Utiliser pleinement mon ordinateur

Thierry (aka tTh) Boudet

3 novembre 2015

Chapitre 1

Introduction

Un nouvel ouvrage, en vrac, complètement en vrac. Des notes informelles sur l'utilisation d'un ordinateur moderne, vous allez en avoir vraiment beaucoup. Mais elles sont en vrac.

Tout cette ouvrage a été écrit avec gedit et LaTeX selon la méthode éprouvée dite de « La Rache », essentiellement sur une machine Debian/x86, et parfois sur OpenBSD/sparc64. Une belle occasion de comparer [1] les avantages et les inconvénients de ces deux respectables systèmes libres.

1.1 legal stuff

Ahemm...

Sous quelle licence est distribué cet ouvrage ? Bonne question. À laquelle je vais donner une réponse provisoire : DWTFYWL [2].

Tout ce que vous pourrez lire dans cet ouvrage n'est absolument pas garanti en bon état de fonctionnement.

Your mileage may vary...

1. Qui a dit *pulseaudio* au fond de la salle ?
2. Do what the fsck you want.

Table des matières

Chapitre 2

Formater du texte

Manipuler du texte, afin de le modifier, de le mettre en forme, de le présenter de façon harmonieuse, tout le monde l'a déja fait, avec des outils divers et variés et des résultats parfois (ou souvent) hasardeux.

2.1 fmt et ses copains

Nous allons voir quelques outils de base, simple et efficace, mais qui ont parfois quelques limitations. Le premier est `fmt` qui permet de remettre en forme un texte.

```
tth@plop:~/Ecriture/Ordinateur$ cat texte.txt
Ce morceau de texte n'est pas
bien
formaté.

Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Phasellus nunc sapien, porta id pellentesque at, elementum et felis.
Curabitur condimentum
ante in metus
iaculis quis congue diam commodo.
tth@plop:~/Ecriture/Ordinateur$ fmt texte.txt
Ce morceau de texte n'est pas bien formaté.

Lorem ipsum dolor sit amet, consectetur adipiscing elit.  Phasellus
nunc sapien, porta id pellentesque at, elementum et felis.  Curabitur
condimentum ante in metus iaculis quis congue diam commodo.
```

2.2 roff

Roff est l'outil qui a permis à quelques hackers légendaire de justifier auprès de leur hiérarchie le développement d'une machine de jeu, il est donc légitime d'en parler, non ?

Après quelques minutes d'expérimentation, un premier souci arrive. Il semble bien que `groff` ne sache pas lire les fichiers encodés en **utf8**. Il faut donc utiliser

un hack avec `recode` en prenant bien soin d'utiliser une redirection vers `stdin` comme ceci :

```
< $BASENAME.txt recode utf8..latin1 | groff -Tutf8
```

2.3 yodl

Yodl is a package that implements a pre-document language and tools to process it. The idea of Yodl is that you write up a document in a pre-language, then use the tools (e.g. yodl2html) to convert it to some final document language. Current converters are for HTML, man, LaTeX SGML and texinfo, a poor-man's text converter and an experimental xml converter.

2.4 zoem

Zoem is a language for transforming text. It supports the creation of light-weight mark-up languages as macro packages built on top of zoem. PUD (Portable Unix Documentation) provides two such mini-languages and is shipped with zoem. The PUD languages are for generating manpages and FAQ documents in both HTML and troff.

Chapitre 3

Éditeurs

3.1 gedit

Gedit [1] est un éditeur de texte simple basé sur Gtk pour le bureau Gnome [2]. Entre autres fonctions basique, il fait la coloration syntaxique [3]. Par contre, contrairement [4], il est très facilement scriptable : il suffit d'écrire quelques lignes de shell. Vous voulez numéroter les lignes d'un paragraphe ? Il suffit d'un tout petit morceau de Awk que voici :

```
#!/usr/bin/awk -f
{ printf "%5d\t%s\n", NR, $0; }
```

Ok ? Pigé le principe ? Il ne reste plus qu'à intégrer ce bout de code dans les outils personnalisables de Gedit.

3.2 emacs

Escape Meta Alt Control Del

3.3 ed

The only true one. C'est de lui que tout est parti.

3.4 nedit

Vieilles marmites, toussa...

Mais où est donc la coloration syntaxique pour POV ?

1. dans sa version 2.30.3
2. On ne rigole pas dans le fond !
3. sauf pour le Postscript, hélas.
4. Si quelqu'un veut me contredire, il sait comment faire

Chapitre 4

Gestion de versions

4.1 SCCS

Un des grands ancêtres.

4.2 CVS

Utilisé par OpenBSD, donc forcément barbu.

4.3 Fossil

> Fossil is a distributed version control system (DVCS) with built-in wiki, ticket tracker, CGI/http interface, and http server.

4.4 Subversion

4.5 Et les autres ?

Nous verrons le truc à la mode en page 61.

Chapitre 5

X11

5.1 Les fontes

Comment fait-on pour connaitre la liste des polices installées ?

```
$ fc-list | awk -F ":" '{print $1}' | sort -u
```

Comment faire pour visualiser la plénitude des formes de mes fontes ?

Un petit *tips* fourni par NG [1]

```
xterm -fa 'DejaVu Sans Mono:size=10:matrix=2 0 0 1'
http://www.freedesktop.org/software/fontconfig/fontconfig-user.html#AEN150
```

5.2 Xephyr

D'après la manpage : *Xephyr is a kdrive server that outputs to a window on a pre-existing "host" X display. Think Xnest but with support for modern extensions like composite, damage and randr.*

Ça semble attirant à première vue, mais l'utilisation n'est pas si simple qua ça. Par exemple, quelle est la méthode à mettre en œuvre pour avoir le clavier en Azerty ?

5.3 Numlock

```
#include <X11/XKBlib.h>
#include <X11/extensions/XKB.h>
#include <X11/keysym.h>
int main(){
     Display *disp = XOpenDisplay (NULL);
     if(disp == NULL) return 1;
     unsigned int nl_mask = XkbKeysymToModifiers           \
                                  (disp, XK_Num_Lock);
```

1. Oui, oui, le fameux Nicolas Georges, dans un message Usenet

```
    XkbLockModifiers (disp, XkbUseCoreKbd, nl_mask, nl_mask);
    XCloseDisplay (disp);
return 0;}
```

5.4 xdotool

D'après la manpage : *xdotool lets you programatically (or manually) simulate keyboard input and mouse activity, move and resize windows, etc. It does this using X11's XTEST extension and other Xlib functions.*

En gros, et pour faire simple, avec ce logiciel, vous pouvez piloter des applications X11, à la manière d'un wm, et ce, à partir de la ligne de commande.

Vous pourrez les déplacer, les redimensionner, leur envoyer des évènements clavier, tout ce genre de choses. Il est donc possible de coder « des sortes d'animations » uniquement basées sur le protocole de X11. *Big win, bro !*

La démarche d'utilisation n'étant pas immédiate, nous allons commencer par un petit exemple : déplacer un Xlogo sur l'écran. La première étape est le lancement de la cible en arrière-plan. La seconde est la récupération de l'indentifiant (interne à X11) de la fenêtre concernée.

```
$ xlogo &
[3] 2674
$ WIN=$(xdotool search -name xlogo)
$ echo $WIN
69206019
$
```

Voilà [2], nous savons maintenant à quelle fenêtre causer : par son numéro. Le problême des multiples fenêtres du même nom sera abordé ultérieurement. Déplaçons donc notre widget...

```
$ xdotool windowmove $WIN 10 10
$
```

C'est simple, non ?

