

Échanges multimédias sur mesure

Guillaume T. – Novembre 2008

Linux a résolument fait son nid chez les serveurs, les centres de calculs et les systèmes embarqués. Son adoption sur le bureau en est à ses balbutiements, tant dans le monde de l'entreprise que chez les particuliers. Il n'est pas nécessaire de donner beaucoup de crédit aux campagnes polémiques Get The Fact pour admettre que les Active Directory, Sharepoint ou encore Exchange – et les dépendances qu'ils créent côté bureau – ont encore de beaux jours devant eux dans les grandes infrastructures. Si l'on s'intéresse aux plus petits déploiements, il existe un créneau où le potentiel de Linux est tangible et encore largement sous-exploité. Il concerne cette partie de la population pour laquelle l'informatique ne représente pas un loisir mais un mal nécessaire.

Le taux de pénétration de l'Internet en France en fait une infrastructure de communication de choix. Cependant, l'avènement du haut débit s'est aussi traduit par une exclusion de certains abonnés téléphoniques. Des études ont confirmé que la fracture numérique était sensible¹.

On peut la comprendre comme se faisant à deux niveaux. D'abord sur des critères de disponibilité technique : 6% des abonnés téléphoniques avec un débit ADSL limité à 512 kbps, 2% sans ADSL possible, cela fait déjà du monde². L'absence de service d'accès à Internet clef en main est une caractéristique toute aussi rédhibitoire. Le chemin de croix est long, entre les obstacles techniques et une qualité de service douteuse – 60 millions de consommateurs, Que Choisir et autres évoquent régulièrement le sujet avec éloquence.

À l'intersection des utilisateurs qui bénéficient d'une infrastructure adéquate et de ceux qui sont bien décidés à en tirer partie, on trouve les acharnés du Net. Les acharnés conservent méticuleusement leurs profils à jour sur Facebook, Myspace, LinkedIn, Gather et autres sites de réseaux sociaux, et s'en servent à loisir pour échanger toutes sortes de données multimédias. Télécharger une vidéo de 1 méga-octet pour qu'elle soit retaillée côté serveur, rien de plus banal pour eux. Que ce soit à coups de Smartphones, Netbooks ou n'importe quel autre gadget à la mode. Cela ne concerne hélas qu'une frange de la population, qui dédie budget et temps à cette activité à part entière.

Il existe ensuite une autre frange de la population, les pragmatiques, qui ne veulent certainement pas investir et entretenir un budget et un temps abyssaux pour un besoin de communication somme toute assez réduit : faire parvenir rapidement et de manière *asynchrone* du texte, des images, des sons et des vidéos aux membres plus ou moins distants de leurs communautés respectives.

Le besoin de communication *asynchrone* vient aussi bien des acharnés (pour sa compatibilité avec un mode de vie multitâche) que des limitations technologiques ou pratiques (différences de fuseau horaire par exemple).

Un certain nombre d'outils permettent de capturer de manière plus ou moins efficace et conviviale des images, sons ou vidéos, nous ne reviendrons pas dessus. L'acquisition du texte est finalement peu privilégiée. L'acharné est généralement ou bien un virtuose du clavier, ou un adepte du langage SMS. Le pragmatique, lui, s'il passait suffisamment de temps en ligne, serait peut-être membre du « komiT contr le langag SMS é lé fot volontR »³. Il sera en tout état de cause peu enclin à taper laborieusement sa correspondance sur un clavier.

Ces dernières années ont vu se développer avec frénésie des outils en ligne qui fournissent une infrastructure aux réseaux sociaux. Leurs licences d'utilisation sont parfois inquiétantes.

La licence d'utilisateur (*EULA*) de Google Chrome n'arrogait-elle pas à Google *a perpetual, irrevocable, worldwide, royalty-free, and non-exclusive license to reproduce, adapt, modify,*

translate, publish, publicly perform, publicly display and distribute any Content which you submit, post or display on or through, the Services ? Voilà de quoi rendre nerveux, eut égard au volume et à la diversité des informations publiées depuis les navigateurs web. La levée de bouclier qui suivit n'est certainement pas étrangère à la reformulation de ce paragraphe⁴.

Chez Facebook, les termes ne sont guère plus engageants⁵ : *By posting User Content to any part of the Site, you automatically grant, and you represent and warrant that you have the right to grant, to the Company an irrevocable, perpetual, non-exclusive, transferable, fully paid, worldwide license (with the right to sublicense) to use, copy, publicly perform, publicly display, reformat, translate, excerpt (in whole or in part) and distribute such User Content for any purpose [...]*.

Scott McNealy, cofondateur de Sun, aurait dit *privacy is dead, deal with it* – ce qui pourrait être traduit par *faites vous bien à l'idée que le respect de la vie privée est mort et enterré*. Il se peut que l'acharné ait déjà fait son deuil, ou bien qu'il s'évertue à préserver une sérénité tout apparente.

Le pragmatique n'ayant pas ce besoin inhérent d'utiliser les outils dans le vent, son sort n'est pas nécessairement scellé. *A fortiori*, lesdits outils – et les amours de l'acharné pour telle ou telle mise en œuvre de réseau social – ont une tendance à la volatilité qui est bien malvenue.

