DB2 magazine

http://db2mag.com/story/showArticle.jhtml?articleID=59301197

The View From Outside

By Lester Knutsen

Unix utilities give a different look at database server performance.

It's not hard to find information on using Informix `ONSTAT` or other database utilities to monitor and tune IBM Informix Dynamic Server (IDS) databases. I've written several articles on that topic myself. (You can find them on my Web site, listed in the resources at the end of this column, if you're interested.) But the Informix utilities only tell part of the story. In order to effectively monitor and tune your database server, you need to know what the IDS server tells you about its performance and what the Unix operating system tells you about the whole server's performance.

How do you step outside the database server and monitor its performance with Unix or Linux utilities? Both operating systems have a rich set of utilities and commands that provide very useful information on the status of your database server.

The areas you'll want to review and monitor on your database server using both IDS and Unix utilities are:

- **CPU performance.** How busy are the CPUs? How big is the backlog of processes waiting to run on your CPUs?
- **Memory usage.** How much memory is being used? Are you swapping out to disk because too much is being used? Do you have unused memory that would allow you to increase the size of your database buffers?
- **Disk output.** What is the disk I/O throughput? Is a particular disk a bottleneck?
- **Network usage.** What's the network utilization? Are there network collisions and errors that may be slowing down your database applications?

Several Unix and Linux utilities can help you get the bigger picture of hardware performance, including:

- `sar`, for CPU, disk, and memory statistics
- `vmstat`, for CPU and virtual memory statistics
- `mpstat`, for per-CPU statistics
- `iostat`, for disk I/O throughput statistics
- `ps`, for Unix processes statistics
- `top`, for top Unix processes statistics
- `netstat`, for network statistics
- `uptime`, for the system uptime and the average load on the system.

In this column, I'll focus on CPU and memory performance and show you which key performance metrics indicate whether your CPUs are overloaded or your system memory is overused. (Both of these factors will slow down a database server.)

Setting up Sar

`Sar`, or System Activity Recorder, is a basic Unix utility (also available for Linux) that lets you collect and monitor database server performance statistics. `Sar` comprises two parts: a data collection and a reporting element. The data collection part is a `cron` job that runs at specific intervals to collect statistics and save them to a file in `/var/adm/sa/`. The default file will be `sa`*XX*, where `XX` is the day of the month for which the data has been collected. Files get overwritten every month; if you want to save data for more than a month, you'll need to copy the files to another location. Depending on your operating system, `sar` is run by `root` or `sys` in `cron`. In most default Unix installations, the `cron` entries are commented out.

My preferred collection interval, to ensure a continuous flow of system data, is to have sar run and collect data every 15 minutes, seven days a week. Sar is the only utility that requires configuration and setup. If it isn't installed on your system, work with your system administrator. Getting sar running usually only requires setting up a cron job. Listing 1 is an example entry that would run every 15 minutes, 24 hours a day, seven days a week.

Listing 1. A sar example for collecting data in 15-minute intervals.

```
EXAMPLE Cron setup to collect
data every 15 minutes:
0,15,30,45 * * * *
/usr/lib/sa/sa1
```

CPU Monitoring

How do you know when your CPUs are overloaded? On your desktop it's easy — you have to wait to do work. The same is true on a database server. If the database server has to wait to run a process, or has too many processes waiting to run, the CPUs are overloaded. A number of factors affect CPU load: the number of CPUs, their speed (older vs. newer CPUs), and the number of processes waiting for CPU time.

The Unix tools for monitoring your CPUs include `sar-u, vmstat, mpstat, top`, and `uptime`. My recommendation for CPU performance? If the CPUs are less than 30 percent busy, you're in great shape; if they're 30 to 60 percent busy, you're in fair shape (and should start planning for an upgrade); if they're more than 60 percent busy, they're in poor shape (overloaded). Think of a freeway. If it's empty, you can run at full speed. If it's 30 percent busy, you can cruise along. Between 30 and 60 percent busy, you're approaching a traffic jam; and when it's 100 percent busy, you're stopped in your tracks. If your CPUs are idle, your application has full use of them, and any bottlenecks you detect will be somewhere else (such as on your disks or network systems).

Listing 2 is an example of `sar` output. The first column shows what time the sample was taken. The next column, `%usr`, is the percent of time that is spent executing user application code. The third column, `%sys`, shows the percent of time spent executing operating system calls, including system calls from your application. Add the `%usr` and `%sys` columns to figure out how busy your CPUs are.

Listing 2. A sample sar output showing CPU usage and other statistics.

```
EXAMPLE output from sar:
00:00:00 %usr %sys %wio %idle
07:00:00  27    3    0    70
07:15:02  61    6    0    33
07:30:01  47    4    0    49
07:45:01  28    3    0    70
```

The column `%wio` shows the percent of time the CPUs spent waiting for disk I/O. A high percentage here means your disk system is slowing down your database server. The last column shows the percent of time the CPU is idle.