2. en français dans le texte

Chapitre 6

Son

6.1 Généralités

6.2 ALSA

ALSA, *Advanced Linux Sound Architecture*, gère à peu près correctement les entrées/sorties sonores d'une machine Linux, aussi bien en audio qu'en MIDI [1].

Nous allons faire le tour de quelques outils associés. En premier, le « centre de contrôle », `alsamixer`, dont la qualitée de l'interface en Curses est époustouflante :

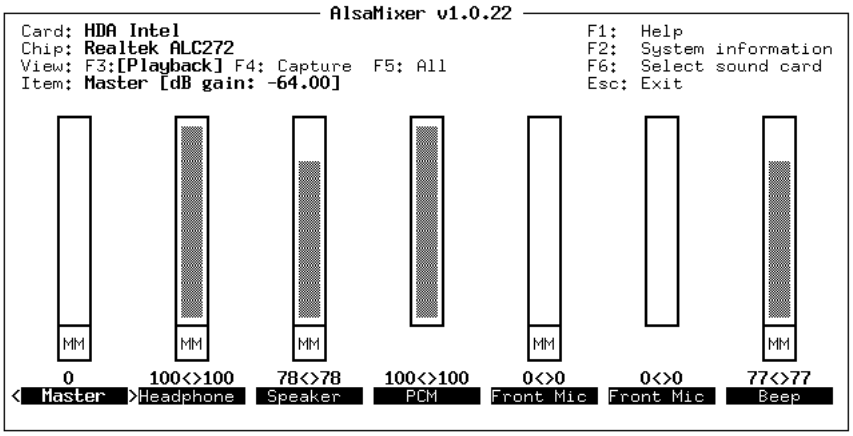

FIGURE 6.1 – Alsamixer

Ensuite, nous avons `aplay` et `arecord`, qui permettent d'écouter et d'enregistrer Justin Bieber [2].

Et quand on regarde par la lorgnette du programmeur, on découvre rapidement que l'utilisation d'ALSA n'est pas si simple que ça, essentiellement à cause

1. Voir page 22 pour en savoir plus.
2. Nan, j'plaisante !

d'une documentation spartiate. Heurensement, pour certains domaines, il existe des surcouches plus simple d'emploi : `libao` pour les sorties sonores, XXX pour le reste.

6.3 pulseaudio

On avait pourtant dit « **pas de trolls** », mais il va quand même falloir y passer : ce machin prétenduement *michu compliant* a parfois tendance à se mettre en travers du chemin des honnètes gens.

6.4 Jack

Jack Audio Connection Kit : ce sous-système [3] est destiné avant tout aux professionnels de la musique. Je vais attendre le retour de mon spécialiste préféré pour lui laisser le soin d'en parler.

Jack peut aussi servir de source sonore pour darkice, lequel pourra feeder un Icecast

6.5 libsndfile

Libsndfile est essentiellement une très complète bibliothèque de lecture/écriture de fichiers sons. Elle permet de gérer facilement un nombre considérable de formats de fichiers [4].

Par contre, elle arrive avec une floppée d'outils en ligne de commande, dont certains sont assez performants.

Citons, entre autres, `sndfile-resample` qui est un convertisseur de fréquence d'échantillonage très performant.

6.6 libao

C'est une bibliothèque de fonctions pour gérer les sorties sonores d'une application (*Libao is a cross-platform audio library that allows programs to output audio using a simple API on a wide variety of platforms.*). Elle supporte une grande variété de formats.

```
/*
 * ao_example.c
 *      Written by Stan Seibert - July 2001
 *
 * Compilation command line (for Linux systems):
 *      gcc -o ao_example ao_example.c -lao -ldl -lm
 */

#include <stdio.h>
```

3. Dans le monde Linux, on aime bien les sous-systèmes pour le son.
4. Mais ceci est une autre histoire, que nous verrons peut-être un de ces jours...

```
#include <string.h>
#include <ao/ao.h>
#include <math.h>

#define BUF_SIZE 4096

int main(int argc, char **argv)
{
ao_device *device;
ao_sample_format format;
int default_driver;
char *buffer;
int buf_size;
int sample;
float freq = 440.0;
int i;

        /* -- Initialize -- */
fprintf(stderr, "libao example program\n");
ao_initialize();

        /* -- Setup for default driver -- */
default_driver = ao_default_driver_id();

memset(&format, 0, sizeof(format));
format.bits = 16;
format.channels = 2;
format.rate = 44100;
format.byte_format = AO_FMT_LITTLE;

        /* -- Open driver -- */
device = ao_open_live(default_driver, &format, NULL);
if (device == NULL) {
        fprintf(stderr, "Error opening device.\n");
        return 1;
}

        /* -- Play some stuff -- */
buf_size = format.bits/8 * format.channels * format.rate;
buffer = calloc(buf_size, sizeof(char));

for (i = 0; i < format.rate; i++) {
        sample = (int)(0.75 * 32768.0 *
        sin(2*M_PI * freq * ((float)i/format.rate)));

        /* Put the same stuff in left & right channel */
        buffer[4*i] = buffer[4*i+2] = sample & 0xff;
        buffer[4*i+1] = buffer[4*i+3] =
                                (sample >> 8) & 0xff;
        }
ao_play(device, buffer, buf_size);

/* -- Close and shutdown -- */
ao_close(device);
```

```
ao_shutdown();

return (0);
}
```

C'est finalement assez clair, assez simple. Nous aurons bien l'occasion d'en reparler, peut-être lors de l'écriture d'un *source client* pour Icecast.

6.7 Audacity

Graphical cross-platform audio editor.

6.7.1 Introduction

6.7.2 Analyse spectrale

6.8 Ladspa

Ladspa, mais c'est quoi ? C'est l'acronyme de « Linux Audio Developers Simple Plugin API », ou, en français : « machin avancé ».

La commande `listplugins`, comme son nom l'indique, fournit la liste de tous les plugins accessibles.

6.9 Sox

Sox est le « couteau suisse » des manipulateurs de données sonores. Sa documentation étant assez ésotérique, je vais faire comme d'hab' : présenter des exemples.

6.9.1 Conversions

Nous avons un process quelconque qui nous délivre des données en format texte, tel que celles-ci, deux valeurs par ligne, normalisées dans l'intervalle $]-1,1[$, et nous voulons obtenir un .Wav vanille.

```
0.286498189155 0.33310689881
0.0500751550941 -0.326965910932
-0.01317994812 -0.392365740013
-0.353560435178 -0.018978716441
```

Voici comment faire dans un contexte de *Makefile* :

```
random.wav: random.dat Makefile
        @echo " conversion " $< $@
        sox --rate 44100 $< $@
```

6.9.2 Génération

Besoin d'un bon coup de ¡beep¿ ? `play -n synth sine 480 sine 620 remix 1-2 fade 0 0.5 delay 0.5 repeat 5` et voilà.

6.9.3 Traitements

`sox --norm source.wav normal.wav` pour normaliser un son, selon quels critères ?

`sox left.wav right.wav -M stereo.wav` pour combiner deux voies mono en un fichier stéréo. `-M` est l'abréviation de `--combine merge`.

Bien entendu, on peut enchainer (mais avec quelle logique ?) les opérations. la commande `sox g.wav d.wav -m --rate 44100 --norm gd.wav` assemble deux mono en un stéréo, ré-échnatillone à 44100 Hz et normalise le niveau.

6.10 espeak

Espeak [5] est un logiciel de synthèse vocale. Certaines pensent qu'il est rudimentaire, je pense plutôt qu'il est incomplet.