Le *nuage* des réseaux sociaux n'est donc pas nécessairement la meilleure réponse aux besoins du pragmatique d'échanger des données multimédias. Nous nous attacherons ici à utiliser des outils libres et simples pour remplir ce besoin. Compresser et envoyer par message électronique remplit ces deux critères, le pragmatique n'aurait que faire de procédés plus alambiqués.

À titre d'exemple, l'acharné conseillera (ou installera) Ubuntu au pragmatique pour combler ses besoins multimédias, et plus encore. La distribution Ubuntu nécessite quelques rajouts pour pouvoir opérer à son plein potentiel, cela dû à des questions de licence. Taper *medibuntu* dans la bar d'adresse de Firefox suffit pour apprendre comment installer les extensions multimédias d'Ubuntu, de l'ajout de la clef de signature numérique à l'installation des paquetages.

Il paraît aussi judicieux de se limiter à la version LTS (*Long Term Support*, version d'Ubuntu qui est maintenue sur une durée de 3 ans pour les ordinateurs de bureau). Moins de mise à jour, moins de risque de casse, plus de stabilité dans l'interface utilisateur.

Clique, Clique, Envoyer

Revenons aux exigences de base de notre utilisateur pragmatique : être capable d'envoyer facilement et rapidement des images, des sons et des vidéos. Il existe une pléthore d'utilitaires sous Ubuntu qui, combinés, permettent d'atteindre ce but. Leur utilisation n'est cependant pas unifiée et rarement adaptée à notre pragmatique. Efficacité et minimum de temps face à la machine sont encore une fois les maîtres mots.

Dès lors qu'Ubuntu 8.04 LTS (Hardy) a été installé, l'insertion d'un périphérique multimédia met en route toute la machinerie nécessaire pour copier le contenu sur l'ordinateur. L'enchaînement des étapes qui conduiront ce fichier au destinataire final dépend ensuite principalement de l'inventivité de l'utilisateur, ce qui est malvenu dans notre cas. Nous prendrons donc comme hypothèse de travail que le pragmatique est parvenu à copier le contenu multimédia sur son bureau, et qu'il souhaite à présent le partager avec ses *proches lointains*, eux-même acharnés ou pragmatiques.

Plusieurs facteurs jouent alors en défaveur d'un simple envoi par messagerie électronique, principalement la bande passante montante et les tailles maxima de mails configurées dans les serveurs SMTP (10, 15 mégaoctets).

Moultes utilitaires avec ou sans interface graphique se chargent déjà de ce travail, quel est donc ce besoin qu'ils ne rempliraient pas ?

Picasa de Google est impressionnant de réactivité, relativement simple d'emploi et peut même compresser les photos avant de les envoyer par mail. Cependant, le support de la vidéo est largement perfectible ; le support du son seul ne semble pas être là. *Exit* donc l'interface simple et unique pour les trois types multimédias considérés. L'évolution de Picasa est aussi une source d'inquiétude. Autant l'acharné sera-t-il ravi des dernières fonctionnalités, autant les changements d'interface graphique auront des chances de bloquer le pragmatique qui veut vraiment juste partager des photos. Le tout aggravé par l'absence de licence libre.

F-Spot – l'utilitaire que Hardy lance par défaut quand un appareil photo est connecté – aurait notre

préférence en tant que logiciel libre. Il est encore loin de Picassa en terme de fonctionnalités, mais plus proche de nos besoins pour ce qui est de l'interface graphique. Hélas, toujours pas de support du son ou de la vidéo au-delà de leur simple copie sur le disque dur.

/// Image: 01. Capture-Importer à partir d'un appareil photo.png ///

Ubuntu réagit à l'insertion d'un périphérique multimédia

/// Fin légende ///

Nous prendrons donc l'hypothèse que l'utilisateur connaît le chemin à suivre pour que ses photos, sons et vidéos arrivent sur son ordinateur, et qu'il sait naviguer jusqu'à leur emplacement avec Nautilus. Le paquetage nautilus-sendto se charge alors d'envoyer le fichier, à l'aide d'un simple clic avec le bouton droit de la souris. Le glisser-déplacer depuis Nautilus vers la fenêtre de courrier électronique marche aussi très bien.

Nous nous intéresserons en particulier à la compression du contenu multimédia, avec pour idée que le destinataire le consultera sur son ordinateur – et n'ira pas par exemple essayer d'imprimer des photos retaillées.

Les données vidéos sont les moins arrangeantes en matière de compression. Certains appareils comme des téléphones portables placent leurs vidéos directement dans un conteneur 3GP (version simplifiée du conteneur MPEG-4). Le fichier est alors généralement de taille raisonnable et peut être envoyé directement par courrier électronique.

Beaucoup d'appareils de conception plus ancienne (ou pas) utilisent le conteneur AVI, fourre-tout développé par Microsoft, et l'utilisent pour une grande palette de formats de compression (du JPEG animé au MPEG-4). Nous supposons que la vidéo de départ n'est pas ou peu compressée – typiquement du JPEG animé ou guère mieux. Nous commencerons par estimer l'effet d'un transcodage simple de la vidéo (changer son codeur / décodeur en espérant que cela réduise suffisamment la taille totale du fichier). Puis, si l'étape précédente de transcodage ne s'est pas avérée suffisante, nous réduirons les dimensions de la vidéo (hauteur et largeur).

