
Power Management Guide

Performance Tuning Whitepapers



Red Hat Inc.
Don Domingo

Abstract

This document explains how to manage power consumption on Red Hat Enterprise Linux 5 systems effectively. The following sections discuss different techniques that lower power consumption (for both server and laptop), and how each technique affects the overall performance of your system.

1. Importance of Power Management	1
2. Power Management Basics	2
3. Using cpufreq Governors	2
3.1. CPUfreq Setup	3
3.2. Tuning CPUfreq Policy and Speed	4
3.3. CPUfreq Governor Types	5
4. Aggressive Link Power Management	6
A. Revision History	7
Index	7

1. Importance of Power Management

At the core of power management is an understanding of how to effectively optimize energy consumption of each system component. This entails studying the different tasks that your system performs, and configuring each component to ensure that its performance just right for the job.

The main concerns surrounding power management are:

- Heat reduction for servers
- Reducing overall power consumption
- Extending battery life for laptops

As a rule, lowering the power consumption of a specific component (or of the system as a whole) will lead to lower heat and naturally, performance. As such, you should thoroughly study and test the decrease in performance afforded by any configurations you make, especially for mission-critical systems.

The following sections will explain how optimal hardware performance benefits your system in terms of energy consumption.

2. Power Management Basics

Effective power management is built on the following principles:

An idle CPU should only wake up when needed.

The Red Hat Enterprise Linux 5 kernel used a periodic timer for each CPU. This timer prevents the CPU from truly going idle, as it requires the CPU to process each timer event (which would happen every few milliseconds, depending on the setting), regardless of whether any process was running or not. A large part of effective power management involves reducing the frequency at which CPU wake-ups are made.

Unused hardware / devices should be disabled completely.

This is especially true for devices that have moving parts (for example, hard disks). In addition to this, some applications may leave an unused but enabled device “open”; when this occurs, the kernel assumes that the device is in use, which can prevent the device from going into a power saving state.

Low activity should translate to low wattage.

In many cases, however, such is not the case. For example, the CPU needs to be manually set configured to use a lower clock frequency during idle times (e.g. after office hours). Not all system components are intelligent enough to lower power usage / heat emission efficiently during idle times; the task of configuring a power-efficient system is usually up to you.

Latency-on-wakeup can offset power savings from idle time.

This is especially true for the CPU, which has different states of sleep (C-states), frequency (P-states), and heat output (T-states or “thermal states”). A CPU running on the lowest sleep state possible consumes the least amount of watts, but it also takes considerably more power to wake it up from that state when needed.

3. Using cpufreq Governors

One of the most effective ways to reduce power consumption and heat output on your system is to use CPUfreq. CPUfreq -- also referred to as CPU speed scaling -- allows the clock speed of the processor to be adjusted on the fly. This enables the system to run at a reduced clock speed to save power. The rules for shifting frequencies, whether to a faster or slower clock speed, and when to shift frequencies, are defined by the CPUfreq governor.

The governor defines the power characteristics of the system CPU, which in turn affects CPU performance. Each governor has its own unique behavior, purpose, and suitability in terms of workload. This section describes how to choose and configure a CPUfreq governor, the characteristics of each governor, and what kind of workload each governor is suitable for.

At the core of power management is an understanding of how to effectively optimize energy consumption of each system component. This entails studying the different tasks that your system performs, and configuring each component to ensure that its performance just right for the job.

The main concerns surrounding power management are:

- Heat reduction for servers
- Extending battery life for laptops

As a rule, lowering the power consumption of a specific component (or of the system as a whole) will lead to lower heat and naturally, performance. As such, you should thoroughly study and test the decrease in performance afforded by any configurations you make, especially for mission-critical systems.

The following sections explain how optimal hardware performance benefits your system in terms of energy consumption.

3.1. CPUfreq Setup

Before selecting and configuring a CPUfreq governor, you need to add the appropriate CPUfreq driver first.

Procedure 1. How to Add a CPUfreq Driver

1. Use the following command to view which CPUfreq drivers are available for your system:

```
ls /lib/modules/[kernel version]/kernel/arch/[architecture]/kernel/cpu/
cpufreq/
```

2. Move to the directory of the CPUfreq drivers.

```
cd /lib/modules/[kernel version]/kernel/arch/[architecture]/kernel/cpu/
cpufreq/
```

3. From there, use **modprobe** to add the appropriate CPUfreq driver.

```
modprobe [CPUfreq driver]
```

When using the above command, be sure to remove the **.ko** filename suffix.



