De Ubuntu à Debian: Usage avancé des outils APT

Lorsqu'il y a quelques mois, j'ai découvert la distribution CrunchBang[1], j'ai su que j'avais enfin trouvé la distribution Linux qui était faite pour moi! Je ne vais pas revenir sur ce qui fait le charme de cette version légère, rapide et ergonomique basée sur Ubuntu Linux[2] et OpenBox[3].

Illustration 1: Bureau de la distribution CrunchBang Linux

En effet, j'ai déjà consacré à cette version de l'environnement Linux un article intitulé « CrunchBang: petit mais Ubuntu!» paru dans l'édition de Juillet/Août 2008 du magazine Linux+[4].

Mon bonheur aurait été complet si CrunchBang était basé sur Debian[5] plutôt que sur Ubuntu. En effet, Ubuntu est une distribution parfaite pour mettre en œuvre simplement et rapidement un environnement de travail complet et pré-configuré. Cet environnement s'adresse à un public intéressé avant tout par l'utilisation quotidienne du PC (que ce soit comme station de travail/jeux ou serveur). De plus, la facilité d'installation, le large inventaire de périphériques reconnu et son ergonomie suffisamment proche de Windows pour ne pas dérouter les débutants a fait de cette version de Linux la star des InstallParty.

Forcément, pour aboutir à ce résultat, des choix techniques ont dû être fait par les équipes de

1http://crunchbang.org/projects/linux/

2http://www.ubuntu-fr.org/

3http://icculus.org/openbox/

4http://lpmagazine.org/prt/view/actualites.htm

5http://www.debian.org/index.fr.html

développement de Canonical[6] et des spécificités à Ubuntu rende le «bricolage» du système parfois plus compliqué que prévu. Je ne me considère pas comme un Geek mais je suis un grand curieux et aussi j'ai très vite décidé que Ubuntu n'était pas pour moi. J'ai donc choisi Debian (dont Ubuntu dérive d'ailleurs) et même la version CrunchBang n'a pas réussi à me faire changer d'avis!

Toutefois, fort du principe que Linux est un système merveilleusement souple et versatile, je me suis dit qu'il devait être possible d'obtenir un bureau graphique très proche de celui de CrunchBang sous Debian. J'ai entrepris cette tâche et ce fût l'occasion d'explorer la gestion de paquets logiciels dans l'univers Debian/Ubuntu.

Cet article retrace les différentes étapes et les outils utilisés pour passer d'une CrunchBang Ubuntu à une CrunchBang Debian.

1) La gestion des paquets logiciels sous Debian/ Ubuntu

Traditionnellement, sous Linux, pour installer un programme, il faut le recompiler à partie de son code source. C'est d'ailleurs pour cela que l'on parle de logiciel « Open Source »; l'utilisateur final a un accès libre au code source qui compose le programme. Par cette méthode vous êtes certains d'avoir un exécutable parfaitement optimisé à votre environnement logiciel et matériel. En contrepartie, cette approche est aussi assez laborieuse. Certaines compétences très techniques et très spécifiques sont requises pour compiler correctement un programme et au final, cette opération peux être assez longue; à titre d'exemple, la compilation d'un noyau Linux peux durer une heure...

1.a) Les paquets « .deb »

Fort heureusement, Debian met à la disposition de ses utilisateurs des logiciels pré compilés. Ils sont livrés sous la forme de « paquets » ayant une extension « .deb ». Ces fichiers contiennent tous les éléments nécessaires pour installer le programme à savoir:

- Les exécutables, librairies, fichiers de configuration... spécifiques au logiciel à installer,
- La liste des autres logiciels/librairies requises pour que ce logiciel fonctionne,
- Des scripts à exécuter avant/après l'installation et la suppression du programme.

Chaque paquet contient un entête incluant (entre autres) les informations suivantes:

- la version du logiciel,
- la version de Debian auquel il est destiner (stable, testing...),
- l'architecture matérielle pour lequel il a été compilé (i386, Alpha, PowerPC...) ,
- un descriptif du programme (une version courte et une version détaillée).

Tous les paquets déjà installés dans votre système sont stockés dans le répertoire «/var/apt/cache/archives ».

```
ls /var/apt/cache/archives/nano*.deb
/var/cache/apt/archives/nano_2.0.2-1etch1_i386.deb
```

Vous pouvez afficher le contenant d'un paquet .deb avec l'outil « dpkg »:

```
dpkg --info
/var/cache/apt/archives/nano_2.0.2-1etch1_i386.deb

nouveau paquet Debian, version 2.0.
taille 547168 octets : archive de contrôle = 3066 octets.
12 octets, 1 lignes conffiles
718 octets, 18 lignes control
```

6http://www.canonical.com/

```
4506 octets, 67 lignes md5sums
579 octets, 21 lignes * postinst #!/bin/sh
160 octets, 5 lignes * postrm #!/bin/sh
379 octets, 20 lignes * preinst #!/bin/sh
288 octets, 14 lignes * prerm #!/bin/sh
Package: nano
Version: 2.0.2-1etch1
Section: editors
Priority: important
Architecture: i386
Depends: libc6 (>= 2.3.6-6), libncursesw5 (>= 5.4-5)
Suggests: spell
Conflicts: nano-tiny (>= 1.0.0-1), pico
Replaces: pico
Provides: editor
Installed-Size: 1624
Maintainer: Jordi Mallach <jordi@debian.org>
Description: free Pico clone with some new features
GNU nano is a free replacement for Pico, the default Pine
editor. pine is copyrighted under a restrictive licence, that
makes it unsuitable for Debian's main section. GNU nano is an
effort to provide a Pico-like editor, but also includes some
features that were missing in the original, such as 'search and
replace', 'goto line' or internationalization support.
```

1.b) La commande « dpkg »

« dpkg » est la commande de base pour travailler avec des paquets « .deb ». « dpkg » nécessite les droits d'administration pour être exécuté.

Comme nous venons de la voir à l'instant, le paramètre « --info » d'un lien vers une fichier « .deb » permets de visualiser le contenu de l'entête d'un paquet.

L'option « -i », ou « --install », installe le ou les paquets indiqués. Ceci oblige évidemment à avoir déjà télécharger l'archive « .deb » correspondant au programme à installer.

```
dpkg -i paquet.deb
```

« dpkg » ne gère pas automatiquement les dépendances. Ceci signifie que si un programme requiert d'autres programmes ou librairies pour fonctionner, il faudra télécharger le fichier « .deb » correspondant à chacun de ces éléments avoir de pouvoir exécuter le programme principal. Sachez toutefois qu'il existe un autre programme nommé « apt-get » que nous allons étudier un peu plus loin dans le cours de cet article, qui est capable de s'acquitter ce cette tâche. Pour pour compléter l'installation d'un programme qui a des dépendances non résolues, exécutez, en tant que « root » dans un terminal :

```
apt-get -f install
```

Le programme « dpkg-reconfigure » permets de relancer le script de configuration d'une application déjà installée.

```
dpkg-rconfigure nom_du_paquet
```

L'option « -r » ou « –remove » supprime le (ou les) paquet(s) indiqué(s) mais pas les fichiers de configuration associés au(x) paquet(s).

```
dpkg -r nom_du_paquet
```

L'option « -P » ou « –purge » permet de supprimer les fichiers de configuration associés en même temps que le paquet indiqué.

```
dpkg -P nom_du_paquet
```

L'usage des options « –force-all » associéeà « –purge » permet de forcer la désinstallation du paquet, ses dépendances et de supprimer les fichiers de configuration associés.

```
dpkg --force-all --purge nom_du_paquet
```

Quelques commandes de base sont résumées ci-dessous. Le détail des fonctions de « dpkg » est disponible en ligne sur « http://www.delafond.org/traducmanfr/deb/man8/dpkg.8.html ».

`dpkg --unpack paquet.deb`	Décompresse le paquet, sans rien configurer.
`dpkg -l`	Affiche la liste des paquets installés.
`dpkg -l chaîne`	Effectue une recherche et affiche une liste des paquets satisfaisant un motif de recherche.
`dpkg -S fichier`	Affiche tous les paquets contenant le fichier indiqué en tant que critère de recherche.
`dpkg -L paquet.deb`	Retourne la liste des fichiers contenus dans le ou les paquets .deb indiqués.
`dpkg --get-selections > liste_paquets`	Retourne la liste de tous les paquets installés sur la machine.
`dpkg --set-selections < liste_paquets` `deselect install`	Marque comme « A installer » les paquets donnés par la liste « liste_paquets ». La commande « dselect » effectuera l'installation effective.

Tableau des options supplémentaires de « dpkg »

1.c) Les outils « apt »

Le soucis avec l'usage de « dpkg » est qu'il faut disposer d'une copie locale du paquet d'installation du programme que vous souhaitez installer et une copie du paquet d'installation de toutes les dépendances non présentes dans votre système. Évidemment, cela peux vite devenir très compliqué et fastidieux. Heureusement, il existe un outils bien plus simple: « apt » pour « Advanced Packaging Tool ». Il s'agit d'un système complet qui s'appuie sur « dpkg » et qui permet:

- une recherche facile et efficace des programmes disponibles pour votre distribution,
- une installation simple et rapide,
- une désinstallation propre des logiciels que vous souhaitez retirer de votre système.

Il permet aussi de tenir à jour votre distribution (c'est à dire d'installer les paquets de version plus récentes que ceux installés) et de passer à une nouvelle version de Debian/Ubuntu, lorsque celle-ci est disponible.

Grâce à « apt », il est possible d'installer des logiciels, les diverses bibliothèques, extensions et autres compléments indispensables pour les faire fonctionner (les dépendances) en une seule ligne de commande. « apt » dispose aussi de nombreuses interfaces graphiques, dont Synaptic[7] et d'interfaces en ligne de commande comme Aptitude, afin d'en rendre l'utilisation plus conviviale.

7http://www.nongnu.org/synaptic/

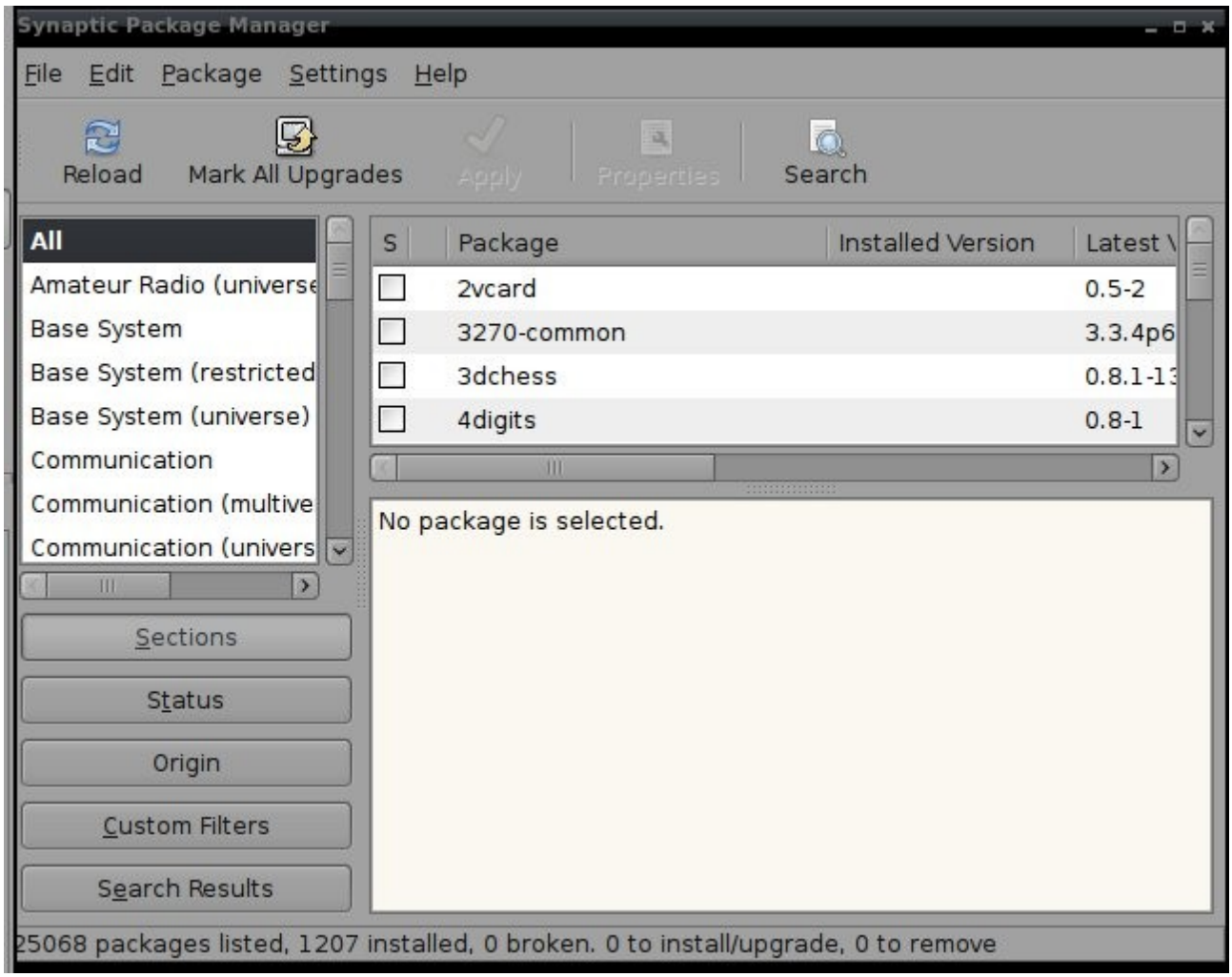


Illustration 2: Alternative graphique aux outils "apt": "synaptic"

1.c.i) Gestion des sources de paquets

La commande « apt » utilise une base de données des programmes disponible pour votre distribution. Pour actualiser cette base d'information, tapez la commande:

```
apt-get update
```

Cette base de logiciels peut être alimentée par plusieurs sources. Les liens vers les serveurs hébergeant des applications présentées sous la forme de paquet « .deb » sont définis dans le fichier « /etc/ap/source.list ».

Des paquets très spécifiques peuvent ainsi être téléchargés depuis Internet quand les distributeurs en fournissent.