La suite mplayer / mencoder fait des merveilles en matière de lecture, transcodage et traitement scriptables de vidéos. Pour les utilisateurs d'Ubuntu, Mplayer est disponible chez Medibuntu (<http://www.medibuntu.org/>), suivre en particulier le lien *Didacticiel dépôt*). Les licences d'utilisation doivent être regardées de près ; le terrain peut être miné suivant l'endroit dans le monde où l'utilisateur habite.

Deux solutions s'offrent à nous : se plonger dans les pages de manuel de mencoder, ou s'inspirer de projets existants. Dans l'optique de s'inspirer de ce que les développeurs libres ont fait de mencoder, AcidRip est un morceau de choix. Le paquetage est disponible directement chez Ubuntu. On récupère ses sources via un `apt-get source acidrip`. Dans cet article, on s'intéresse à la version 0.14, livrée avec Ubuntu 8.04 LTS. AcidRip est sous licence GPL, ce qui nous donne la permission d'accéder à son code source et l'obligation de rester sous GPL si nous choisissons d'intégrer toute ou partie de son code dans notre projet.

Comme beaucoup d'interfaces graphiques bien faites, AcidRip a l'avantage de mettre en exergue les paramètres importants de la tâche à accomplir. Ici, pour une compression : dimensions de la vidéo (nombre de pixels en ligne, `hs`, et en colonne, `ws`), débit vidéo (`br`), efficacité de la compression en bits par pixel (`bpp`). Un autre paramètre clef est caché par l'interface graphique : le nombre d'images par seconde (`fps`). On trouve l'équation qui les relie à la ligne 1073 de `AcidRip/acidrip.pm` :

```
$bpp = sprintf( "%.3f", ( $br * 1000 ) / ( $hs * $ws * $fps ) ) if $ws * $hs;
```

Le nombre de bits par pixel est un bon indicateur de la qualité de la vidéo, à résolution et nombre d'images par seconde constants. Les valeurs intéressantes de `bpp` sont encadrées dans les lignes 188 et suivantes de `signals.pm`. On peut les résumer dans un tableau – traduction aux bons soins de votre serveur :

BPP	Qualité apportée par ce débit en bits par pixel (BPP)
0-0,10	Beaucoup trop basse ! Augmentez la taille du fichier ou diminuez la résolution.
0,10-0,15	Assez basse, il est probable que le résultat ne soit pas folichon, augmentez la taille du fichier ou diminuez la résolution.
0,15-0,20	Bon, on verra un peu les blocs MPEG, mais ça ira.
0,20-0,25	Super, le résultat devrait vraiment être bon.
0,25-0,30	Trop élevée, pas grand chose à gagner de ce côté là. Considérez augmenter un peu la résolution.

Nous ne prendrons la décision désespérée de modifier les dimensions de la vidéo que si le transcodage n'est pas suffisant. Le nombre d'images par seconde est aussi une qualité intrinsèque qui a un impact direct sur la perception de qualité de la vidéo, gardons-nous de le modifier – sauf si un codec nous y force. La dernière inconnue est alors le débit vidéo, que nous pouvons dériver facilement de l'égalité citée précédemment. Avec les bonnes unités :

$$br = bpp \times hs \times ws \times fps$$

On peut alors envisager d'écrire notre réducteur de fichiers multimédias, pourquoi pas en Perl puisque AcidRip semble si bien s'en tirer. Appelons-le **2petit.pl** et plaçons-le sous les bons auspices de la licence libre GPL. On commence par les invocations usuelles :

```
#!/usr/bin/perl -w
use strict;
```

Le script utilise certaines constantes, qu'on définit en tête de fichier pour pouvoir les modifier facilement si besoin est :

```
use constant {
    MPLAYER      => "/usr/bin/mplayer",
    MENCODER     => "/usr/bin/mencoder",
    CONVERT      => "/usr/bin/convert",
    LAME         => "/usr/bin/lame",
    EVOLUTION    => "/usr/bin/evolution",
    GNOMEVFSINFO => "/usr/bin/gnomevfs-info",
    ZENITY       => "/usr/bin/zenity",
    TMP          => "/var/tmp",
    NEWBPP       => 0.2,
    NEWAUDIOBPS => 192000,
    NEWMAXVKBYTES => 8000,
    NEWIMGSIZE   => 1024,
    NEWIMGQUAL   => 85,
};
```

NEWBPP est l'efficacité de la compression en bits par pixel. Le débit audio visé **NEWAUDIOBPS** est pris à 192 kbps (qui vaut bien 192 000 bps, pas de 1 024 qui tiennent ici). On se fixe la taille maximale des vidéos compressées **NEWMAXVKBYTES** à 8 000 kilooctets (un peu moins de 8 mégaoctets), et la plus grande dimension des images compressées **NEWIMGSIZE** à 1 024 pixels.

Occupons nous rapidement de la petite maintenance : la fonction **petit** s'occupe de définir le nom du fichier compressé en rajoutant *_petit* avant l'extension.

```
sub petit {
    my $file = shift();
    my $ext  = shift();
    my $newfile = $file;
    $newfile =~ s/^(.*)\.[^.]+$/$_1_petit.$ext/;
    return $newfile;
}
```

quote se charge d'isoler les noms de chemin et de fichier par des apostrophes, utile quand ils sont envoyés au shell :

```
sub quote {
    my $string = shift();
    return "'" . $string . "'";
}
```

Nous rentrons alors dans le corps du sujet, la compression vidéo. Le fichier d'exemple `/usr/share/doc/mplayer/examples/midentify` – ou bien `acidrip.pm` – donne les bons paramètres pour demander à mplayer d'extraire les caractéristiques de la vidéo de départ. On citera `ID_VIDEO_WIDTH` (largeur de la vidéo en pixel), `ID_VIDEO_HEIGHT` (sa hauteur), `ID_VIDEO_FPS` (nombre d'images par seconde).