Important

When choosing an appropriate CPUfreq driver, always choose **acpi-cpufreq** over **p4-clockmod**. While using the **p4-clockmod** driver reduces the clock frequency of a CPU, it does not reduce the voltage. **acpi-cpufreq**, on the other hand, reduces voltage along with CPU clock frequency, allowing less power consumption and heat output for each unit reduction in performance.

4. Once the CPUfreq driver is set up, you can view which governor the system is currently using with:

```
cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

You can also view which governors are available for use for a specific CPU using:

```
cat /sys/devices/system/cpu/[cpu ID]/cpufreq/scaling_available_governors
```

Some CPUfreq governors may not be available for you to use. In this case, use **modprobe** to add the necessary kernel modules that enable the specific CPUfreq governor you wish to use. These kernel modules are available in `/lib/modules/[kernel version]/kernel/drivers/cpufreq/`.

Procedure 2. Enabling a CPUfreq Governor

1. If a specific governor is not listed as available for your CPU, move to the directory of the CPUfreq drivers:

```
cd /lib/modules/[kernel version]/kernel/drivers/cpufreq/
```

2. Use **modprobe** to enable the governor you wish to use. For example, if the **ondemand** governor is not available for your CPU, use the following command:

```
modprobe cpufreq_ondemand
```

3. Once a governor is listed as available for your CPU, you can enable it using:

```
echo [governor] > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

3.2. Tuning CPUfreq Policy and Speed

Once you've chosen an appropriate CPUfreq governor, you can further tune the speed of each CPU using the tunables found in `/sys/devices/system/cpu/[cpu ID]/cpufreq/`. These tunables are:

- **cpuinfo_min_freq** — Shows the CPU's available minimum operating frequency (in KHz).
- **cpuinfo_max_freq** — Shows the CPU's available maximum operating frequency (in KHz).
- **scaling_driver** — Shows what CPUfreq driver is used to set the frequency on this CPU.
- **scaling_available_governors** — Shows the CPUfreq governors available in this kernel. If you wish to use a CPUfreq governor that is not listed in this file, refer to [Enabling a CPUfreq Governor](#) in [Section 3.1, "CPUfreq Setup"](#) for instructions on how to do so.
- **scaling_governor** — Shows what CPUfreq governor is currently in use. To use a different governor, simply use **echo [governor] > /sys/devices/system/cpu/[cpu ID]/cpufreq/scaling_governor** (refer to [Enabling a CPUfreq Governor](#) in [Section 3.1, "CPUfreq Setup"](#) for more information).
- **cpuinfo_cur_freq** — Shows the current speed of the CPU (in KHz).
- **scaling_available_frequencies** — Lists available frequencies for the CPU, in KHz.
- **scaling_min_freq** and **scaling_max_freq** — Sets the *policy limits* of the CPU, in KHz.



Important

When setting policy limits, you should set **scaling_max_freq** before **scaling_min_freq**.

- **affected_cpus** — Lists CPUs that require frequency coordination software.
- **scaling_setspeed** — Used to change the clock speed of the CPU, in KHz. You can only set a speed within the policy limits of the CPU (as per **scaling_min_freq** and **scaling_max_freq**).

To view the current value of each tunable, use **cat [tunable]**. For example, to view the current speed of cpu0 (in KHz), use:

```
cat /sys/devices/system/cpu/cpu0/cpufreq/cpuinfo_cur_freq.
```

To change the value of each tunable, use **echo [value] > /sys/devices/system/cpu/[cpu ID]/cpufreq/[tunable]**. For example, to set the minimum clock speed of cpu0 to 360 KHz, use:

```
echo 360000 > /sys/devices/system/cpu/cpu0/cpufreq/scaling_min_freq
```

3.3. CPUfreq Governor Types

This section lists and describes the different types of CPUfreq governors available in Red Hat Enterprise Linux 5.

cpufreq_performance

The Performance governor forces the CPU to use the highest possible clock frequency. This frequency will be statically set, and will not change. As such, this particular governor offers *no power saving benefit*. It is only suitable for hours of heavy workload, and even then only during times wherein the CPU is rarely (or never) idle.

cpufreq_powersave

By contrast, the Powersave governor forces the CPU to use the lowest possible clock frequency. This frequency will be statically set, and will not change. As such, this particular governor offers maximum power savings, but at the cost of the *lowest CPU performance*.

The term "powersave" can sometimes be deceiving, though, since (in principle) a slow CPU on full load consumes more power than a fast CPU that is not loaded. As such, while it may be advisable to set the CPU to use the Powersave governor during times of expected low activity, any unexpected high loads during that time can cause the system to actually consume more power.

