The Linux Keyboard HOWTO

Andries Brouwer, aeb@cwi.nl
v2.1, 8 Novembre 1995 - version française Bruno Viaris

Ce document traite de l'utilisation du clavier, de la console et de caractères non ASCII sous Linux.

Contents

1 Programmes utiles

Les paquetages suivants contiennent des programmes en rapport avec le clavier ou la console.

`kbd-0.90.tar.gz` contient `loadkeys`, `dumpkeys`, `showkey`, `setmetamode`, `setleds`, `setfont`, `showfont`, `mapscrn`, `kbd_mode`, `chvt`, `resizecons`, `disalloc`, `getkeycodes`, `setkeycodes`.

`util-linux-2.5` contient `setterm`, `kbdrate`.

`shellutils-1.8` (ou `stty.tar.gz`) contient `stty`.

`open-1.1.tgz` contient `open`. (Voir aussi `dynamic-vc-1.1.tar.gz`.)

`SVGATextMode-0.9.tar.gz` contient `SVGATextMode`.

La distribution standard de X contient `xmodmap`, `xset`, `kbd_mode`. (Voir aussi X386keybd(1).)

2 Le clavier - généralités

Quand on appuie sur une touche, le contrôleur clavier envoie des scancodes au pilote clavier du noyau. Certains claviers sont programmables, mais en général les scancodes sont fixes. Quand le pilote clavier est en *mode scancode* (dit aussi *mode raw*), le noyau se contente de transmettre les codes tels quels à l'application, c'est ce qui se passe sous `X`. Sinon le flux de scancodes est décomposé en keycodes correspondant à l'appui ou au relâchement d'une touche. (Le simple fait d'appuyer sur une touche peut produire jusqu'à 6 scancodes, réduits ici en un seul keycode.) Si le pilote clavier est en *mode keycode* (ou *mode mediumraw*), l'application reçoit les keycodes (ce mode est utilisé, par exemple, par `showkey`). Sinon, les keycodes sont traduits d'après une table (keymap), ensuite, soit un caractère (ou un chaîne) est envoyé à l'application, soit une action est déclenchée. (Exemple: si on appuie et relâche la touche `q`, le clavier envoie les scancodes 0x1E et 0x9E, ce qui est converti en keycodes 30 et 158, puis en la valeur 113, qui est le code ASCII ou Latin-1 de `q` (avec un clavier Azerty et une keymap adaptée). Si on appuie et relâche la touche `Suppr`, le clavier envoie les scancodes 0xE0 0x53 0xE0 0xD3, convertis en keycodes 111 et 239, ce qui donne finalement la séquence ESC `[ 3 ~`. Ctrl-Alt-Suppr est une combinaison de touches qui déclenche une action.)

La conversion de scancodes exotiques en keycodes peut être programmée avec l'utilitaire `setkeycodes`, mais peu de gens ont besoin. La table de traduction des keycodes en caractères, chaînes ou actions, i.e. la keymap, se manipule grâce à `loadkeys`. Pour plus de détails, voir getkeycodes(8), setkeycodes(8), dumpkeys(1), loadkeys(1).

Dans ce paragraphe, le terme 'envoyé à l'application' signifie en fait 'transmis au pilote du terminal'. Le texte est ensuite traité de la même façon que s'il venait d'une ligne série. Les détails de ce traitement sont ajustés par le programme `stty`.

3 La console - généralités

A l'inverse, quand on envoie des caractères à la console, ils subissent un traitement standard (réglé par `stty`), puis sont envoyés au pilote de la console. Le pilote de la console émule un terminal VT100, et scrute les données reçues pour intercepter les séquences escape (du genre déplacement du curseur, effacement de l'écran, etc.). Les autres octets sont, après une éventuelle conversion par la table de conversion de la console, écrits dans la mémoire vidéo. La carte vidéo affiche ces caractèrs en utilisant une police stockée dans sa mémoire. On peut changer cette police en utilisant `setfont`, et la table de conversion de la console peut être modifiée avec `mapscrn`. Pour plus de détails, voir plus bas.

4 Réinitialiser le terminal

L'écran est plein de cochonneries, tout ce qu'on tape apparaît sous forme de petits dessins. Que faire ?

De nombreux programmes retracent l'écran quand on tape ^L, ce qui peut être utile en cas de modem bruité ou de messages intempestifs à l'écran. La commande `clear` efface l'écran.

La commande `reset` réinitialise le pilote de la console. Elle est bien utile quand l'écran est plein de ces jolis caractères graphiques illisibles, ou quand l'affichage est réduit à la ligne du bas. Si vous ne disposez pas de cette commande, ou si elle fait autre chose, voici le remède: mettez ces deux lignes dans un fichier exécutable `reset` quelque part dans votre PATH:

```
#!/bin/sh
echo -e \\033c
```

ce qui envoie la séquence ESC c à la console. Si vous avez chargé une police bizarre et que vous voulez revenir à la police par défaut,

```
% setfont
```

fera l'affaire (à condition que la police par défaut soit à sa place). Sur de vieux terminaux, l'utilisation de tabs peut demander un délai, exécuter alors

```
% stty tab3
```

(voir stty(1)). Pour changer de mode vidéo, utiliser `resizecons` ou `SVGATextMode`. Ceci règle généralement la partie affichage. Côté clavier, il peut y avoir encore pas mal de problèmes. Si `X`, `DOOM` ou tout autre programme travaillant en mode raw (ou mediumraw) se plante, le clavier peut rester dans ce mode et il est alors difficile d'exécuter la moindre commande. (Voir "Comment sortir du mode raw" plus loin.)