Le contenu du fichier « /etc/ap/source.list ». de la distribution CrunchBang Linux est la suivante:

deb cdrom:[Ubuntu 8.04.1 _Hardy Heron_ - Release i386 (20080701)]/ hardy main restricted

```
#deb cdrom:[Ubuntu 8.04.1 _Hardy Heron_ - Release i386 (20080701)]/ hardy
main restricted
# See http://help.ubuntu.com/community/UpgradeNotes for how to upgrade to
# newer versions of the distribution.

deb http://gb.archive.ubuntu.com/ubuntu/ hardy main restricted
deb-src http://gb.archive.ubuntu.com/ubuntu/ hardy main restricted

## Major bug fix updates produced after the final release of the
## distribution.
```

```
deb http://gb.archive.ubuntu.com/ubuntu/ hardy-updates main restricted
deb-src http://gb.archive.ubuntu.com/ubuntu/ hardy-updates main restricted

## N.B. software from this repository is ENTIRELY UNSUPPORTED by the Ubuntu
## team, and may not be under a free licence. Please satisfy yourself as to
## your rights to use the software. Also, please note that software in
## universe WILL NOT receive any review or updates from the Ubuntu security
## team.
deb http://gb.archive.ubuntu.com/ubuntu/ hardy universe
deb-src http://gb.archive.ubuntu.com/ubuntu/ hardy universe
deb http://gb.archive.ubuntu.com/ubuntu/ hardy-updates universe
deb-src http://gb.archive.ubuntu.com/ubuntu/ hardy-updates universe

## N.B. software from this repository is ENTIRELY UNSUPPORTED by the Ubuntu
## team, and may not be under a free licence. Please satisfy yourself as to
## your rights to use the software. Also, please note that software in
## multiverse WILL NOT receive any review or updates from the Ubuntu
## security team.
deb http://gb.archive.ubuntu.com/ubuntu/ hardy multiverse
deb-src http://gb.archive.ubuntu.com/ubuntu/ hardy multiverse
deb http://gb.archive.ubuntu.com/ubuntu/ hardy-updates multiverse
deb-src http://gb.archive.ubuntu.com/ubuntu/ hardy-updates multiverse

## Uncomment the following two lines to add software from the 'backports'
## repository.
## N.B. software from this repository may not have been tested as
## extensively as that contained in the main release, although it includes
## newer versions of some applications which may provide useful features.
## Also, please note that software in backports WILL NOT receive any review
## or updates from the Ubuntu security team.
# deb http://gb.archive.ubuntu.com/ubuntu/ hardy-backports main restricted
universe multiverse
# deb-src http://gb.archive.ubuntu.com/ubuntu/ hardy-backports main
restricted universe multiverse

## Uncomment the following two lines to add software from Canonical's
## 'partner' repository. This software is not part of Ubuntu, but is
## offered by Canonical and the respective vendors as a service to Ubuntu
## users.
# deb http://archive.canonical.com/ubuntu hardy partner
# deb-src http://archive.canonical.com/ubuntu hardy partner

deb http://security.ubuntu.com/ubuntu hardy-security main restricted
deb-src http://security.ubuntu.com/ubuntu hardy-security main restricted
deb http://security.ubuntu.com/ubuntu hardy-security universe
deb-src http://security.ubuntu.com/ubuntu hardy-security universe
deb http://security.ubuntu.com/ubuntu hardy-security multiverse
deb-src http://security.ubuntu.com/ubuntu hardy-security multiverse
```

Contenu de « /etc/apt/source.list » de CrunchBang

On y recense à la fois des serveur Ubuntu officiels (« archive.canonical.com ») et des serveurs spécifiques à CrunchBang (« gb.archive.ubuntu.com »). Ces serveurs sont aussi appelés « dépôts »,

Chaque ligne est de la forme « deb serveur distribution univers » où:

- Les lignes débutant par « deb » pointent vers des sources de paquets compilés, les lignes débutant par « deb-src » pointent vers les code sources des applications,
- « serveur » est l'URL du serveur internet http, ftp ou le cdrom,
- « distribution » fait référence à la version de la distribution linux – dans le listing ci-dessus, on voit clairement de CrunchBang Linux est basé sur Ubuntu 8.04.1 _Hardy Heron,
- « univers » fait référence à un classement des logiciels: « main » = nécessaire, « non-free » logiciels non gratuits...

Les lignes commençant par # sont ignorées.

1.c.ii) Mise à jour de tous les logiciels installés: apt-update / apt-upgrade

Bien évidemment, il faut tout d'abord s'assurer que la base de données des paquets « .deb » est bien à jour avec la commande:

```
apt-get update
```

Ensuite, lancez la recherche et l'installation de toutes les mises à jour disponibles avec la commande:

```
apt-get update
```

La mise des mise à jour disponible vous est d'abord présentée. Tapez « O » si vous souhaitez procéder à la mise à jour qui peux être assez longue car plusieurs Gigaoctets e données peuvent alors téléchargés depuis internet. L'option « -y » lance l'installation sans poser de questions.

```
apt-get -y update
```

Tous les fichiers « .deb » téléchargés sont gardés dans un cache disque situé dans «/var/apt/cache/archives ». Ainsi, si vous devez réinstaller un logiciel ultérieurement et qu'il n'existe pas de version plus récente de celui-ci, alors la copie locale sera utilisé. Cela évite de la télécharger à nouveau. Évidemment, ce cache occupe beaucoup de place sur votre disque dur. Vous pouvez le purger avec la commande:

```
apt-get clean
```

Notez que cela n'a aucun impact sur les paquets installés.

L'option « autoclean » permet de ne supprimer que les copies des paquets désinstallés.

```
apt-get autoclean
```

L'option « dist-upgrade » met à jour tous les paquets installés vers les dernières versions en installant de nouveaux paquets si nécessaire, par opposition à « upgrade » simple qui n'ajoute pas de nouveaux paquets. Utilisez cette commande si vous voulez passer à la nouvelle version de la distribution Debian.

```
apt-get dist-upgrade
```

1.c.iii) Rechercher un logiciel à installer: apt-cache

L'option « search » passée à l'utilitaire « apt-cache » permet de chercher les paquets contenant certain mots-clefs. Par défaut, la recherche se fait sur les noms de programmes et sans le descriptif associé. Les réponses dépendent bien évidemment des dépôts configurés dans « /etc/apt/source.list ». L'usage de « apt-cache » ne nécessite pas les droits d'administration.

```
apt-cache search <word1 word2 ...>
```

Vous pouvez limiter la recherche aux noms de programmes seulement avec l'option « --name-only » ou « -n ».

```
apt-cache search -n <chaîne>
```

Quelques unes des autres options utiles de « apt-cache » sont résumées dans le tableau ci-dessous.

`apt-cache showpkg <paquet(s)>`	Affiche des informations sur les paquets indiqués.
`apt-cache dumpavail`	Affiche une liste des paquets disponibles.
`apt-cache show <paquet(s)>`	Affiche les informations sur un paquet

Options supplémentaires de « apt-cache »

1.c.iv) Installer un logiciel: apt-get install

L'option « install » passée à l'utilitaire « apt-get » permet d'installer les paquets indiqués.

- apt-get install <paquet(s)>
  Quand une installation échoue pour diverses raisons, vous pouvez forcer celle-ci ainsi en installant les dépendances !

```
apt-get -f install
```

1.c.v) Supprimer un logiciel: apt-get remove

L'option « remove » permet de désinstaller les paquets indiqués. Ceci laisse toutefois en place les fichiers de configuration de ces paquets. Elle laisse aussi en place les dépendances même si elles ne sont plus utilisées par d'autres programme

```
apt-get remove <paquets(s)>
```

L'option « autoremove » permet de désinstaller les paquets indiqués avec leurs dépendances logicielles. Ceci laisse toutefois en place les fichiers de configuration de ces paquets.

```
apt-get autoremove <paquets(s)>
```

L'option « –purge », passée à la commande « remove » ou « autoremove », supprime les paquets indiqués et leurs fichiers de configuration.

```
apt-get remove --purge <paquets(s)>
apt-get autoremove --purge <paquets(s)>
```

1.d) Choisir les sources les plus rapides (Debian seulement)

Comme nous l'avons vu plus tôt, les commandes « apt » se basent sur une base de données alimentée par des sources internet. Lorsque vous installez une application avec « apt-get » elle est téléchargée depuis l'une de ces sources internet. Bien évidemment, plus le débit de ce lien est rapide, plus le temps nécessaire au téléchargement du logiciel sera court. En réalité, il existe une grande quantité de source internet pour les fichiers de base de la distribution Debian/Ubuntu. Ce sont des sites que l'on appelle « mirroirs ». Leur contenu est identique aux sources officielles Debian simplement, ils ont l'avantage d'être plus proche de chez vous, moins sollicités et donc plus rapides.

Il existe une commande nommée « netselect-apt » qui est référence tous les « miroirs », en compare les vitesses de téléchargement respectives et ajuste le fichier «/etc/apt/sources.list » afin que le fonctionnement de « apt » soit le plus rapide possible.

Toute d'abord, il vous faut télécharger « netselect-apt ».

```
apt-get update
apt-get upgrade
apt-get install -y netselect-apt
Install netselect-apt in debian using the following command
```

La syntaxe d'utilisation de « netselect-apt » est de la forme:

```
 netselect-apt option distributions
```

où

« options »:

- « -a » architecture (i386 par défaut)
- « -s » rechercher des mirroirs contenant les codes sources
- « -n » inclure les mirroirs contenant des logiciels « non-free » c'est à dire non -gratuit (commerciaux, shareware...
- « -u » force la recherche de sources de téléchargement non-US.

« distribution »:

- stable|testing|unstable|experimental|woody|sarge|etch|sid qui sont les noms de versions

actuelles etfutures de Debian. Par default la version stable nommée « etch » est utilisée.

La commande par défaut est va trouver et trier près de 400 miroirs et générer un fichier « sources.list » optimisé dans le répertoire courant. Sauvegardez le fichier original et copie le nouveau fichier dans « /etc/apt/ ».

```
netselect-apt
cp /etc/apt/sources.list /etc/apt/sources.list.bak
cp sources.list /etc/apt/sources.list
```

1.e) Suppression de paquets devenus inutiles: DebOrphan

DebOrphan est un programme qui permet de connaître les paquets devenu inutiles; les bibliothèques auxquelles aucun programme ne fait appel, ceux qui ne sont plus utilisés... On qualifie ces paquets de « orphelins ».

L'installation de « deborphan » vous est désormais acquise:

```
apt-get install -y deborphan
```

Les amateurs d'interface graphique pourront installer le paquet « « gtkorphan qui permet de connaître et supprimer sélectivement les paquets inutiles.

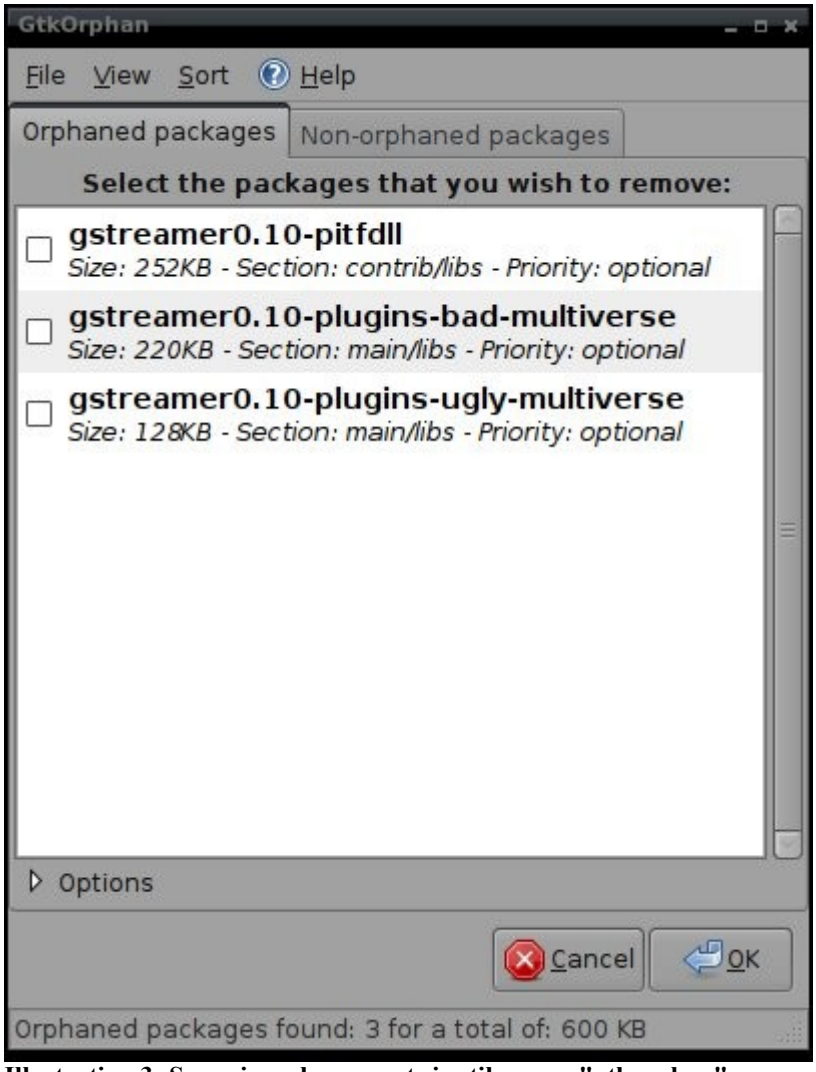


Illustration 3: Supprimer les paquets inutiles avec "gtkorphan"

La commande listera les paquets inutiles.

```
deborphan
```

En combinant la sortie de la commande « deborphan » avec la commande « apt-get remove

--purge », vous pouvez demande en une ligne la suppression des tous les paquets orphelins.

```
apt-get remove --purge $(deborphan)
```

1.f) Optimisez les paquets pour votre matériel: apt-build

Un peu plsu tôt dans cet article, je vous ai dit que les outils « dpkg » étaient une alternative à la compilation des programmes à partir des codes source. Mais, il y aura toujours des « plus malins » pour vous rétorquer que votre distribution est pas optimisée.

Grace à l'outils « apt-build », vous etes libres de compiler les logiciels que vous voulez utiliser sur votre Debian/Ubuntu, tout en utilisant le système « apt ».

Pour installer « apt-build », on opère comme d'habitude :

```
apt-get install apt-build
```

En fin d'installation, vous allez devoir répondre à quelques question concernant l'architecture de votre processeur. Toutes ces informations sont inscrites dans le fichier « /etc/apt/apt-build.conf », si vous avez fait une erreur en répondant vous pourrez donc revenir en arrière. Voici un exemple :

```
cat /etc/apt/apt-build.conf
build-dir = /var/cache/apt-build/build
repository-dir = /var/cache/apt-build/repository
Olevel = -O3
march = -march=pentium2
mcpu = -mcpu=pentium2
options =
```

Fichier de configuration de « apt-build »

Pour rappel, la commande la plus simple pour identifier architecture de son CPU est:

```
cat /proc/cpuinfo
```

La liste des processeurs acceptés par « apt- build » est très longue (plus d'une vingtaine de CPU!) (« ttp://gcc.gnu.org/onlinedocs/gcc/i386-and-x86_002d64-Options.html »).

