Tune system performance

Linux #redhat #processes #daemons

Linux system comes fined tunned by default when you install, however there are a few tweaks that can be done based on system performance and application requirements

In this note:

- Optimize system performance by selecting a tuning profile managed by the `tuned` daemon
- Prioritize or de-prioritize specific processes with the `nice` and `renice` commands

Optimize system performance

What is `tuned`?

- Tuned pronounced as tune-d
- **Tune** is for system tuning and **d** is for daemon
- It is `systemd` service that is used to tune Linux system performance
- It is installed in CentOS/Redhat version 7, 8, and 9 by default.
- `tuned` package name is `tuned`
- The `tuned` service comes with pre-defined profiles and settings.
- Based on selected profile the `tuned` service automictically adjust system to get the best performance. E.g. `tuned` will adjust networking if you are downloading a large file or it will adjust IO settings if it detects high storage read/write
- The `tuned` daemon applies system settings when the service starts or upon selection of a new tuning profile.

`tuned` profiles

Tuned profile	Purpose
balanced	deal for systems that require a compromise between power saving and performance
desktop	Derived from the balanced profile. Provides faster response of interactive applications
Throughput-performance	Tunes the system for maximum throughput
Latency-performance	Ideal for server systems that require low latency at the expense of power consumption
network-latency	Derived from the latency-performance profile. It enables additional network tuning parameters to provide low network latency
Network-throughput	Derived from the throughput-performance profile. Additional network tuning parameters are applied for maximum network throughput
powersave	Tunes the system for maximum power saving
oracle	Optimized for Oracle database loads based on the throughput-performance profile
virtual-guest	Tunes the system for maximum performance if it runs on a virtual machine
virtual-host	Tunes the system for maximum performance if it acts as a host for virtual machines

- These are some `tuned` profiles, there could be added more but depending on the application.

- **Check if `tuned` is installed**
  - You can run `rpm -qa | grep tuned`
    - Returned `tuned-2.22.1-1.el9.noarch`
- **Install tuned package if NOT installed already**
  - `dnf install tuned`
- **Check `tuned` service status**
  - `systemctl status|enable|start tuned`
  - When started, it will ask you to authenticate your user account.
- **Command to change setting for `tuned` daemon**
  - `tuned-adm`
    - Command used to set the profile based on your system requirement
- **To check which profile is active**
  - `tuned-adm active`

Example using `tuned-adm` command:

```
[user@localhost ~]$ tuned-adm active
```

Output:

```
Current active profile: virtual-guest
```

- By default it has selected this profile, and the system will change `tuned` settings based on the profile it has selected.
- **To list all available profiles**
  - `tuned-adm list`

Example using `tuned-adm` command:

```
[user@localhost ~]$ tuned-adm list
```

Output:

```
Available profiles:
- accelerator-performance     - Throughput performance based tuning with disabled higher latency STOP
states
- aws                         - Optimize for aws ec2 instances
- balanced                    - General non-specialized tuned profile
- desktop                     - Optimize for the desktop use-case
- hpc-compute                 - Optimize for HPC compute workloads
- intel-sst                   - Configure for Intel Speed Select Base Frequency
- latency-performance         - Optimize for deterministic performance at the cost of increased power
consumption
- network-latency             - Optimize for deterministic performance at the cost of increased power
consumption, focused on low latency network performance
- network-throughput          - Optimize for streaming network throughput, generally only necessary on
older CPUs or 40G+ networks
- optimize-serial-console     - Optimize for serial console use.
- powersave                   - Optimize for low power consumption
- throughput-performance      - Broadly applicable tuning that provides excellent performance across a
variety of common server workloads
- virtual-guest               - Optimize for running inside a virtual guest
```

```
- virtual-host               - Optimize for running KVM guests
Current active profile: virtual-guest
```

- **To change to desired profile**
  - `tuned-adm profile [profile-name]`

Example using `tuned-adm` command:

```
[root@localhost ~]# tuned-adm profile balanced
```

- Note we need to be root user in order to change the `tuned` profile.
- In this line we are attempting to change to the balanced `tuned` profile.
- **Check for `tuned` recommendation**
  - `tuned-adm recommend`
  - It will tell you what profile you should select based on the system that you are running.

Example using `tuned-adm` command:

```
[root@localhost ~]# tuned-adm recommend
```

Output:

```
virtual-guest
```

- As we are running the system in a virtual machine, this is the best option we can select as profile.
- **Turn off `tuned` setting daemon**
  - `tuned-adm off`
  - If you don't want the system to select any profile, you want to do things manually, you can turn `tuned` off.