4.1 Keyboard hardware reset

Le problème peut parfois se situer à un niveau encore plus bas. Il y a au moins deux niveaux (le clavier et le contrôleur clavier) où l'on peut envoyer la commande "désactiver le clavier" au matériel. De plus les claviers ont souvent trois tables de scancodes utilisables au choix.

Je n'ai cependant jamais entendu parler de cas où cela ait posé un problème.

Certains claviers ont une fonction de reprogrammation des touches. Stormy Henderson (*stormy@Ghost.Net*) écrit:

> 'Si votre clavier a été accidentellement reprogrammé, vous pouvez (avec un clavier Gateway AnyKey) appuyer sur control-alt-suspend_macro pour remettre les choses en place.'

5 Delete et Backspace

5.1 Comment choisir le caractère à utiliser pour effacer le dernier caractère tapé

```
% stty erase ^?
```

Si le caractère est effacé, mais d'une manière bizarre, c'est que les paramètres du terminal sont mauvais. Si `echoprt` est activé, les caractères effacés s'affichent entre \ et `/`. Si `echoe` est désactivé, les caractères ne sont plus effacés à l'écran, mais leur effacement est signalé par l'affichage du caractère d'effacement, exemple: `Libux###nux` (avec # comme caractère d'effacement).

On utilisera donc généralement `stty echoe -echoprt`. La commande `stty sane` activera ces paramètres et bien d'autres. La commande `stty -a` affiche les paramètres actuels. Pourquoi ces paramètres ne sont pas activés par défaut ? Il suffit d'utiliser le bon `getty`.

Il est à noter que beaucoup de programmes (comme `bash`, `emacs` etc.) ont leur propres assignations de touches (définis dans `~/.inputrc`, `~/.emacs`, etc.) et ne tiennent pas compte de ces paramètres de terminal.

5.1.1 'Getty comprenait bien DEL et BS, mais plus maintenant ?'

Jadis, la console traduisait DEL (`\177`) en BS Espace BS (`\010\040\010`).

Les DEL sont désormais ignorés (comme le devrait toute émulation vt100 fidèle). Choisissez un meilleur getty, i.e., qui n'affiche pas les DEL.

5.1.2 'Login ne fait pas la même chose au premier et au deuxième essai?'

Au premier essai, on parle à `getty`, au deuxième à `login`, deux programmes distincts qui peuvent se comporter différemment.

5.2 Comment dire à Linux quel code générer à l'appui sur une touche

Quand on utilise la console, ou plus précisément, quand le clavier n'est pas en mode (medium)raw, il faut utiliser:

```
% loadkeys monclavier.map
```

Sous X, utiliser:

```
% xmodmap monclavier.xmap
```

Remarque: depuis XFree86-2.1, X initialise son clavier à partir de la configuration du clavier de la console. Bien que les deux systèmes ne soient pas 100% compatibles, cette technique rend généralement l'utilisation de `xmodmap` superflue.

Si, par exemple, si vous voulez que la touche Backspace génère le code BackSpace (ˆH), au lieu du code par défaut (Delete), faites:

```
% loadkeys
keycode 14 = BackSpace
%
```

5.2.1 'Pourquoi la touche BackSpace ne génère pas BackSpace par défaut ?'

(i) Parce que les terminaux VT100 ont une touche Delete au dessus de la touche Entrée.

(ii) Parce que Linus en a décidé ainsi.

5.3 Comment dire à X d'échanger Delete et Backspace

```
% xmodmap -e "keysym BackSpace = Delete" -e "keysym Delete = BackSpace"
```

Ou, si vous voulez juste que Backspace génère le code BackSpace:

```
% xmodmap -e "keycode 22 = BackSpace"
```

Ou, si vous voulez juste que Delete génère le code Delete:

```
% xmodmap -e "keycode 107 = Delete"
```

(mais généralement c'est déjà la disposition par défaut).

5.4 Comment dire à emacs ce qu'il doit faire des codes Delete et Backspace

Mettez dans votre `.emacs` les lignes:

```
(global-set-key "\?" 'delete-backward-char)
(global-set-key "\C-h" 'help-command)
```

Bien sûr, vous pouvez associer d'autres commandes à d'autres touches de la même manière.

5.5 Comment dire à emacs de permuter Delete et Backspace

Mettez dans votre `.emacs` les lignes:

```
(setq keyboard-translate-table (make-string 128 0))
(let ((i 0))
(while (< i 128)
    (aset keyboard-translate-table i i)
    (setq i (1+ i))))
(aset keyboard-translate-table ?\b ?\^?)
(aset keyboard-translate-table ?\^? ?\b)
```

5.6 Comment dire à kermit d'échanger Delete et Backspace

Mettez dans votre `.kermrc` les lignes:

```
set key \127 \8
set key \8 \127
```

5.7 Comment régler xterm selon vos modes de terminal préférés

Normalement xterm hérite des modes de terminal du processus qui l'appelle. Pour `xdm`, les caractères erase et kill sont `#` et `@`, comme sur les bons vieux Unix Version 6. Si ça ne vous plaît pas, vous pouvez mettre

```
XTerm*ttymodes: erase ^? kill ^U intr ^C quit ^\ eof ^D susp ^Z start ^Q stop ^S eol ^@
```

dans `/usr/lib/X11/app-defaults/XTerm` ou dans `$HOME/.Xresources`, à condition d'exécuter la commande

```
xrdb $HOME/.Xresources
```

dans votre `$HOME/.xinitrc`.