On se propose d'archiver ces paramètres dans une table de hachage, avec en clef le nom du paramètre et en élément sa valeur :

```
sub vidparam {
    my $file = shift();
    my $param;
    open PARAMVID, MPLAYER . " -demuxer 35 -nocache -identify -ao null -vo null -frames 0
$file 2>/dev/null |";
    while (<PARAMVID>) {
        $param->{$1} = $2 if (/^(\\w_+)=((\\d\\.)+)$/);
    }
    close PARAMVID;
    return ($param);
}
```

Le `-demuxer 35` force l'utilisation du démultiplexeur de libavformat. Ce choix vient de problèmes rencontrés par votre serviteur avec le démultiplexeur que mplayer choisit de sa propre initiative – par exemple `-demuxer 3`, le démultiplexeur avi. `mplayer -demuxer help` fournit la liste des démultiplexeurs disponibles. Une des évolutions possibles de `vidparam` serait d'utiliser d'autres démultiplexeurs quand celui de libavformat échoue à la tâche.

On peut alors mettre en œuvre l'équation établie plus haut et qui fournit le débit vidéo en bits par seconde :

```
sub newvideobps {
    my $param = shift();
    my $reductionf = shift() || 1;
    my $ws = int( $param->{ID_VIDEO_WIDTH} / $reductionf );
    my $hs = int( $param->{ID_VIDEO_HEIGHT} / $reductionf );
    my $fps = $param->{ID_VIDEO_FPS};
    return ( int( NEWBPP * $hs * $ws * $fps ) );
}
```

Cette fonction prend en paramètres la table de hachage créé précédemment et un facteur optionnel de réduction à appliquer sur les dimensions (largeur et hauteur) de la vidéo.

On est alors capable de calculer la taille en kilooctets du fichier vidéo compressé. La taille n'est finalement que le produit de la durée de la vidéo par le débit multimédia (audio plus vidéo), en tenant compte des unités.

```
sub newfilesizekB {
    my $param = shift();
    my $reductionf = shift() || 1;
    my $newvideobps = newvideobps( $param, $reductionf );
    my $lengths = $param->{ID_LENGTH};
    return ( int( ( ( NEWAUDIOBPS + $newvideobps ) * $lengths ) / 8192 ) );
}
```

```
}
```

Il nous manque alors une fonction, celle qui va calculer le paramètre de réduction des dimensions (largeur et hauteur) de la vidéo pour qu'elle rentre dans la débit vidéo voulu.

Si on repart de

$$br = bpp \times hs \times ws \times fps$$

en appliquant le facteur f à hs et ws , on obtient

$$br = f^2 \times bpp \times hs' \times ws' \times fps$$

d'où le

$$f = \sqrt{\frac{br}{bpp \times hs' \times ws' \times fps}}$$

Mise en œuvre :

```
sub getvideoredfac {  
  my $param = shift();  
  my $desiredvideorate = shift();  
  my $oldws = $param->{ID_VIDEO_WIDTH};  
  my $oldhs = $param->{ID_VIDEO_HEIGHT};  
  my $fps = $param->{ID_VIDEO_FPS};  
  return ( sqrt( $desiredvideorate / ( NEWBPP * $oldws * $oldhs * $fps ) ) );  
}
```

On se jette alors à l'eau, la compression vidéo à proprement parler :

```
sub vidrecompress {  
  my $file = shift();  
  my $param = vidparam($file);  
  my $scale = "";
```

Si la taille (calculée) de la vidéo compressée dépasse la limite fixée en tête de script, alors

- on calcule le débit vidéo correspondant à la taille de fichier maximum, compte tenu de la durée de la vidéo et de son débit audio une fois compressée ;
- on en tire le paramètre de réduction optimal, ce qui permet de calculer les dimensions (largeur et hauteur) de la vidéo compressée ;
- on arrondit ces dimensions aux premiers multiples de 16 inférieurs – les blocs JPEG et MPEG ont *souvent* pour taille 16 par 16. Tant qu'on est à enlever des pixels, autant tomber sur des dimensions qui arrangent les codecs. Pour multiplier par 16 (2^3), on décale de trois bits, d'où le >> (la division par 16 se fait en décalant dans l'autre sens, d'où le <<). L'inconvénient majeur de cette démarche est que les proportions initiales risquent de ne pas être respectées à la lettre. Pas de problème *a priori* pour mplayer qui peut gérer des pixels qui ne sont pas carrés, d'autres lecteurs vidéos ne s'en tireront pas aussi bien.