The Powersave governor is, in simple terms, more of a "speed limiter" for the CPU than a "power saver". It is most useful in systems and environments where overheating can be a problem.

cpufreq_ondemand

The Ondemand governor is a dynamic governor that allows the CPU to achieve maximum clock frequency when system load is high, and also minimum clock frequency when the system is idle. While this allows the system to adjust power consumption accordingly with respect to system load,

it does so at the expense of *latency between frequency switching*. As such, latency can offset any performance/power saving benefits offered by the Ondemand governor if the system switches between idle and heavy workloads too often.

For most systems, the Ondemand governor can provide the best compromise between heat emission, power consumption, performance, and manageability. When the system is only busy at specific times of the day, the Ondemand governor will automatically switch between maximum and minimum frequency depending on the load without any further intervention.

cpufreq_userspace

The Userspace governor allows userspace programs (or any process running as root) to set the frequency. This governor is normally used in conjunction with the **cpuspeed** daemon. Of all the governors, Userspace is the most customizable; and depending on how it is configured, it can offer the best balance between performance and consumption for your system.

cpufreq_conservative

Like the Ondemand governor, the Conservative governor also adjusts the clock frequency according to usage (like the Ondemand governor). However, while the Ondemand governor does so in a more aggressive manner (i.e. from maximum to minimum and back), the Conservative governor switches between frequencies more gradually.

This means that the Conservative governor will adjust to a clock frequency that it deems fitting for the load, rather than simply choosing between maximum and minimum. While this can possibly provide significant savings in power consumption, it does so at an ever *greater latency* than the Ondemand governor.



Note

You can enable a governor using **cron** jobs. This allows you to automatically set specific governors during specific times of the day. As such, you can specify a low-frequency governor during idle times (e.g. after work hours) and return to a higher-frequency governor during hours of heavy workload.

For instructions on how to enable a specific governor, refer to *Enabling a CPUfreq Governor* in Section 3.1, “CPUfreq Setup”.

4. Aggressive Link Power Management

Aggressive Link Power Management (ALPM) is a power-saving technique that helps the disk save power by setting a SATA link to the disk to a low-power setting during idle time (i.e. when there is no I/O). ALPM automatically sets the SATA link back to an active power state once I/O requests are queued to that link.

Power savings introduced by ALPM come at the expense of disk latency. As such, you should only use ALPM if you expect the system to experience long periods of idle I/O time.

ALPM is only available on SATA controllers that use the *Advanced Host Controller Interface (AHCI)*. For more information about AHCI, refer to <http://www.intel.com/technology/1serialata/1ahci.htm>.

When available, ALPM is enabled by default. ALPM has three modes:

min_power

This mode sets the link to its lowest power state (SLUMBER) when there is no I/O on the disk. This mode is useful for times when an extended period of idle time is expected.

medium_power

This mode sets the link to the second lowest power state (PARTIAL) when there is no I/O on the disk. This mode is designed to allow transitions in link power states (e.g. during times of intermittent heavy I/O and idle I/O) with as small impact on performance as possible.

medium_power mode allows the link to transition between PARTIAL and fully-powered (i.e. "ACTIVE") states, depending on the load. Note that it is not possible to transition a link directly from PARTIAL to SLUMBER and back; in this case, either power state cannot transition to the other without transitioning through the ACTIVE state first.

max_performance

ALPM is disabled; the link does not enter any low-power state when there is no I/O on the disk.

**Warning**

Setting ALPM to **min_power** or **medium_power** will automatically disable the "Hot Plug" feature.