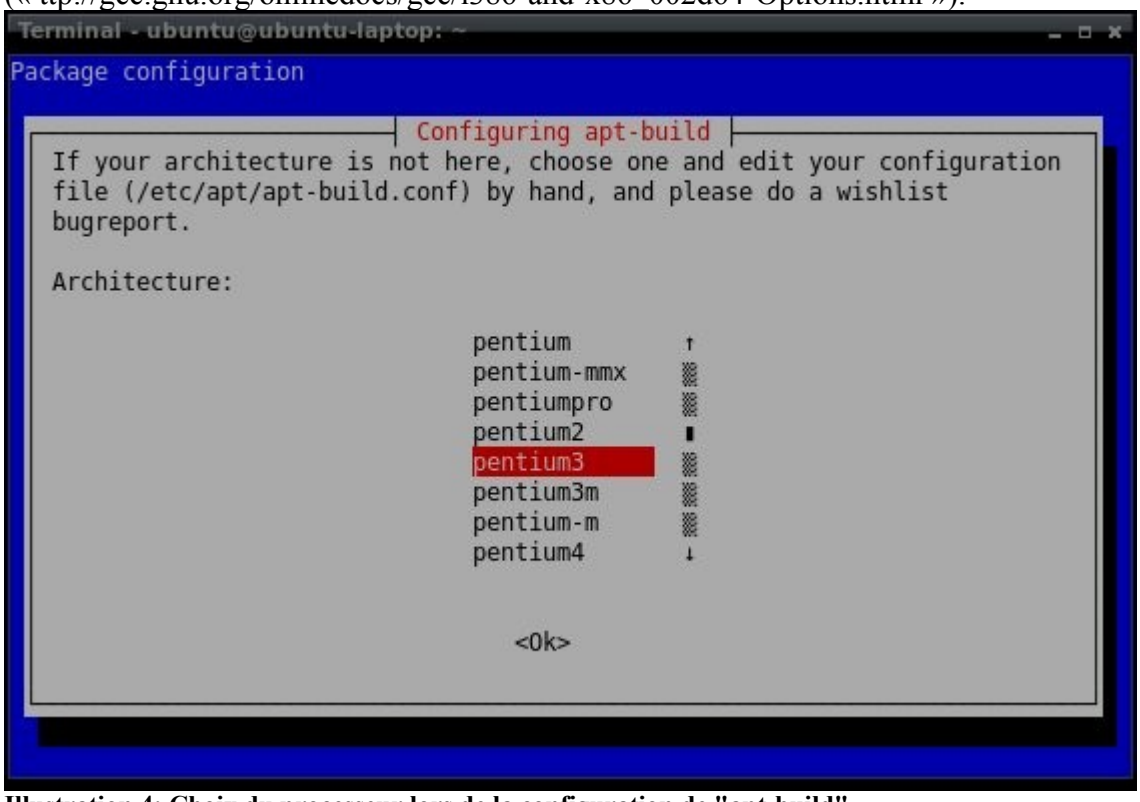


Illustration 4: Choix du processeur lors de la configuration de "apt-build"

Pour les paramètres « build-dir » et « repository-dir », assurez vous de donner un répertoire sur une partition où vous avez de la place. Enfin, « apt-build » requière que les serveurs contenant les

sources de programmes soient présents dans le fichier « /etc/apt/sources.list » - ce sont ceux dont la ligne débute par « deb-src ». « apt-build » les ajoutent automatiquement à vos dépôts lors de son installation

L'option « source » passée à « apt-get » permet de télécharger les paquets de codes sources indiqués.

```
apt-get source <paquet(s)>
```

Vous êtes dorénavant prêt à utiliser « apt-build » ! Tout comme « apt-get », « apt-build » utilise les options « update », « upgrade » et « install ». Pour télécharger et compiler le programme « memstat », i l vous tapez la série de commande suivante:

```
apt-build update
apt-build install memstat
-----> Installing build dependencies (for memstat) <-----
Reading Package Lists... Done
Building Dependency Tree... Done
0 packages upgraded, 0 newly installed, 0 to remove and 0  not
upgraded.
-----> Downloading memstat source (memstat) <-----
Reading Package Lists... Done
Building Dependency Tree... Done
Need to get 22.4kB of source archives.
Get:1 ftp://127.0.0.1 1.0.4/main memstat 0.4-1 (dsc) [482B]
Get:2 ftp://127.0.0.1 1.0.4/main memstat 0.4-1 (tar) [21.9kB]
Fetched 22.4kB in 0s (322kB/s)
dpkg-source: extracting memstat in memstat-0.4
-----> Building memstat <-----
...
...
...
dpkg-genchanges: binary-only upload - not including any source
code
dpkg-buildpackage: binary only upload (no source included)
-----> Moving packages to repository <-----
-----> Updating repository <-----
Using: -O3 -mcpu=pentium2 -march=pentium2
...
...
Reading Package Lists... Done
Building Dependency Tree... Done
Reading Package Lists... Done
Building Dependency Tree... Done
The following NEW packages will be installed:
  memstat
```

On remarque que « apt-build » procède en plusieurs étapes :

•l'installation des paquets nécessaires pour la compilation,
•le téléchargement des sources proprement dites,
•la compilation
•l'installation du paquet produit.

Il existe une option intitulé « world » pour « apt-build ». Avec elle, il est possible en une seule ligne de commande de pour recompiler et optimiser tout notre système ou presque.

Il faut au préalable créer la liste des paquets à recompiler et à installer à la place des paquets actuels. De cette liste, il est préférable de retirer des paquets trop sensibles comme « gcc »,

« libc6 ». Une première ébauche de cette liste peux être réalisée avec la commande « dpkg ».

```
dpkg --get-selections | awk '{if ($2 == "install") print $1}'
> /etc/apt/apt-build.list
```

Ensuite, vous pouvez lancer la commande:

```
apt-build world
-----> Rebuilding the world ! <-----
You should read README.Debian first
```

Attention, car si vous utilisez « apt-build » pour construire vos propres paquets, une mise à jour avec « apt-get dist-upgrade » va remplacer vos paquets par des paquets issus des serveurs officiels. Vous pouvez éviter cela en donnant une priorité plus importante aux paquets construits avec « apt-build ». Pour cela, il vous faut modifier le fichier « /etc/apt/preferences » comme suit :

```
Package: *
Pin: release o=apt-build
Pin-Priority: 990
```

La commande « apt-cache policy » qui vous donnera la possibilité de vérifier les priorités entre dépôts.

```
apt-cache policy
Package Files:
 100 /var/lib/dpkg/status
     release a=now
 500 http://ftp2.de.debian.org sid/non-free Packages
     release o=Debian,a=unstable,l=Debian,c=non-free
     origin ftp2.de.debian.org
 500 http://ftp2.de.debian.org sid/contrib Packages
     release o=Debian,a=unstable,l=Debian,c=contrib
     origin ftp2.de.debian.org
 500 http://ftp2.de.debian.org sid/main Packages
     release o=Debian,a=unstable,l=Debian,c=main
     origin ftp2.de.debian.org
 990 file: apt-build/main Packages
     release o=apt-build,a=apt-build,l=apt-build,c=main
Pinned Packages:
```

Pour plus d'information sur l'usage de la commande « apt-build », je vous invite à lire le ien « http://www.andesi.org/paquets:apt-build-optimisez-les-paquets-debian-pour-votre-systeme ».

1.g) Chercher le paquet dont provient un fichier: apt-file

Une distribution Debian/Ubuntu, c'est plus ou moins:

•un noyaux assurant le démarrage de la distribution,
•une série de logiciels installés au travers d'outils de gestion de paquets pré-compilés,
•des fichiers de configuration.
Vous pouvez être amené à vous demander quel paquets à installé tel programme, fichier, icône, image...

L'utilitaire « apt-file » permets de répondre très facilement à cette question.

L'installation de « apt-files » est désormais simple:

```
apt-get install apt-files
```

Ensuite vous devez mettre à jour la base de données de « apt-file » - celle-ci est basée sur le contenu du fichier « /etc/apt/sources.list ». Les résultats que retournera « apt-files » ne sont donc pas limités aux logiciels déjà installés dans votre ordinateur. Ainsi, s'il vous manque un fichier, « apt-files »

saura vous dire quel paquet installer pour obtenir ce fichier.

```
apt-file update
```

La commande pour utiliser « apt-file » ressemble à cette de « apt-cache ».

apt-file search <nomdefichier>

Si « apt-file » affiche trop d'entrées, essaye z la commande ci-dessous qui ne vous donnera que les fichiers contenant le nom du fichier comme un seul mot:

```
apt-file search nomdefichier | grep -w nomdefichier
```

La commande suivante ne listera plus que les fichiers situés dans les répertoires comme « /bin » ou « /usr/bin ».

```
apt-file search filename | grep /bin/
```

L'utilitaire « grep » est un programme en ligne de commande initialement écrit pour Unix utilisant l'algorithme d'Aho-Corasick. Le comportement habituel de « grep » est de recevoir une expression rationnelle en ligne de commande, de lire les données sur l'entrée standard ou dans une liste de fichiers, et d'écrire les lignes qui contiennent des correspondances avec l'expression rationnelle sur la sortie standard. « grep » est un filtre, ce qui lui permet d'être combiné avec d'autres commandes, sous la forme d'un pipeline. (« http://fr.wikipedia.org/wiki/Grep»). Typique si vous avez un fichier texte « test.txt » contenant les mots « chien », « chat » et « oiseau », la commande ci-dessous affichera « chat » uniquement:

```
cat test.txt|grep chat
```

1.h) Etudier l'arborescence des dépendances: apt-rdepend

L'outils « apt-rdepends » liste l'arborescence des paquets dont dépends un programme. Chaque dépendance est ensuite analysée et ainsi récursivement jusqu'à ce que l'arborescence est été complètement parcourue. Le résultat est affiché soit sous la forme d'un listing soit sous la forme d'un fichier compatible avec le générateur de graph « dotty ».
L'installation de « apt-rdepends » est réalisée avec la commande:

```
apt-get install -y apt-rdepends
```

La commande suivante listera toutes les dépendances du logiciel « bash »:

```
apt-rdepends bash
bash
  Depends: base-files (>= 2.1.12)
  Depends: debianutils (>= 2.15)
  PreDepends: libc6 (>= 2.4)
  PreDepends: libncurses5 (>= 5.6+20071006-3)
base-files
  Depends: awk
  Depends: base-passwd (>= 2.0.3.4)
  Depends: libpam-modules (>= 0.79-3ubuntu3)
awk
base-passwd
  Depends: libc6 (>= 2.6.1-1)
libc6
  Depends: libgcc1
libgcc1
  Depends: gcc-4.2-base (= 4.2.3-2ubuntu7)
  Depends: libc6 (>= 2.7-1)
gcc-4.2-base
libpam-modules
  Depends: libc6 (>= 2.4)
  Depends: libdb4.6
```

```
  Depends: libpam0g (>= 0.99.7.1)
  Depends: libselinux1
libdb4.6
  Depends: libc6 (>= 2.7-1)
libpam0g
  Depends: debconf (>= 0.5)
  Depends: debconf-2.0
  Depends: libc6 (>= 2.4)
  Depends: libpam-runtime
debconf
  Depends: debconf-english
  Depends: debconf-i18n
  PreDepends: perl-base (>= 5.6.1-4)
debconf-english
  Depends: debconf
debconf-i18n
  Depends: debconf
  Depends: liblocale-gettext-perl
  Depends: libtext-charwidth-perl
  Depends: libtext-iconv-perl
  Depends: libtext-wrapi18n-perl
liblocale-gettext-perl
  Depends: libc6 (>= 2.2)
  Depends: perl-base (>= 5.8.8-6)
  Depends: perlapi-5.8.8
perl-base
  PreDepends: libc6 (>= 2.6.1-1)
perlapi-5.8.8
libtext-charwidth-perl
  Depends: libc6 (>= 2.5-0ubuntu1)
  Depends: perl-base (>= 5.8.8-7)
  Depends: perlapi-5.8.8
libtext-iconv-perl
  Depends: libc6 (>= 2.4-1)
  Depends: perl-base (>= 5.8.7-10ubuntu2)
  Depends: perlapi-5.8.7
perlapi-5.8.7
libtext-wrapi18n-perl
  Depends: libtext-charwidth-perl
debconf-2.0
libpam-runtime
libselinux1
  Depends: libc6 (>= 2.7-1)
debianutils
  PreDepends: coreutils (>= 4.5.8-1)
  PreDepends: libc6 (>= 2.7-1)
  PreDepends: mktemp
coreutils
  PreDepends: libacl1 (>= 2.2.11-1)
  PreDepends: libc6 (>= 2.7-1)
  PreDepends: libselinux1
libacl1
  Depends: libattr1 (>= 2.4.4-1)
```

```
  Depends: libc6 (>= 2.6.1-1)
libattr1
  Depends: libc6 (>= 2.6.1-1)
mktemp
  PreDepends: libc6 (>= 2.7-1)
libncurses5
  Depends: libc6 (>= 2.7-1)
```

On apprends ainsi que « bash » dépends de « base-files » qui lui même dépends de « awk »....