La chaîne de caractères `$scale` se charge de retailler la vidéo (changer ses dimensions) si besoin est.

```
if ( newfilesizeKB($param) > NEWMAXVKBYTES ) {  
  my $lengths = $param->{ID_LENGTH};  
  my $videorate = int( ( 8192 * NEWMAXVKBYTES / $lengths ) - NEWAUDIOBPS );  
  my $redfac = getvideoredfac( $param, $videorate );  
  my $newws = $param->{ID_VIDEO_WIDTH} * $redfac;  
  my $newhs = $param->{ID_VIDEO_HEIGHT} * $redfac;  
  $newws = $newws >> 3 << 3;  
  $newhs = $newhs >> 3 << 3;  
  $scale = "-vf scale=" . $newws . ":" . $newhs . " ";  
  $param->{ID_VIDEO_WIDTH} = $newws;  
  $param->{ID_VIDEO_HEIGHT} = $newhs;  
}
```

On est alors en mesure de composer la ligne de commande `mencoder`, avec retaillage des dimensions de la vidéo si besoin est. On choisit de recompresser en deux passes : la première passe

écrit un fichier de statistiques, la deuxième prend des décisions de contrôle de débit en conséquence.

```
my $newvideobps = newvideobps($param);
print "New targetted file size is ", newfilesizeB($param), "kB\n";
my $mencoder1 = MENCODER
. " '$file' -oac mp3lame -lameopts abr:br="
. int( NEWAUDIOBPS / 1000 )
. " -ovc lavc -lavcopts vcodec=mpeg4:vhq:v4mv:vqmin=2:vbitrate="
. $newvideobps
. ":vpass=1 "
. $scale
. "-o /dev/null";
`$mencoder1`;
my $mencoder2 = MENCODER
. " '$file' -oac mp3lame -lameopts abr:br="
. int( NEWAUDIOBPS / 1000 )
. " -ovc lavc -lavcopts vcodec=mpeg4:vhq:v4mv:vqmin=2:vbitrate="
. $newvideobps
. ":vpass=2 "
. $scale . "-o "
. quote( petit( $file, "avi" ) ) . "";
`$mencoder2`;
unlink("divx2pass.log");
return ( petit( $file, "avi" ) );
}
```

On s'arrêtera là pour le traitement de la vidéo. Il y a beaucoup de choses à en redire – on s'est comporté de manière quelque peu cavalière avec les proportions de la vidéo, on pourrait augmenter le nombre de passes, il existe d'autres codecs, d'autres conteneurs qu'AVI, etc. – mais l'essentiel est écrit.

Il est assez rapide de faire un sort à des images statiques, en s'aidant de la très bonne suite ImageMagick :

```
sub jpgrecompress {
my $file = shift();
my $size = NEWIMGSIZE;
my $cmd = CONVERT
. " -quality 90 -geometry "
. $size . "x"
. $size . " "
. quote($file) . " "
. quote( petit( $file, "jpg" ) );
`$cmd`;
return ( petit( $file, "jpg" ) );
}
```

Beaucoup d'appareils d'enregistrement de son ont la mauvaise idée de ne pas proposer d'alternative à MP3 comme format de compression. L'excellent RockBox 3.0 (<http://www.rockbox.org>) aide quelque peu à se passer de ce format *encombré*, mais dans un but d'interopérabilité il serait malvenu de ne pas tenir compte de MP3 :

```
sub mp3recompress {
my $file = shift();
my $cmd = LAME . " -v -V 9 " . quote($file) . " " . quote( petit( $file, "mp3" ) );
`$cmd`;
return ( petit( $file, "mp3" ) );
}
```

Il faut alors une fonction qui nous informe de la nature du fichier multimédia. Pour se faire, il est possible de regarder l'extension du fichier. Un mécanisme plus puissant existe depuis la RFC 2045, les types MIME. Gnome a la bonne idée de fournir un utilitaire qui se charge d'extraire le type MIME de tout fichier, `gnomevfsinfo`. On écrit une fonction Perl autour :

```

sub getmimetype {
    my $file = shift();
    open MIMETYPE, GNOMEVFSINFO . " " . quote($file) . " |";
    my $mimetype;
    while (<MIMETYPE>) {
        $mimetype = $1 if (/^MIME type\s*:\s*([\s]+)\s*$/);
    }
    close MIMETYPE;
    return ($mimetype);
}

```

Nous pouvons alors écrire le cœur du script. Le nom du ou des fichiers à traiter vient en argument, le type MIME nous dira qu'en faire. L'excellent paquetage Zenity nous permet d'indiquer à l'utilisateur que sa demande a été prise en compte, en affichant une petite interface graphique. L'option `--progress` crée une barre de progression, prenant en entrée standard le progrès de la tâche en pourcentage. Une fois le programme Zenity lancée depuis Perl, il suffit donc de lui fournir l'avancement de la tâche en cours pour qu'il se charge de mettre à jour la barre de progrès.

```

/// Image: Capture-2petit.png ///

```

Barre de progrès créée par Zenity

```

/// Fin légende ///

```

```

open ZEN, "| " . ZENITY . ' --auto-close --progress --text="Compression en cours"
--title="2petit" --auto-kill';
my $i = 0;

foreach my $file (@ARGV) {
    $i++;
    print ZEN int( 100 * $i / ( $#ARGV + 1 ) - 1 ) . "\n";
    my $newfile;
    my $mimetype = getmimetype($file);
    SWITCH: for ($mimetype) {
        m{^video/(mpeg|quicktime|x-msvideo)} && do { $newfile =
vidrecompress($file); last; };
        m{^image/(jpeg|tiff)} && do { $newfile = jpgrecompress($file);
last; };
        m{^audio/(x-wav|mpeg|midi)} && do { $newfile = mp3recompress($file);
last; };
        warn "Unsupported MIME type: $mimetype";
    }
}
close ZEN;

```

Une expression régulière sur le type MIME se charge d'appeler la fonction de compression adaptée. Là encore, plusieurs améliorations sont envisageables, à commencer par une gestion plus convaincante de l'action du bouton *Annuler*.

