NAME

cpufreqd.conf − configuration file for cpufreqd(1)

DESCRIPTION

cpufreqd.conf is a simple text file containing rules to be used by **cpufreqd(1).**

cpufreqd.conf is divided into sections enclosed in tags (eg.: [General][/General]). You need at least one [General] section and one or more [Profile] and [Rule] sections. Each [Rule] depends on previously defined [Profile] subsections. Some plugins also require a proper configuration section.

Some notes to better understand how to write appropriate rules:

− the score of a rule is made up of the percentage of entries that match the current system state as reported by plugins + the number of matching entries. In this way even a single−entry rule can reach 100% but more accurate rules are preferred.

− in case of 2 or more rules having the same score the first one (as found in the configuration file) or the one already applied if applicable is kept and set.

− each directive is handled by a single plugin that will determine if the state described matches the current system state.

− if no rule matches the current system status, no action is performed.

What to keep in mind when writing rules:

− the −V switch is (hopefully) your friend, test your configuration slightly increasing**cpufr eqd** verbosity and look what happens (−V6 will report rules' scores, −V7 will report which entry matched and which not).

− if you want a rule to be preferred over another just describe the system state more accurately.

A good approach to write cpufreqd rules is to first describe the basic parameters you want for a general usage (e.g.: define at least an "AC−on" rule and an "AC−off" rule), then proceed and describe all the special cases by describing the system state more accurately (e.g.: "AC−off but running mplayer" or "AC−on but temperature too hot").

Note that no white space is allowed between name and value pairs. Characters after a '#' are considered comments and ignored.

SECTIONS

Acceptable configuration tokens and values include:

[General]

poll_interval

A float larger than 0.15, measures the interval between system status reading in seconds. Note: the lower bound has been set in order to try to avoid trashing your system if using a too low value. (default: 1.0)

enable_plugins

A list of plugins separated by comma. As of cpufreqd 2.1.0 this option is useless, cpufreqd will load all available plugins and remove those remaining unused after the configuration file is read.

pidfile Specifies the file to write as its process identification file. (default: /var/run/cpufreqd.pid)

enable_remote
Make cpufreqd open a local UNIX socket and listen for command to be executed. See cpufreqd-set(1) and cpufreqd-set(1) for two very simple clients.

remote_group
Make the socket readable and writeable to the specified group. Useful to allow simple users to tweak cpufreqd with cpufreqd-set and cpufreqd-get.

double_check
Make cpufreqd check if the requested policy has been correctly applied by re-reading the corresponding kernel attributes.

verbosity
Verbosity level from 0 (less verbose) to 7 (most verbose), the default value only prints warning/error/critical messages. (default: 4)

[Profile]

name An arbitrary and unique name for your profile. [REQUIRED]

minfreq
An integer value representing the minimum frequency to set in /proc/cpufreq. This value can be both a percentage of the CPU full capacity or frequency in kHz. [REQUIRED]

maxfreq
An integer value representing the maximum frequency to set. This value can be both a percentage of the CPU full capacity or frequency in kHz. [REQUIRED]

policy Can be any of the available governor's name as shown in /sys/devices/.../cpufreq/scaling_available_governors, this means that if you compiled governors as modules in your kernel, you need to load them before running cpufreqd. [REQUIRED]

other plugin entries
Other Profile directives are available according to the enabled plugins.

[Rule]

name An arbitrary and unique name for your rule. [REQUIRED]

profile A character string that must match a [Profile] section name property. A Rule can also associate profiles to single cpus providing a list of the format CPU%d:%s separated by semicolons (";"), e.g.: profile=CPU0:profile0;CPU1:profile1. The keyword "ALL" can be used to indicate that all cpus must have the profile applied. The "ALL" keyword has a lower priority so you can mix up CPU%d and ALL meaning that if no specific profile is supplied, the "ALL" one will be used. [REQUIRED]

other plugin entries
Other Rule directives are available according to the enabled plugins.

PLUGINS

Plugins extend cpufreqd in order to be able to cope with the most exotic system paramters. Currently available plugins are listed below along with the configration directives they provide and their configuration section description if available.

acpi plugin

This plugin includes all the acpi monitoring functionalities previously available as separate plugins. It allows monitoring battery, temperature, ac and interacting with acpid to immediately react on ACPI events.