En effet, si il connait plusieurs « voix » de diverses régions du monde :

```
tth@plop:~/RADIO$ espeak --voices=fr
Pty Language Age/Gender VoiceName       File        Other Langs
 5  fr             M  french            fr
 7  fr             M  french-mbrola-1   mb/mb-fr1
 7  fr             F  french-mbrola-4   mb/mb-fr4
 5  fr-be          M  french (Belgium)  fr-be
```

Il n'a pas de choix entre, par exemple, des voix de femmes et des voix d'homme. Quoique : « *Use the -v option with a language code and optionally a variation, +m1 to +m6 for a male voice, and +f1 to +f4 for a female voice.* ». À tester...

Espeak comprend un certain nombre d'options et de paramètres (Attention, il y a parfois des différences entre la manpage et `espeak -h`, je vais donc utiliser la sortie du -h). Voici celles dont on a besoin pour démarrer :

-f blabla.txt : Le fichier texte à lire. *quel encodage ?*

-s nnn : Vitesse en mots par minute, entre 80 et 390, 170 par défaut.

-g nnn : Pause entre les mots, exprimée en "10ms".

-a nnn : Amplitude entre 0 et 200, 100 par défaut.

-w fichier.wav : Écriture dans un fichier (mono 16b 22050).

-p nnn : Hauteur [6] du son, entre 0 et 99.

5. Version utilisée : 1.43.03, il y a peut-être mieux depuis...
6. En anglais, c'est le *pitch*.

6.10.1 SSML

Speech Synthesis Markup Language (SSML) Version 1.0 W3C Recommendation 7 September 2004

Rapidement, quelques exemples, est-ce possible ?

6.11 Pour finir...

En conclusion, le son avec Linux, c'est un peu la grande foire. On arrive parfois même à y voir du XML dedans et c'est en page 50 pour les courageux. On peut aussi carrément utiliser un langage spécialisé comme *liquidsoap*.

Chapitre 7

Joueurs de musique

Vous pouvez aussi aller voir la page 14, qui traite d'autres problématiques liées au son.

7.1 mocp

Le nom « Mocp » est l'abréviation de Music on-consol Player. Personnellement, j'aime beaucoup son look, et son ergonomie parfois déroutante.

7.2 mpd

Le nom « Mpd » est l'abréviation de Music Player Daemon. Il peut théoriquement [1] *feeder* un serveur Icecast.

7.3 Audacious

7.4 Effets sonores

Ladspa ?

7.5 Les playlists

Une *playlist* est (en général) un fichier contenant une liste de musiques à écouter.

1. Théoriquement, parce que je n'ai pas encore essayé :)

Chapitre 8

Puredata

8.1 Premiers pas

Hop, on y va... Attention, la première marche semble assez rude.

8.2 Lire un fichier WAV

Chapitre 9

Musical Instrument Digital Interface

Dans un premier temps, je vais me concentrer sur les services MIDI offerts par la couche ALSA d'un système Linux. Ensuite, nous passerons aux *raw devices* (page 23).

9.1 Outils de base

9.1.1 aseqdump

Cet outil permet de voir de façon relativement claire **tous** les évènements MIDI qui rentre par un chemin donné, et, en première étape, surtout de lister ces chemins.

```
tth@plop:~/Muzak/Samples$ aseqdump -l
 Port    Client name                      Port name
  0:0    System                           Timer
  0:1    System                           Announce
 14:0    Midi Through                     Midi Through Port-0
```

9.1.2 aconnect

Maintenant que nous savons ce qui rentre dans notre machine, il est temps de savoir comment le faire ressortir. Il faut donc créer une connexion quelque part au tréfond des tables obscures de la kernelle.

9.2 ALSA tips

cat /proc/asound/seq/clients

```
Client info
  cur  clients : 4
  peak clients : 4
```

```
  max  clients : 192

Client   0 : "System" [Kernel]
  Port   0 : "Timer" (Rwe-)
  Port   1 : "Announce" (R-e-)
    Connecting To: 15:0
Client  14 : "Midi Through" [Kernel]
  Port   0 : "Midi Through Port-0" (RWe-)
Client  15 : "OSS sequencer" [Kernel]
  Port   0 : "Receiver" (-we-)
    Connected From: 0:1
Client 128 : "TiMidity" [User]
  Port   0 : "TiMidity port 0" (-We-)
  Port   1 : "TiMidity port 1" (-We-)
  Port   2 : "TiMidity port 2" (-We-)
  Port   3 : "TiMidity port 3" (-We-)
  Output pool :
    Pool size          : 500
    Cells in use       : 0
    Peak cells in use  : 0
    Alloc success      : 0
    Alloc failures     : 0
  Input pool :
    Pool size          : 1000
    Cells in use       : 0
    Peak cells in use  : 0
    Alloc success      : 0
    Alloc failures     : 0
```

9.3 Raw devices

Plutôt que de passer par la couche ALSA, il est aussi possible d'attaquer directement le *device*, et de blaster soi-même tous les petits octets directement sur le fil. Voici un petit exemple en Python [1], facilement transposable dans un langage plus sérieux :

```
#!/usr/bin/env python

import time

def note_on(note):
        tx.write("%c%c%c" % (0x90, note, 100))
        tx.flush()
def note_off(note):
        tx.write("%c%c%c" % (0x90, note, 0))
        tx.flush()
def progchange(prog):
        tx.write("%c%c" % (0xc0, prog))
        tx.flush()

tx = open("/dev/midi", "w")
```

1. Oui, je sais, c'est un langage de djeunz, mais bon, ça fait plaisir à mon gamin

```
for valeur in range(30,90):
        progchange(valeur)
        note_on(valeur)
        time.sleep(1)
        note_off(valeur)
        time.sleep(1)

tx.close()
```

9.4 Et en Perl ?

En Perl [2], il existe un certain nombre de modules très évolués pour jouer avec les fichiers MIDI, ce qui est un domaine légèrement différent, il faut le reconnaitre.

```
MIDI (3pm)           - read, compose, modify, and write MIDI files
MIDI::Event (3pm)    - MIDI events
MIDI::Filespec (3pm) - MIDI File Specification
MIDI::Opus (3pm)     - functions and methods for MIDI opuses
MIDI::Score (3pm)    - MIDI scores
MIDI::Simple (3pm)   - procedural/OOP interface for MIDI composition
MIDI::Track (3pm)    - functions and methods for MIDI tracks
```

2. Oui, je sais, c'est un langage de vieux, mais bon, il faut bien se faire plaisir de temps en temps. . .

Chapitre 10

Processing

10.1 Installation

10.2 Premiers pas

Chapitre 11

Vidéo

11.1 mplayer

Une belle grosse usinagaz...

`mplayer tv:// -tv driver=v4l2:width=640:height=480:fps=25:device=/dev/video0`

`mencoder tv:// -tv driver=v4l2:width=640:height=480:fps=25:device=/dev/video0:forceaudio:alsa=1:adevice=default -ovc lavc -oac mp3lame -lameopts cbr:br=64:mode=3 -o`

Pour faire une planche contact d'une vidéo : `mplayer -benchmark -nosound -quiet -zoom -vf scale=320:-3 -vo jpeg:outdir=/tmp -sstep 60 monfichier.ts`

11.2 ffmpeg

D'après certaines rumeurs des intertubes, ce serait un peu le remplaçant de mencoder pour pas mal de trucs...

Pour convertir une sextape prise avec un *iMachin* :

`ffmpeg -i tthsf.mov -target film-dvd sextape.mpeg`

Extraire le son d'une vidéo Youtube :

`ffmpeg -i myvideo.flv -f mp3 -vn -acodec copy myvideo.mp3`

`ffmpeg -i fichiervideo.flv -vn -ar 44100 -ac 2 -f wav fichierson.wav`

`mplayer -vo png -ss 3:46 -frames 20 stairs.mp4` # Write out 20 png format frames from a video starting at 3 minutes, 46 seconds.

