



RED HAT ENTERPRISE LINUX: UNDERSTANDING CPUfreq POWER GOVERNORS

BRIAN MALY



EXECUTIVE SUMMARY

With today's rising energy costs, controlling power consumption is increasingly important. Power usage affects everyone—from a laptop user trying to stretch battery life to the corporate data center with many machines. More systems, more applications, more storage, and more servers require a lot of electricity, and they generate a lot of heat. Add the cost of cooling, and overall power costs for any IT infrastructure become significant rather quickly. Then there's still cooling to consider.

The most effective way to reduce power demand and cool down a Linux system is to use CPUfreq technology. CPUfreq, also referred to as CPU speed scaling, allows the clock speed of the processor to be adjusted dynamically—on the fly. This enables the system to run at a reduced clock speed for the purpose of saving 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 how the power characteristics of the system will behave. The governor also influences the performance of the system. There are many governors, and each has its own unique behavior and purpose. There is no one-size-fits-all solution. In this whitepaper, we briefly discuss each of CPUfreq's governors and how best to utilize them with respect to individual computing needs.

YOUR HARDWARE

Your hardware must support CPUfreq technology in order to use any of the governors discussed in this paper. CPUfreq technology has been around for a number of years, and nearly all recently manufactured mainstream processors support this technology, therefore a system that is brand new, or even several years old can most often use CPUfreq technology. The CPUfreq driver must also be employed. Since it is most often built into the kernel, you likely won't have to do anything special to use this technology.

THE GOVERNORS

Which governor to use depends a lot on hardware and workload. Everyone has different needs and expectations from their system, and there is no one-size-fits-all governor. Each governor has its own individual advantages and disadvantages. Understanding how each governor behaves is the key to best leveraging the most power savings and performance from your hardware. Let's take a look at each of the five governors in more detail.

THE PERFORMANCE GOVERNOR

This governor forces the CPU to use the highest available clock frequency. The frequency is statically set and does not change. Using this governor will yield maximum performance; however no power savings is employed. This governor is complementary to and most often utilized in conjunction with the *powersave* governor.

THE POWERSAVE GOVERNOR

This governor forces the CPU to use the lowest available clock frequency. The frequency is statically set and does not change. Using this governor will yield maximum power savings, but at the cost of performance. This governor is the complementary governor to the *performance* governor.



THE ONDEMAND GOVERNOR

This governor is a dynamic governor. It is the best of both worlds; it allows the system to achieve maximum performance if the system load is high and allows the system to achieve maximum power savings if the system is idle. The *ondemand* governor will step from one frequency to the next with respect to system load. The downside of this governor is latency. The system requires some time to ramp up or down in clock frequency.

THE CONSERVATIVE GOVERNOR

This governor is a dynamic governor. It is similar to the *ondemand* governor; however it attempts to change speed more gradually. The *conservative* governor will gracefully step from one frequency to the next with respect to system load, whereas *ondemand* will jump much more aggressively to the best frequency to match the load. Expect even more latency with the *conservative* governor than with the *ondemand* governor, but also you'll also save even more power. This governor is very well suited for battery-powered systems, such as laptops.

THE USERSPACE GOVERNOR

This governor allows userspace or any process running as root to specifically set the frequency manually. This governor is often, but not always, used in conjunction with the *cpuspeed* daemon. The *userspace* governor allows the user to define policy. It is the most customizable of any of the governors. It may save even more power, or yield even more performance, but this is largely dependent on how it is configured.

EVERYONE USES POWER DIFFERENTLY

Not everyone uses power the same way. Each of us uses the hardware a little differently, and how a computer is used largely defines how power is consumed. One computer may be used as a workstation, surfing the net, or using a spreadsheet. That computer might sit idle most of the time with some occasional and brief spikes of activity. In a case like this, a very large amount of power can be saved by reducing CPU frequency during the idle periods. Another computer might be a server in a corporate enterprise's server room. This machine might be idle all night, but when the business day begins, workload jumps exponentially and stays high until the close of the business day. A very large amount of power can be saved by reducing CPU speed overnight. Evaluating how a system is used best determines which governor will be best for your situation. Keep in mind the potential disadvantages of using one governor over another. There is no one-size-fits-all governor. At best, there are one-size-fits-most scenarios.

DECIDING WHICH GOVERNOR IS BEST

So which governor would be best for your system? It all depends on what the system is used for. Let's explore some common scenarios and see what each governor might offer.

WORKSTATION USER

The average workstation user is not absolutely concerned with power usage. Saving power might seem interesting, but if the system is plugged into a wall outlet, there are an infinite supply of electrons. If you could save energy without consciously thinking about CPU frequencies, then why not save a little or a lot



of energy? The *ondemand* governor just happens to be perfect in a situation such as this. It is completely automatic. The computer just picks the best CPU speed for any given system load and saves power whenever possible. This is a great governor to use because you never even have to think about it. It just works, and you may even forget it is there. But still you will save power.

LAPTOP USER

Laptop users are much more inclined to worry about power usage. Once the battery charge is depleted, recharging is necessary, and sometimes you can't recharge.

There are a few governors that might make life a little easier for laptop users. If you are most often using AC power and rarely running on battery, the *ondemand* governor is a good solution. You will get performance when you need it, save power when you can, and never have to think about it.

Conversely, the *conservative* governor might be more optimal if you are running on battery more of the time. It will save even more power or battery life than the *ondemand* governor. The cost of using this governor is that you have greater latency. The system takes a little longer to increase CPU frequency, but the benefit of choosing this governor is maximized battery life.

