



## lm\_sensors, contrôle et monitoring

Le 25 juillet 2008, à 0:56 par Ulhume...

Rapidement fatigué de l'inesthétisme de mon ancien boîtier dans notre nouvel intérieur, et attiré par la magie d'avoir enfin un garage entier pour bricoler, j'ai décidé de transformer un de nos meubles de salon en boîte à serveur. Le problème, c'est que le bois ça fait aussi bien sonner les belles guitares que rendre assourdissant les ventilateurs... Alors retour en atelier, installation de suspensions pour les disques, et de ventilateurs 12cm à basse vitesse de rotation. Ne restait donc que ce maudit ventilateur de CPU qui moulinait comme un malade.

### lm\_sensors

Il se trouve que c'est le même outil qui permet à la fois de lire les senseurs internes de la bécane et de contrôler certains éléments comme la vitesse de rotation des ventilateurs. Et cet outil c'est [lm\\_sensors](#) [1]. Pour l'installer, l'enfance de l'art, il est forcément dans votre dépôt. Donc `urpmi/apt-get` selon vos tendances.

Pour le configurer il faut lancer `sensors-detect` en tant que root. L'outil va tester toutes les combinaisons bus/puce pour terminer par écrire le paramétrage de ce qu'il aura détecté. Dans le cas de ma Gigabyte P35 - Q6600, il a détecté une puce SuperIO ITE IT8718F et les 4 senseurs de température des 4 coeurs du processeur.

Ceci fait, un nouveau service est alors utilisable qui se lance par `/etc/init.d/lm_sensors start`. Cela va charger les modules correspondant à ce qui a été trouvé, dans mon cas `it87` et `i2c_i801`. Ensuite, pour lire les valeurs des capteurs, vous devez lancer la commande `sensors`. Apparaissent alors 5 sections, correspondant à ce qui a été trouvé par `sensors-detect` :

```
root# sensors
it8718-isa-0290
Adapter: ISA adapter
in0: +1.12 V (min = +0.00 V, max = +4.08 V)
in1: +1.90 V (min = +0.00 V, max = +4.08 V)
in2: +3.36 V (min = +0.00 V, max = +4.08 V)
in3: +2.99 V (min = +0.00 V, max = +4.08 V)
in4: +0.26 V (min = +0.00 V, max = +4.08 V)
in5: +0.05 V (min = +0.00 V, max = +4.08 V)
in6: +0.10 V (min = +0.00 V, max = +4.08 V)
in7: +3.10 V (min = +0.00 V, max = +4.08 V)
in8: +3.26 V
fan1: 1523 RPM (min = 0 RPM)
fan2: 0 RPM (min = 0 RPM)
fan3: 0 RPM (min = 0 RPM)
fan4: 0 RPM (min = 0 RPM)
temp1: +51°C (low = +127°C, high = +127°C) sensor = thermistor
temp2: +34°C (low = +127°C, high = +127°C) sensor = diode
temp3: -2°C (low = +127°C, high = +127°C) sensor = thermistor
vid: +0.513 V
coretemp-isa-0000
Adapter: ISA adapter
```

```
Core 0: +43°C (high = +100°C)
coretemp-isa-0001
Adapter: ISA adapter
Core 1: +46°C (high = +100°C)
coretemp-isa-0002
Adapter: ISA adapter
Core 2: +42°C (high = +100°C)
coretemp-isa-0003
Adapter: ISA adapter
Core 3: +42°C (high = +100°C)
root#
```

Sur le fond, `lm_sensors` est un très bon outil, le problème vient plus de la difficulté à le paramétrer. En effet, si vous obtenez une température de CPU de 150°C, ne jetez pas tout de suite votre jus d'orange pour stopper la combustion, c'est sûrement un problème d'échelle. Par exemple dans mon cas, je doute fort qu'il y ait un point sur la carte mère à -2°C... Pas évident de faire son tris là dedans.

Le problème vient du fait que les constructeurs ne jouent pas le jeu (oh !), genre ASUS, au hasard, qui ne publie pas les datasheet de ses composants. Du coup les calibrages se font au doigt mouillé. J'ai par exemple un ventilateur sur une A7N8X qui est censé tourner à 88730 rpm... Rassurez-vous, j'ai du ancré le PC au sol 😊

Il faut donc aller trifouiller dans `/etc/sensors.conf` pour améliorer un peu les choses. Le meilleur moyen que j'ai trouvé est de redémarrer le PC, noter les valeurs données par le BIOS et opérer les ajustements de retour sous Linux. On peut aussi utiliser un thermomètre infrarouge qui permettra de faire cela avec plus de précision.

## sensors.conf

Pour bien comprendre comment fonctionne le fichier `sensors.conf`, il faut déjà saisir d'où viennent les valeurs qu'il manipule.