5.8 Comment dire à xmosaic que la touche Backspace génère une DEL

Mettre

```
*XmText.translations: #override\n\
       <Key>osfDelete: delete-previous-character()
*XmTextField.translations: #override\n\
        <Key>osfDelete: delete-previous-character()
```

dans votre `$HOME/.Xresources` devrait faire l'affaire.

Cependant, la FAQ de netscape dit:

```
Pourquoi ma touche Backspace ne marche pas dans les boites texte ?
Par defaut, Linux et XFree86 ont leur touches Backspace et Delete
mal configurees. Tous les programmes Motif (y compris Netscape
Navigator) auront les memes problemes.

Les specifications Motif disent que Backspace doit effacer le
```

```
caractere precedent et Delete le caractere suivant. Par defaut sous
Linux et XFree86 Backspace et Delete generent le code DEL.

Le probleme peut etre resolu en utilisant un programme tel que
xmodmap, xkeycaps, ou loadkeys pour que la touche Backspace
genere le code BackSpace au lieu de Delete.

Une autre solution consiste a avoir un fichier .motifbind;
voir la page de manuel de VirtualBindings(3).

Note: n'utilisez pas les ressources *XmText.translations ou
*XmTextField.translations pour essayer de resoudre ce probleme.
Si vous le faites, vous annulez toutes les autres programmations
de touches dans les boites textes de Netscape Navigator.
```

5.9 Et les fichier termcap et terminfo?

Les gens qui ont des problèmes avec backspace ont tendance à aller voir dans leur fichier termcap (ou terminfo), et effectivement la ressource `kb` (ou `kbs`) décrit le code généré par la touche Backspace. Cependant peu de programmes utilisent ces bases de données pour rechercher le code envoyé par backspace, donc, à moins de n'avoir des problèmes qu'avec certains programmes, il faut en général chercher ailleurs. Cela dit, il vaut mieux de toute façon avoir un termcap (ou terminfo) correct. Voir aussi “La variable TERM” plus loin.

6 Les tables de caractères de la console

Le noyau a quatre tables de conversion des octets en symboles écran: a) Latin1 -> PC, b) VT100 graphique -> PC, c) PC -> PC, d) utilisateur.

Il y a deux tables de caractères, appelées G0 et G1, dont l'une est la table courante. (Initialement G0.) Taper ^N fait passer à la table G1, ^O à la table G0.

Ces variables G0 et G1 pointent sur des tables de conversion et peuvent êtres changées par l'utilisateur. Initialement, elles pointent respectivement sur les tables a) et b). Les séquences ESC (B , ESC (0 , ESC (U et ESC (K font respectivement pointer G0 sur les tables de conversion a), b), c) et d). Les séquences ESC) B , ESC) 0 , ESC) U et ESC) K font respectivement pointer G0 sur les tables de conversion a), b), c) et d).

La séquence ESC c provoque une réinitialisation complète du terminal, ce qui peut être nécessaire si l'écran est tout brouillé. La commande `echo ^V^O`, souvent conseillée, ne fera que rappeler la table G0, mais rien ne garantit que G0 pointe sur la table a). Certaines distributions ont un programme reset(1) qui fait juste un `echo ^[c`. Si votre termcap pour la console est correcte (et a une entrée `:rs=\Ec:`), alors vous pouvez utiliser `setterm -reset`.

La table de conversion d) peut être définie par mapscrn(8). Le résultat est que si un symbole c est affiché, c'est le symbole `s = map[c]` qui est mis dans la mémoire vidéo. L'image du caractère `s` se trouve dans la mémoire de la carte vidéo et peut être modifiée avec setfont(8).

7 Changer de console

Par défaut, le changement de console se fait par Alt-Fn ou Ctrl-Alt-Fn. Sous `X` (ou les versions récentes de `dosemu`), seule la combinaison Ctrl-Alt-Fn marche.

XFree86 1.3 ne sait pas que la touche Alt est enfoncée quand on rappelle la fenêtre X. Il faut donc relâcher Alt avant de pouvoir rechanger de console. Pourtant, ça pourrait marcher: le noyau mémorise toujours l'état appuyé/relâché des touches. (Autant que faire se peut: certains clavier n'émettent pas de scancode quand on appuie dessus (ex: les touches PFn d'un FOCUS 9000) ou quand on les relâche (ex: la touche Pause de la plupart des claviers).)

XFree86 1.3 sauve les polices chargées sur la carte vidéo quand il est démarré, et les restaure lors d'un changement de console. Donc le résultat d'un `setfont` sur une console virtuelle est annulé par un aller retour dans la fenêtre X. L'utilisation de `setfont` sous X donne des résultats bizarres...

Le changement de console par programme est possible grâce à la commande `chvt`.

7.1 Changer le nombre de Consoles Virtuelles

Cette question est toujours posée de temps en temps, mais la réponse est: il y en a déjà suffisament. Depuis la version 1.1.54 du noyau, il y a entre 1 et 63 consoles virtuelles. Une nouvelle console est créée dès qu'elle est ouverte. Elle peut être supprimée avec `disalloc` (mais il faut que plus aucun processus ne lui soit associé).

Pour les vieux noyaux, changer la ligne

```
#define NR_CONSOLES     8
```

dans `include/linux/tty.h` (ne pas dépasser 63), et recompiler. Il est possible que vous ayez à créer le périphérique correspondant avec `MAKEDEV` ou `mknod tty`N `c 4` N où N est le numéro du terminal. Si vous voulez avoir des `getty` sur ces nouvelles consoles, ajoutez des lignes dans `/etc/inittab`.