Afin d'obtenir une représentation graphique, il faut installer la pack logiciel « graphviz »[8] dont « dot » fait partie.

```
apt-file update
apt-file search dot| grep /usr/bin/
apt-get install -y graphviz
```

Vous exportez le résultat de « apt-rdepends » sous la forme d'un fichier compatible avec la commande « dot » en ajoutant l'option « -d » à la commande:

```
apt-rdepends -d bash > bash.dot
```

Le fichier « bash.dot » est ensuite convertie en image au format vectoriel « svg » avec la commande:

```
cat bash.dot|dot -Tsvg > bash.svg
```

L'image résultante peux être visualiser avec la visionneuse « mirage ».

```
mirage bash.png
```

8 http://www.graphviz.org/

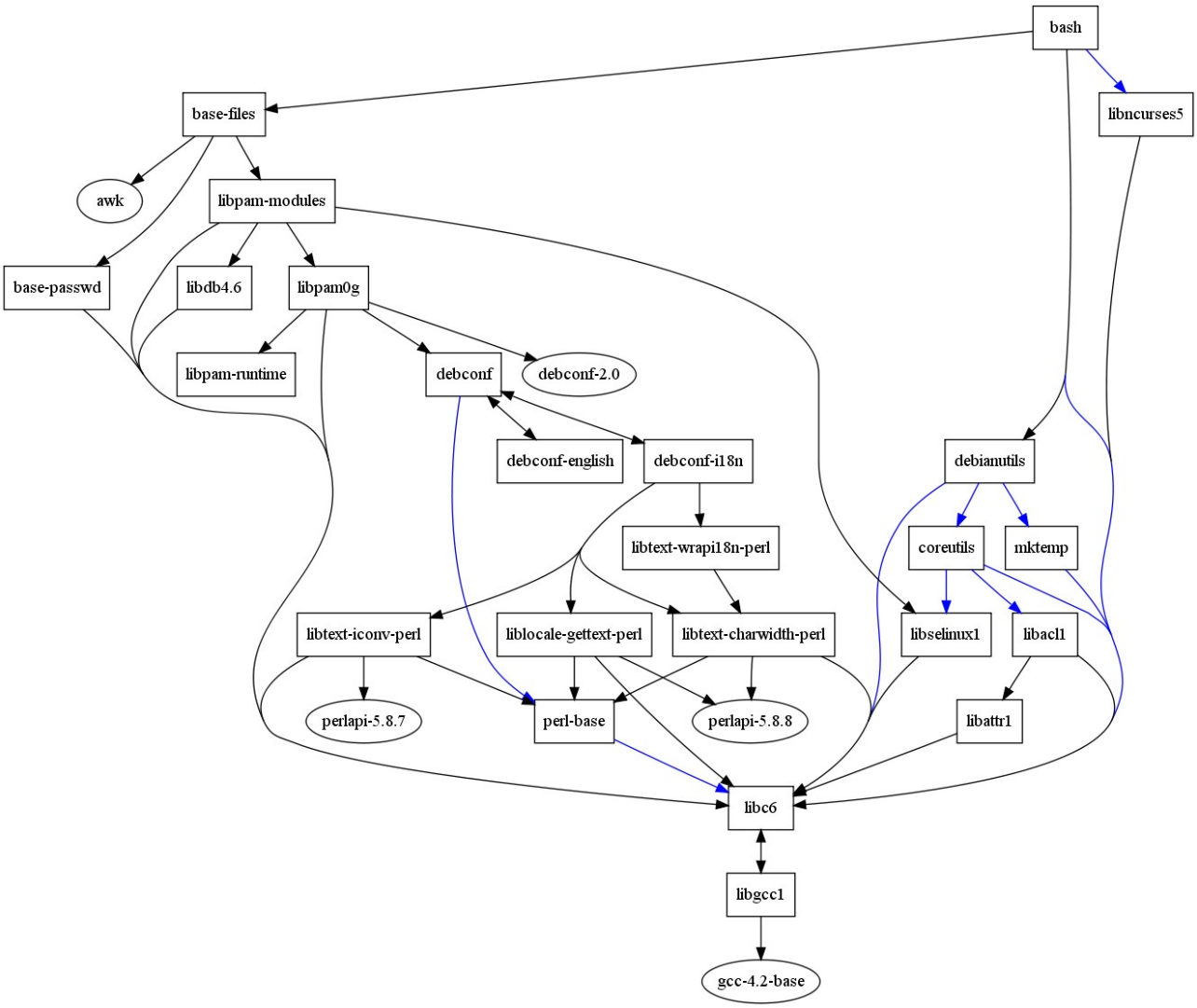


Illustration 5: Dépendances de "bash" calculée par "apt-rdepends"

Il est également possible de ne suivre que les paquets effectivement installé dans la distribution avec les options « --state-follow=Installed --state-show=Installed » ce qui réduit considérablement la complexité de l'arborescence.
Enfin, « apt-rdepends » peux également générer l'arborescence inverse d'un paquet c'est à dire lister l'ensemble des programmes ayant pour dépendance directe « bash » ou dont les dépendances dépendance à leur tour de « bash ». Ceci est réalisé avec l'option « -r » et la représentation graphique est reproduite ci-dessous.

```
apt-rdepends -d --state-follow=Installed -state-show=Installed
bash > rbash.dot
cat rbash.dot|dot -Tsvg > rbash.svg
mirage rbash.svg
```

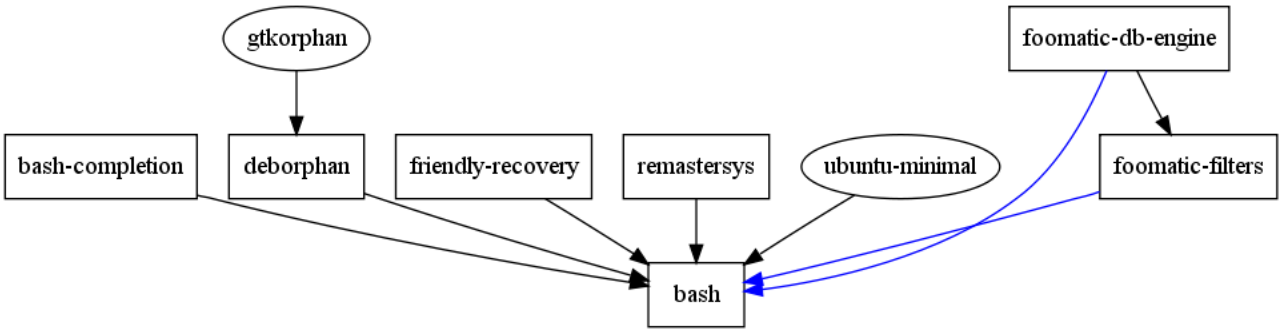


Illustration 6: Arborescence inverse de "bash" générée par "apt-rdepends"

1.i) Mettre en œuvre un proxy de paquets: apt-cacher

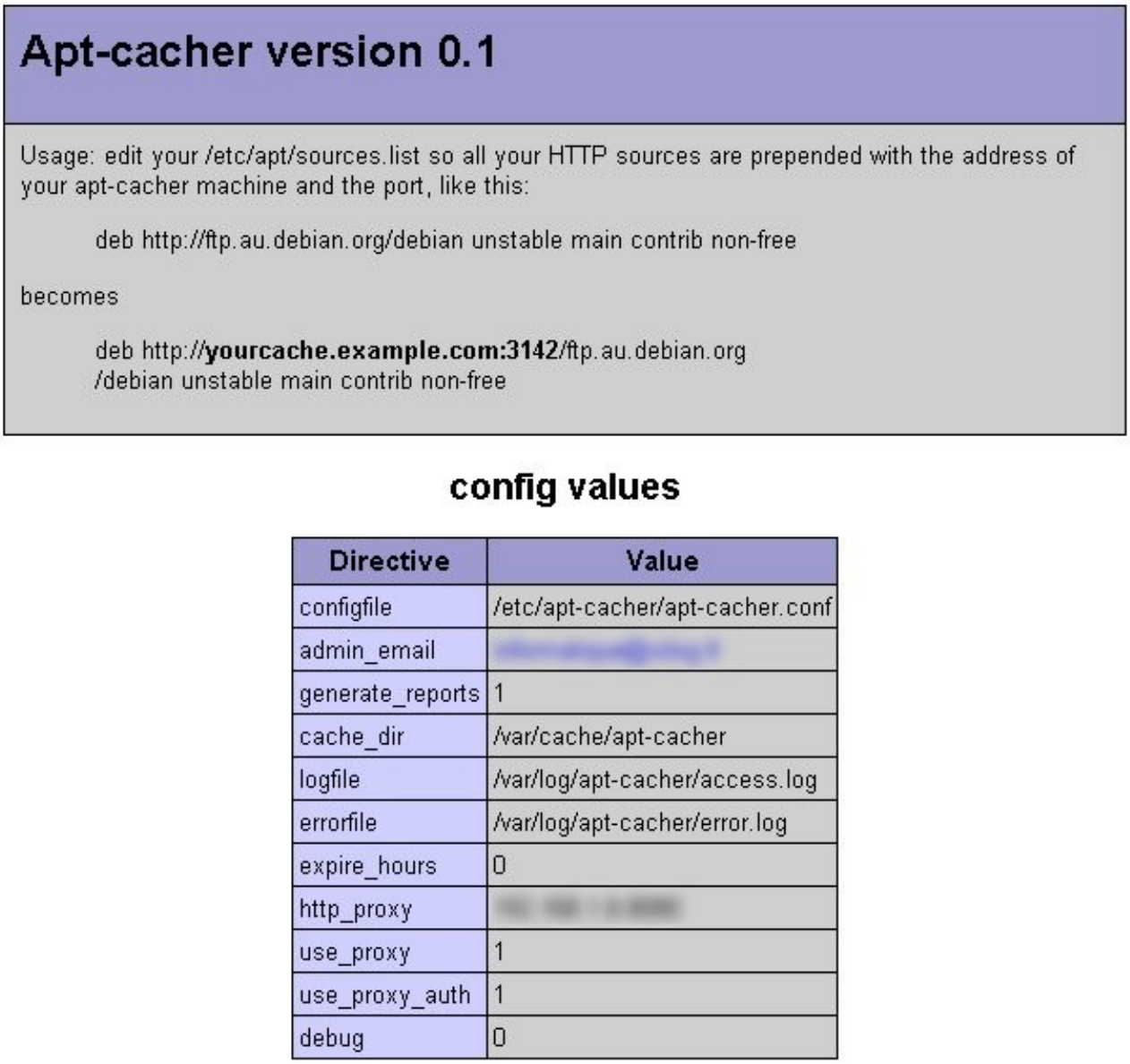
Vous l'avez désormais bien compris, « apt » est l'outil idéal pour trouver, télécharger et installer vos programmes sous Debian/Ubuntu. D'ailleurs, une des méthode d'installation de Debian repose sur un « netinstall » CD c'est à dire un CDROM de 160Mb à peine qui se charge de démarrer l'installer, d'initialiser le réseau et installe ensuite votre distribution à partir des derniers paquets logiciels présents dans les dépôts. Du coup, chaque nouvelle installation peux nécessiter le téléchargement de plusieurs centaines de mégaoctets de paquets. Si vous êtes dans un réseau, il peux être intéressant de mettre en place un proxy[9] dédié au paquets logiciels. Ainsi, lors de la première installation de votre distribution, le CD « netinstall » va demander au proxy s'il dispose déjà du paquet en cache. Évidemment, comme le cache est pour l'instant vide, le proxy va télécharger ce paquet depuis internet puis le faire suivre à votre PC. Au final, il faudra prévoir une petite heure pour installer la poste bureautique complet. Par contre, si vous installez ensuite un nouveau PC, celui-ci va également s'adresser au proxy. Le proxy va juste vérifier que la version du paquet qui est déjà présent dans son cache est bien la version la plus récente et si c'est bien le cas, il va simplement le faire suivre votre nouveau PC. Si la version en cache n'est plus d'actualité, il va d'abord télécharger la nouvelle version. Si tous les paquets requis sont déjà en cache, alors l'installation de votre station de travail ne prendra que 10 minutes. De plus, si vous faites un « apt-get upgrade » sur le premier PC, celui-ci sera mis à jour quasi instantanément puisque tous les derniers paquets logiciels sont déjà dans le proxy.

Un serveur de proxy de paquets Debian connu est « apt-cacher. Pour l'installer, il suffit de taper la commande:

```
apt-get install -y apt-cacher
```

Ce logiciel installe un service WEB répondant sur le port 3142. Aussi, lancez ensuite un navigateur internet et rendez-vous à l'adresse « http://DebianServer:3142/apt-cacher/ » où « DebianServer » est le nom de votre PC.

9 Un serveur mandataire ou proxy (de l'anglais) est un serveur informatique qui a pour fonction de relayer des requêtes entre un poste client et un serveur (http://fr.wikipedia.org/wiki/Proxy)



Apt-cacher version 0.1

Usage: edit your /etc/apt/sources.list so all your HTTP sources are prepended with the address of your apt-cacher machine and the port, like this:

deb http://ftp.au.debian.org/debian unstable main contrib non-free

becomes

deb http://**yourcache.example.com:3142**/ftp.au.debian.org /debian unstable main contrib non-free

config values

Directive	Value
configfile	/etc/apt-cacher/apt-cacher.conf
admin_email	[illegible]
generate_reports	1
cache_dir	/var/cache/apt-cacher
logfile	/var/log/apt-cacher/access.log
errorfile	/var/log/apt-cacher/error.log
expire_hours	0
http_proxy	[illegible]
use_proxy	1
use_proxy_auth	1
debug	0

Illustration 7: Etat des paramétres de "apt-cacher"

Le fichier de configuration de « apt-cacher » est « /etc/default/apt-cacher ». Il y a très peu de paramètres à renseigner. Tout d'abord, il faut mettre AUTOSTART à 1 de sorte que «apt-cacher » se lance automatiquement avec le PC.

Le paramètre « cache_dir » indique l'endroit où seront enregistrés les paquets récupérés depuis Internet. Prévoyez quand même plusieurs Gigaoctets!

Le contenu de ce cache est contrôlé toute les 24 heures si l'option « clean_cache » est à 1. Cette opération est assez lourde pour le PC mais supprime du cache tous les fichiers qui ne sont plus maintenus dans les dépôts (version obsolète, logiciels plus maintenus, changement de licence...).

