

screen, pour lancer de longs programmes et pouvoir les suivre

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 18 Mai 2008

<http://www.bortzmeyer.org/screen.html>

Sur une machine Unix, on a souvent envie de lancer des programmes qui durent longtemps, parfois des jours ou des semaines, et qu'on souhaite regarder de temps en temps, sans pour autant bloquer un terminal (d'autant plus qu'on travaille depuis différents postes). screen est un outil assez peu connu pour assurer cette tâche. Il permet de suivre un programme en cours d'exécution, puis de se **détacher** de lui tout en le laissant tourner et de se **rattacher** ensuite à volonté.

Si le programme est prévu pour tourner sans surveillance, directe, si c'est un démon, screen n'est pas nécessaire. Mais si on lance une grosse compilation sur sa machine Gentoo ou FreeBSD ? Si on a lancé un client BitTorrent pour télécharger des fichiers et qu'on veut voir de temps en temps où il en est ? Si c'est sur la machine de son bureau, on peut toujours laisser un xterm ouvert. Mais si c'est sur une machine dédiée hébergée chez OVH ou Slicehost (<http://www.bortzmeyer.org/slicehost-debut.html>) ?

Avec la solution du xterm ouvert, je ne peux pas regarder dans la journée au bureau et le soir à la maison. Même chose avec des trucs comme nohup. Avec screen, par contre, je peux lancer le programme, regarder s'il tourne, me **détacher** du programme (screen continuera à tourner, même sans terminal, et le programme avec lui) puis plus tard me **ré-attacher**, depuis une toute autre machine.

Les trois commandes de base avec screen dont donc :

- screen pour lancer une nouvelle session. On peut aussi taper screenMONPROGRAMME pour lancer le programme immédiatement (screen se terminera lorsque le programme MONPROGRAMME se terminera).
- Control-ad (par défaut, mais c'est modifiable) pour se détacher.
- screen-r pour reprendre une session en cours.

Personnellement, j'utilise aussi beaucoup l'option -s qui permet de donner un nom à une session screen. screen-list permet ensuite d'afficher les sessions et screen-rNOMSESSION de reprendre celle désirée :

```
~ % screen -S toto
[Ici, on fait un Control-a d]
[detached]
~ % screen -list
There are screens on:
      8116.toto      (Detached)
      3509.BitTorrent (Detached)
2 Sockets in /var/run/screen/S-stephane.
```

(screen permet aussi d'avoir plusieurs fenêtres dans une même session mais je n'utilise pas cette possibilité.)

L'un des intérêts de screen est de surveiller des programmes bavards (comme une grosse compilation) et il peut donc être utile de remonter dans les messages précédemment affichés. `Control-a<escape>` fera passer en mode "Copy" et on pourra chercher en arrière avec `Control-r`.

screen peut se configurer, notamment via son fichier `~/ .screenrc`. Par exemple, certains utilisateurs d'Emacs n'apprécie pas le `Control-a` qui commence les commandes (dans Emacs, cette séquence permet d'aller au début de la ligne) et le reconfigurent dans `.screenrc` :

```
# Control-z, plutôt
escape ^Zz
```

L'excellent site `dotfiles.org` a une section sur des exemples de tels fichiers (<http://dotfiles.org/.screenrc>).

Un dernier truc amusant que `paparoot@paparoot.org` m'a signalé. screen peut aussi s'attacher à un terminal physique et peut donc servir d'émulateur de terminal, de concurrent de minicom. Ici, la machine `preston` est accessible par une liaison série :

```
% screen -S preston /dev/ttyS0
NetBSD/sparc64 (preston) (console)
login:
[Control-A d et on est détaché]
```

J'ai aussi écrit un article sur screen spécifique à BitTorrent (<http://www.bortzmeyer.org/screen-bittorrent.html>). On peut trouver une excellente introduction à screen <http://linuxgazette.net/147/appaiah.html> ou une plus courte en <http://www.mattcutts.com/blog/a-quick-tutorial-on-screen/> ou bien <http://www.redhatmagazine.com/2007/09/27/a-guide-to-gnu-screen/>.