Comme nous l'avons vu plus haut, la fonction `sensors-detect` va trouver une série de puces. Chaque puce est reliée à une série de capteurs ou de contrôles. Dans le cas de ma Gigabyte, l'outil a trouvé 5 puces, 4 correspondant à la température de chacun des cœurs du CPU (`coretemp-isa-XXXX`), et un pour le reste des valeurs (`it8718-isa-0290`). Pour ces deux catégories de puce, nous avons deux pilotes kernel correspondant : `coretemp` et `it87`.

Une fois montés en mémoire, ces pilotes vont chacun créer des entrées dans le dossier `/sys/class/hwmon`. Dans ce dossier nous allons trouver 5 sous-dossiers `hwmon0` à `hwmon4` : 4 pour `coretemp` et 1 pour `it87`.

Dans chaque dossier `hwmon0-4`, nous avons un sous-dossier `device`. Et dans ce dossier nous avons quelque chose de genre :

```
root# ls /sys/class/hwmon/hwmon0/device/
alarms fan4_min in3_input in6_min pwm2 temp2_input
bus@ hwmon:hwmon0@ in3_max in7_input pwm2_enable temp2_max
cpu0_vid in0_input in3_min in7_max pwm2_freq temp2_min
driver@ in0_max in4_input in7_min pwm3 temp2_type
fan1_input in0_min in4_max in8_input pwm3_enable temp3_input
fan1_min in1_input in4_min modalias pwm3_freq temp3_max
fan2_input in1_max in5_input name subsystem@ temp3_min
```

```

fan2_min in1_min in5_max power/ temp1_input temp3_type
fan3_input in2_input in5_min pwm1 temp1_max uevent
fan3_min in2_max in6_input pwm1_enable temp1_min vrm
fan4_input in2_min in6_max pwm1_freq temp1_type
root#

```

Comme vous le voyez, tout y est. D'abord la vitesse des ventilateurs ( `fanX_input` ) et la valeur minimum à ne jamais atteindre ( `fanX_min` ). Les `inX_input` représentent des tensions avec une valeur min ( `inX_min` ) et max ( `inX_max` ). Enfin les températures ( `tempX_input` ), et la aussi des valeurs min ( `tempX_min` ) et max ( `tempX_max` ). Enfin, nous le verrons un peu plus loin, les `pwmX` nous permettrons de changer la vitesse de rotation des ventilateurs.

Donc si lors de l'exécution de la commande `sensors`, vous aviez une vitesse affiché pour `In1`, vous pouvez avoir cette vitesse en tapant :

```

cat [2] sys class hwmon hwmon0 device in1_input

```

La seule chose est que la valeur retournée n'a généralement que peu de rapport avec celle de `sensors`. C'est tout l'intérêt de `sensors.conf`, transformer ces valeurs brute en réalité compréhensible. Dans le vocabulaire `lm_sensors`, `in1_input`, `fan3_min`, `pwm2`, etc, sont des fonctionnalités (features). Ce sont ces fonctionnalités que nous allons travailler dans notre fichier de configuration. Le fichier `sensors.conf` est composé de directives :

- **chip** "nom\_1" "nom\_2" ... "nom\_n", par exemple `chip "it87-*" "it-8712-*"`. Cette directive indique que tout ce qui suit jusqu'à la prochaine directive `chip` ou la fin du fichier, s'applique aux puces dont le nom commence par `it87-` ou `it-8712`. Le nom de la puce est celui affiché lorsque vous lancez `sensors`.
- **ignore** fonctionnalité, par exemple `ignore temp3`. Comme son nom l'indique, le but de cette directive est d'indiquer à `lm_sensors` que la fonctionnalité est à zapper car généralement elle n'est connecté à rien du tout.
- **label** fonctionnalité "description", par exemple `label temp1 "Température du CPU"`. Ici le but est de donner aux fonctionnalités un nom un peu plus sexy que celui choisi par le pilote.
- **set** fonctionnalité expression, par exemple `set temp1_over 60`. Ici il s'agit de donner à des valeurs du système, typiquement le min et le max d'un capteur. Il est possible de mettre de petites expressions mathématiques qui peuvent même inclure des noms de capteurs. Faites cependant attention à ne pas gaffer aux références circulaires... Pour que ces valeurs soient prises en compte dans les alarmes, il faut soit lancer `sensors -s`, soit redémarrer le service `lm_sensors`.
- **compute** fonctionnalité expression\_out expression\_in, par exemple `compute in3 ((6.8/10)+1)*@ , @/((6.8/10)+1)`. Cette directive définit deux expressions permettant respectivement de transformer la valeur brute venant du capteur en valeur lisible, et inversement. Le `@` représente la valeur brute dans le premier cas, et une valeur lisible dans le second. C'est ici que tout le jeu se passe pour arriver à trouver la formule la plus proche possible de la réalité. En effet, les puces sont construites pour lire un interval fixe de tensions, par exemple 0 à 5v. Donc pour pouvoir mesurer la tension 12v, le constructeur de la carte mère va ajouter des diviseurs de tension. C'est ce type de magouille que les formules cherchent à caractériser.