« apt-cacher » propose deux méthodes pour déterminer si un fichier nécessite d'être actualisé:

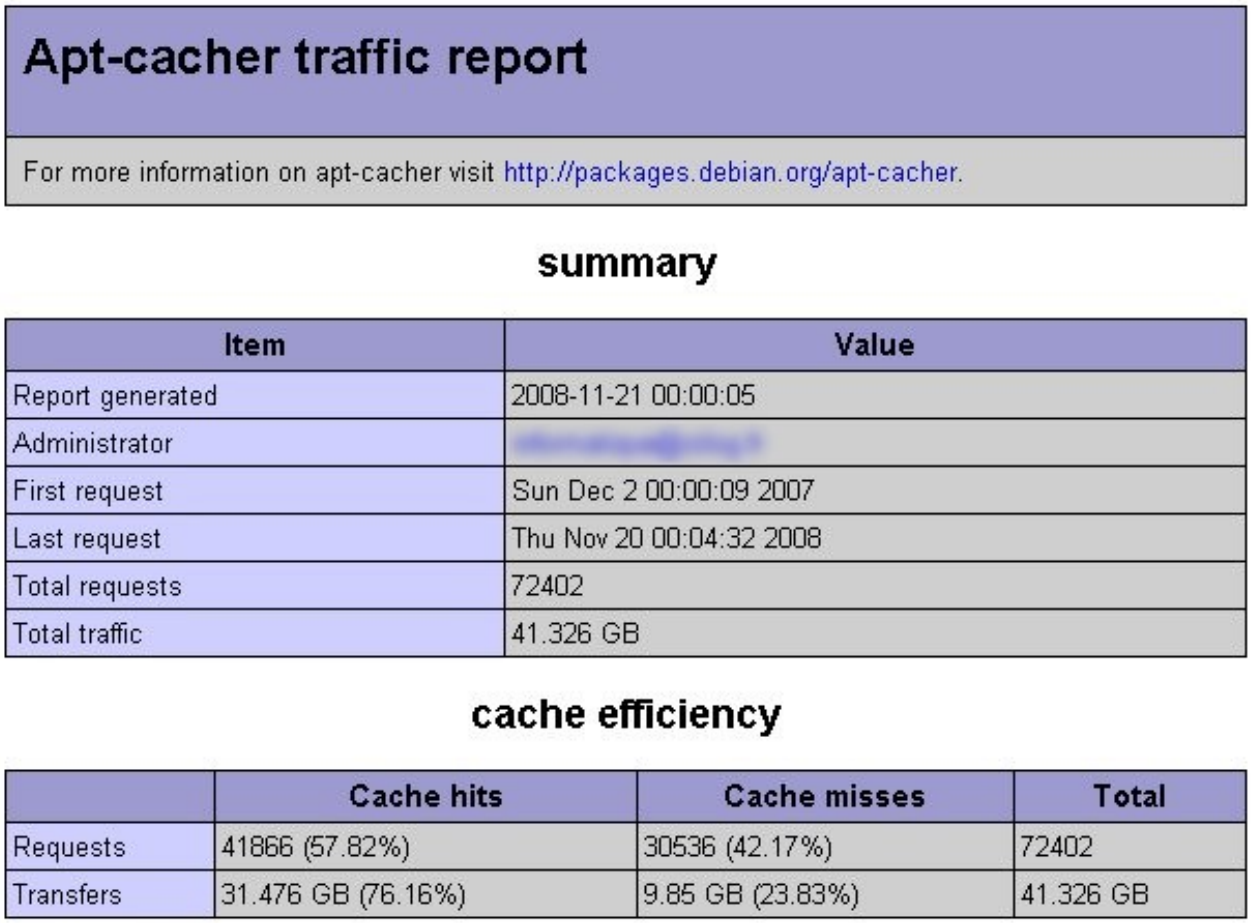
- Méthode A: il regarde l'age du fichier dans le cache,
- Méthode B: il faut appelle aux dépôts pour vérifier la version actuelle.

Évidemment, la seconde méthode B est plus précise mais elle nécessite une connexion à internet pour que sollicitation du proxy. Le paramètre « expire_hours » défini le nombre d'heures de présence en cache au bout de laquelle un paquet est considéré comme périmé (mettre la valeur « 0 » pour la méthode B).

Les paramètres « http_proxy » et « http_proxy_auth » ne vous concerne que si votre réseau dispose

d'un proxy HTTP comme SQUID[10].

Enfin, « apt-cacher » génère un rapport sur l'état d'utilisation de son cache toute les 24 heurs si le paramètre « generate_reports » est mis à 1. Ce rapport est disponible depuis l'URL « http://DebianServer/apt-cacher/report ».



Apt-cacher traffic report

For more information on apt-cacher visit http://packages.debian.org/apt-cacher.

summary

Item	Value
Report generated	2008-11-21 00:00:05
Administrator	[redacted]
First request	Sun Dec 2 00:00:09 2007
Last request	Thu Nov 20 00:04:32 2008
Total requests	72402
Total traffic	41.326 GB

cache efficiency

	Cache hits	Cache misses	Total
Requests	41866 (57.82%)	30536 (42.17%)	72402
Transfers	31.476 GB (76.16%)	9.85 GB (23.83%)	41.326 GB

Illustration 8: Statistiques d'utilisation de "apt-cacher"

```
################################################################
# This is the config file for apt-cacher. On most Debian systems
# you can safely leave the defaults alone.
################################################################

# cache_dir is used to set the location of the local cache. This can
# become quite large, so make sure it is somewhere with plenty of space.
cache_dir=/var/cache/apt-cacher

# The email address of the administrator is displayed in the info page
# and traffic reports.
admin_email=xxxx@yyyy.fr

# For the daemon startup settings please edit the file /etc/default/apt-
cacher.

# Daemon port setting, only useful in stand-alone mode. You need to run the
# daemon as root to use privileged ports (<1024).
daemon_port=3142

# optional settings, user and group to run the daemon as. Make sure they have
# sufficient permissions on the cache and log directories. Comment the
settings
```

10 http://www.squid-cache.org/

```
# to run apt-cacher as the native user.
group=www-data
user=www-data

# optional setting, binds the listening daemon to one specified IP. Use IP
# ranges for more advanced configuration, see below.
# daemon_addr=localhost

# If your apt-cacher machine is directly exposed to the Internet and you are
# worried about unauthorised machines fetching packages through it, you can
# specify a list of IPv4 addresses which are allowed to use it and another
# list of IPv4 addresses which aren't.
# Localhost (127.0.0.1) is always allowed. Other addresses must be matched
# by allowed_hosts and not by denied_hosts to be permitted to use the cache.
# Setting allowed_hosts to "*" means "allow all".
# Otherwise the format is a comma-separated list containing addresses,
# optionally with masks (like 10.0.0.0/22), or ranges of addresses (two
# addresses separated by a hyphen, no masks, like
'192.168.0.3-192.168.0.56').
allowed_hosts=*
denied_hosts=

# And similiarly for IPv6 with allowed_hosts_6 and denied_hosts_6.
# Note that IPv4-mapped IPv6 addresses (::ffff:w.x.y.z) are truncated to
# w.x.y.z and are handled as IPv4.
allowed_hosts_6=fec0::/16
denied_hosts_6=

# This thing can be done by Apache but is much simplier here - limit access
to
# Debian mirrors based on server names in the URLs
#allowed_locations=ftp.uni-kl.de,ftp.nerim.net,debian.tu-bs.de

# Apt-cacher can generate usage reports every 24 hours if you set this
# directive to 1. You can view the reports in a web browser by pointing
# to your cache machine with '/apt-cacher/report' on the end, like this:
#      http://yourcache.example.com/apt-cacher/report
# Generating reports is very fast even with many thousands of logfile
# lines, so you can safely turn this on without creating much
# additional system load.
generate_reports=1

# Apt-cacher can clean up its cache directory every 24 hours if you set
# this directive to 1. Cleaning the cache can take some time to run
# (generally in the order of a few minutes) and removes all package
# files that are not mentioned in any existing 'Packages' lists. This
# has the effect of deleting packages that have been superseded by an
# updated 'Packages' list.
clean_cache=1

# The directory to use for apt-cacher access and error logs.
# The access log records every request in the format:
# date-time|client ip address|HIT/MISS/EXPIRED|object size|object name
# The error log is slightly more free-form, and is also used for debug
# messages if debug mode is turned on.
# Note that the old 'logfile' and 'errorfile' directives are
# deprecated: if you set them explicitly they will be honoured, but it's
# better to just get rid of them from old config files.
logdir=/var/log/apt-cacher

# apt-cacher can use different methods to decide whether package lists need
to
# be updated,
```

```
# A) looking at the age of the cached files
# B) getting HTTP header from server and comparing that with cached data.
This
# method is more reliable and avoids desynchronisation of data and index
files
# but needs to transfer few bytes from the server every time somebody
requests
# the files ("apt-get update")
# Set the following value to the maximum age (in hours) for method A or to 0
# for method B
expire_hours=0

# Apt-cacher can pass all its requests to an external http proxy like
# Squid, which could be very useful if you are using an ISP that blocks
# port 80 and requires all web traffic to go through its proxy. The
# format is 'hostname:port', eg: 'proxy.example.com:8080'.
http_proxy=proxy.example.com:8080

# Use of an external proxy can be turned on or off with this flag.
# Value should be either 0 (off) or 1 (on).
use_proxy=0

# External http proxy sometimes need authentication to get full access. The
# format is 'username:password'.
http_proxy_auth=proxyuser:proxypass

# Use of external proxy authentication can be turned on or off with this
flag.
# Value should be either 0 (off) or 1 (on).
use_proxy_auth=0

# Rate limiting sets the maximum bandwidth in bytes per second to use
# for fetching packages. Syntax is fully defined in 'man wget'.
# Use 'k' or 'm' to use kilobits or megabits / second: eg, 'limit=25k'.
# Use 0 or a negative value for no rate limiting.
limit=0

# Debug mode makes apt-cacher spew a lot of extra debug junk to the
# error log (whose location is defined with the 'logdir' directive).
# Leave this off unless you need it, or your error log will get very
# big. Acceptable values are 0 or 1.
debug=0

# Adapt the line in the usage info web page to match your server
configuration
#
example_sources_line=deb http://<b>my.cacher.server:3142/</b>ftp.au.debi
an.org/debian unstable main contrib non-free

# Print a 410 (Gone) HTTP message with the specified text when accessed via
# CGI. Useful to tell users to adapt their sources.list files when the
# apt-cacher server is beeing relocated (via apt-get's error messages while
# running "update")
#cgi_advise_to_use = Please use http://cacheserver:3142/ as apt-cacher access
URL
#cgi_advise_to_use = Server relocated. To change sources.list, run perl -pe
"s,/apt-cacher\??,:3142," -i /etc/apt/sources.list

# Server mapping - this allows to hide real server names behind virtual paths
# that appear in the access URL. This method is known from apt-proxy. This is
# also the only method to use FTP access to the target hosts. The syntax is
simple, the part of the beginning to replace, followed by a list of mirror
urls, all space separated. Multiple profile are separated by semicolons
```

```
# path_map = debian ftp.uni-kl.de/pub/linux/debian
ftp2.de.debian.org/debian ; ubuntu archive.ubuntu.com/ubuntu ; security
security.debian.org/debian-security ftp2.de.debian.org/debian-security
# Note that you need to specify all target servers in the allowed_locations
# options if you make use of it. Also note that the paths should not overlap
# each other. FTP access method not supported yet, maybe in the future.
```

Une fois tous ces réglages terminé, activez les en relançant « apt-cacher » avec la commande:

```
/etc/init.d/apt-cacher restart
```

Afin d'informer vos PC de la présence de ce proxy, vous devez éditer le fichier « /etc/apt/apt.conf » afin d'y ajouter la ligne:

```
Acquire::http::Proxy "http://DebianServer:3142/apt-cacher/";
```

Lors d'une nouvelle installation à partir de "netinst" CD, vous devez saisir la valeur « http://DebianServer:3142/apt-cacher/» dans le masque « mandataire HTTP ».

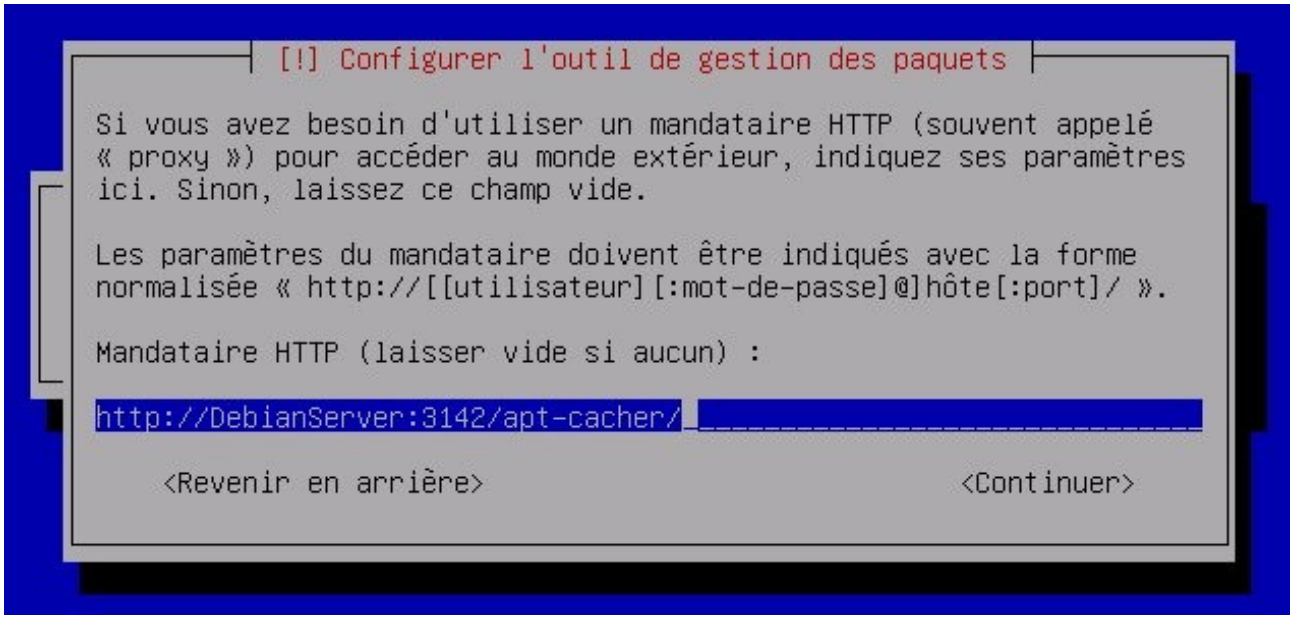


Illustration 9: Configuration de "apt-cacher" à l'installation de Debian

1.j) Dépôts collaboratifs pour les paquets: debTorrent

On parle d'une relation client-serveur, c'est à dire que personne d'autre n'entre en relation avec vous et la machine qui vous envoie les données.

Lorsque vous téléchargez une application avec la commande « apt-get », votre ordinateur demande les fichiers nécessaires à l'installation aux serveurs listés dans « /etc/apt/sources.list » . Ce serveur réponds peut-être au même moment à des requêtes provenant de centaines d'utilisateurs. Du coup, le téléchargement peux être considérablement ralenti.

Lors d'un téléchargement en « peer 2 peer », il n'y a pas (ou presque) de serveur. Si vous demandez un paquet, le programme va chercher d'autres utilisateurs disposant déjà du même fichier et va leur demander de le renvoyer: le téléchargement est relayé par une multitude d'utilisateurs, pas par un serveur potentiellement saturé!

Illustration 10: Illustration d'un réseau Peer2Peer de partage de paquets ".deb"

Il existe deux utilitaires permettant de mettre en œuvre cette technologie

- « apt-P2P »[11] fonctionnant exclusivement sous Ubuntu,
- « debTorrent »[12] fonctionnant sous Debian et Ubuntu.

« Apt-P2P » est compatible avec les clients « debTorrent ». Tout deux utilisent le protocole «BitTorrent ». Les techniques utilisées sont : le téléchargement Poste à poste depuis différents « pairs » (*peer*) pour un même fichier (on nomme cette technique « multisourcing ») et le morcellement du fichier en blocs qui le permet. Les blocs peuvent arriver dans un ordre quelconque depuis des sources multiples, le fichier étant réputé téléchargé lorsque la totalité des blocs sera parvenue, quel que soit l'ordre d'arrivée de ceux-ci ou leurs provenances - qui n'ont de fait aucune importance. L'efficacité du réseau est maximale lorsqu'il y a beaucoup d'utilisateurs, car tous ceux qui téléchargent partagent par construction ce qu'ils téléchargent[13].