Quand les consoles sont allouées dynamiquement, il est généralement plus simple de n'avoir qu'un ou deux `getty`. D'autres consoles sont ouvertes par `open -l -s bash`. Les consoles inutilisées (sans processus associé) sont désallouées par `disalloc`.

Si vous disposez de `spawn_login` (inclus dans `kbd-0.90.tar.gz`) et que vous mettez

```
loadkeys << EOF
alt keycode 103 = Spawn_Console
EOF
spawn_login &
```

dans `/etc/rc.local`, taper Alt-FlècheHaute créera (et affichera) une nouvelle console avec un `login`. Voir aussi `open-1.3.tgz`.

Vous ne pouvez vous loger sous "root" que sur les terminaux listés dans `/etc/securetty`.

8 Ctrl-Alt-Del et autres combinaisons spéciales

8.1 Ctrl-Alt-Del (Boot)

Quand on appuie sur Ctrl-Alt-Del (ou toute autre combinaison à laquelle loadkeys associe le keycode Boot), soit la machine reboote immédiatement (sans sync), soit le signal SIGINT est envoyé à `init` (ce qui est le comportement par défaut, changeable par l'appel système reboot(), voir ctrlaltdel(8)). Certains `init` changent le défaut. Ce qui se produit quand `init` reçoit SIGINT dépend de la version du `init` utilisée et est souvent déterminé par le ligne `pf` du fichier `/etc/inittab` (on peut dans ce cas lancer n'importe quel programme). Dans les versions actuelles du noyau Ctrl-AltGr-Del n'est plus assigné par défaut à Boot.

8.2 Autres combinaisons

Combinaisons par défaut avant utilisation de loadkeys, donc en qwerty.

```
Nom du keycode  Combinaison
-------------------------------
Show_Memory     Shift-Scrollock
Show_Registers  AltGr-ScrollLock
Show_State      Ctrl-ScrollLock
Console_n       Alt-Fn and Ctrl-Alt-Fn  (1 <= n <= 12)
Console_{n+12}  AltGr-Fn                (1 <= n <= 12)
Incr_Console    Alt-FlecheDroite
Decr_Console    Alt-FlecheGauche
Last_Console    Alt[Gr]-ImprEcran
Scroll_Backward Shift-PgPrec
Scroll_Forward  Shift-PgSuiv
Compose         Ctrl-.          (Ctrl-: sur clavier azerty)
Caps_On         <pas defini>    (Par defaut CapsLock est un va-et-vient,
Caps_Shift      <pas defini>    ces Keycodes servent a recreer le desagreable
                                comportement MS-Keyb Fr)
```

8.3 Combinaisons sous X

```
Ctrl-Alt-Fn           Affiche la console virtuelle n
Ctrl-Alt-KP+          Resolution suivante
Ctrl-Alt-KP-          Resolution precedente
Ctrl-Alt-Backspace    Tue le serveur X
```

Sur certaines cartes-mères, Ctrl-Alt-KP- et Ctrl-Alt-KP+ émulent le bouton Turbo. Ces séquences produisent les scancodes 1d 38 4a ca b8 9d et 1d 38 4e ce b8 9d, et toutes deux changent la vitesse de Turbo (>= 25MHz) à non-Turbo (8 or 12 MHz) et vice-versa. (souvent ce comportement peut être désactivé par un cavalier sur la carte-mère.)

Perry F Nguyen (*pfnguyen@netcom22.netcom.com*) écrit:

> Les BIOS AMI permettent de bloquer le clavier et de faire clignoter les LEDs quand on appuie sur Ctrl-Alt-Backspace et qu'un mot de passe BIOS est défini, et ce jusqu'à ce que celui-ci soit tapé.

8.4 Combinaisons sous Dosemu

```
Ctrl-Alt-Fn     Rappelle la console n (versions >0.50; avant Alt-Fn)
Ctrl-Alt-PgDn   Tue dosemu (quand le calvier est en mode RAW)
(et bien d'autres, voir la documentation de dosemu)
```

8.5 Composition de symboles

Un symbole peut être construit en utilisant plusieurs frappes.

- En mode Ascii, taper un code Ascii en décimal sur le pavé numérique, tout en maintenant la touche Alt enfoncée, produit le caractère en question quand on relâche Alt. (En mode Unicode, il faut taper le code hexadécimal à 4 chiffres du symbole)
- L'appui sur un accent mort puis sur une lettre produit cette lettre accentuée (si possible, sinon l'accent suivi de la lettre). Les touche mortes sont définissables par l'utilisateur (grâce à loadkeys(1)). Il y a cinq possibilités:

```
accent                keycode mort          keycode normal
------------------------------------------------------
accent grave          dead_grave            grave
accent aigu           dead_acute            apostrophe
accent circonflexe    dead_circumflex       asciicircum
tilde                 dead_tilde            asciitilde
trema                 dead_diaeresis        diaeresis
```

  Par défaut aucune touche du clavier n'est morte. Le résultat est celui de Compose + accent + lettre (et est donc redéfinissable, cf plus bas).

  > NDT: pour des francophones utilisant un clavier Azerty, il vaut généralement mieux se contenter de définir comme touches mortes l'accent circonflexe et le tréma de la touche à droite du 'P'. Il est sinon fastidieux voire parfois impossible (en particulier sous X) de taper les symboles tilde et apostrophe, bien utiles sous Unix. Or les tables standard pour clavier Latin-1 activent en général toutes les touches mortes possibles. Il faut donc pour ressusciter ces touches, remplacer le keycode mort par le keycode normal dans la table chargée au démarrage par loadkeys.