Tu peux générer les images de transition avec ImageMagick : `convert a.png b.png -alpha set -channel A -evaluate set 10% +channel -compose src-atop -composite -alpha off out.png`

Tu peux assez facilement insérer ça dans une boucle et piper les images à ffmpeg :

```
for image in ...; do
  next=... dépend de tes noms de fichiers
  for a in {0..24}; do
    convert $image $next -alpha set -channel A -evaluate set $[i*4]% \
```

```
    +channel -compose src-atop -composite -alpha off png:-
  done
done |
ffmpeg -f image2pipe -c png -i - ...
```

Plus quelques trucs en vrac :

```
fichier .avi -> mpeg ( qualité DVD )
    ffmpeg -i fichier.avi -target pal-dvd fichier.mpeg

fichier .avi -> mpeg ( qualité DVD, bitrate 7Mbps, 16/9 )
    ffmpeg -i fichier.avi -target pal-dvd -aspect 16:9 -b 7000000
fichier.mpeg

fichier .avi -> mpeg ( qualité dvd, en 2 passes, 16/9 )
    ffmpeg -i fichier.avi -target pal-dvd -aspect 16:9 -pass 1
fichier_tempo.mpeg
    ffmpeg -i fichier.avi -target pal-dvd -aspect 16:9 -pass 2
fichier_final.mpeg

préciser le "framerate" ( images par secondes : ici 25 )
    ffmpeg -i fichier.avi -r 25 fichier.mpeg

préciser la taille ( défaut = taille du source ( option -s ))
    ffmpeg -i fichier.avi -s 720x576 fichier.mpeg

les deux précédents :
    ffmpeg -i fichier.avi -r 25 -s 640x480 fichier.mpeg

préciser le coef de qualité ( qscale : 1 = le mieux / 3 = correcte )
    ffmpeg -i fichier.dv -target pal-dvd -qscale 2 Fichier.mpeg
```

11.3 Autres trucs

```
  flvstreamer -r rtmp://lacantine.live-fstreams.cdn02.rambla.be:1935/live-lacantine/live-lacant
| mplayer -
```

Chapitre 12

Webcam

Sous Linux, c'est quand même un peu tordu, dans le même genre que la pile son, quoi... Nous allonc commencer par le plus simple : juste tenter de prendre une photo avec une *webcam* classique en USB, et d'enregistrer cette image.

12.1 vgrabbj

D'après la manpage : « *vgrabbj is a program that will grab images from any v4l-capable device which supports one of the rgb24, rgb32, yuv420, yuv420p, yuyv, or yuv422 palettes and saves the image as a .jpg, .png, or .pnm file.* »

Quelques options à connaitre : `-S` échange des deux composantes R et B ; `-s <device>` affiche les capacités du périphérique v4l.

Chapitre 13

SCSI

Acronyme de *Small Computer System Interface*. Ce bus SCSI a été initialement créé par la société Shugart en 1979, et a subit par la suite de nombreuses évolutions.

13.1 lecteurs de DAT

DAT = Digital Audio Tape.

Pour contrôler ce genre de matérien, on utilisera la commande `mt(1)` : control magnetic tape drive operation.

13.2 les scanners

13.3 le bas niveau

Chapitre 14

trucs de dino

Je vais, dans ce chapitre, parler de choses que les moins de vingt ans ne peuvent pas connaitre. Des choses qui existaient *bien avant* le Web !

14.1 inetd

Aussi connu sous le nom de « superserveur Internet », ce démon a fait ses premiers pas dans l'Unix 4.3BSD et depuis est maintenant remplacé par `xinetd`, décrit en page 39.

Inetd a pour but de gérer l'accès à des services réseaux, qui peuvent être internes ou externes. Un service interne est entièrement géré par Inetd : on y trouve par exempme daytime qui permet de connaitre l'heure qu'il est à l'autre bout du monde :

```
tth@plop:~$ grep daytime /etc/inetd.conf
daytime         stream  tcp     nowait  root    internal
tth@plop:~$ nc uind.cispeo.fr daytime
29 JUL 2014 21:10:30 CEST
```

Dans ce cas, c'est directement Inetd qui répond. Mais il sait faire une autre chose, bien plus constructive : toujours répondre à un appel venant du grand Ternet, mais maintenanr nous allons déléguer le traitement de la demande à un autre participant.

Il ne reste plus qu'à le mettre en œuvre, en réutilisant un exemple très ancien [1], et très simple à comprendre.

14.1.1 qotd

Le protocole « Quote Of The Day » (`qotd`) a été normalise dans le `rfc865`, écrit en 1983 par John Postel. Une rapide implémentation du moteur de quote peut être codée en Shell :

```
#!/bin/sh
```

1. Août 1998, lors d'une réunion du CULTe, à Toulouse

```
/usr/games/fortune -a -n 500
```

Maintenant, il nous faut dire à Inetd d'appeler ce moteur chaque fois qu'une connection tcp entrante arrive sur le port 17, réservé au service `qotd`. Vérifions d'abord si la correspondance entre le nom et le numéro de port est correcte :

```
tth@plop:~$ grep qotd /etc/services
qotd            17/tcp          quote
```

Parfait, nous pouvons donc rajouter cette ligne dans le fichier de configuration de Inetd, `/etc/inetd.conf` :

```
qotd
```

14.2 gopher

Sur le port 70.

> The Internet Gopher(tm) software and documentation is copyright (c) 1991, 1992, 1993, 1994 by the University of Minnesota.

Chapitre 15

Makefile

15.1 À quoi ça sert ?

C'est un truc qui va bosser à votre place.

Une sorte de peti esclave bienveillant.

15.2 Ça marche pas !

En cas de souci du genre « mais pourquoi cet imbécile me reconstruit sans cesse `essai.tga` ? », la première étape, c'est d'essayer de comprendre à partir de quoi `make` prend ces décisions.

```
$ script toto
$ make -d essai.tga
$ ^D
$ less toto
```

Oups...

Chapitre 16

MINITEL

Une très belle réussite technologique française, dont la fin de vie est annoncée pour 2014. Certains geeks pensent qu'il présente encore un grand intérêt, à la fois historique et ludique Le Tetalab [1], par exemple, les interface avec des Arduino afin de diffuser des images pornographiques ou de contrôler des appareils étranges.

16.1 Xtel

Xtel est un émulateur de MINITEL pour les systèmes Unix/X11, écrit par Pierre Ficheux vers 1997.

Il est architecturé selon le modèle client/serveur, comme l'explique son README :

> Il est utilisable en réseau, c'est a dire qu'une des machines du réseau commande les MODEMS et dialogue avec les services Télétel, les utilisateurs de XTEL dialoguent avec cette machine "serveur" par socket internet (utilisation d'un service TCP et du super-démon "inetd").

1. Un hackerspace à Toulouse

Chapitre 17

Threads

Ahemmm, je ne sais pas par quoi commencer [1] ce chapitre tant le sujet est vaste et plutôt fourbe ? Peut-être un peu d'histoire pour mieux comprendre la suite ?

Qui a inventé ces fameux threads ? Ahemm.

> Wikipédia, help me...

Puis un extrait de la page `pthreads(6)` ? Voilà, ça aussi, c'est bien :

> POSIX.1 specifies a set of interfaces (functions, header files) for threaded programming commonly known as Pthreads. A single process can contain multiple threads, all of which are executing the same program. These threads share the same global memory (data and heap segments), but each thread has its own stack (automatic variables).

Et c'est là que les ennuis potentiels débarquent : *share the same global memory*, ce qui veut dire en gros « se partagent la même mémoire globale ». Que l'on peut traduire par « ils vont se marcher sur les pieds ».