L'installation de « debTorrent » est réalisé par la commande:

```
apt-get install debtorrent apt-transport-debtorrent
```

Ce paquet ne requière à priori aucune configuration. Néanmoins, le fichier de configuration est situé dans « /etc/debtorrent/debtorrent-client »
Pour utiliser « debtorrent », vous devez simplement éditer le fichier de dépôts « /etc/apt/sources.list » afin de remplacer toutes les occurrences de « deb http:// » par « deb debtorrent://localhost:9988/ » puis relancer la mise à jour de la base de paquets avec la commande:

```
apt-get update
```

Les sources ne sont pas gérées par « debtorrent » aussi il est inutile de modifier les lignes débutant par « deb-src ».
Vous pouvez visualiser la progression de vos téléchargement depuis l'URL « http://localhost:9988/ ».

2) CrunchBang Linux

Cette revue des différents outils de gestion des paquets sous Debian/Ubuntu n'a rien d'exhaustif mais il nous fourni un rappel des fonctions que nous allons utiliser pour passer d'une distribution basée sur Ubuntu à une distribution basée sur Debian. Il existe de nombreux sites où trouver de l'information complémentaires sur l'usage avancé des outils « APT ». Je citerai par exemple: http://www.luxpopuli.fr/Systeme/Debian/APT-gerer-les-packages-debian.

Revenons donc à CrunchBang Linux! Cette distribution a été conçue pour fonctionner sur des configurations matérielles minimales sur lesquelles permets un excellent équilibre entre fonctionnalités et performances. Pour arriver à ce résultat, CrunchBang utilise le gestionnaire de

11 http://www.camrdale.org/apt-p2p/install/
12 http://debtorrent.alioth.debian.org/
13 http://fr.wikipedia.org/wiki/BitTorrent

fenêtre OpenBox[14] et par conséquent ne consomme que 40 Mo de mémoire vive.

CrunchBang est très clairement destinée à un usage bureautique, graphique, internet et multimédia. D'ailleurs, le support des fichiers MP3, Adobe Flash et la lecture des DVD est déjà activé.

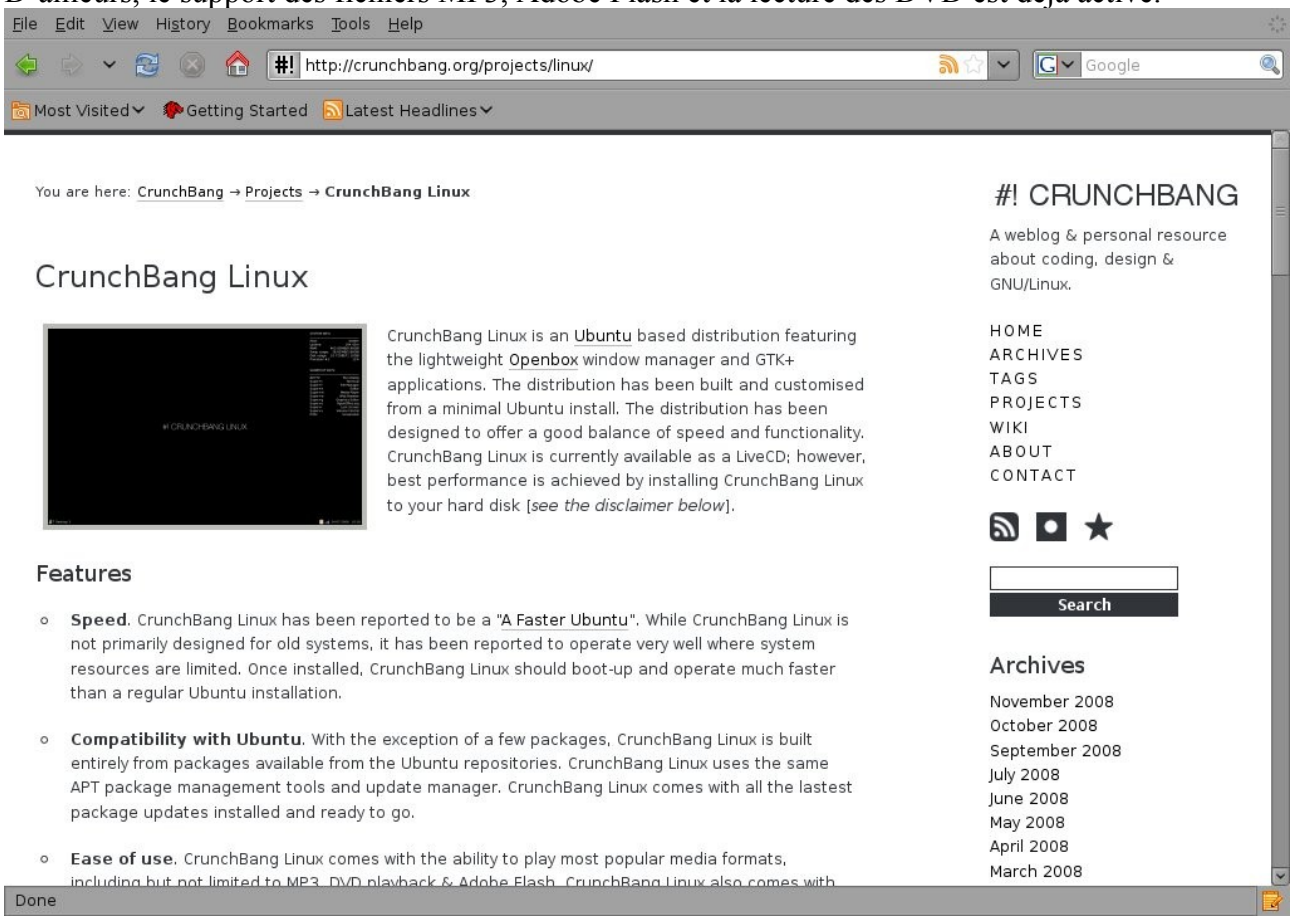


Illustration 11: Site de la distribution CrunchBang Linux

Comme Ubuntu, CrunchBang est disponible sous la forme d'un CD-ROM de 580MB est un LiveCD mais vous ne pourrez en tirer pleinement partie qu'après l'avoir installé sur votre PC.
Il existe un raccourci vers les programmes de d'installation de CrunchBang depuis le menu d'applications disponibles lorsque l'application fonctionne en mode LiveCD.
Cliquer sur bureau avec le bouton droit de la souris et choisissez dans l'arborescence des menus « System » puis « Install CrunchBang Linux ». Validez vos choix en cliquant sur le bouton « entrée » avec le bouton gauche de la souris.

14http://icculus.org/openbox/index.php/Main_Page

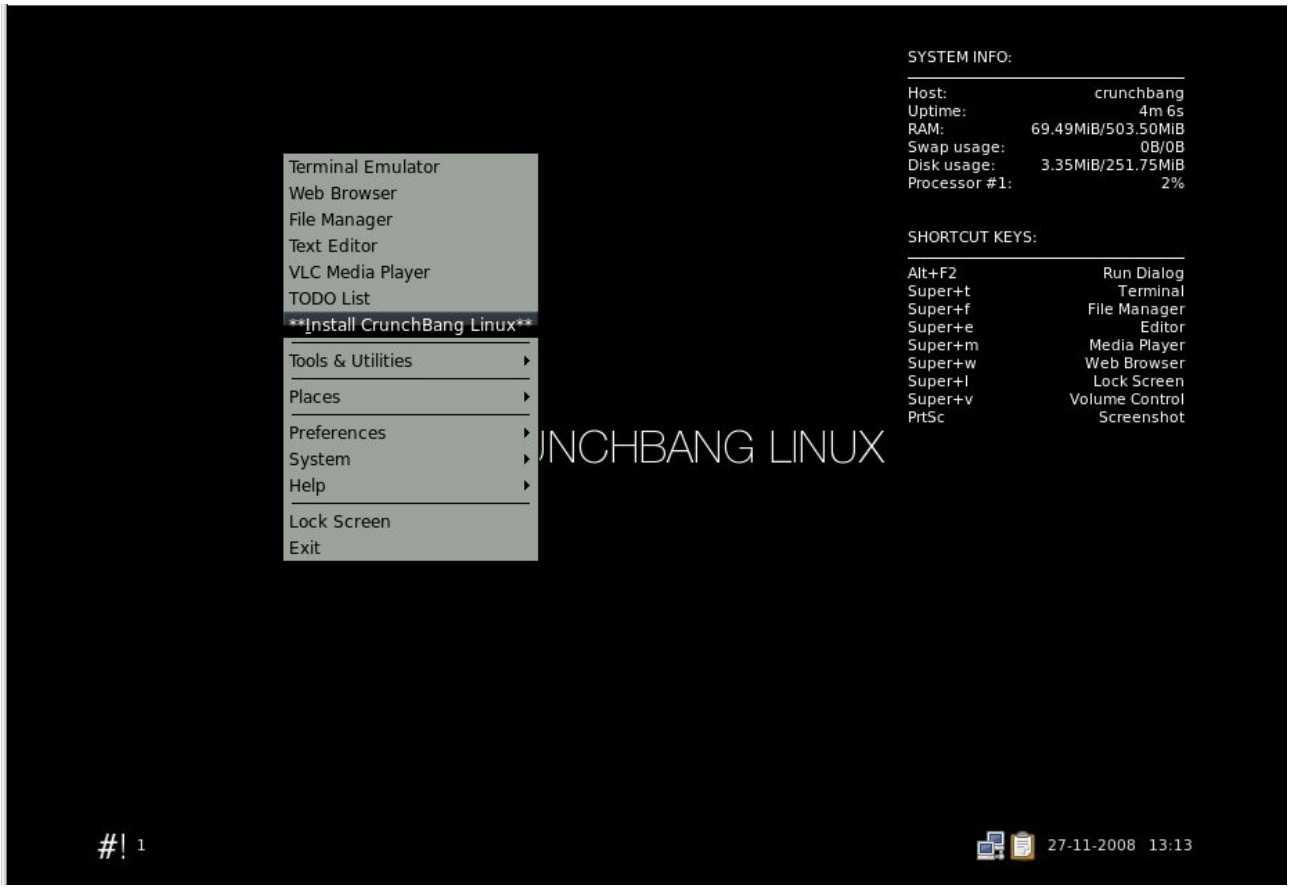


Illustration 12: Raccourci vers le menu d'installation de CrunchBang Linux

Depuis peu, une version Lite de cette distribution est disponible depuis http://crunchbang.org. Cette version comprends tout le système de base mais très peu d'applications. C'est avec la version Lite que nous allons travailler au cours de cet article afin de ne pas nous embarrasser de paquets logiciels non-significatifs.

Je vous invite donc à télécharger ce CDROM de 403M et de l'installer sur un PC.

Vous allez ensuite, depuis un terminal et en la qualité de « root » demander la liste des paquets installés. Afin d'étudier le contenu de cette liste à loisir, vous allez exporter le résultat de la commande « dpkg » dans un fichier texte nommé « liste.txt ». Pour cela, il vous suffit d'utiliser l'opérateur de redirection « > ». Afin d'être bien certain de ne prendre en compte que les fichiers effectivment installé, vous allez filtrer les lignes contenant la séquence de caractère '^ii'.

```
dpkg -l |grep '^ii'|  awk '{print $2}' > liste.txt
```

« awk » est un langage de traitement de lignes. Il est principalement utilisé pour la manipulation de fichiers textuels pour des opérations de recherches, de remplacement et de transformations complexes. La syntaxe est inspirée du C :

```
awk [options] [programme] [fichier]
```

où la structure du programme est :

```
'motif1 { action1 } motif2 { action2 } ...'
```

Chaque ligne du fichier est comparée successivement aux différents motifs (le plus souvent des expressions rationnelles, et globalement une expression booléenne) et l'action du premier motif renvoyant la valeur vraie est exécutée. Dans ce cas, ou si aucun motif n'est accepté, le programme lit la ligne suivante du fichier et la compare aux motifs en partant du premier. Dans la commande ci-dessus, « awk '{print $2}' » affiche la seconde colonne de chaque ligne résultant du traitement précédent.[15] Le pipe « | » engendre l'enchaînement les traitements.

Le contenu du liste « liste.txt » peut être listé avec la commande « cat »[16] ou edité avec le

15 http://fr.wikipedia.org/wiki/Awk
16 http://en.wikipedia.org/wiki/Cat_(Unix)

programme « nano »[17].

```
cat liste.txt
nano liste.txt
```

Vous allez utiliser la commande « wc »[18] pour compter le nombre de paquets qui composent la distribution CrunchBang:

```
wc -l liste.txt
```

Le résultat est sans appel: cette distribution est composée de 862 paquets! Afin d'aborder sereinement le projet de migration, vous allez devoir réduire au maximum le nombre de logiciels et librairie à migrer de Ubuntu vers Debian.
En effet, sur les 862 paquets listés ci-dessus, il y a fort à parier qu'un grande nombre de ces paquets ne sont en réalité que des dépendances de paquets « supérieurs » et que par conséquent, si l'on se contente de migrer les programmes de haut niveau, les dépendances requises seront automatiquement installées par les outils « apt ».
Installez l'outils apt-rdepends » afin d'étudier les dépendances des logiciels.

```
apt-get install -y apt-rdepends
```

Prenons le programme «bash» listé. La commande suivante retourne un fichier « rdep.txt » d'une longueur de xx lignes.

```
apt-rdepends -r --state-follow=Installed -state-show=Installed
bash > rdep.txt
```

En effet, ce produit est listé comme dépendance logiciels de 7 paquets dont « ubuntu-minimal ». Par contre, la commande ci-dessous ne produit qu'un fichier de 1 lignes. En effet, « ubuntu-minimal » est un paquet de niveau supérieur dont aucun autre logiciel ne dépends. Cette appproche sous-entends que tous les dépendances d'un paquets sont bien installées ce qui n'est pas forcément le cas si la distribution est très optimisée pour réduire son espace disque.

```
apt-rdepends -r --state-follow=Installed -state-show=Installed
ubuntu-minimal > rdep.txt
wc -l rdep.txt
1
```

Il ne nous reste qu'on plus qu'à appliquer « apt-rdepends » à chacun des fichiers listés dans « liste.txt » et ne garder que les fichiers générant un fichier ne contenant que 3 lignes. Bien évidemment, il est hors de question de réaliser cette tâche à la main. Pour cela, nous allons réaliser un petit script «bash ». Un script est un fichier texte contenant une série de commandes qui en temps normal seraient exécutées manuellement par l'utilisateur d'un terminal. Traditionnellement, on donne à ces fichiers l'extension « .sh ». Par exemple, créez un fichier nommé « go.sh » ave l'éditeur « nano »;

```
nano  go.sh
```

Dans ce fichier, tapez simplement la ligne « ls » qui corresponds à la commande de listage du répertoire courant. Sauvegardez le fichier et donnez lui le droit d'être exécuté avec la commande:

```
chmod +x go.sh
```

Enfin, vous pouvez lancer ce script avec la commande:

```
./go.sh
```

Sans surprise, il va afficher le contenu du répertoire courant.