Another measure of CPU performance is the average number of processes waiting to run. If this number is high, your CPUs may be overloaded. Several tools can give you a quick snapshot, including `uptime`, which displays how long your system has been running as well as the load average over the last one minute, five minutes, and 15 minutes.

Listing 3 indicates that my system has been up for two days and had an average of .03 processes waiting to run one minute ago, .04 processes waiting to run five minutes ago, and .04 processes waiting to run 15 minutes ago.

Listing 3. Uptime output showing system uptime and load average over various intervals.

```
EXAMPLE Uptime CPU Load Average:
lester@atlas >uptime
9:58pm up 2 day(s), 5:52, 4 users, load average: 0.03, 0.04, 0.04
```

My guideline is that a load average of less than two per CPU is excellent, two to four per CPU is fair, and more than four per CPU is poor. On a four-way machine, a load average of less than eight (234 CPUs) is good. You can also use `sar` with the `-q` option to report the number of processes ready to run in the column `runq-sz`.

Monitoring Memory Usage

How can you tell if you're using too much memory, or not using all of it? If more memory is required than is physically available, your server will swap or page-out parts of memory to disk. This process will slow down the database server, because the server has to wait while the operating system loads and unloads parts of memory to and from disk.

The key to monitoring memory usage is making sure that your system isn't swapping out to disk. Most modern Unix operating systems will take extra memory and use it for file system cache, so you will rarely see a lot of free memory on a system. But could that memory be better used for database buffers or virtual memory than for file system cache?

Several tools let you monitor operating system paging or swapping. `Sar` with the `-p` option shows paging. `Vmstat` also shows paging and can be run in real time to monitor current activity on a system.

The example in Listing 4 shows `vmstat` run every five seconds, for five times. The column to monitor is labeled po, which means page-outs. A 0 indicates that there is no paging-out on the system. Any number other than 0 indicates that your server doesn't have enough memory

to run all the processes in memory. To solve this problem, reduce the size of your database buffers or virtual segments, or look for other applications that are using up memory.

LISTING 4. Monitoring operating system paging or swapping with `vmstat`.

```
EXAMPLE using vmstat:
lester@ >vmstat 5 5
```

procs			memory		page						disk				faults			cpu		
r	b	w	swap	free	re	mf	pi	po	fr	de	sr	s0	s1	s2	s3	in	sy	cs	us	sy	id
0	0	0	4350896	573168	0	0	0	0	0	0	0	0	1	1	7	4294967196	0	0	-5	-1	-104
0	0	0	3749680	370888	106	68	0	0	0	0	0	0	0	1	0	237	835	839	9	1	90
0	0	0	3748784	369728	3	3	0	0	0	0	0	0	0	1	0	233	368	728	25	0	75
0	0	0	3748816	369760	1	1	0	0	0	0	0	0	0	0	0	233	287	692	25	0	75
0	0	0	3748816	369760	0	0	0	0	0	0	0	0	0	1	0	226	278	715	9	2	89

A couple of utilities let you monitor how much memory each process is using. One of my favorites is `top`, which shows the top process using the CPU or memory. Another option is the basic Unix `ps` command, which lists all the running processes on your system. If you use the command `ps-el`, you'll get a column labeled `sz`, which indicates how much memory the process is using.

As I mentioned, paging-out processes from memory to disk makes a database server perform more slowly than it should. Your options are to add more memory to the server or to reduce the amount of memory your programs use.

The Informix configuration parameters that most directly affect memory are:

- `BUFFERS`, which determines the number of shared memory buffers
- `SHMVIRTSIZE`, which sets initial virtual shared memory segment size
- `SHMADD`, which sets the size of new shared memory segments
- `SHMTOTAL`, which determines the total size of shared memory.

If your server has unused memory, you can increase the size of the memory that IDS uses. I would start with increasing the number of buffers to improve performance. As you increase the number of buffers, continue to monitor your system paging to make sure you don't use up more memory than you have available.

Another factor that affects memory is whether you're running a 32- or 64-bit version of IDS. A 32-bit version is limited to about 3.6GB of memory on Solaris, 2GB on AIX, and 2GB on Windows. I once managed a large data warehouse server with 12GB of memory. We weren't able to use all the memory on the machine until we upgraded to a 64-bit version of IDS.

The Whole Story

IDS utilities alone don't present a complete performance picture of your database server. Without using some Unix utilities, you'd never know if your CPUs were idle or overloaded or if your system memory was being used effectively.

Keep an eye out for my next column, in which I'll share tips on using Unix tools for monitoring disk I/O and network performance.

Lester Knutsen is president of Advanced DataTools Corp., an IBM Informix consulting and training partner specializing in data warehouse development, database design, performance tuning, and Informix training and support. He is president of the Washington D.C. area Informix User Group, a founding member of the International Informix Users Group, and an

IBM Gold Consultant.

Resources

More Informix Articles
www.advancedatatools.com

IBM Informix
ibm.com/informix

Return to Article