17.1 ITC

Inter Thread Communication

Imaginons une application avec un thread qui fait le travail effectif (nous le nommerons *worker*), et plusieurs autres threads qui attendent des ordres de l'extérieur (telnet, MIDI, joystick...) et qui doivent passer ces commandes au *worker*.

Comment allons-nous procéder [2] ?

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
```

1. Par de l'aspirine ?
2. Et c'est une vraie question !

```
void * slave(void *vptr)
{
int     foo;

for (foo=0; foo<10; foo++) {
        printf("vptr = %p, foo = %d\n", vptr, foo);
        sleep(1);
        }


return NULL;
}

int main(int argc, char *argv[])
{
int             foo;
pthread_t       thread;
int             control2worker[2];

foo = pthread_create(&thread, NULL, slave, NULL);
printf("create thread -> %d\n", foo);
sleep(20);
return 0;
}
```

Chapitre 18

Dosbox

Dosbox est un émulateur de machine MS-DOS avec le son et le graphique. Il ne nécessite pas de système d'exploitation.

18.1 Popcorn

Popcorn est un jeu de cassebrique écrit dans les années 80 pour tirer la quintessence des cartes graphiques CGA.

Chapitre 19

database

19.1 PostgreSQL

19.2 Gdbm

The GNU database manager. Includes dbm and ndbm compatability.

> GNU dbm is a library of routines that manages data files that contain key/data pairs. The access provided is that of storing, retrieval, and deletion by key and a non-sorted traversal of all keys. A process is allowed to use multiple data files at the same time.

19.3 MySQL

Chapitre 20

Photographie

20.1 gphoto2

Notes écrites en se basant sur la version 2.4.5 (Ubuntu 10.04), les choses peuvent être (rarement) différentes chez vous. L'appareil photo que j'utilise actuellement est un Canon *PowerShot A520*.

```
gphoto2 --list-ports
gphoto2 --list-cameras
```

Deux commandes pour obtenir la description des modes de connexion possibles, et la liste des appareils gérés.

```
gphoto2 --auto-detect
gphoto2 --port usb:002,002 -P
```

Détection d'un éventuel appareil connecté, puis récupération bestiale de toutes les photos (l'option `-P` peut être remplacée par `--get-all-files`).

L'option `--new` permet de se limiter au téléchargement des nouvelles photos.

Pour les drogués de la ligne de commande, l'option `--shell` démarre un environnement interactif, qui permet d'explorer le contenu de votre appareil, et d'interagir avec lui.

Il doit exister plein d'autres tricks'n'tips, il faut juste prendre le temps de les découvrir.

20.2 ufraw

Ufraw (Unidentified Flying Raw) est un convertisseur de fichier images *RAW*, c'est à dire en général les données brutes en sortie du capteur de l'appareil photo.

Chapitre 21

Les intertubes

Ah, le grand Internet sauvage, il serait temps qu'on en parle un peu. Nous allons voir dans ce chapitre quelques utilisations que l'on peut qualifier de « créatives ».

Si vous attendiez des choses plus sérieuses, j'ai aussi ça en magasin : DNS page 44, SSH page 46, streaming page 50, et probablement bien d'autres à venir.

21.1 xinetd

Xinetd est un *super-server daemon* qui gère l'accueil et le lancement pour certains services Internet essentiels tels que `daytime` ou `echo`. Par rapport à son ancêtre `inetd`, décrit en page 30, il offre plus de souplesse dans la configuration et la gestion des droits d'accès.

21.1.1 qotd

À titre d'exercice, nous allons reprendre l'exemple du serveur de *quote of the day* que nous avons déja utilisé avec Inetd, et lancer le petit script par l'intermédiaire de Xinetd.

Pour cela, il faut créer dans le répertoire `/etc/xinetd.d/` [1] un fichier nommé `qotd` et contenant ceci :

```
service qotd
{
disable         = yes
type            = UNLISTED

socket_type     = stream
protocol        = tcp

user            = nobody
server          = /usr/local/bin/qotd.sh
}
```

1. Attention, ce répertoire est peut-être un debianisme, ymmv.

Bien entendu, le script `qotd.sh` décrit en page 30 sera placé dans le bon répertoire. Ensuite, il faut demander à Xinetd de relire ses fichiers de configuration par un signal bien senti et au bon endroit.

21.2 Translation d'adresse

Voir en page 60 pour découvrir comment utiliser une machine connectée en WiFi pour donner accès à Internet à votre réseau local Ethernet...

Mais cette translation d'adresse peut avoir aussi d'autres défits à relever pour en faire une solution complexe et amusante à des problèmes inexistants et datant du siècle dernier. Par exemple la comparaison de la mise en œuvre du NAT avec `iptable` et `pfctl` est une source de trolls sans fin.

21.3 Iodine

Iodine, c'est quoi ? C'est un ensemble de logiciels qui permettent de se brancher sur l'Intertube même si on n'a pas d'accès « autorisé » par la voisine qui a déménagée vers la lointaine campagne avec sa friboite.

> This is a piece of software that lets you tunnel IPv4 data through a DNS server. This can be usable in different situations where internet access is firewalled, but DNS queries are allowed. The name iodine was chosen since it starts with IOD (IP Over DNS) and since iodine has atomic number 53, which happens to be the DNS port number.

À ce jour (2 septembre 2013) ce que vous allez lire est basé sur la version `0.6.0-rc1` et sera probablement plus ou moins incompatible avec les versions précedentes ou suivantes, ymmv.

```
root@plop:iodine-0.6.0-rc1# bin/iodine teton.mooo.com -f -P m
Opened dns0
Opened UDP socket
Sending DNS queries for teton.mooo.com to 192.168.1.1
Autodetecting DNS query type (use -T to override).................
iodine: No suitable DNS query type found. Are you connected to a network?
iodine: If you expect very long roundtrip delays, use -T explicitly.
iodine: (Also, connecting to an "ancient" version of iodined won't work.)
```

21.3.1 Coté serveur

Je pense qu'il faut commencer par voir du coté du DNS pour déléguer une sous-zone à la machine qui va faire tourner le serveur iodine. Mais j'en suis pas certain. RTFM.

21.3.2 Coté client

Je n'en sais pas plus non plus...

```
sudo bin/iodine -f -P s3cr3tp4ssw0rd i.buvette.org
```

21.4 outils de diagnostic

iftop, iptraf ...

21.4.1 tcpdump

C'est clairement un outil dédié au voyeurisme :)

21.4.2 nmap

C'est clairement un outil dédié au voyeurisme :)

21.5 netcat

> The nc (or netcat) utility is used for just about anything under the sun involving TCP or UDP. It can open TCP connections, send UDP packets, listen on arbitrary TCP and UDP ports, do port scanning, and deal with both IPv4 and IPv6. Unlike telnet(1), nc scripts nicely, and separates error messages onto standard error instead of sending them to standard output, as telnet(1) does with some.

Chapitre 22

Internet Relay Chat

22.1 hybrid

C'est le serveur utilisé par Smeuh. Une vrai galère à configurer.

On se retrouve vite dans des situations critiques, du genre un #canal où il n'y a plus de @op, et donc, on ne peut plus changer le topix, et là, **ça craint vraiment** !

22.2 irssi

> Irssi is a modular Internet Relay Chat client. It is highly extensible and very secure. Being a fullscreen, termcap based client with many features, Irssi is easily extensible through scripts and modules.

```
/recode add #gcu ISO-8859-15
pinpin0: çhârsèt ?

