly set, you should look at the setup for volume groups, logical volumes (LVs) and filesystems. I try to have fewer volume groups with several disks in them, as it gives me the flexibility to easily move filesystems using mirroring technology. This lets me move a filesystem from one hot disk to another without an outage. I've seen setups where there's one volume group per filesystem and it's not conducive to flexibility in solving performance problems.

Asynchronous and Concurrent I/O

Asynchronous I/O (AIO) is set at the OS level and Oracle needs to know that it's configured. AIO is used to improve performance for I/O to raw LVs as well as filesystems. It's set up differently on AIX 5.3 and AIX 6.1/7. In 5.3, use chdev to set three parameters: minservers, maxservers and maxreqs. In 6.1 and 7, use ioo. If you do an "ioo -a | grep aio" on a 6.1/7 system, you'll see the new parameters. Gather statistics using the "iostat -A" command. From these you can tell how many AIOs are in use. Additionally, if you see maxg getting close to maxreqs, then you should increase maxreqs.

In 6.1 and 7, AIO changes and the subsystems are now loaded by default. The aio0 device goes away as does the new aioo command. Additionally, AIO is enabled by default and the only value that normally needs tuning is the aio_maxservers or posix_aio_maxservers option. This is done with the ioo command.

Concurrent I/O (CIO) is a feature of AIX with JFS2 that bypasses the buffer caching and reduces double buffering, where an I/O comes into memory, is stored there and then copied into the application buffer. CIO also removes inode locking for the filesystem during write operations, so it should only be used where the application takes care of data serialization. For Oracle, this means CIO should be used for DataBase files (DBFs), redo logs and control files, and flashback log files. It shouldn't be used for Oracle binaries or archive log files. Use of CIO, for a mixed or random-access workload, can make a significant difference in memory usage (reducing paging), CPU utilization (no more copying memory pages between the two memory locations) and performance in general. However, since it bypasses readahead, sequential operations may not perform as well.

Network

In Figure 1, I provide recommendations for setting network tunables. These are based on recommendations for Gbit network cards. In the netstat -v output you want to look for errors like overflows and memory allocation failures. As an example, if "Software Xmit Q Overflows" are occurring, then packets are overflowing the transmit queue on the adapters. Another indication of this is packets being dropped "due to memory allocation failure." To determine what the transmit queue is called, it'll be necessary to run the "lsattr -El

ent0" (or ent1 ....) command. The field to increase varies from 5.3 to 6 to 7 and is also different depending on whether the adapter is 100 Mb, 1Gb or 10Gb.

When setting network tunables, you should set them globally using no, but you will also need to check the individual adapters as the system can override the global settings. Use "ifconfig -a" to see if any of the parameters you set for the network in Figure 1 are being overridden. If they are smaller in the ifconfig output, then consider using ifconfig to set them to the new values.

Another network tuneable worth considering is tcp_nodelay, which is disabled by default. This can cause large delays for request/response workloads that might only send a few bytes and then wait for a response. TCP implements delayed acknowledgments, as it expects to piggy back a TCP acknowledgment on a response packet. The delay is normally 200 ms. In order to reduce this delay, tcp_nodelay should be enabled. This is done using the no command.

Gathering Performance Data

I have been using nmon for many years to gather historical performance data and to monitor the systems. Now that it's integrated into the system, I use the one IBM provides and kick off a new cronjob every night at midnight. The script it runs consists of the following:

```
#!/bin/ksh
#
cd /usr/local/perf
/usr/bin/nmon -ft -A -M -L -^ -s 150 -c 576
#
```

The above will gather information about the system for 24 hours, including AIO, large pages and the top processes. The resulting .nmon files can then be downloaded and processed using nmon analyzer, nmon consolidator and a number of other tools, including rrd.

Simple Changes, Large Impact

As you can see AIX 6 and 7 have vastly reduced the number of memory tunables you need to work on. However, it's still necessary to pay attention to I/O buffers, I/O in general and network tuning. Simple changes can have a major impact on your system so these should all be tested before going into production, but in many cases the changes above should help performance a great deal.