Example using `tuned-adm` command:

```
[root@localhost ~]# tuned-adm active
```

Output:

```
No current active profile.
```

- If the `tuned-adm off` command was ran previously, when trying to get the current profile, it will output this.

Change profile through web console

- Login to https://192.168.1.x:9090
  - Remember to start the `cockpit` service, as well as stopping the `firewalld`.
- Overview -> Configuration -> Performance profile
- Then select the profile you want to set, and it will be applied right away.

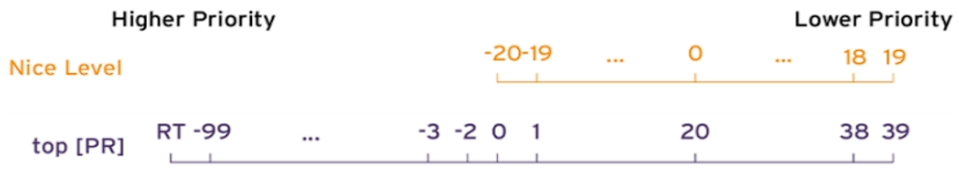
Prioritize or de-prioritize processes.

- Another way of keeping your system fine-tuned is by prioritizing processes through `nice` and `renice` command
- If a server has 1 CPU then it can execute 1 computation/process at a time as they come in (first come first served) while other processes must wait.
- With `nice` and `renice` commands we can make the system to give preference to certain processes than others
- This priority can be set at 40 different levels
- The nice level values range from -20 (highest priority) to 19 (lowest priority) and by default, processes inherit their nice level from their parent, which is usually 0.
  - The negative sign gives you the highest priority
- **To check process priority**
  - `top`

Output from `top`:

```
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
  824 rtkit     21   1  154108   3328   3072 S   0.3   0.0   0:02.63 rtkit-daemon
 3835 root      20   0       0      0      0 I   0.3   0.0   0:03.89 kworker/u16:2-events_unbound
    1 root      20   0  174564  17812  10756 S   0.0   0.2   0:15.90 systemd
    2 root      20   0       0      0      0 S   0.0   0.0   0:00.31 kthreadd
```

- When you run `top`, you will see two columns named **PR** and **NI**. (3rd and 4th column respectively)
- The **PR** column is the system priority level
- The **NI** is the actual user setting, the `nice` level that you actually set
- What is the difference?



- The system has its own priority that actually aligns it to the user nice level.
- Nice value is a user-space and priority PR is the process's actual priority that use by Linux kernel. In Linux system priorities are 0 to 139 in which 0 to 99 for real time and 100 to 139 for users
- **Process priority can be viewed through ps command as well with the right options**
  - `ps axo pid,comm,nice,cls --sort=-nice`

Example using `ps` command:

```
[root@localhost ~]# ps axo pid,comm,nice,cls --sort=-nice
```

Output:

```
  PID COMMAND         NI CLS
   80 khugepaged      19  TS
  846 alsactl          - IDL
   79 ksmd             5  TS
  824 rtkit-daemon     1  TS
    1 systemd          0  TS
```

```
      2 kthreadd          0  TS
     12 rcu_tasks_kthre   0  TS
...
```

- You'll see all the processes that are running in your system with priority level assigned to them.

The `nice` and `renice` commands

- **To set the process priority**
  - `nice -n [#] [process-name]`
    - e.g. `nice -n -15 top`

Example using `nice` command:

```
[root@localhost ~]# nice -n -15 top
```

- Changing the nice level of the `top` process to -15
- The `-n` options adds integer N to the niceness (default is 10)

Output from `top` (it was ran a second time):

```
    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
  12173 root       5 -15  226056   4352   3456 R   0.3   0.0   0:00.03 top
```

- Note that the new PR value is 5 and the new nice value is -15 (set previously)
- **To change the process priority**
  - What if you wanted to change the process priority but without stopping the process. Use the `renice` command.
  - `renice -n [#] [PID]`
    - e.g. `renice -n 12 12173`
      - In this case you cannot use the process name, you have to use the process ID (PID).

Example using `renice` command:

```
[root@localhost ~]# renice -n 12 12173
```

- Number 12173 is the PID of the `top` process.
- We are attempting to change the process priority from the `top` process to a priority of 12, without having to stop and run the process again.

Output:

```
12173 (process ID) old priority -15, new priority 12
```

Output from `top` (never stopped):

```
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
12173 root      32  12  226056   4352   3456 R   0.3   0.0   0:00.03 top
```