/SET autoclose_windows OFF
```

Chapitre 23

Usenet

Je suis un vieux dino du grand USENET mondial. Mes premiers posts doivent dater de quelques mois après l'arrivée de COMPU$ERVE en France. C'est dire... Et quand il m'arrive de parler de cet ancêtre des réseaux sociaux, la question : « Usenet, c'est quoi ? » revient parfois [1]. On va essayer d'y répondre rapidement, avant de passer aux outils qui le font fonctionner.

Usenet was born approximately three decades ago, in 1979. It all began as a small communication network between a few universities in the United States used for the purposes of trading information, sharing news, discussing new and exciting developments, and collaborating on research results and projects.

23.1 slrn

> slrn is an easy to use but powerful NNTP / spool based newsreader. It is highly customizable, supports scoring, free key bindings and can be extended using the embedded S-Lang interpreter.

```
printf 'set wrap_flags 7\nset wrap_width 72\n' >> .slrnrc
```

23.2 leafnode

Leafnode est l'outil essentiel si on désire troller au fond du jardin, en dehors du rayon d'action de sa box.

23.3 INN

Voilà, c'est le moment de parler de l'engin fondamental de ce réseau de discussions. INN.

1. Mais pas assez souvent, à mon gout...

Chapitre 24

Domain Name Service

24.1 host

C'est vraiment la commande de base pour tout ce qui touche à la résolution de nom. Elle propose les premières options de base, avec une syntaxe facile à retenir. Elle existe, sous des formes légèrement déviantes, dans une grande partie des OS [1] existants de nos jours.

```
tth@plop:~ $ host sigfood.dinorama.fr
sigfood.dinorama.fr has address 91.121.221.123
tth@plop:~ $ host -t mx dinorama.fr
dinorama.fr mail is handled by 20 mx1.smeuh.org.
tth@plop:~ $
```

Ces deux exemples sont encourageants. Est-il possible d'aller plus loin ?

24.2 nslookup

> Nslookup is a program to query Internet domain name servers. Nslookup has two modes : interactive and non-interactive. Interactive mode allows the user to query name servers for information about various hosts and domains or to print a list of hosts in a domain. Non-interactive mode is used to print just the name and requested information for a host or domain. »

24.3 dig

Dig (*domain information groper*) est un outil d'interrogation des serveurs DNS. Par défaut, il questionne les serveurs listés dans `/etc/resolv.conf`.

Pour résoudre récursivement un nom de domaine à partir de la racine : $ dig +trace ¡nom.de.domaine¿

Pour le reverse d'une adresse IP : $ dig +trace -x ¡adresse.ip¿

1. même dans le sale, c'est dire.

24.4 delv

Nouvel outil en rapport avec DNSSEC.

24.5 checkresolv

Outil très basique conçu et bricolé par Tonton Th au siècle dernier lors d'un séjour à Frontignan-plage.

24.6 dnsmasq

dnsmasq is a lightweight DNS, TFTP and DHCP server. It is intended to provide coupled DNS and DHCP service to a LAN.

Chapitre 25

ssh

Secure Shell.

D'après le site `openssh.org` : *OpenSSH is a* free *version of the SSH connectivity tools that technical users of the Internet rely on. Users of telnet, rlogin, and ftp may not realize that their password is transmitted across the Internet unencrypted, but it is. OpenSSH encrypts all traffic (including passwords) to effectively eliminate eavesdropping, connection hijacking, and other attacks. Additionally, OpenSSH provides secure tunneling capabilities and several authentication methods, and supports all SSH protocol versions.*

25.1 sshfs

D'après la manpage Linux : *SSHFS (Secure SHell FileSystem) is a file system for Linux (and other operating systems with a FUSE implementation, such as Mac OS X or FreeBSD) capable of operating on files on a remote computer using just a secure shell login on the remote computer. On the local computer where the SSHFS is mounted, the implementation makes use of the FUSE (Filesystem in Userspace) kernel module. The practical effect of this is that the end user can seamlessly interact with remote files being securely served over SSH just as if they were local files on his/her computer. On the remote computer the SFTP subsystem of SSH is used.*

Hélas, ça ne marche pas trop bien depuis un OpenBSD [1] :

```
~ $ uname -a
OpenBSD vierge.thsf.net 5.5 GENERIC#159 sparc64
~ $ sshfs tth@10.20.0.23:/tvbruits 23/
fuse_mount: Permission denied
```

Grace à `<semarie>` [2] : pour un montage en tant que yuser il faut deux choses : les bons droits sur `/dev/fuse*` et un `sysctl kern.usermount=1` bien senti.

La semaine prochaine, nous verrons le *mapping* entre les UID locaux et distants.

1. OpenBSD 5.5 sur sparc64
2. du Groupuscule des Contributeurs d'Unix

Chapitre 26

Hardware

26.1 Diagnostics

dmesg, lshw, lsusb, lspci...

26.2 Ports série

Comment détecter si un *device* correspond bien à un port série ?

```
#include <stdio.h>
#include <fcntl.h>
#include <termios.h>

#define TTY "/dev/ttyS18"

int main(void) {
        int fd = open(TTY, O_RDWR | O_NOCTTY);
        if (fd < 0) {
                perror("open␣" TTY);
                return 1;
        }

        struct termios tio;
        if (tcgetattr(fd, &tio)) {
                perror("tcgetattr␣" TTY);
                return 1;
        }

        return 0;
}
```

26.3 Disques durs

Iozone ? Bonnie ?

26.4 Interface réseaux

On a parfois besoin de savoir dans quel état est une interface réseau (UP, RUNNING dans la sortie de ifconfig). Parfois, je n'aime pas trop *parser* celle-ci à grand coup de *regexp*, en particulier dans un contexte de type Busybox. Et pour ce cas-là, je préfère utiliser un binaire brassé à la maison, et dont voici un fragment de source :

```
int get_if_flags(char *ifname, short *p_flags)
{
int                     fd;
int                     retcode;
struct  ifreq           req;

#if DEBUG_LEVEL
fprintf(stderr, ">>> %s \"%s\" to %p\n",     \
                    __func__, ifname, p_flags);
#endif

/* Sanity check */
if ( strlen(ifname) > (IFNAMSIZ-1) )
        {
        fprintf(stderr, "name %s to long\n", ifname);
        abort();
        }

fd = socket(PF_INET, SOCK_DGRAM, 0);
if (fd < 0)
        {
        perror("socket bla...");
        return -2;
        }

/* populate the struct for the request */
memset(&req, 0, sizeof(req));
strcpy(req.ifr_name, ifname);

/* do the call */
retcode = ioctl(fd, SIOCGIFFLAGS, &req);
if (retcode < 0)
        {
        perror("ioctl SIOCGIFFLAGS");
        close(fd);
        return -1;
        }
#if DEBUG_LEVEL
/* display the result */
fprintf(stderr, "flags = 0x%04x\n", req.ifr_flags);
#endif

close(fd);

*p_flags = req.ifr_flags;
```

```
return 0;
}
```

Hélas, je n'ai pas pu trop tester ce truc sur beaucoup de systèmes, et je vous propose de ne pas trop lui faire confiance pour une application critique...

Chapitre 27

streaming

Avertissement : pour le moment [1], nous n'allons nous pencher que sur la diffusion de flux audios, avec des encodages libres comme *ogg* ou *speex*.

Tout simplement parce que le monde de la vidéo est un gigantesque foutoir de codecs, de containers, de brevets logiciels et d'incompatiblités diverses et variées.

27.1 icecast2

> Icecast, the project, is a collection of programs and libraries for streaming audio over the Internet. This includes : Icecast, the program that streams audio data to listeners, libshout, a library for communicating with Icecast servers and IceS, a program that sends audio data to Icecast servers

Icecast est, semble-t-il, **le** serveur de stream de référence. Il se configure facilement avec des fichiers en XML. Première approche d'une ergonomie différente, vous êtes avertis.

more to come...

27.2 idjc

> Internet DJ Console is an Internet radio application for making a live radio show or podcast. Features include two main media players with a crossfader, a jingle player, microphone signal processing (compressor and noise gate)

27.3 liquidsoap

> Liquidsoap is an extensible and flexible audio stream generator, mainly used for streaming to a shoutcast/icecast server. It is also

1. En juillet 2014

> possible to output in a file, to speakers or as raw PCM stream. liquidsoap can perform any audio processing, mixing streams together, applying filters, generating sound procedurally, there is no limit but your imagination. Input files can be accessed through the network, using samba, ftp or speech synthesis. The system features lots of programming operators, including user requests, time-based selection, sum...

Un étonnant langage de script pour gérer tous ces octets musicaux en les promenant un peu dans tous les sens, la possibilité de commander Savonet avec un simple telnet. Vite, un exemple d'utilisation, vous créez un fichier texte avec cez quelques lignes dedans et vous le sauvez sous le nom `savon.liq` :