Un script peux contenir des instructions de tests conditionnels et de boucle.

17 http://www.gentoo.org/doc/en/nano-basics-guide.xml
18 http://en.wikipedia.org/wiki/Wc_(Unix)

2.a.i) Exécution conditionnelles dans un script: if/then/elif

Lorsque certains conditions sont validée, il peux être nécessaire de réaliser un traitement spécifique. La syntaxe de la commande « si » ... « alors » ... « sinon » en « bash » est:

```
if condition
then
        traitement1
else
        traitement2
fi
```

traitement1 est exécutés si la condition est vraie, sinon, c'est traitemetn2 qui est éxécuté.

La condition est généralement la forme: [operand1 opérateur operand2]. A titre d'exemple, si le contenu de la variable $USER est « gnulamp » alors lancer le trainement: «echo "User is gnulamp" » sinon ne rien faire.

```
#!/bin/bash
if [ $USER = "gnulamp" ]; then
echo "User is gnulamp"
fi
```

Le tableau ci -dessous résume les principaux opérateurs:

opérateur	Retourne true si...	Nombre d'opérandes
-n	Opérande de longueur supérieure à 0	1
-z	opérande de langueur nulle	1
-d	Il existe un répertoire du nom de l'opérande	1
-f	Il existe un fichier du nom de l'opérande	1
-eq	Les opérandes sont des entiers et sont égaux	2
-neq	L'inverse de « -eq »	2
=	Les chaines des opérandes sont identiques	2
!=	L'inverse de « = »	2
-lt	operand1 est strictement inférieur à operand2 (les deux opérandes sont des entiers)	2
-gt	operand1 est strictement supérieur à operand2 (les deux opérandes sont des entiers)	2
-ge	Operand1 est supérieur ou égal à operand2 (les deux opérandes sont des entiers)	2
-le	operand1 est inférieur ou égal à operand2 (les deux opérandes sont des entiers)	2

2.a.ii) Les boucles

Les boucles sont des constructions qui permettent d'exécuter plusieurs fois un même jeu d'instruction en faisant varier les paramètres. Il existe deux type d boucle dans « bash »:

- les boucles « for »
- les boucles « while »

boucle « for «

La syntaxes des boucles « for » est:

```
#!/bin/bash
```

```
for X in red green blue
do
        echo $X
done
```

La boucle « for » va s'exécuter trois fois: une fois pour la valeur « red », une seconde fois pour la valeur « green » et une dernière fois pour la valeur « blue ». A chaque étape, la variable X est initialisée avec la valeur courante. Ainsi, lors de la première itération, X contient la valeur « red ». la commande « echo $X » affiche dans le terminal le contenu de la variable X. résultat du traitement précédent est:

```
red
green
blue
```

boucle « while »

La syntaxes des boucles «while» est:

```
#!/bin/bash
X=0
while [ $X -le 4 ]
do
        echo $X
        X=$((X+1))
done
```

La variable est initialisée avec la valeur 0 . Tant que le contenu de la variable X reste inférieur ou égal à la valeur 4, alors la valeur de X est affichée puis X (« echo $X »)est augmenté de 1 (« X=$ ((X+1)) ». Le résultat de ce traitement sera:

```
0
1
2
3
4
```

Il existe un très grand nombre de tutoriels sur Internet pour s'initier à la programmation de scripts bash. Personnellement, je vous invite à lire http://www.ibm.com/developerworks/library/l-bash.html et http://www.gnulamp.com/bashprogramming.html.

Revenons maintenant à notre script de traitement des dépendances. Saisissez le script ci-dessous dans un fichier nommé « go.sh »

```
#!/bin/bash
echo "" > sup.txt
echo "" > inf.txt
for i in `cat liste.txt`; do
 apt-rdepends -r --state-follow=Installed --state-show=Installed $i >apt_temp
 if [ $(wc -l < apt_temp) -eq 1 ]
 then
  echo $i >> sup.txt
 else
  echo $i >> inf.txt
 fi
done;
```

Listing « go1.sh »

Ce petit programme crée tout d'abord deux fichiers vides (« echo « » > »):

- « sup.txt » qui contiendra la liste des pquets de niveaux supérieurs,
- « inf.txt » qui contiendra la liste des paquets de niveau inférieur.

Ensuite, pour chacun des éléments (« for i in ... do ») listés dans le fichier « liste.txt » (« cat liste.txt »), la commande « apt-rdepends -r --state-follow=Installed --state-show=Installed $i ») est

exécutée. « $i » est remplacé par la valeur de l'élément courant. Plutôt que d'afficher le résultat dans la console, la sortie de la commande est redirigées vers un fichier temporaire : « apt_temp ». Si ce fichier n'existe pas, il est crée automatiquement par l'opérateur de redirection « > ». Ensuite, le nombre de lignes de ce fichier (« wc -l < apt_temp ») est comparé avec la valeur « 1 » (« [$(wc -l < apt_temp) -eq 1] »). Si « apt_temp » ne contient qu'une ligne, alors le nom du paquets courant contenu dans la variable « $i » est ajouté à la fin du fichier « sup.txt » grâce à l'opérateur « >> ». Si le nombre de ligne est différent de « 1 » alors « $i » est ajouté à la fin du fichier « inf.txt ».
Lancez le script avec la commande:

```
chmod +x go.sh
./go.sh
```

Le script ne devrait prendre que quelques minutes pour s'exécuter. Comparez ensuite le nombre de ligne:

- « liste.txt » (« wc -l liste.txt ») => 862 paquets
- « sup.txt » (« wc -l sup.txt ») => 130 paquets (j'ai retiré 1 car la première ligne est vide)
- « inf.txt » (« wc -l inf.txt ») => 732 paquets (j'ai retiré 1 car la première ligne est vide)

Ceci signifie que notre distribution est en réalité composée de 130 logiciels; tous les autres paquets seront installés automatiquement au travers des outils « apt ».

3) Debian Lenny

Le projet Debian est une association de personnes quji ont décidées de travailler en commun à la création d'un système opérationnel gratuit. Ce système s'appelle Debian GNU/Linux. Debian utilise le noyau Linux initialement crée par Linus Torvalds. De nombreux projets utilise Debian comme base de développement, le plus répandue étant probablement « Ubuntu ».Le. nom de code de la prochaine édition majeure de Debian après Etch est Lenny .

Cette version a été initialisée à partir d'une copie de Etch, et se trouve pour l'instant dans une phase dite « de test » . Cela signifie que vous ne devriez pas souffrir des mêmes problèmes qu'avec les distributions instable ou expérimentale, car les paquets n'entrent dans cette distribution qu'après une certaine période de test, et s'ils n'ont pas de bogues critiques. C'est cette version que installerons sur notre PC de test

Avant toute chose, il vous faut télécharger une version du support d'installation de la distribution Debian depuis le site http://www.debian.org/CD/netinst/. Il existe une grande variété de CD/DVD/disquette d'installation. Vous choisirez le version « netinst » pour plateforme « i386 » ; il s'agit d'un CDROM de seulement 180Mo qui va vous permettre d'installer rapidement le cœur du système – les applications supplémentaire seront téléchargées depuis internet. Pour information, il existe également une version « businesscard » de 40Mo mais alors l'essentiel de la distribution est téléchargée à la volée ce qui peut être assez long en fonction du débit internet dont vous disposez. Enfin, si vous n'avez pas la possibilité de vous connecter à internet au moment de l'installation, sachez que vous pouvez télécharger l'intégralité de tous les paquets disponibles sous la forme de quatre DVD.

Votre image ISO se nomme http://cdimage.debian.org/cdimage/lenny_di_rc1/i386/iso-cd/debian-testing-i386-netinst.iso.

Après quelques minutes pendant lesquelles votre système de base est installé sur votre disque dur, l'installeur va revenir vers vous pour vous demander si vous souhaitez compléter votre distribution par des programmes additionnels. Par défaut, il vous propose l'environnent graphique GNOME (http://www.gnomefr.org/) et tous les utilitaires bureautiques et Internet traditionnels. Il est inutile d'installer d'autre éléments que le système standard.

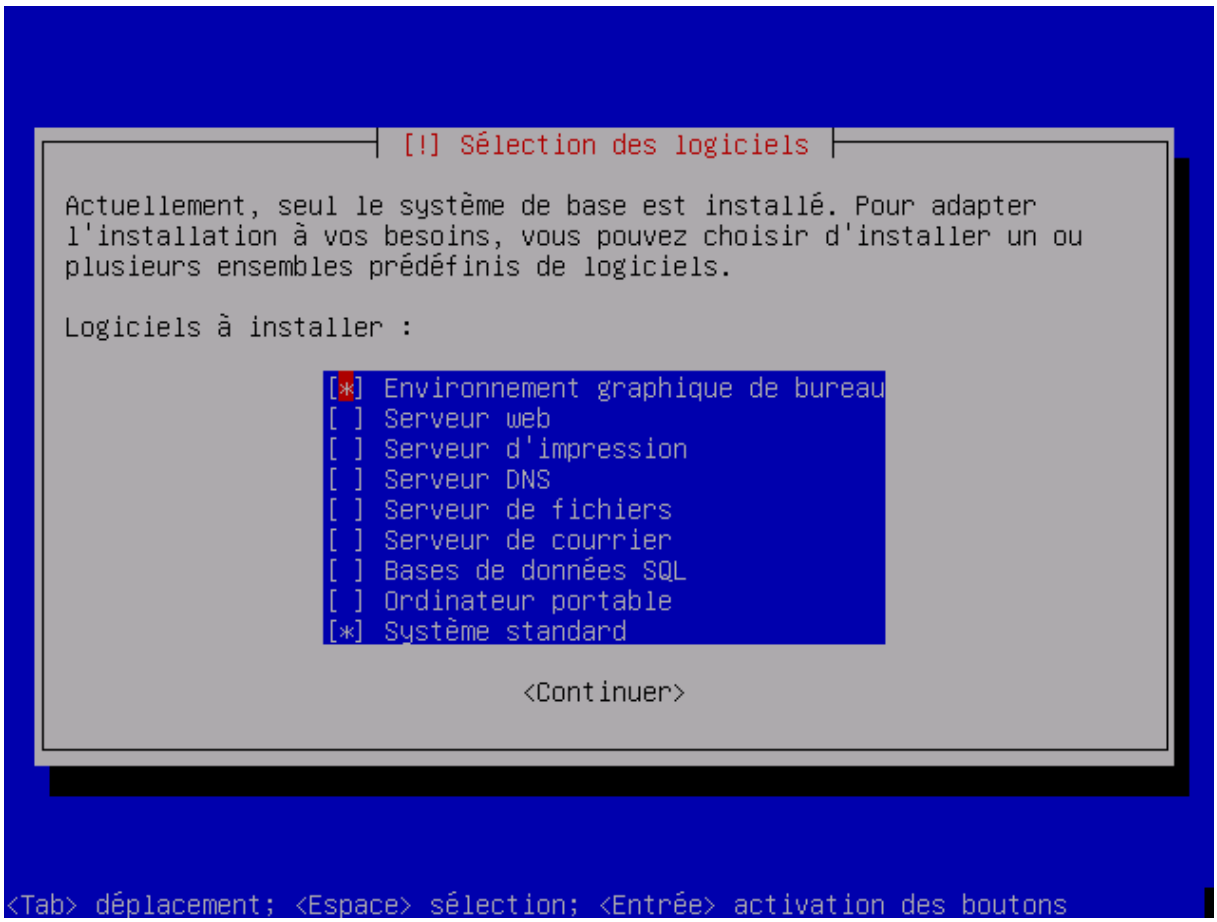


Figure 1: Choix des applications à installer

Un fois que votre nouveau système est prêt, copiez y le fichier « sup.txt » que vous avez récupérez depuis le poste équipé de CrunchBang Linux. Vous allez créer un script dont la structure est fort similaires à script étudié plus haut sauf que l'opbejctif de celui-ce sera de tester la disponibilité des paquets listes dans « sup.txt » dans l'environnement « Lenny ». Pour cela, vous allez utilisez la commande « apt-get -y install» mais en y ajoutant l'option « -s » de sorte que rien ne soit réellement installé. Seul la disponibilité du paquets est validée. Comme précédemment, le résultat de la commande est renvoyé vers un fichier temporaire « apt_temp ». Si le paquet mentionné dans la variable « $i » n'existe pas sous « Lenny » alors « apt_temp » contiendra 3 lignes et le nom du paquets sera ajouté à un fichier nommé « notok.txt ».

```
#!/bin/bash
echo "" > notok.txt
echo "" > ok.txt
for i in `cat sup.txt`; do
  apt-get -y -s install $i >apt_temp
  if [ $(wc -l < apt_temp) -eq 3 ]
  then
    echo $i >> notok.txt
  else
    echo $i >> ok.txt
  fi
done;
```

Listing « go2.sh »

Le résultat est très encourageants:

- « sup.txt » (« wc -l supp.txt ») => 130 paquets
- « notok.txt » (« wc -l notok.txt ») => 23 paquets (j'ai retiré 1 car la première ligne est vide)
- « ok.txt » (« wc -l ok.txt ») => 107 paquets (j'ai retiré 1 car la première ligne est vide)

Ceci signifie que seul 24 paquets logiciel présents sur CrunchBang n'existent pas sous Lenny
24 paquets son dispos sous Lenny. Jetez un coup d'œil au contenu de « notok.txt ».

```
apparmor-utils
```

```
command-not-found
crunchbang-look
firefox-gnome-support
friendly-recovery
gstreamer0.10-pitfdll
gstreamer0.10-plugins-ugly-multiverse
jockey-gtk
language-selector
linux-generic
medibuntu-keyring
python-exo
remastersys
tablaunch
transset
transset-df
ubiquity-frontend-gtk
ubiquity-ubuntu-artwork
ubuntu-minimal
ubuntu-standard
usplash-theme-crunchbang
xfce4-governor-plugin
xfce4-mcs-plugins-extrau
```

Il est clair que des paquets comme « ubuntu-standard » avaient peu de chance de trouver un équivalent sous « Lenny »!
Retournez un instant et regardez ce que contiennent chacun de ces paquets. Prenons le cas de « ubuntu-standard ».

```
apt-rdepends --state-follow=Installed --state-show=Installed
ubuntu-standard
```

