Chapter 1

System Performance Reporting

1.1 Understanding System Performance Parameters

The definition of performance of a system is dependent upon the expectations from a system. For example, **low latency** is desired from *database servers*, while **high throughput** is needed from *file servers*.

Actual performance has to be judged on the basis of performance level agreements. This has to be clearly defined for anyone - "*The web server should always react within 10 seconds*" is better than "*generic load should be less than 60%*", because that's what the end user will care about!

Thus, first we need to decide upon which metrics we want to measure, and then collect baseline data for it via monitoring systems.

1.1.1 Typical Performance Focus Areas

Factor	Description
Memory	The single most important factor that affects server performance. When enough memory isn't available, swap has to be used to house the excess pages and then the IO performance suffers, thus bogging down the entire system. It even affects the network throughput.
Disk	Another very important factor in overall server performance. When the disk is slow, too much memory is wasted to buffer data that's waiting to be written to disk. Processes will also have to wait longer to access data from the disk.
Network	Network is no longer a significant bottleneck, since most network connections aren't 10Mbps anymore - enterprise infrastructure uses Gigabit connections as a standard.
CPU	While the CPU has many tunables, in general it is not a very significant factor in performance deterioration. It is only for certain workloads that CPU becomes a factor in performance. The gain from CPU optimizations can be expressed in nanoseconds.

1.1.2 Common Performance Monitoring Tools

Terms	Description
top	While it's a very basic tool, it's also very rich in features. It provides an excellent generic overview of everything going on in the system. Typical use case for `top` is to detect problems and then use a more specialized tool to diagnose further.
iostat	A dedicated tool to detect Input/Output problems. It shows statistics about I/O. To detect which process is creating a high I/O load, a valuable tool is **iotop**.
vmstat	This tool shows statistics about virtual memory usage.
sar	The **System Activity Reporter** specializes in providing long-term data about what the system has been doing and long term performance statistics.

1.2 Understanding top

This is perhaps the single most important performance monitoring utility due to the kind of data it provides. There are alternatives to top such as `htop`, but top is programmed efficiently and doesn't have too much overhead. Comparing the two - htop uses about 5 times as much system resources as top!

The first feature of interest in the output of top is the **load average**, which consists of three numbers: the load average for the last 1, 5 and 15 minutes. The load average is the average of the number of processes in a runnable state, i.e., currently being executed by the CPU or waiting for CPU, over the concerned period of time. Optimally, all CPUs should be utilized as much as possible, but no process should be waiting for the CPU. The output of the `nproc` command tells us the effective number of CPUs available ($=$ Physical CPUs $\times$ logical cores per CPU).

The individual CPU utilization per CPU core can be shown by pressing the `1` key. A typical output is:

```
1 # top
2 %Cpu0  :  5.5 us,  3.3 sy,  0.0 ni, 90.7 id,  0.0 wa,  0.5 hi,  0.0 si,  0.0 st
```

Here, the number after the CPU indicates the core number. The *us* value refers to CPU usage in percentage in user space, i.e., by processes started by the end user without administrative privileges. The *sy* does the same, but for processes started by the users with root privileges. The *id* value is the percentage of time the processor remains idle. The next important metric is the number before *wa* which represents the waiting time, i.e., percentage of time processes spend waiting for I/O. A high value here indicates that the there's something wrong with the I/O channel and may indicate imminent disk failure.

Next, the memory statistics are shown, which includes the amount of memory completely free and amount of memory used to cache files that are frequently requested. Buffers contain data that needs to be written to disk during high I/O loads. While these are technically *non-essential*, it's suggested that 30% of the total memory be dedicated to buffers/cache usage.

We can also toggle the fields being shown by pressing the `f` key. If we quit top using the `q` key, the edits to the configuration are gone the moment we quit. However, if we quit using `Shift + W`, then the configuration is written to the `.toprc` file.

1.3 Understanding iostat

The iostat tool is a part of the `sysstat` package, which needs to be installed to use the `iostat` command. The command by itself provides a snapshot of the I/O statistics at the time of the invocation of the command. However, it takes two arguments in the syntax: `iostat <interval> <loops>`. The interval refers to the gap between displaying statistics and the loops refer to the number of times the command should show its output. Typical output for the command is:

```
# iostat 3 2
Linux 3.10.0-693.17.1.el7.x86_64 (vmPrime.somuVMnet.local)         Tuesday 27
↪  February 2018         _x86_64_        (1 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.50    0.00    0.64    0.49    0.00   98.37

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda               1.20        54.56         3.33     584199      35622
scd0              0.00         0.10         0.00       1054          0
dm-0              1.11        51.31         3.13     549442      33537
dm-1              0.01         0.21         0.00       2228          0
sdb               0.00         0.10         0.00       1044          0
sdc               0.00         0.10         0.00       1044          0
sdd               0.00         0.03         0.00        336          0

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           5.44    0.00    1.36    0.00    0.00   93.20

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda               0.68         0.00        10.88          0         32
scd0              0.00         0.00         0.00          0          0
dm-0              2.04         0.00        32.48          0         95
dm-1              0.00         0.00         0.00          0          0
sdb               0.00         0.00         0.00          0          0
sdc               0.00         0.00         0.00          0          0
sdd               0.00         0.00         0.00          0          0
```

In the output, **tps** refers to the number of transactions per second. The *kB_read/s* and the *kB_wrtn/s* values are self explanatory. The next two columns show the total kBs read and written respectively.

1.3.1 Usage scenario

Let us consider a scenario where top shows us that processes spend 60% of their execution time waiting for I/O. Let us consider that the concerned server is connected to 6 different disks or other storage devices. We can use the output of the iostat command to determine which disk is so slow.

If we consult the output from the command, we can see that dm-0 has the greatest tps. To find out which device is dm-0, we can simply go to the `/dev/mapper` directory and see what links to it:

```
# \ls -l /dev/mapper
total 0
crw-------. 1 root root 10, 236 Feb 27 20:53 control
lrwxrwxrwx. 1 root root       7 Feb 27 20:53 rhel-root -> ../dm-0
lrwxrwxrwx. 1 root root       7 Feb 27 20:53 rhel-swap -> ../dm-1
```

1.3.2 iotop

The **iotop** command needs to be installed using `yum -y install iotop`. It shows the processes that are doing the most amount of I/O in descending order. Typical output looks like:

```
1         # iotop
2         Total DISK READ :      45.37 M/s | Total DISK WRITE :       0.00 B/s
3         Actual DISK READ:      45.37 M/s | Actual DISK WRITE:       0.00 B/s
4         TID  PRIO  USER     DISK READ  DISK WRITE  SWAPIN     IO>    COMMAND
5         5696 be/4 root       45.37 M/s    0.00 B/s  0.00 % 73.97 % dd if=/dev/sda
  ↪   of=/dev/null
6         5450 be/4 root        0.00 B/s    0.00 B/s  0.00 % 12.66 % [kworker/0:2]
7         1 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % systemd --switched-root
  ↪   --system --deserialize 21
8         ...
```

Here we can see that the `dd if=/dev/sda of=/dev/null` is performing the most amount of I/O by copying the entire hard disk to `/dev/null`.

1.4 Understanding vmstat

1.5 Understanding sar components

1.6 Setting up sar

1.7 Analyzing sar data