L'ébauche de script proposée nécessite l'installation d'un certain nombre de paquetages d'Ubuntu et de Medibuntu. Une méthode générale est assez intéressante dans ce cas de figure : `auto-apt`, fourni par le paquetage du même nom, se charge d'installer automatiquement le paquetage contenant un fichier exécutable.

Une fois le fichier `sources.list` mis à jour avec les entrées de Medibuntu, il convient d'informer `auto-apt` de la liste des fichiers contenus dans chaque paquetage. La commande `$ sudo auto-apt update` s'en charge.

Il suffit alors de lancer en ligne de commande le script qui nécessite de nouveaux paquetages pour qu'`auto-apt` se charge de les installer :

\$ sudo auto-apt run 2petit.pl

Le sudo n'est pas obligatoire pour installer les paquetages dans la mesure où auto-apt invoque sudo lui-même, mais dans ce cas le script **2petit.pl** échoue à la première invocation. Il faut aussi noter qu'auto-apt n'installe pas les paquetages qui sont seulement suggérés par la base de données apt – w32codecs et libdvdcss2 font partie des victimes collatérales dans notre cas particulier.

Toutes ces contorsions sont éducatives, mais il est totalement proscrit de les imposer au pragmatique. *Clique, clique, envoyer*, dit le concept.

Le paquetage nautilus-actions offre une manière élégante de lancer notre script. Ce logiciel permet en effet de rajouter des actions au menu contextuel de Nautilus (clic droit de la souris). Le menu **Configuration des actions de Nautilus** dans **Système, Préférences** permet de créer de nouvelles actions, d'en importer à partir de fichiers XML ou encore d'exporter des actions existantes. Il en existe déjà pléthore sur le web⁶.

On suppose qu'on a installé le script de compression sous `/usr/local/bin/2petit.pl`. Le script appartient idéalement à `root:root`, avec permission d'exécution pour tout le monde (755 par exemple).

/// Image: Capture-Édition de l'action « 2petit ».png ///

Action 2petit

/// Fin légende ///

Les paramètres du script sont ici assez simples : %M représente le chemin dans le système de fichiers local. On se prend à rêver d'utiliser les URI (littéralement *identifiant uniforme de ressource*) de Gnome, pour que **2petit.pl** puisse fonctionner sur des systèmes de fichiers distants. Hélas, les logiciels de manipulation multimédia dont nous nous servons dans le script ne permettent pas une intégration aussi fine avec Gnome.

/// Image: Capture-Édition de l'action « 2petit »-1.png ///

Conditions

/// Fin légende ///

L'action n'apparaît dans le menu contextuel que si certaines **conditions** sont remplies. Les conditions peuvent porter sur l'extension du fichier ou sur le type MIME. On prétend ici prendre en charge les vidéos Quicktime (video/quicktime), MPEG (video/mpeg) et AVI (x-msvideo) ainsi que tous les sons (audio/*) et toutes les images (image/*). La définition exhaustive des types MIME se trouve dans `/etc/mime.types`.

Le dernier onglet permet de préciser le protocole d'accès au fichier. Par souci de simplicité, on s'arrête aux fichiers locaux.

/// Image: Capture-Édition de l'action « 2petit »-2.png ///

Conditions avancées

/// Fin légende ///

Il faut alors redémarrer nautilus pour qu'il prenne connaissance de ses nouvelles actions.

2petit.pl peut être amélioré d'une multitude de manières : en précisant le lieu où le fichier compressé doit être sauvegardé (Zenity à la rescousse), en proposant l'envoi direct du courrier

électronique (utilisation de `mailto:?attach=[Chemin vers le fichier compressé]` en argument d'Evolution), etc.. Ce début de script fournit cependant une compression aisée des fichiers multimédia trouvés communément, permettant au pragmatique de les échanger par courrier électronique sans trop de contorsions.

Virtualiser pour une meilleure aide à distance

Même si notre pragmatique se retrouve avec tous les outils dont il a besoin à portée de souris, le problème de la maintenance à distance demeure plein et entier. Tout pragmatique, fut-il aux besoins particulièrement bien définis, rencontrera à un moment donné une difficulté qui nécessitera une forme de support à distance. Quel acharné n'a jamais eu à offrir un service de maintenance à un *John Doe* ou une *Madame P...*

Pour restreindre les interventions, choisir une distribution qui ne se mette à jour que rarement est salvateur. Par exemple, se cantonner aux version LTS (*Long Term Support*, support à long terme) d'Ubuntu. Cela garantit 3 ans de support, et donc une mise à jour au plus tous les trois ans.

D'une part, l'effort d'apprentissage pour se mettre aux toutes dernières techniques sera moindre. Par exemple, voir le gestionnaire de photos f-spot arriver par défaut avec la version 8.04 d'Ubuntu était une bonne nouvelle pour les acharnés, mais a aussi potentiellement impliqué un changement dans le flux de travail du pragmatique. D'autre part, la mise à jour d'un système d'exploitation est une opération essentiellement risquée – en particulier si elle est effectuée sans accès physique.

Deux outils sont donc nécessaires pour aider efficacement un pragmatique : un système jumeau et un accès à distance.