- Compose suivi de deux symboles produit une combinaison des deux. Cette combinaison est redéfinissable. Il y a à ce jour 68 combinaisons par défaut qui peuvent être affichées par `dumpkeys | grep compose`.
- Depuis la version 1.3.33 du noyau, il est possible d'avoir des touches "collantes" (plus ou moins comme Escape) grâce aux codes SControl et SAlt. Par exemple, on peut taper ^C par Scontrol puis C et Ctrl-Alt-BackSpace par SControl puis SAlt puis BackSpace.

Il est à noter qu'il existe au moins trois mécanismes de composition distincts:

1. Le pilote clavier de Linux, configuré par loadkeys.
2. Le mécanisme de X - voir X386keybd(1), ou XFree86kbd(1).
   Sous X11R6 : éditer `/usr/X11R6/lib/X11/locale/iso8859-1/Compose`.
3. Le mécanisme d'emacs, activé en chargeant `iso-insert.el`.

Pour X, l'ordre des deux symboles n'a pas d'importance: Compose-,-c et Compose-c-, donnent tous deux un c-cédille; pour Linux et emacs, seule la première combinaison est correcte. Les combinaisons de X sont fixes, celles de Linux et emacs sont reconfigurables. Les trois listes par défaut sont assez similaires, mais il y a quelques différences.

9 Diverses propriétés de la console

Voir loadkeys(1), setleds(1), setmetamode(1) pour les codes générés par les diverses touches et l'utilisation des leds quand on n'est pas sous X. Sous X, voir xmodmap(1).

Voir setterm(1), kbdrate(8) pour les propriétés telles que les couleurs de l'écriture et du fond, l'économiseur d'écran et la vitesse de répétition des touches quand on n'est pas sous X. Sous X, voir xset(1), qui s'occupe aussi du clic des touches et du volume des bips.

Le fichier `/etc/termcap` définit les séquences escape utilisées par de nombreux programmes utilisant la console (ou tout autre terminal). Une version plus moderne se trouve dans `/usr/lib/terminfo`. (voir terminfo(5). Les fichiers terminfo sont créés par le compilateur terminfo `/usr/lib/terminfo/tic`, voir tic(1).)

(Sur ma machine) `/dev/console` est lien symbolique sur `/dev/tty0`, et le noyau considère `/dev/tty0` comme un synonyme de la console virtuelle courante. XFree86 1.3 change le propriétaire de `/dev/tty0`, mais ne le restaure qu'à la sortie. Or dumpkeys risque de ne pas marcher si quelqu'un d'autre est propriétaire de `/dev/console`; dans ce cas, lancer X et en sortir peut arranger les choses.

10 Comment sortir du mode RAW

Si un programme utilisant le mode `K_RAW` du clavier se termine sans remettre le calvier en mode `K_XLATE`, alors il est très difficile de faire quoi que ce soit - même Ctrl-Alt-Del. Cependant il est parfois possible d'éviter d'appuyer sur Reset (et même souhaitable: vos utilisateurs risquent de ne pas apprécier qu'on leur tue leur Nethack, et surtout il est possible d'endommager le système de fichiers). Les solutions simples consistent à se loger à partir d'un autre terminal ou une autre machine et faire `kbd_mode -a`. La procédure suivante suppose que X ne tourne pas, que l'écran est en mode texte, que vous êtes à un prompt shell, que Ctrl-C est le caractère d'interruption et que vous n'avez pas un clavier trop exotique (i.e. la touche '=' est 2 touches à droite de '0').

1ère étape: Lancer X. Appuyez sur la combinaison 2-F12-= : appuyez sur 2 puis F12 (sans lâcher 2) et puis = (sans lâcher les deux autres), le tout assez rapidement! Ça lance X. (Explication: si l'appui sur une touche produit le keycode K, alors son relâchement produit K+128. Votre shell risque de na pas apprécier ces caractères, donc on évite de les lui envoyer en ne lâchant pas les touches. '2' produit un ^C qui annule tout ce qui a été tapé avant. F12 produit un X et '=' le code entrée. Et tout ça vite fait avant que la répétition des touches ne commence...)

Votre écran est alors probablement gris, puisqu'aucun `.xinitrc` n'a été spécifié. Cependant Ctrl-Alt-Fn marche, et vous pouvez appeler une autre console. (Ctrl-Alt-Backspace marche aussi, mais ça restaure l'état initial du clavier qui justement n'est pas bon.)

2ème étape: Programmer le changement de mode du clavier. (Par exemple, avec `sleep 5; kbd_mode -a`.)

3ème étape: Quitter X. Alt-Fx (souvent Alt-F7) retourne sous X, ensuite Ctrl-Alt-Backspace tue X. Dans les 5 secondes le clavier redevient utilisable.

Si vous voulez vous préparer pour cette occasion, faites de `\215A\301` (3 symboles) un alias de `kbd_mode -a`. Désormais, l'appui sur F7 en mode RAW remettra tout en ordre.

11 La variable TERM

De nombreux programmes utilisent la variable `TERM` et la base de données `/etc/termcap` ou `/usr/lib/terminfo/*` pour trouver quelle chaîne de commande efface l'écran, déplace le curseur, etc., et parfois pour savoir quelle chaîne est envoyée par la touche backspace, ou les touches de fonction, etc. Cette variable est d'abord définie par le noyau (pour la console). Généralement elle est redéfinie par `getty`, en utilisant `/etc/ttytype` ou l'argument spécifié dans `/etc/inittab`. Parfois elle est encore changée par `/etc/profile`.