Now, you may also be interested in taking a more hands-on approach. The *performance* and the *powersave* governors can also be used. When you plug into AC power, select the *performance* governor manually and have the maximum CPU speed at your disposal at any given moment. When you unplug from AC power and go on battery, you can then manually select the *powersave* governor. The system may seem a bit more sluggish at moments but will save a very large amount of power. Because battery capacities increase with a smaller draw of current and decrease with a larger draw of current, the system may stay up longer with the *powersave* governor than with any other governor.

Consider, for example, that a battery can deliver more power if it is discharged more slowly. A battery might deliver 2.25 amp-hour (AH) if drawn over one hour or 2.50AH if drawn over a period of five hours. Using the *powersave* governor will minimize the draw on the battery. You will get a larger percentage of the charge you put into your battery with this governor. But since this governor is statically set to the minimum CPU speed, it is possible you may get less done – though not likely. Keep in mind that many systems used as workstations sit idle the majority of the time, but it all depends on how you use your computer.

SERVER USER

Servers often consume more power than any other type of system. Servers usually have a larger number of buses, peripherals, and processors. There is huge potential for power savings on multiprocessor systems with CPU speed scaling. Not only does each processor consume energy, but each makes a lot of heat. The BTU output of some large multiprocessor systems is enough to heat a small house. So imagine how much power can be saved in a server room with hundreds of systems turning lots of energy into lots of heat.

The economic benefits of using CPU speed scaling can be exponential, but the governor must be selected carefully. While choosing the right governor can save you a lot of money, choosing the wrong governor can have a negative impact. Governor selection should be per-machine, based on what the system is used for, and it is unlikely you would pick one governor for your whole infrastructure. The biggest caveat of using an automated governor, like the *ondemand* governor, is latency. Some systems might not be affected by latency, like perhaps a system that does some type of batch processing that isn't time sensitive. In this case,



the *ondemand* governor would be a perfect fit. However if you are a trading house on Wall Street, latency is an important issue. Fortunately, there are ways of saving power no matter what your scenario. If latency is a concern, the *performance* and *powersave* governors should be used instead. Use the *performance* governor during business hours, and the *powersave* governor after hours to save power overnight.

CUSTOM USER

You are a custom user if none of the current governors completely suits your specific needs. If you need to define your own custom policy to specify how you want your governor to behave, the *userspace* governor and a management daemon such as the *cpuspeed* daemon should be used. The *userspace* governor allows you to automate and tweak your system's policy above and beyond what would otherwise be possible with one of the standardized governors.

For example, if using the *cpuspeed* daemon, it is possible to configure your system to automatically select the maximum processor speed if you are on AC wall power, then to automatically select the minimum processor speed when switching to battery power. Policy may also be set based on thermal parameters. If your system's temperature goes too high, you can decrease the processor speed until the temperature drops. Additionally, the *userspace* governor allows userspace to make changes, so it is possible to write your own application or scripts to accomplish exactly what you need to do. This provides a high degree of flexibility to meet almost any requirement, while integrating well into unique and highly proprietary IT frameworks.

MAKING IT HAPPEN

First, make sure the governor you wish to use is available. The CPUfreq driver status information is available through the */sys* filesystem and is located in */sys/devices/system/cpu/*. There is a separate directory for each CPU. For the sake of simplicity, we will use a uniprocessor system as our example. If your system is a multiprocessor system, you will need to repeat each change you make for every processor on the system.

EXAMPLE 1

First, see which governor your system is currently using:

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

This will report which governor is currently active.

EXAMPLE 2

Next, find out which governors are currently available on your system:

```
→ cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_governors
```

This will give you a list of the governors available for you to use. You can use any governor listed. You might wish to use a governor and not see it listed. Keep in mind that some governors are not built into the kernel and are instead available as modules. If this is the case, you will need to "modprobe cpufreq_ondemand" or "modprobe cpufreq_powersave", etc. You should then see the governor you wish to use in */sys/devices/system/cpu/cpu0/cpufreq/scaling_available_governors*



EXAMPLE 3

Select the governor that will be the current active governor:

```
→ echo ondemand > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

In this case, the ondemand governor has been selected. Any of the available governors that appear in `/sys/devices/system/cpu/cpu0/cpufreq/scaling_available_governors` can be used as the current governor.

EXAMPLE 4

Selecting governors can be done manually at the command line or with your favorite CPU frequency scaling applet for your window manager. The governor can be selected via a pull-down menu. This works well if you are using a laptop, but what about on a server? You can write scripts or cron jobs to do this for you as well.

8:00 a.m. cron job:

```
→ echo performance > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

6:00 p.m. cron job:

```
→ echo powersave > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

In this case, power is being saved during the off hours while providing maximum performance during business hours and also avoiding the potential latency issues associated with the automatic governors. There are also more creative approaches like using scripts to drive these changes from your batch manager when there is work to be done or other approaches to integrate control within a custom IT infrastructure. This is entirely scriptable and easily integrated.

MORE INFORMATION

CPUfreq has a lot of flexibility, far more than the scope of this whitepaper. For a better understanding of the full potential of the CPUfreq driver, its governors, the cpuspeed daemon, and all the configurable settings, review the official documentation at [Documentation/cpu-freq/](https://www.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/working_with_cpu_frequencies/).

RED HAT SALES AND INQUIRIES

US/Canada
1 800 RED HAT 1
www.redhat.com