A. Revision History

Revision History

Revision 1.0

updated to build on Publican 0.36

DonDomingo ddomingo@redhat.com

Index

A

acpi-cpufreq

 CPUfreq

 setup, 3

adding a driver

 CPUfreq

 setup, 3

adding a governor

 CPUfreq

 setup, 4

Advanced Host Controller Interface

 Aggressive Link Power Management, 6

affected_cpus

 CPUfreq

 tuning, 5

Aggressive Link Power Management

- Advanced Host Controller Interface, 6

- AHCI, 6

- ALPM, 6

- hot-plug, 7

- max_performance, 7

- medium_power, 7

- min_power, 7

- PARTIAL, 7

- SATA-AHCI, 6

- SLUMBER, 7

- AHCI

- Aggressive Link Power Management, 6

- ALPM

- Aggressive Link Power Management, 6

B

- basics

- power management, 2

- principles behind, 2

C

- concerns surrounding power management

- CPUfreq

- governors, 3

- Conservative governor

- CPUfreq

- governor types, 6

- CPUfreq, 2, 3, 4

- cron, 6

- governor types, 5

- Conservative governor, 6

- cpufreq_conservative, 6

- cpufreq_ondemand, 5

- cpufreq_performance, 5

- cpufreq_powersave, 5

- cpufreq_userspace, 6

- Ondemand governor, 5

- Performance governor, 5

- Powersave governor, 5

- Userspace governor, 6

- governors, 2

- concerns surrounding power management, 3

- definition, 2

- setup, 3

- acpi-cpufreq, 3

- adding a driver, 3

- adding a governor, 4

- enabling a governor, 4

- modprobe, 4

- p4-clockmod, 3

- selecting a driver, 3

- tuning, 4
 - affected_cpus, 5
 - cpuinfo_cur_freq, 4
 - cpuinfo_max_freq, 4
 - cpuinfo_min_freq, 4
 - policy, 4
 - policy limits, 5
 - scaling_available_frequencies, 4
 - scaling_available_governors, 4
 - scaling_driver, 4
 - scaling_governor, 4
 - scaling_min_freq, 4
 - scaling_setspeed, 5
- types of governors, 5
- cpufreq_conservative
 - CPUfreq
 - governor types, 6
- cpufreq_ondemand
 - CPUfreq
 - governor types, 5
- cpufreq_performance
 - CPUfreq
 - governor types, 5
- cpufreq_powersave
 - CPUfreq
 - governor types, 5
- cpufreq_userspace
 - CPUfreq
 - governor types, 6
- cpuinfo_cur_freq
 - CPUfreq
 - tuning, 4
- cpuinfo_max_freq
 - CPUfreq
 - tuning, 4
- cpuinfo_min_freq
 - CPUfreq
 - tuning, 4
- cron
 - CPUfreq, 6
 - scheduled governor switching, 6

D

- definition
 - CPUfreq
 - governors, 2

E

- enabling a governor
 - CPUfreq
 - setup, 4

G

governor types

- CPUfreq, 5

 - Conservative governor, 6

 - cpufreq_conservative, 6

 - cpufreq_ondemand, 5

 - cpufreq_performance, 5

 - cpufreq_powersave, 5

 - cpufreq_userspace, 6

 - Ondemand governor, 5

 - Performance governor, 5

 - Powersave governor, 5

 - Userspace governor, 6

governors

- CPUfreq, 2

 - concerns surrounding power management, 3

 - definition, 2

H

hot-plug

- Aggressive Link Power Management, 7

I

importance

- power management, 1

M

max_performance

- Aggressive Link Power Management, 7

medium_power

- Aggressive Link Power Management, 7

min_power

- Aggressive Link Power Management, 7

modprobe

- CPUfreq

 - setup, 4

O

Ondemand governor

- CPUfreq

 - governor types, 5

P

p4-clockmod

- CPUfreq

 - setup, 3

PARTIAL

- Aggressive Link Power Management, 7

Performance governor

- CPUfreq

- governor types, 5
- policy
 - CPUfreq
 - tuning, 4
- policy limits
 - CPUfreq
 - tuning, 5
- power management
 - basics, 2
 - importance, 1
- Powersave governor
 - CPUfreq
 - governor types, 5
- principles behind
 - basics, 2

S

- SATA-AHCI
 - Aggressive Link Power Management, 6
- scaling_available_frequencies
 - CPUfreq
 - tuning, 4
- scaling_available_governors
 - CPUfreq
 - tuning, 4
- scaling_driver
 - CPUfreq
 - tuning, 4
- scaling_governor
 - CPUfreq
 - tuning, 4
- scaling_min_freq
 - CPUfreq
 - tuning, 4
- scaling_setspeed
 - CPUfreq
 - tuning, 5
- scheduled governor switching
 - cron, 6
- selecting a driver
 - CPUfreq
 - setup, 3
- setup
 - CPUfreq, 3
 - acpi-cpufreq, 3
 - adding a driver, 3
 - adding a governor, 4
 - enabling a governor, 4
 - modprobe, 4
 - p4-clockmod, 3
 - selecting a driver, 3

SLUMBER

Aggressive Link Power Management, 7

T

tuning

CPUfreq, 4

affected_cpus, 5

cpufreq_cur_freq, 4

cpufreq_max_freq, 4

cpufreq_min_freq, 4

policy, 4

policy limits, 5

scaling_available_frequencies, 4

scaling_available_governors, 4

scaling_driver, 4

scaling_governor, 4

scaling_min_freq, 4

scaling_setspeed, 5

types of governors

CPUfreq, 5

U

Userspace governor

CPUfreq

governor types, 6