Les anciens systèmes utilisent `TERM=console` ou `TERM=con80x25`. Les plus récents (avec ncurses 1.8.6) utilisent la valeur plus spécifique `TERM=linux` ou `TERM=linux-80x25`. Cependant, les vieilles versions de `setterm` cherchent une valeur du type `TERM=con*` et ne marchent donc pas avec `TERM=linux`.

Depuis la version 1.3.2 du noyau, la valeur par défaut est `TERM=linux`.

Si votre termcap n'a pas d'entrée 'linux', il suffit d'ajouter l'alias linux dans la description de 'console':

```
console|con80x25|linux:\
```

et de copier (ou faire un lien symbolique) `/usr/lib/terminfo/c/console` en `/usr/lib/terminfo/l/linux`.

11.1 Terminfo

Il manque, dans le fichier terminfo de la console linux fourni avec ncurses 1.8.6, la ligne: `kich1=\E[2~`, nécessaire à certains programmes. Editez ce fichier et compilez-le avec `tic`.

12 Comment faire pour que d'autres programmes acceptent les caractères non-ASCII

C'était jadis un véritable calvaire. Il fallait convaincre individuellement chaque programme de travailler en 8 bits. Les choses ne sont pas encore idéales, mais récemment de nombreux utilitaires GNU ont appris à reconnaître les variables `LC_CTYPE=iso_8859_1` ou `LC_CTYPE=iso-8859-1`. Essayez d'abord ça, et si ça ne suffit pas essayez les trucs ci-dessous.

Tout d'abord, le huitième bit doit survivre au processus d'entrée du noyau, assurez-vous-en donc avec `stty cs8 -istrip -parenb`.

A. Pour `emacs`, mettez les lignes

```
(standard-display-european t)
(set-input-mode nil nil 1)
(require 'iso-syntax)
```

et peut-être aussi

```
(load-library "iso-insert.el")
(define-key global-map [?\C-.] 8859-1-map)
```

dans votre `$HOME/.emacs`. (Cette dernière ligne marche dans un `xterm`, en utilisant `emacs -nw`, mais il faut alors mettre

```
XTerm*VT100.Translations:       #override\n\
Ctrl <KeyPress> . : string("\0308")
```

dans votre `.Xresources`.) NDT: fichiers pour clavier Qwerty, à vérifier pour azerty.

B. Pour `less`, mettez `LESSCHARSET=latin1` dans l'environnement.

C. Pour `ls`, mettez l'option `-N`. (A priori en faisant un alias.)

D. Pour `bash` (version 1.13.*), mettez

```
set meta-flag on
set convert-meta off
```

et, selon le Danish-HOWTO,

```
set output-meta on
```

dans votre `$HOME/.inputrc`.

E. Pour `tcsh`, définissez les variables:

```
setenv LANG     fr_FR   (ou fr_CA, fr_CH, fr_BE...)
setenv LC_CTYPE iso_8859_1
```

Si `nls` est installé, les routines correspondantes sont utilisées. Sinon `tcsh` agit en iso_8859_1, quelle que soit les valeurs données à LANG et LC_CTYPE. voir la section NATIVE LANGUAGE SYSTEM de tcsh(1). (d'après le Danish-HOWTO: `setenv LC_CTYPE ISO-8859-1; stty pass8`)

F. Pour `flex`, donnez l'option `-8` si l'analyseur généré doit accepter les entrées 8-bits. (Bien sur qu'il doit le faire !)

G. Pour `elm`, mettez `displaycharset` à `ISO-8859-1`. (Danish HOWTO: `LANG=C` et `LC_CTYPE=ISO-8859-1`)

H. Pour les programmes utilisant curses (comme `lynx`) David Sibley dit:

> La version standard de curses utilise le huitième bit pour la vidéo inversée (voir le flag _STANDOUT défini dans `/usr/include/curses.h`). Cependant `ncurses` semble fonctionner en 8-bits et affiche le iso-latin-8859-1 correctement.

I. Pour les programmes utilisant `groff` (comme `man`), utilisez le `-Tlatin1` au lieu de `-Tascii`. Les vieilles versions de `man` utilisent aussi `col`, et le point suivant s'applique aussi.

J. Pout `col`, assurez-vous 1) qu'il a été corrigé et fait un `setlocale(LC_CTYPE,"");` et 2) de définir `LC_CTYPE=ISO-8859-1` dans l'environnement.

K. Pour `rlogin`, utilisez l'option `-8`.

L. Pour `joe`, `sunsite.unc.edu:/pub/Linux/apps/editors/joe-1.0.8-linux.tar.gz` devrait marcher après édition du fichier de configuration. J'ai aussi lu: `joe`: mettez l'option `-asis` dans `/usr/lib/joerc` en première colonne.

M. Pour LaTeX: `\documentstyle[isolatin]{article}`.
Pour LaTeX2e: `\documentclass{article}\usepackage{isolatin}` ou `isolatin.sty` est disponible à `ftp://ftp.vlsivie.tuwien.ac.at/pub/8bit` .

Une belle discussion sur le thème de l'ISO-8859-1 et sur comment manipuler les caractères 8-bits est disponible dans `ftp://grasp.insa-lyon.fr/pub/faq/fr/accents` (en français). Une autre, en anglais, peut être trouvée à `ftp://rtfm.mit.edu/pub/usenet-by-group/comp.answers/character-sets/iso-8859-1-faq` . Encore une autre(?): `ftp://ftp.vlsivie.tuwien.ac.at/pub/8bit/FAQ-ISO-8859-1` .

