Performance Analysis

How to improve performance?

- The following are general guidelines for achieving a higher performance level on a typical Linux box:
  - Make sure that you have enough memory to serve the running applications
  - Use software or hardware load balancing systems. They not only provide faster responses from network applications, but they also provide redundancy should one of the servers go down undexpectedly.
  - Review the application specific documentation and configuration files. Some settings may dramatically boost application performance like turning on caching in webservers or running multiple instances of a network application.
  - Avoid storage I/O bottlenecks by installing faster disks like SSD's (Solid State Disks), which do not depend on mechanically moving parts to offer much higher read/write speed than their old counterparts.
  - Use technologies like RAID to distribute I/O evenly on disks (like striping). However, not all applications/databases benefit from striping and RAID and sometimes this may lead to negative results. Application and database vendor and/or documentation should be consulted before moving to RAID.
  - Keep an eye on the network bandwidth and errors to ensure that the bandwidth is not saturated and that the error rate is at the minimum

Possible causes of bottlenecks

- Hardware-wise, performance is affected mainly by one or more of the following system components: CPU, memory, and disk and network I/O.
- Processes running on the system must access the above components. They compete to have, for example a CPU cycle or an I/O from the disk to read to write data. If the component is busy, the process will have to wait for its turn to be served. This wait time means that the system will run slower and implies that are having a performance issue.
- A common misconception is that a faster CPU will eliminate all performance-related problems. The CPU is already the fastest component of the system. If the process is waiting for an I/O to return with data, it will *waste* many CPU cycles waiting for the disk or the network to respond. Also when memory becomes full, the OS will start using the paging space on the disk to swap in and out memory pages. This will slow down things even more because the disk is the slowest part of the system. Buying an SSD disk and/or more RAM – in most cases – will mean much more performance boost than just upgrading the CPU.

Check your resources

- Before addressing a performance degradation problem, you must fist check your assets to have an estimate of the upper bound for the system's general performance level.
- The following files provide hardware information:
  - /proc/cpuinfo: take note of the vendor ID, cpu family, model and model name. Each processor core will have a stanza of its own. Useful information can be extracted from the CPU flags like ht which means that the CPU is using the hyper threading technology
  - /proc/meminfo: details on total, used, and free memory
  - /proc/diskstats: disk devices statistics
- Another useful command for this purpose is dmidecode. This will print a lot of hardware information about the machine like the motherboard type, BIOS version, installed memory among many other information.

Using vmstat to measure CPU utilization

- When measuring CPU performance, you may want to determine the overall CPU utilization to know whether or not the overall clock speed is the problem, load averages may also aid you in this. In addition, you may want to check per-process CPU consumption to know which process really hogging the CPU.
- Running vmstat gives you the information you need. It takes the number of seconds and the number of reports as the first and second arguments to determine the number of seconds for which the tool will calculate the averages. The first line of output represents the averages since the systems boot time. The subsequent lines present the average per n seconds.
- The right most column is for CPU readings. Us, sy, id, and wa represent the user, system, idle time, and wait time for the CPU.
- A high us means that the system is busy doing computational tasks, while a high sy time means the system is making a lot of system calls and/or making a lot of I/O requests. A system – typically – should be using no more than 50% in user time, no more than 50% in system time, and have a non-zero idle time.
- The cs is short for context switches per interval. That is how many times the kernel switched the running process per interval. The in is short for interrupts, it shows the number of interrupts per interval. A high cs or in rate may be an indication to a malfunctioning hardware device.

CPU load average and per-process

- Using the uptime command, it essentially provides the total time spent since the system was booted, but it also offers a CPU load average for the same period.
- The load average consists of three values that represent 5, 10, and 15 minutes averages.
- A load average that stays the same on a “good performance” and on a “performance degraded” one is an indication that you have to look elsewhere, perhaps at the network bandwidth, disk I/O, or the intalled memory.
- Other commands that offer real time view of the CPU per-process load is ps –aux and top. You may find a single process using more than 50% of the available CPU time. Using nice to decrease the execution prioroty of this process may help boost performance.