Section [acpi]

acpid_socket The path to Acpid open socket (default: /var/run/acpid.socket). When available cpufreqd will wake up immediately upon event arrival and battery and ac status updates are forced.

battery_update_interval

The number of seconds that have to elapse before polling the battery again. In such period the battery value will be estimated based on reported power consumption (default: 30).

battery_interval

The rule will have a higher score if battery percentage is between the values provided. Can be of the form %d-%d or simply %d for a fixed value (e.g.: battery_interval=10-100) or %s:%d-%d or %s:%d where the string represents the battery name that must match (look at 'ls /proc/acpi/battery' for available names).

ac Can be on or off. The rule will have a higher score if the A/C adapter is on or off as defined in this setting.

acpi_temperature

The rule will have a higher score if the temperature percentage corresponds to the provided values. Can be of the form %d-%d or simply %d for a fixed value (e.g.: acpi_temperature=10-100) or %s:%d-%d or %s:%d where the string represents the thermal zone name that must match (look at 'ls /proc/acpi/thermal_zone' for available names).

apm plugin

Monitors values reported by the APM subsystem. Available Rule entries:

ac Can be on or off. The rule will have a higher score if the A/C adapter is on or off as defined in this setting.

battery_interval

The rule will have a higher score if battery percentage is between the values provided. Must be of the form %d-%d (e.g.: battery_interval=10-100).

pmu plugin

Monitors values reported by the PMU subsystem. Available Rule entries:

ac Can be on or off. The rule will have a higher score if the A/C adapter is on or off as defined in this setting.

battery_interval

The rule will have a higher score if battery percentage is between the values provided. Must be of the form %d-%d (e.g.: battery_interval=10-100).

tau plugin

Support for the Thermal Assist Unit to read the CPU temperature from /proc/cpuinfo.

tau_temperature

The rule will have a higher score if the temperature is between the values provided. Must be of the form %d-%d (e.g.: tau_temperature=30-60).

cpu plugin

Monitors the cpu usage. Available Rule entries:

cpu_interval

The rule will have a higher score if cpu usage is between the values provided. Must be of the form %d-%d (e.g.: cpu_interval=10-100) or %d:%d-%d to monitor a specific cpu in SMP/SMT systems (e.g.: cpu_interval=1:50-100). Moreover you can combine multiple cpus giving multiple intervals on the same line separated by semicolon (';'), if any of them matches the full directive will match (e.g.: cpu_interval=0:50-100;1:0-60). Additionally you can use the strings "ALL" and "ANY" to request that all cpus or any cpu matches respectively (e.g.: cpu_interval=ANY:50-100). It is possible to specify the scale to calculate niced processes cpu usage with the form %d-%d,%f or %d:%d-%d,%f (e.g.: cpu_interval=1:70-100,1.5), default is 3, in this way niced processes will be considered 1/3 of their real value. Rules with overlapping cpu_intervals are allowed.

exec plugin

Executes command on Rule/Profile selection. Available Rule and Profile entries:

exec_pre

exec_post

You can give commands that you want to be executed when a Rule or Profile is applied. As the names suggest, **exec_pre** will be run before a Rule or Profile is applied, **exec_post** will be run after.

programs plugin

Monitors active processes. Available entries:

programs

The rule will have a higher score if one of the listed processes is running. This is a comma separated list. No white space is allowed between values. cpufreqd will try to match each process name with the configured process list. If you need to match against program from a spe- cific location you have to supply the full path as search pattern.

nforce2_atxp1 plugin

Allows you to change Vcore of the CPU on the fly if you own a NForce2 board with atxp1 voltage regulator (and its module loaded). The use of this plugin will allow a new Profile directive and requires a configuration section.

Section [nforce2_atxp1]

vcore_path Defines the interface file created by atxp1 module which will be used to change Vcore.

vcore_default As NForce2 boards only initialize the atxp1 on power-on, you need to put back default Vcore before reboot. This value will be used to set Vcore on exit.

vcore Will set Vcore to this value (given in mV) when the corresponding Profile is applied. Due to safety reasons range is limited from 1200 to 1850.

nvclock plugin

Allows you to tweak the core an memory clock for NVidia cards. The use of this plugin will allow new Profile directives. **NOTE: you MUST use this plugin ONLY with supported cards.** See also the nvclock homepage (http://www.linuxhardware.org/nvclock).

nv_core

Sets the core clock in MHz. Must be of the form %d:%d where the first integer represents the card number, the second the desired frequency in MHz.

nv_mem

Sets the memory clock in MHz. Must be of the form %d:%d where the first integer represents the card number, the second the desired frequency in MHz.

sensors plugin

Allows you to specify lm-sensors features to watch, see 'sensors −u' to find out which sensors are available on your system. A configuration section is also available to tell cpufreqd which sensors.conf file to use. If not specified it will take the first on the default locations.