13 Que fait exactement XFree86-2.1 à l'initialisation de sa keymap?

Depuis la version 2.1, XFree86 initialise sa keymap d'après celle de Linux, dans les limites du possible. Linux a 16 entrées par touches (une pour chaque combinaison de Shift, AltGr, Ctrl, Alt; en fait il en a même 256), alors que X n'en a que 4 (une pour chaque combinaison de Shift et Mod), il y a donc forcément des informations perdues.

D'abord `X` lit le fichier `Xconfig`, où il trouve les correspondances entre les touches Control, Alt et ScrollLock avec les codes X Meta, ModeShift, Compose, ModeLock et ScrollLock - voir X386keybd(1), ou XFree86kbd(1).

Par défaut, c'est la colonne LeftAlt qui sert pour Mod, sauf si CtlDroit est défini comme ModeShift ou ModeLock, dans ce cas ce sont les entrées RightCtl qui servent pour Mod. (Sauf si AltGr est défini comme Mod dans Xconfig, auquel cas c'est la colonne RightAlt qui sert.) Ceci détermine comment les 4 entrées de XFree86 sont choisies parmi les 16 de Linux. Notons que par défaut Linux ne fait pas la différence entre les deux touche Control ou Shift. `X` fait la dufférence.

Les touches "action" Show_Memory, Show_State, Show_Registers, Last_Console, Console_n, Scroll_Backward, Scroll_Forward, Caps_On et Boot sont ignorées, de même pour les touches mortes, NumLock, ScrollLock et Alt+code-ASCII.

Ensuite, les définitions de `Xconfig` sont utilisées. (Donc une définition de Compose dans `Xconfig` annulera celle trouvée dans la keymap du noyau.)

Que deviennent les chaînes associées aux touches des fonctions ? Rien, ce concept n'existe pas sous X. (Mais il est possible de définir des chaînes associées aux touches de fonction dans `xterm` - mais elles ne doivent pas être interceptées par le gestionnaire de fenêtres.)

Je ne sais pas comment convaincre `xterm` qu'il devrait utiliser la keymap de X quand Alt est enfoncé. Il semble qu'il ne réagisse qu'en fonction de sa ressource `eightBitInput`, et selon qu'elle est à vrai ou faux, soit il met à 1 le huitième bit, soit il génère un caractère escape devant le caractère (comme le fait setmetamode(1) pour la console).

14 Touches et claviers particuliers

Les deux touches ImprEcran/Syst et Pause/Attn sont spéciales car elles ont deux keycodes: la première produit le keycode 84 quand Alt est enfoncé et 99 sinon; la seconde 101 si Ctrl est enfoncé, 119 sinon. (Il est donc inutile d'assigner des fonctions à Alt-Keycode99 ou Ctrl-Keycode119.)

Si votre clavier a des touches étranges qui ne génèrent aucun code sous Linux (ou génèrent des messages du genre "unrecognized scancode"), vous pouvez, à partir du noyau 1.1.63, utiliser setkeycodes(1) pour dire au noyau quel keycode assigner à ces touches. (Leur utilisation sous X sera cependant impossible.) Une fois qu'elles ont un keycode grâce à `setkeycodes`, on peut leur associer une fonction avec `loadkeys`.

15 Exemples d'utilisation de loadkeys and xmodmap

Permuter ScrollLock et Control (en suposant que vous utilisez les keymaps 0-15; vérifiez avec `dumpkeys | head -1`)

```
% loadkeys
keymaps 0-15
keycode 58 = Control
keycode 29 = Caps_Lock
```

```
%
```

Les permuter sous X seulement:

```
% xmodmap .xmodmaprc
```

où `.xmodmaprc` contient les lignes

```
remove Lock = Caps_Lock
remove Control =  Control_L
keysym  Control_L  =  Caps_Lock
keysym  Caps_Lock  = Control_L
add Lock = Caps_Lock
add Control = Control_L
```

Qu'en est-il de la numérotation des touches? Backspace a le numéro 14 sous Linux et 22 sous X... En fait, la numérotation est plus ou moins arbitraire. Le numéro sous Linux peut être visualisé avec showkey(1), et le numéro sous X avec xev(1). Souvent le numéro sous X est 8 de plus que le numéro sous Linux.

15.1 'Je ne peux taper qu'avec un seul doigt'

Les touches Shift, Ctrl et Alt peuvent-elles être des commutateurs stables ? Oui, en faisant:

```
% loadkeys
keycode 29 = Control_Lock
keycode 42 = Shift_Lock
keycode 56 = Alt_Lock
%
```

les Control, Shift et Alt de gauche deviennent stables. Les numéros à utiliser sont donnés par showkey (et sont généralement 29 et 97 (Control), 42 et 54 (Shift), 56 et 100 (Alt)) et les fonctions possibles sont Control_Lock, Shift_Lock, Alt_Lock, ALtGr_Lock.

Et le touches 'collantes' ? Il n'y a pas encore eu de nouvelle version du paquetage kbd depuis leur introduction dans le noyau 1.3.33, il faut donc utiliser leurs codes hexa, par example:

```
% loadkeys
keymaps 0-15
keycode 54 = 0x0c00
keycode 97 = 0x0c02
keycode 100 = 0x0c03
%
```

rend les Shift, Ctrl et Alt de droite collantes.