Memory management

- When an application requests memory to operate, the kernel offers this memory in the form of "pages". In Linux, a page size is 4 KiB.
- The kernel serves those pages from physical storage hardware (either RAM or SWAP space on the disk).
- The kernel shuffles pages between the SWAP space together with RAM. Memory that is not accessed for a specific period of time is moved into SWAP space (paged) to free more space for rather more frequently accessed memory.
- As more and more processes demand memory, the kernel tries to fulfil the requests by paging in and out memory pages from and to the SWAP space. And because the disk is the slowest component of the system, as the paging rate increases, performance is degraded as processes will have to wait longer before they can have their requested memory and things start to get slower.
- Finally, if the system runs out of both physical memory and SWAP space, the kernel resorts to more drastic measures: it kills the least important process with an out-of-memory killer function, a situation that should be avoided at all costs by anticipating the need to install more memory early enough.

Using vmstat to measure memory utilization

- Vmstat is used the same way it was used to measure CPU utilization.
- The swap in (si) and swap out (so) columns in the SWAP area of the output are of the most importance here. Pages that are read from disk into memory are "swapped in" while those which are ejected by the kernel into the disk are "swapped out". A high rate of si and so may be an indication that the system is using SWAP space extensively and that it might need more physical memory to be installed.
- Such a decision should not be reached by the si and so rates alone as they system *normally* does page in and page out operations. Only if is is accompanied by slow system response and user complaints.

Disk I/O measurement and analysis

- Disk I/O is measured using the iostat command. it is used the same way as vmstat is used with memory and CPU analysis. You can add –d argument to make the output specific for disk performance. –x is used for extended statistics.
- You will have to take note of the the following fields:
  - svctm: the average service time, which is how many milliseconds were spent serving I/O requests issued through the report period.
  - %util: the percentage of CPU cycles spent during issuing I/O requests to the device
  - r/s and w/s: read and write requests per second to the device,
- If one or more of the preceding field values is high for a long period of time, this means that the disk device(s) is causing a bottleneck and it should be replaced with a faster one, or a load-distribution mechanism should be applied like striping-RAID.

Optimizing disk I/O with "scheduler"

- The I/O scheduler is a kernel module that organizes and controls processes access to the disks and their I/O requests. There are three possible options for I/O scheduler:
  - CFQ: Completely Fair Queuing. This is the default algorithm and it is recommended for general purpose servers. It distributes the load evenly on all I/O bandwidth. It provides a balance between throughput and latency.
  - Deadline: it caps the maximum latency for an I/O request and provides the maximum possible throughput. This is best suited for disk-intensive applications like databases.
  - NOOP: it uses a FIFO (First In First Out) technique to handle I/O requests. It assumes that performance has been already optimized by the disk controller (if you are using SAN). This algorithm is more suited to SSD disks.
- You can change the default I/O scheduler at boot time by editing /etc/grub.conf and adding the option eleveator=*option* at the end of the kernel line. For example:

```
kernel /vmlinuz-3.8.13-118.2.2.el6uek.x86_64 ro
root=/dev/mapper/vg_redhat01-lv_root rd_NO_LUKS KEYBOARDTYPE=pc
KEYTABLE=us LANG=en_US.UTF-8 rd_NO_MD SYSFONT=latarcyrheb-sun16
rd_NO_DM rd_LVM_LV=vg_redhat01/lv_swap
rd_LVM_LV=vg_redhat01/lv_root rhgb quiet elevator=deadline
```

- This can be changed dynamically by using the following command:

```
echo sched_name > /sys/block/<sdx>/queue/scheduler
```

where `sched_name` is the algorithm name and `<sdx>` is the device name

A slow system quick diagnosis and remedy

- If you find that the system is suddenly running slower than before and users start complaining, you can examine the resources discussed in this section for bottlenecks.
- For example, running ps –auxww will show you the CPU utilization per process. If you find that a single process is using more than 50% of the CPU for a long time, this might be an indication of a fault in the process itself. Also check the load average with uptime to determine whether or not the CPU is contended.
- Check the paging activity with vmstat. If there are a lot of page-outs this means the the physical memory is overloaded. Additionally, if there is a lot of disk activity without paging this means the a process is extensively using the disk for read and write requests. If this is not the normal behavior (e.g. a database), the process activity should be further examined.
- It is difficult to know exactly which process is using the disk I/O the most, but using kill –STOP to temporarily suspend the susceptible process can narrow down the possibilities
- If a process is identified as resource intensive, a number of actions can be taken: if it is CPU intensive you can use the renice command to decrease its priority. You can also ask the user to run it later. If the process is hogging the disk and/or the network, renice will not solve the problem, but you can tune the process itself to optimize its behavior (for example webservers).