```
set("log.file.path","/tmp/savon.log")
set("log.stdout",true)

output.icecast (%vorbis,
                host = "10.20.0.23",
                port = 1984,
                password = "hopla!",
                mount = "savon.ogg",
                single("loveme.ogg"))
```

Avec ce petit programme, nous pouvons envoyer en boucle un fichier Ogg vers notre serveur. Les deux premières lignes permettent de voir et d'enregistrer les diverses actions. Et la grosse instruction `output.icecast` envoie la musique dans votre serveur Icecast2 en passant les paramètres nécessaires.

27.3.1 10.20.0.23

Maintenant, voyons un exemple plus complet et plus réaliste, largement inspiré d'un tutoriel de la documentation officielle, et qui deviendra un de ces jours la base de Radio-Myrys :

```
set("log.file.path",   "/tmp/savon.log")
set("log.stdout",      false)

default = single("loveme.ogg")

openbsd = playlist(mode="random", "files/openbsd.pls")
others  = playlist(mode="random", "files/others.pls")
jingles = playlist(               "files/jingles.pls")

radio = fallback([  switch([({ 5h-11h }, openbsd),
                           ({ 11h-5h }, others)]),
                    default])

radio = random(weights=[1,12],[ jingles, radio ])

output.icecast( %vorbis,
                host      = "localhost",
                port      = 1984,
```

```
                password  = "hopla!",
                mount     = "savon.ogg",
                radio)
```

Ah bah oui, ça commence à être robuste, mais en fait ça semble assez simple à comprendre, et un peu plus compliqué à mettre en œuvre, quand on ne connait encore rien au truc..

On se retrouve vite face à un problème classique : les différences de niveau sonore entre les divers morceaux. Et comme on veut avancer sur d'autres choses, on rache un petit bout de Bash avec de la poudre verte dedans :

```
#!/bin/bash/
echo "que faire ?"
```

Cette poudre verte pourrait bien être `vorbisgain(1)`...

27.3.2 Rotation des playlists

```
pl2014 = playlist("2014")
pl2013 = playlist("2013")
...
jingles = playlist("jingles")
s = rotate([pl2014, pl2013, ..., jingles])
output.icecast(..., s)
```

27.3.3 Évolutions

Pour découvrir d'autres machins, nous allons maintenant partir d'un exemple très simple, un peu analogue au second que nous avons vu sur 10.20.0.23, et nous allons y intégrer une nouvelle fonctionnalité, puis une autre...

La première va répondre à cette question : « Comment insérer une petit pause silencieuse d'une certaine durée entre chacun des morceaux d'une playlist ? »

Avec l'instruction `blank`, probablement...

Et la seconde question sera : « Comment changer la durée de cette pause en temps réel ? ». Pour cela, nous découvrirons l'interface `telnet` du savon.

Voici le script de départ, qui joue une playlist en séquence :

```
set("log.stdout",true)

dflt = single("files/jingles/Imperial.ogg")
shorts = playlist(mode="normal", "files/shorts.pls")
essai = fallback( [ shorts, dflt ] )

output.icecast (%vorbis,
                host = "10.20.0.23",  port = 1984,
                mount = "plop.ogg",   password = "******",
                essai )
```

Pour inserer cette pause à durée fixée entre les morceaux, cette incantation [2] insérée au bon endroit fera l'affaire :

2. fournie par Pierre B. de la *ML* savonet

```
essai = append ( essai, fun (a) -> blank(duration = 4.2) )
```

Oui, je sais, c'est un peu étrange... Il doit y avoir du Ocaml là-dessous.

27.3.4 trucs divers

Que signifie ce message : **That source is fallible.** ? Par exemple, que le fichier que vous aviez soigneusement désigné comme `default` n'existe plus sur le disque, le `loveme.ogg` de l'exemple précédent, par exemple.

Comment envoyer vers plusieurs icecasts ? Bah c'est très facile : il suffit de mettre plusieurs "blocs" `output.icecast` à la suite les uns des autres ;)

Comment insérer une petit pause silencieuse entre chacun des morceaux d'une playlist ? Avec l'instruction `blank`, probablement...

27.4 Source clients

27.4.1 Ezstream

> Ezstream is an audio and video streaming client which feeds any icecast server with audio data and metadata from a standard input without reencoding. It thus requires very little CPU resources. Some playlists can be looped and composed with MP3, Ogg Vorbis or Ogg Theora media files.

27.4.2 darkice

> DarkIce as a live audio streamer. It records audio from an audio interface (e.g. sound card), encodes it and sends it to a streaming server.

27.4.3 VLC

27.4.4 oggfwd

Chapitre 28

Gadgets

28.1 Fortune

Un rapide petit exemple, avant de passer aux choses techniques :

```
tth@plop:~$ fortune
You will outgrow your usefulness.
tth@plop:~$ fortune
Future looks spotty.  You will spill soup in late evening.
tth@plop:~$ fortune
You are deeply attached to your friends and acquaintances.
tth@plop:~$
```

Bref, vous voyez le principe : c'est un machin à raconter des trucs plus ou moins aléatoires [1]. Il y a tout un tas d'options intéressantes, la principale étant `-o` pour avoir les fortunes offensives qui peuvent choquer les âmes sensibles.

Nous allons donc maintenant le plier à notre volonté en créant nous même le contenu adéquat. Pour cela, nous devons tout d'abord préparer un fichier texte contenant tous nos aphorismes, séparés par une ligne contenant uniquement le caractère `%`.

Ensuite, il faut utiliser `strfile` qui va préparer une version indexée de notre fichier texte afin qu'il soit utilisable par fortune.

Le pendant graphique est `xcowsay`.

28.2 /usr/games/

On trouvera dans ce répertoire une collections de jeux anciens, dont certains datent d'une époque où la `vt102` était un terminal de luxe. Certains remontent même à l'antique Unix de Berkeley. Et quelques-uns d'entre eux sont devenus, au fil du temps, de grands classiques.

1. et jamais revenus, il faut bien le préciser.

28.2.1 les générateurs

Les décideurs pressés seront heureux de pouvoir enrichir leurs PointPuissant avec de très écoles-anciennes messages cryptiques avec ces quelques outils ergonomiques et pipelinables avec d'autres fidèles serviteurs.