Comme vous le voyez, il est relativement simple de modifier ce fichier de configuration, un peu moins d'arriver à obtenir les bonnes valeurs si les formules par défaut ne collent pas...

Prenons le cas de ma puce `it8718-isa-0290`. Déjà, je ne trouve pas de directive `chip` qui lui correspond. En revanche, j'ai bien un `chip "it87-*" "it-8712-*"`. Comme je sais que l'`it8718` fait

peu ou prou la même chose que l'ensemble de la gamme `it87`, je vais donc simplement rajouter ma puce à la liste `chip "it87-*" "it-8712-*" "it-8718-*"`. Ensuite je vais faire un peu de ménage sur les valeurs non pertinentes par `ignore`, renommer proprement les fonctionnalités avec `label` et bidouiller deux trois formules pour coller avec la réalité. J'ajouterais enfin des directives `set` pour fixer les valeurs min et max de certaines valeurs critiques (arrête du ventilateur, CPU en fusion, etc.).

Au final, on obtiens un résultat de `sensors` le plus juste possible. Ensuite il existe de nombreux enrobages graphiques à `lm_sensors` comme par exemple le [gnome-applet-sensors](#) [3] me permettant de remonter tout cela sur mon bureau.

Si vous voulez prendre en charge les alarmes, faites un tour sur `man sensord`. `sensord` est le démon lancé par le service `lm_sensors` qui va auditer toutes ces valeurs et vous prévenir si quelque chose part en vrille. En revanche si cette fonctionnalité ne vous intéresse pas, ou si vous monitorisez ces valeurs via nagios, il peut être malin d'empêcher le lancement de ce démon en torpillant le fichier `/etc/init.d/lm_sensors`.

## Contrôle des ventilateurs



S'amuser à arrêter un ventilateur sur un processeur en activité est le meilleur moyen pour le faire griller. Ne faites jamais cela plus de 5 secondes, et seulement si vous savez ce que vous faites.

Maintenant que les valeurs sont au point, il nous reste toujours à réduire la vitesse de nos ventilateurs & bien sur la carte mère le permet. Comme nous l'avons vu plus haut, ce sont les fonctionnalités `pwmX` qui sont responsables de ce comportement. Basiquement pour activer le contrôle il faut envoyer une valeur 1 au `pwmX_enable` qui va bien, et ensuite une valeur de 0 (arrêt) à 255 (fullspeed) à `pwmX`. En somme, si l'on connaît le bon `pwm`, réduire de moitié la vitesse d'un ventilateur se résume à

```
echo 1 sys class hwmon hwmon0 device pwm3_enable
echo 127 sys class hwmon hwmon0 device pwm3
```

Après le jeu est donc de trouver le bon `pwm` associé au ventilateur que l'on cherche à modérer. Vous pouvez faire cela à la main, ou utiliser `pwmconfig` qui est fait pour cela. Ce dernier pour chaque `fanX` activé, va tester les `pwmX` les uns après les autres jusqu'à trouver celui qui fonctionne. Une fois que vous avez le numéro, vous pouvez faire le reste à la main.

Vous pouvez aussi laisser `pwmconfig` poursuivre, il vous proposera alors de calibrer la valeur de `pwm` avec la vitesse obtenue sur le `fanX` correspondant. Ceci fait, il vous permettra de paramétrer un démo n appelé `fancontrol` que vous pourrez utiliser pour ajuster automatiquement la vitesse avec la température du processeur.

## Et le disque dur ?

La température du disque dur ne fait pas parti du domaine de compétence de `lm_sensors` mais de celui du système `S.M.A.R.T.`. Cette norme de contrôle des disques durs introduite par IBM permet de remonter de l'unité une foulditude d'informations permettant de prévenir les avaries. Et l'une de ces informations est la température. Pour la lire vous avez deux solutions, soit l'artillerie lourde avec

et la commande `smartctl -A /dev/sda`, ou alors plus simple avec l'utilitaire `hddtemp` et la commande `hddtemp /dev/sda`.

## Conclusion

Fin du petit tour d'horizon de ce qu'il est possible de faire avec `lm_sensors` et début de la zone de silence 😊

### Liens:

[1] <http://www.lm-sensors.org/>

[2] <http://pwet.fr/man/linux/commandes/cat>

[3] <http://sensors-applet.sourceforge.net/>