Le système jumeau peut être créé et maintenu via une des nombreuses solutions de virtualisation. Cela permet de ne pas mobiliser un véritable ordinateur qui aurait pour but unique de ressembler trait pour trait à la machine du pragmatique. L'autre avantage est de pouvoir tester des mises à jour ou des opérations potentiellement destructrices, puis de revenir en arrière si besoin est.

De nombreuses solutions libres existent, de Xen à VirtualBox en passant par QEMU.

QEMU associé avec QtEmu (interface graphique) permet de créer rapidement une machine virtuelle avec très peu de modifications sur le système hôte. Dans QtEmu, il suffit de préciser le nom du périphérique sur lequel le CD d'installation d'Ubuntu est monté (ou bien l'emplacement du fichier ISO), d'indiquer que l'on souhaite démarrer depuis le CD ROM et l'affaire est entendue. Dans certains cas, il peut être nécessaire d'ajouter certaines options au noyau invité comme `acpi=force`. *module-assistant* peut alors être utilisé pour compiler le module noyau `kqemu`, qui améliore nettement les performances d'un invité i386 sur un hôte i386. Il conviendra alors d'installer le paquet `kqemu-common` et de s'assurer que le périphérique d'accélération est accessible par le group `adm` par exemple, en éditant un des fichiers `/etc/udev/rules.d` (ou en en créant un qui concentre les spécificités locales) :

```
KERNEL=="kqemu", NAME="%k", MODE="0660", GROUP="adm"
```

Pour savoir si l'accélération `kqemu` est opérationnelle, il suffit de presser `<CTRL>+<ALT>, <2>` sous QEMU puis de taper `info kqemu`.

L'accélération optimale est obtenue en lançant `qemu` avec l'option `-kernel-kqemu`. L'interface graphique de QtEmu autorise l'ajout de telles options.

VirtualBox, dans sa version non GPL, demande moins de contorsions pour des performances époustouflantes. Le mode d'installation indiqué à http://www.virtualbox.org/wiki/Linux_Downloads permet d'aboutir à un système opérationnel en peu de temps sur Ubuntu 8.04. Une fois le paquetage `virtualbox-2.0` installé, le menu **Outils Systèmes** permet de lancer directement l'interface graphique très intuitive de VirtualBox. On n'oubliera pas d'ajouter les utilitaires pour système invité. À défaut d'être libre, cette solution est gratuite pour utilisation personnelle – ce qui inclut une utilisation *personnelle à but commercial*⁷.

La version GPL de VirtualBox se trouve dans le paquetage `virtualbox-ose` chez Ubuntu 8.04 ; elle ne fournit pas le support RDP, USB, iSCSI et Serial ATA. Le projet VirtualBox ne construit pas de

paquetages, évitant ainsi de dupliquer le travail des distributions. Des paquetages tout faits associés à quelques fonctionnalités supplémentaires orientées monde de l'entreprise pour la version *payante*, juste le code source pour la version GPL : ces gens de Sun connaissent décidément bien le monde du libre.

La cerise VirtualBox sur le gâteau de la virtualisation vient sans nulle doute de l'intégration très fine des gestionnaires de fenêtres de l'hôte et de l'invité en mode *seamless*.

Dans le monde de *l'accès à distance*, les outils ne manquent pas. Parmi eux, celui fourni de base avec Ubuntu a beaucoup des fonctionnalités que l'on attendrait – si ce n'est qu'il n'offre pas le chiffrement des données transmises sur le réseau. Cela peut poser un problème par exemple si le pragmatique demande de l'aide à la navigation sur des sites confidentiels (commerce en ligne, gestion bancaire...). Une intervention à distance que le pragmatique peut suivre en temps réel sur son écran présente toutefois des vertus pédagogiques certaines.

Pour ce qui est de la *maintenance à distance*, un simple accès console via SSH permet de résoudre beaucoup de problèmes. Si besoin est, il peut être adjoint d'un accès graphique freeNX⁸ (dont les clefs SSH auront été régénérées). L'authentification par clefs publiques a été couverte à de multiples reprises dans la littérature, nous n'y reviendrons pas. Des incidents de sécurité arrivent cependant rarement mais régulièrement avec SSH, le dernier en date concernant le caractère aléatoire des clefs créées⁹. Il vaut donc toujours mieux garder un œil sur SSH, avec deux mesures en particulier :

- Filtrage des adresses IP source *via* netfilter, le pare-feu Linux ;
- Surveillance de l'activité SSH *via* fail2ban.

Dès son installation, fail2ban commence à surveiller les fichiers de journaux `/var/log/auth.log`. Il se charge alors de bloquer les indésirables insistants en modifiant dynamiquement les règles du pare-feu netfilter. On le voit à l'action dans le fichier `/var/log/fail2ban.log`, une première fois par un

```
YYYY-MM-DD HH:MM:SS,NNN fail2ban.actions: WARNING [ssh] Ban A.B.C.D
```

puis quelque temps plus tard (configurable) par un

```
YYYY-MM-DD HH:MM:SS,NNN fail2ban.actions: WARNING [ssh] Unban A.B.C.D
```

La méthode recommandée pour configurer fail2ban est de rajouter un fichier `/etc/fail2ban/jail.local` qui contient les modifications désirées par rapport au comportement par défaut. Si l'on veut par exemple bannir les indésirables pendant 24 heures après trois essais infructueux et activer un peu plus de méthodes de détections, on peut commencer par ceci :

```
[DEFAULT]
bantime = 86400
maxretry = 3
```

```
[ssh]
enabled = true
```

```
[pam-generic]
enabled = true
```

```
[ssh-ddos]
enabled = true
```

[ssh] met en œuvre le filtre sshd (cf. `jail.conf`) et se charge de réagir aux messages du genre `^%(__prefix_line)s[il](?:!legal|nvalid) user .* from <HOST>\s*$`

La configuration complète du filtre se situe dans `/etc/fail2ban/filter.d/sshd.conf`.