```
tth@plop $ morse -s we make porn | tr '\n' ' ' ; echo
 .--  .    --  .-  -.-  .    .--.  ---  .-.  -.    ...-.-
tth@plop $ ppt we make porn | tr 'o' '*'
-----------
| *** .***|
| **  .* *|
|  *  .   |
| ** *.* *|
| **  .  *|
| ** *. **|
| **  .* *|
|  *  .   |
| *** .   |
| ** *.***|
| *** . * |
| ** *.** |
-----------
```

Certains esprits ouverts pensent que `fortune` est aussi un générateur de données futilement complexes. D'autres songent à MXDoctor ou Camélia.

28.2.2 Les animations

Ahem, là, c'est délicat, parce que je ne sais pas vraiment [2] d'animation imprimable.

2. Enfin, si, j'essaye de confectionner un soft pour fabriquer des flipbooks à partir de portnawak

Chapitre 29

S-lang

29.1 C'est quoi ?

S-lang est une bibliothèque de fonctions multi-plateforme pour faire des trucs sympathiques dans des consoles texte. L'exemple le plus éblouissant étant le célèbre `slrn` : "s-lang news reader", dont on cause plus longuement dans les pages consacrées à Usenet (page 43).

29.2 slsh

Autour de s-lang, on trouve un interpréteur de scripts.

Voyons un premier exemple d'utilisation, inspiré de l'exemple de la manpage [1], et qui montre l'utilisation en mode interactif :

```
tth@plop:~/Essais/Slang$ slsh
slsh version 0.8.4-1; S-Lang version: 2.2.2

slsh> 3*10;
30
slsh> x = [1:20];
slsh> sum (sin(x)-cos(x));
0.458613317486777 14
slsh> print x;
Usage: print (OBJ [,&str|File_Type|Filename]);
Qualifiers: pager[=pgm], nopager
slsh> ^D
tth@plop:~/Essais/Slang$
```

Cool...

1. vous lisez bien les pages de man ?

Chapitre 30

curses

En fait, nous allons plutôt parler de **ncurses** qui est l'implémentation actuelle de l'ancêtre `curses`. Voic ce qu'en dit une page de man :

> The ncurses library routines give the user a terminal-independent method of updating character screens with reasonable optimization. This implementation is "new curses" (ncurses) and is the approved replacement for 4.4BSD classic curses, which has been discontinued.

30.1 premier exemple

Bien entendu, nous commencerons par l'ECM traditionnel de rigueur.

```
#include <stdio.h>
#include <stdlib.h>
#include <ncurses.h>
int main(int argc, char *argv[])
{
int             key;

if (2 != argc)  exit(1);

initscr();              /* first initialization */
cbreak();               /* no line buffering */
noecho();               /* be silent on input */
keypad(stdscr, TRUE);   /* acces touches curseur */

mvaddstr(10, 3, argv[1]);
refresh();

key = getch();

endwin();
printf("code␣touche␣%d\n", key);

return 0;
}
```

La première étape consiste à initialiser et configurer le moteur interne de ncurses, et à régler l'interaction clavier/écran.

Ensuite, avec `mvaddstr`, nous écrivons le texte passé en paramètre à une position fixée ligne,colonne dans un écran virtuel qui sera ensuite transféré dans l'écran réel par le `refresh()`. Ce principe d'écran virtuel est central dans la manière d'agir de ncurses.

30.2 astuces

Dans une boucle interactive, on va utiliser `getch()`, qui va bloquer tant qu'on n'a pas tapé sur une touche. Mais que faire si on veut utiliser, en plus du clavier, un autre périphérique de saisie, genre un Joystick ?

Bien entendu, on peut utiliser `select(2)`, mais cela nécessite de connaitre le *file descriptor* auquel est attaché le clavier. Dans le contexte classique, on peut assumer que ce sera le `fd` associé à *stdin*, mais il est imprudent de compter là-dessus.

voir `newterm(3)` qui peut donner quelques pistes...

Chapitre 31

WiFi

WIreless FIdelity.

31.1 wicd

Wired and Wireless Network Connection Manager.

Wicd est constitué de deux éléments : un *daemon* qui est responsable de gérer les connections, et plusieurs interfaces utilisateurs. Ces deux parties communiquent par D-Bus.

```
root@plop:~# wicd-cli -S -y
root@plop:~# wicd-cli -l -y
#       BSSID                   Channel ESSID
0       00:25:15:A6:XX:XX       1       NEUF_2ACC
1       CA:25:15:A6:XX:XX       1       SFR WiFi Mobile
2       CA:25:15:A6:XX:XX       1       SFR WiFi Public
3       A0:21:B7:D5:XX:XX       6       NUMERICABLE-35E0
4       00:17:33:D4:XX:XX       11      NEUF_39B4
5       F4:CA:E5:95:XX:XX       10      FreeWifi_secure
6       F4:CA:E5:95:XX:XX       10      FreeWifi
7       16:10:18:47:XX:XX       11      <hidden>
8       00:24:D4:CE:XX:XX       1       FreeWifi
9       00:24:D4:CE:XX:XX       1       FreeWifi_secure
10      16:10:18:47:XX:XX       11      FreeWifi_secure
11      00:24:D4:CE:XX:XX       1       freeboxwifi
12      00:0E:9B:24:XX:XX       7       poisson_bulle
```

La première ligne demande au *daemon* de scanner (option `-S` ou `--scan`) les réseaux disponibles en wireless. Puis la seconde commande demande l'affichage de la liste.

31.2 Un coup de NAT

Une machine connectée en wifi sur une friboite [1] peut facilement faire translateur d'adresse pour un réseau branché sur son eth, à condition de connaitre les incantations magiques (qui sont ici très linuxisantes) :

```
ifconfig eth0 10.9.9.200/24
echo 1 > /proc/sys/net/ipv4/ip_forward
modprobe iptable_nat
iptables -t nat -A POSTROUTING -o wlan0 -j MASQUERADE
```

Ensuite sur les machines du lan, donc du coté de l'ethernet, il faut incanter ce genre de trucs :

```
ifconfig eth0 10.9.9.199/24
route add default gw 10.9.9.200
echo "nameserver 192.168.0.254" >> /etc/resolv.conf
```

Normalement, ça devrait marcher. Sauf avec OpenBSD, mais Arthur ne comprend pas pourquoi.

1. ou le Cisco sauvage de la salle d'AG, hein, mais ça ne sera pas le même nameserver.

Chapitre 32

GIT

Git est un système de gestion de version, très bien expliqué aux quatre coins des interwebs. Je ne vais donc pas m'étendre plus que ça, pour passer tout de suite [1] à un cas particulier que je n'arrive pas encore à vraiment maitriser.

32.1 Démarrage

```
$ mkdir Foo
$ cd Foo/
$ git init
$ > README.txt
$ git add README.txt
```

32.2 Setup du serveur

Sur le serveur distant, dans un endroit accessible par le protocole `HTTP`, il faut créer le dépot.

```
$ mkdir Foo
$ cd Foo/
$ git init --bare
```

Et ensuite :

```
$ git update-server-info
```

32.3 Pousser le code

Notre commit local semble pertinent, il est temps de le faire connaitre au monde.

```
$ git push yuser@serveur:/chemin/vers/le/depot/Foo
```

1. Pour être honnète, je n'arrive pas à me faire à la logique de Git, un peu overkill pour mon usage.

Chapitre 33

Trucs divers

33.1 Permissions

Suite à un accident d'origine indéterminé, Captain ROOT a crée un fichier dans le $HOME d'un pauvre péone. Et celui-ci découvre avec effroi qu'il ne peut ni lire ni modifier ce fichier. Ce que peut confirmer un `ls -l` qui montre bien que seul root a des droits sur ce fichier :

```
-rwx------  1 root  root    70 2012-03-15 12:16 coin.pan
```

33.2 ntp

Network Time Protocol.

Comment monitorer l'écart de date entre deux machines ?

33.3 Prompt

```
printf '\033]11;lemonchiffon2\a'
```

33.4 fdupes

Un utilitaire bien pratique pour découvrir les fichiers dupliqués dans votre disque dur qui se remplit trop vite à votre gout.

Chapitre 34

On est bientôt arrivé ?

Index