Section [sensors_plugin]

sensors_conf Define this directive to the sensors.conf file you want cpufreqd to use to load the sensors library.

sensor The rule will have a higher score if the given sensor feature reports a value between the two defined. Must be of the form %s:%f-%f where the string represents the feature name or label and the two decimal numbers the interval into which the directive is valid (e.g.: sensor=temp1:0-50).

governor_parameters plugin

Allows you to specify parameters for governors in [Profile] sections. Currently only the 'ondemand' and 'conservative' governors support parameters. The description of the parameters below is basically a summary of the information found in the file 'governors.txt' in the documentation of kernel versions 2.6.16 or later.

sampling_rate

How often the governor checks the CPU usage. Specify in micro-seconds or percentage of mimimum and maximum available values. Supported suffixes: **'%'** for a percentage, **'s'** for a value in seconds, **'m'** for a value in milli-seconds, or **'u'** for a value in micro-seconds (the default),

up_threshold

What the average CPU usage needs be at least to make the governor decide to switch to a higher frequency. Though the value is interpreted as percentage by the governor, you should not append a '%' in cpufreqd.conf for this parameter.

down_threshold ('conservative' governor only)

What the average CPU usage needs be at most to make the governor decide to switch to a lower frequency. This is the opposite of**up_thr eshold** (see above).

sampling_down_factor

How quickly the frequency will be decreased in respect to how quickly it will be increased. E.g. when set to 5, the frequency will go down 5 times 'less easy' than it will go up.

ignore_nice, ignore_nice_load

Whether 'nice' processes should be considered as CPU usage by the governor. This is a boolean value (e.g. value is either 0 or 1). When set to 1 'nice' processes will not be counted as CPU usage by the governor. **Note:** 'ignore_nice' was renamed to 'ignore_nice_load' in kernel version 2.6.16. Both names are accepted in cpufreqd.conf, regardless the version of the running kernel.

freq_step ('conservative' governor only)

How much the frequency should be changed (up or down) each time the governor decides the frequency should go up or down. The value is the percentage of the maximum available frequency you want the frequency to increase or decrease each time. The actual frequency your CPU runs at will only change when the desired frequency reaches the next available frequency. Though the value is interpreted as percentage by the governor, you should not append a '%' in cpufreqd.conf for this parameter.

EXAMPLE

```
# cpufreqd.conf sample
# this is a comment
[General]
pidfile=/var/run/cpufreqd.pid
poll_interval=2
verbosity=5 #(if you want a minimal logging)
[/General]

[Profile]
name=hi
minfreq=100%
maxfreq=100%
policy=performance
[/Profile]

[Profile]
name=medium
minfreq=66%
maxfreq=66%
policy=performance
[/Profile]

[Profile]
name=lo
minfreq=33%
maxfreq=33%
policy=performance
[/Profile]

[Profile]
name=ondemand_hi
minfreq=0%
maxfreq=100%
policy=ondemand
[/Profile]

[Profile]
name=ondemand_lo
minfreq=0%
maxfreq=66%
policy=ondemand
ignore_nice=1
sampling_rate=80%
[/Profile]

# full power when AC
# max score 101%
[Rule]
name=AC_on
ac=on
profile=hi
[/Rule]
```

```
# conservative mode when not AC
# max score 101%
[Rule]
name=AC_off
ac=off
profile=ondemand_hi
[/Rule]

# low battery
# max score 102%
[Rule]
name=lo_battery
ac=off
battery_interval=0-40
profile=ondemand_lo
[/Rule]

# need big power (not if battery very low)
# max score 103%
[Rule]
name=hi_cpu
ac=off
battery_interval=40-100
cpu_interval=ANY:70-100
profile=hi
[/Rule]

# slow down a little if overheated
# max score 103%
[Rule]
name=overheat
acpi_temperature=55-100
cpu_interval=ANY:0-100
battery_interval=40-100
profile=medium
[/Rule]

# full power when watching DVDs and not AC
# can reach a 105% score
[Rule]
name=dvd_watching
ac=off
battery_interval=0-100
acpi_temperature=0-100
cpu_interval=ANY:0-100
programs=xine,mplayer
profile=hi
[/Rule]
```

SEE ALSO

cpufreqd(8),**cpufreqd-set**(1),**cpufreqd-get**(1)

AUTHOR

Mattia Dongili <malattia@linux.it>

George Staikos <staikos@0wned.org>