[pam-generic] réagit à tous les messages d'authentification ratée *via* PAM. [ssh-ddos] s'intéresse en particulier aux messages en `Did not receive identification string from` – un trop grand nombre d'entre eux peut indiquer une tentative d'attaque par dénis de service.

Le dernier service que nous aborderons sera celui de la sauvegarde. Dans l'optique de rendre la maintenance de l'ordinateur aussi transparente que possible, il n'est guère raisonnable de demander au pragmatique de penser à faire des sauvegardes – et de les faire manuellement. Il est encore moins raisonnable de ne pas faire de sauvegardes.

Le paquetage `rsnapshot` a l'avantage d'être simple à appréhender et à mettre en œuvre. `rsnapshot` organise des sauvegardes incrémentales dans des répertoires, et utilise des liens en dur (hard links) pour que les fichiers qui ne changent pas d'une itération de sauvegarde à l'autre ne prennent pas de place sur le disque dur. Peu de modifications sont nécessaires au fichier de configuration par défaut, `/etc/rsnapshot.conf`.

Il peut être utile de franciser les intervalles de sauvegarde, par exemple en

```
interval [TAB] quotidien [TAB] 7
interval [TAB] hebdomadaire [TAB] 4
interval [TAB] mensuel [TAB] 6
```

Il convient alors de modifier le fichier de cron en conséquence (`/etc/cron.d/rsnapshot`).

On aura un peu plus bas dans `/etc/rsnapshot.conf` :

```
backup [TAB] /home/ [TAB] local/
```

Toutes les autres directives `backup` sont laissées en commentaire.

Le résultat est l'arborescence suivante à partir de `rsnapshot_root` :

```
quotidien.0/local/home
quotidien.1/local/home
...
hebdomadaire.0/local/home
hebdomadaire.1/local/home
...
mensuel.0/local/home
mensuel.1/local/home
```

Quand à la perspective de renommer *home* en *maison*... L'effort d'internationalisation d'Ubuntu n'a pas (encore) osé aller jusque là.

Une solution élégante au problème du stockage des sauvegardes est d'ajouter un second disque dur, connecté par USB – ou mieux, SATA – à l'ordinateur sauvegardé. Si `snapshot_root` se trouve sur un disque qui n'est pas monté en permanence, il est très utile de préciser

```
no_create_root [TAB] 1
```

Cela permet d'abandonner la sauvegarde dès lors que le disque de sauvegarde est indisponible.

Pour éviter de ralentir le système pendant la sauvegarde, la stratégie n'est pas tant d'éviter la monopolisation du CPU que celle du système d'entrée sortie (I/O). Dans le script cron qui lance les sauvegardes (par défaut `/etc/cron.d/rsnapshot`), on fera donc précéder l'appel à `rsnapshot` par `/usr/bin/ionice -c3` pour que `rsnapshot` n'ait d'accès I/O qu'une fois que tous les autres processus se soient servis.

Enfin, il est possible de laisser à cron la latitude de monter le disque avant la sauvegarde, et le démonter à la fin. Auquel cas il est aussi envisageable de mettre le disque en veille une fois qu'il a effectué sa tâche (essentiellement pour éliminer le bruit du à sa rotation). Pour un disque de sauvegarde monté en deuxième disque SATA, la mise en veille se ferait par

```
/sbin/hdparm -y /dev/sdb
```

Nous avons donc jeté les bases d'ajustements à apporter à un Linux multimédia sur le bureau, adopté depuis belle lurette par des acharnés de la chose informatique, pour en faire un outil de pragmatiques. La robustesse et l'ubiquité du protocole SMTP font de la messagerie électronique un service de choix pour relier acharnés et pragmatiques, en attendant qu'une plateforme libre s'attache à relier les nœuds les moins favorisés de la société de l'information.

- 1 Fracture numérique, http://fr.wikipedia.org/wiki/Fracture_num%C3%A9rique#La_France_et_la_fracture_num.C3.A9rique
- 2 Zones d'ombre, zones blanches et nouveaux NRA,
<http://www.arcep.fr/fileadmin/reprise/dossiers/collectivites/pdf/info-res-zblanches-0406.pdf>
- 3 http://fr.wikipedia.org/wiki/Nouvelles_formes_de_communication_%C3%A9crites
- 4 <http://googleblog.blogspot.com/2008/09/update-to-google-chromes-terms-of.html>
- 5 <http://www.facebook.com/home.php#/terms.php?ref=pf>
- 6 <http://doc.ubuntu-fr.org/nautilus-action#actions>
- 7 http://www.virtualbox.org/wiki/Licensing_FAQ
- 8 <http://freenx.berlios.de/>
- 9 <http://lists.debian.org/debian-security-announce/2008/msg00153.html>