16 Changer le mode vidéo

Pour autant que je sache, il y a 6 manières de changer de mode résolution:

1. A la compilation: changer la ligne

```
SVGA_MODE=      -DSVGA_MODE=NORMAL_VGA
```

dans `/usr/src/linux/Makefile`.

1A. Après la compilation: utiliser `rdev -v` - une affreuse magouille, mais bon, ça marche.

2. Au démarrage: mettre `vga=ask` dans le fichier de config de lilo, qui demandera au boot le mode voulu. Une fois décidé, remplacer par `vga=`LePlusJoliMode.

3. En cours de route: A. Utiliser la commande `resizecons`. (C'est un programme très primitif utilisant l'ioctl VT_RESIZE.) B. Utiliser `SVGATextMode`. (C'est une version moins primitive)

4. Pas "sur la console": Sous `dosemu`, ou avec svgalib etc. on peut changer le mode vidéo de la carte écran sans que le pilote de la console s'en aperçoive. C'est parfois utile pour configurer `resizecons` ou `SVGATextMode`:sous `dosemu` se mettre dans le mode vidéo voulu grâce à un programme DOS utilisant ce mode, puis dans une autre console, récupérer les paramètres de ce mode. Il ne reste plus qu'à utiliser ces données pour l'initialisation de `resizecons` et `SVGATextMode`. Dans certains cas la carte vidéo se retrouve dans un mode inutilisable, le moyen le plus simple pour se sortir de là est de lancer `dosemu`, laisser le BIOS mettre un mode vidéo correct, puis tuer `dosemu` (avec `kill -9`).

16.1 Instructions pour l'utilisation de resizecons

Récupérer svgalib et compiler le programme `restoretextmode`. Booter la machine dans tous les modes vidéo possibles (en mettant `vga=ask` dans le fichier config de lilo), et sauvegarder les registres vidéo dans des fichiers CxL (C=Colonnes, L=Lignes), par exemple 80x25, 132x44, etc. Placer ces fichiers dans `/usr/lib/kbd/videomodes`. Désormais `resizecons 132x44` changera le mode vidéo (et enverra le signal SIGWINCH à tous les processus qui ont besoin de savoir que la résolution a changé, et chargera une nouvelle police si nécessaire).

A présent, `resizecons` ne change de mode que s'il y a assez de mémoire pour contenir à la fois l'ancienne et la nouvelle console.

17 Changer la vitesse de répétition du clavier

Au démarrage, le noyau met la vitesse de répétition à sa valeur maximale. Pour la plupart des claviers c'est raisonnable, mais sur certains il devient quasiment impossible d'effleurer une touche sans avoir trois fois le même caractère. Dans ce cas utiliser le programme kbdrate(8) pour changer la vitesse de répétition ou si ça ne suffit pas supprimer la section:

```
! set the keyboard repeat rate to the max

    mov     ax,#0x0305
    xor     bx,bx           ! clear bx
    int     0x16
```

de `/usr/src/linux/[arch/i386/]boot/setup.S`.

18 Economiseur d'écran

`setterm -blank` *nn* règle le délai d'extinction de l'écran à *nn* minutes d'inactivité. (Avec $nn = 0$, l'économiseur d'écran est désactivé.)

L'option `s` de xset(1) règle les paramètres de l'économiseur d'écran de X.

Les modes d'économie d'énergie du moniteur peuvent être activés/désactivés par le programme `setvesablank` donné dans les commentaires au début du fichier `/usr/src/linux/drivers/char/vesa_blank.c`.

19 Quelques propriétés du VT100 - mode application

: Parfois les touches de curseur produisent des codes bizarres?

Quand le terminal est en `mode application`, les touches de curseur produisent les codes Esc O x et sinon Esc [x, avec x = A,B,C ou D. Certains programmes mettent le termnial en mode application et si on les tue avec un `kill -9`, ou s'il se plantent, le terminal restera dans ce mode.

```
% echo -e '\033c'
```

réinitialise les propriétés du terminal courant. Si on veut passer en mode application:

```
% echo -e '\033[?1h'
```

et si on veut en sortir:

```
% echo -e '\033[?1l'
```

20 Incompatibilité matérielle

Quelques personnes ont noté des pertes de caractères tapés lors d'un accès disquette. Il semblerait que ce soit un problème avec les cartes mères Uni-486WB. (SVP envoyez moi un mail pour confirmer [Oui, j'ai le même problème], infirmer [Non, tout va bien avec ma carte Uni-486WB], ou modifier [Ma machine Xyzzy a le même problème].)

Certaines personnes ont eu des blocages aléatoires du clavier - parfois associés à une activité disque dur ou une autre entrée/sortie.

ulf@rio70.bln.sni.de

(Ulf Tietz) écrit:

> 'J'avais ce genre de problèmes quand ma carte mère avait des réglages trop rapides. En remettant les délais (CLK, wait-states, etc.) à des valeurs plus raisonnables, tout est rentré dans l'ordre.'

bhogan@crl.com (Bill Hogan) écrit:

> 'Si vous avez un BIOS AMI, vous pouvez essayer de mettre le paramètre Gate A20 emulation sur 'chipset' (si cette option existe). Quand cette option était sur n'importe quoi d'autre ('fast', 'both', 'disabled') j'avais souvent des blocages du clavier.'

———————————————————————————————

Additions et corrections sont les bienvenues.

Andries Brouwer - *aeb@cwi.nl*

(Bruno Viaris - Viaris@Yoko.ENS-Cachan.Fr pour la traduction)