Vous trouverez ci-dessous les listes des paquets dont dépends « ubuntu-standard ». Certains paquets comme celui-ci n'installe aucun programme propre mais ne sont qu'un raccourci pour installer une longue de liste des programmes dont l'usage est lié: on appelle ce type des paquets des « métapaquets ».

```
  Depends: at
  Depends: cpio
  Depends: cron
  Depends: dmidecode
  Depends: dnsutils
  Depends: dosfstools
  Depends: ed
  Depends: file
  Depends: ftp
  Depends: hdparm
  Depends: info
  Depends: inputattach
  Depends: iptables
  Depends: logrotate
  Depends: lshw
  Depends: lsof
  Depends: ltrace
  Depends: man-db
  Depends: memtest86+
  Depends: mime-support
  Depends: nano
  Depends: parted
  Depends: popularity-contest
  Depends: psmisc
  Depends: rsync
  Depends: strace
  Depends: time
  Depends: ufw
  Depends: w3m
```

```
  Depends: wget
```

Ajoutez ces paquet à « ok.txt » puis répétez cette opération pour les 22 autres paquets. Le script suivant « go4.sh » va vous assister dans cette tâche en créant un fichier pour chaque paquet. Ce fichier contient la liste des dépendances.

```
#!/bin/bash
for i in `cat notok.txt`; do
 apt-rdepends --state-follow=Installed --state-show=Installed $i >$i
done;
```

Script go4.sh

Copier la version complétée du fichier « sup.txt » en nommant le résultat « liste2.txt ».

```
cp ok.txt liste2.txt
```

Lancez le script « go5.sh » avec sur « Lenny ». Ce script va vérifier la disponibilité des paquets cités dans « liste2.txt » sous Debian.

```
#!/bin/bash
echo "" > notok2.txt
echo "" > ok2.txt
for i in `cat liste2.txt`; do
  apt-get -y -s install $i >apt_temp
  if [ $(wc -l < apt_temp) -eq 3 ]
  then
    echo $i >> notok2.txt
  else
    echo $i >> ok2.txt
  fi
done;
```

Script go5.sh

Il ne reste plus que 24 fichiers (« wc -l notok2.txt ») non compatibles avec Debian (à ajouter à ceux de « notok.txt »):

```
crunchbang-gdm-theme
crunchbang-theme
crunchbang-wallpapers
tango-icon-theme-common
tango-icon-theme-extras
apparmor
command-not-found-data
firefox-3.0-gnome-support
jockey-common
language-selector-common
linux-image-generic
linux-restricted-modules-generic
```

```
casper
ubiquity
libffi4
ubiquity
ubiquity-ubuntu-artwork
startup-tasks
system-services
ubuntu-keyring
upstart-compat-sysv
upstart-logd
inputattach
liblame0
```

Pour ces logiciels, il vous faudra comprendre leurs rôles, juger de la pertinence de les réinstaller sous Debian et trouver un équivalent.

Réduisez la taille de « ok2.txt » en éliminant les paquets de niveau inférieur avec le script « go6.sh ».

```
#!/bin/bash
echo "" > sup2.txt
echo "" > inf2.txt
for i in `cat ok2.txt`; do
 apt-rdepends -r --state-follow=Installed --state-show=Installed $i >apt_temp
 if [ $(wc -l < apt_temp) -eq 1 ]
 then
  echo $i >> sup2.txt
 else
  echo $i >> inf2.txt
 fi
done;
```

Script go6.sh

Toute la distribution tient dans 259 paquets de niveau supérieurs (« wc -l sup2.txt »).

Désormais, il ne reste plus qu'à les installer dans la distribution Debian minimale que vous avez préparé plus tôt. Pour cela, vous pouvez vous aider du script « go3.sh ».

```
#!/bin/bash
for i in `cat sup2.txt`; do
  apt-get -y install $i
done;
```

Listing go3.sh

L'installation ne prends que quelques minutes; vous relancez le PC et là: Déception!!!! Votre nouveau bureau ressemble bien plus à GNOME qu'à OpenBox.

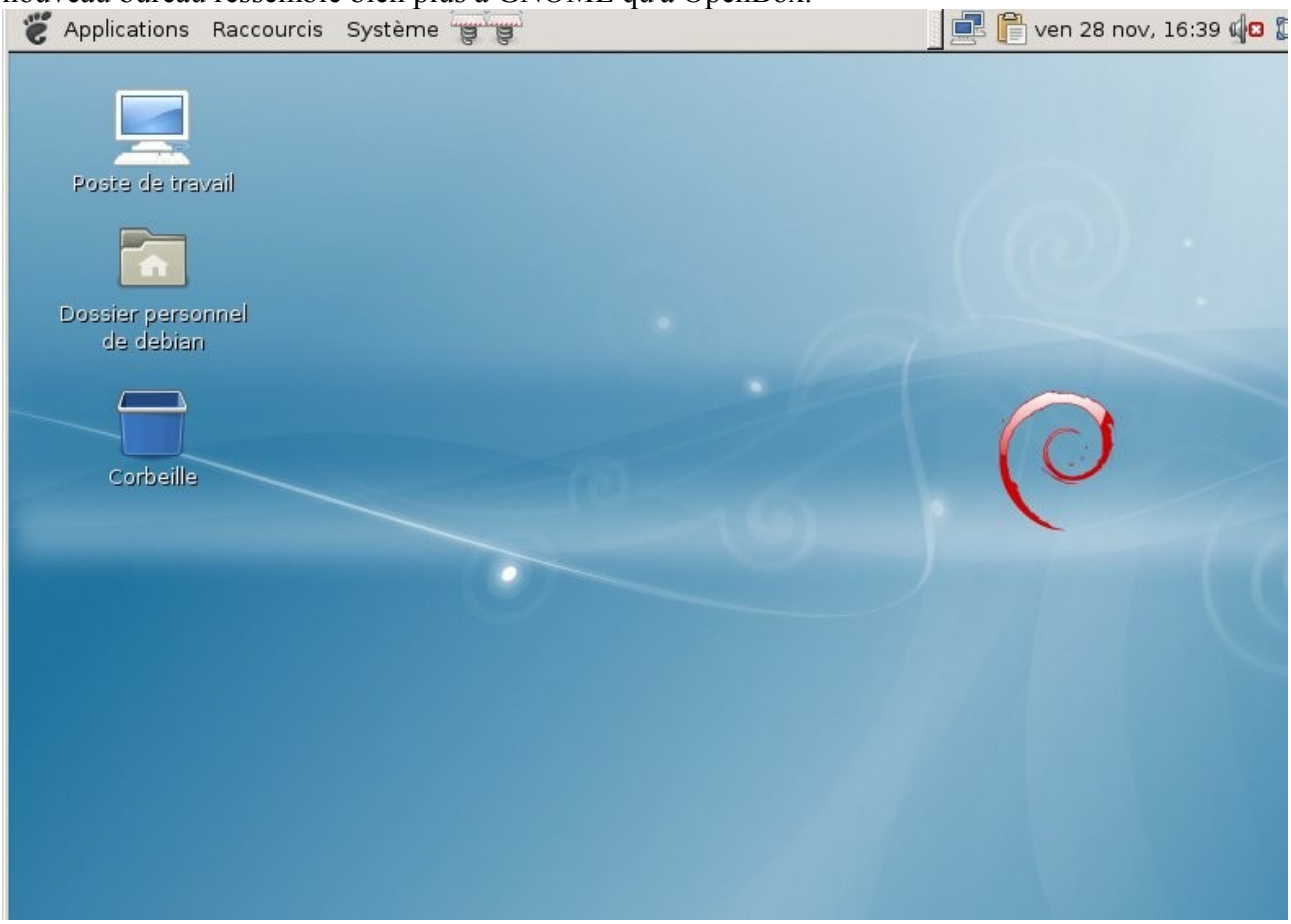


Illustration 13: Bureau Lenny après installation "personnalisée"

Certes vous avez installé tous les programmes nécessaires à votre distribution personnalitée mais avec les fichiers de configuration par défaut lesquels sont plutôt orientés vers le bureau GNOME[19]. Prenez le cas du sélection de session GDM[20]. Au moment de vous identifier, cliquez sur le bouton « session » et choisissez un session de type « openbox session ». Vous verrez que votre bureau sera déjàbeaucoup plus proche de vos aspirations.

19 http://www.gnome.org/
20 http://fr.wikipedia.org/wiki/GNOME_Display_Manager

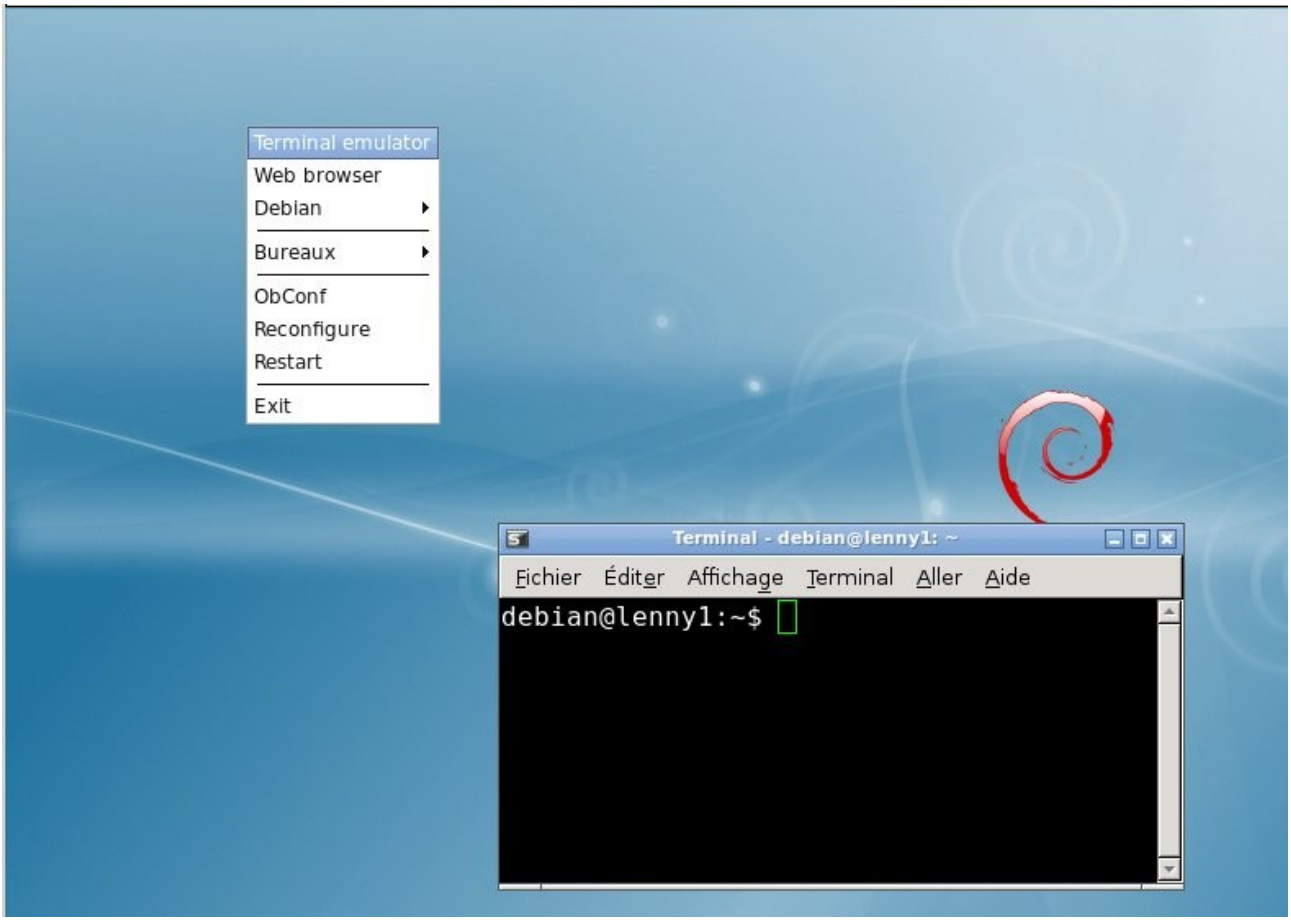


Illustration 14: Le bureau "openbox" sous Lenny

Vous allez devoir persévérer un pe!. OpenBox repose sur trois fichiers de configuration principaux situé dans le répertoire « ~/.config »:

- menu.xml : configuration du menu applicatif
- rc.xml : configuration des paramètres de OpenBox
- autostart.sh : script de lancement des programmes démarrant avec la session graphique

« conky » - le petit moniteur système qui s'incruste dans le fond d'écran – repose sur le fichier de configuration « ~/.conkyrc ».

Le thème très sombre et sobre de CrunchBang est composé de fichiers placés dans l'arborescence « /usr/share/themes/CrunchBang ».

Copiez ces fichiers et vous aurez alors un résultat bien plus convainquant!

4) Conclusion

Au terme de cet article, vous devez disposer d'un environnement reprenant tous les avantages de la distribution CrunchBang mais non plus lié à la plateforme Ubuntu. Bien évidemment, il reste beaucoup de réglages fins à réaliser mais, plutôt que de dépendre d'une distribution pré-configurée, vous ête désormais aux « commandes ». Profitez-en pour créer votre propre environement de travail car Linux est d'une très grande souplesse et offre de multiples variantes. Vous n'êtes pas prêt de vous ennuyer!

L'approche un peu brutale présentée dans le cadre de cet article présente quand même un désavantage: dans le fond, à moins de scruter le contenu de « ok2.txt » et de comprendre le rôle que chacun des programmes qui y sont listés, vous ne savez toujours pas de quoi est composée votre ditribution. Il est fort probable qu'un grande nombre de paquets inutiles pour votre usage ont été installé. Sachez qu'il existe de nombreux projets de bureau léger pour Lenny. Le premier se nomme LXDE[21]. LXDE a pour but de proposer un nouvel environnement de bureau léger, rapide et utilisant peu de ressources, au détriment du nombre de fonctionnalités. Il se veut modulaire : ses composants

21 http://lxde.org/

dépendent peu les uns des autres. LXDE utilise GTK+ modifié pour une plus grande rapidité. Son gestionnaire de fenêtres par défaut est Openbox, et son gestionnaire de fichiers par défaut PCManFM.

Cet environnement de bureau comporte également :

- LXPanel, panneau de bureau
- LXSession, gestionnaire de session
- LXAppearance, gestionnaire de thèmes
- leafpad, éditeur de texte
- xarchiver, gestionnaire d'archives
- GPicView, visionneuse d'images
- LXTerminal, émulateur de terminal
- LXTask, gestionnaire de tâches
- LXNM gestionnaire de réseau sans fil (pour Linux) [22]

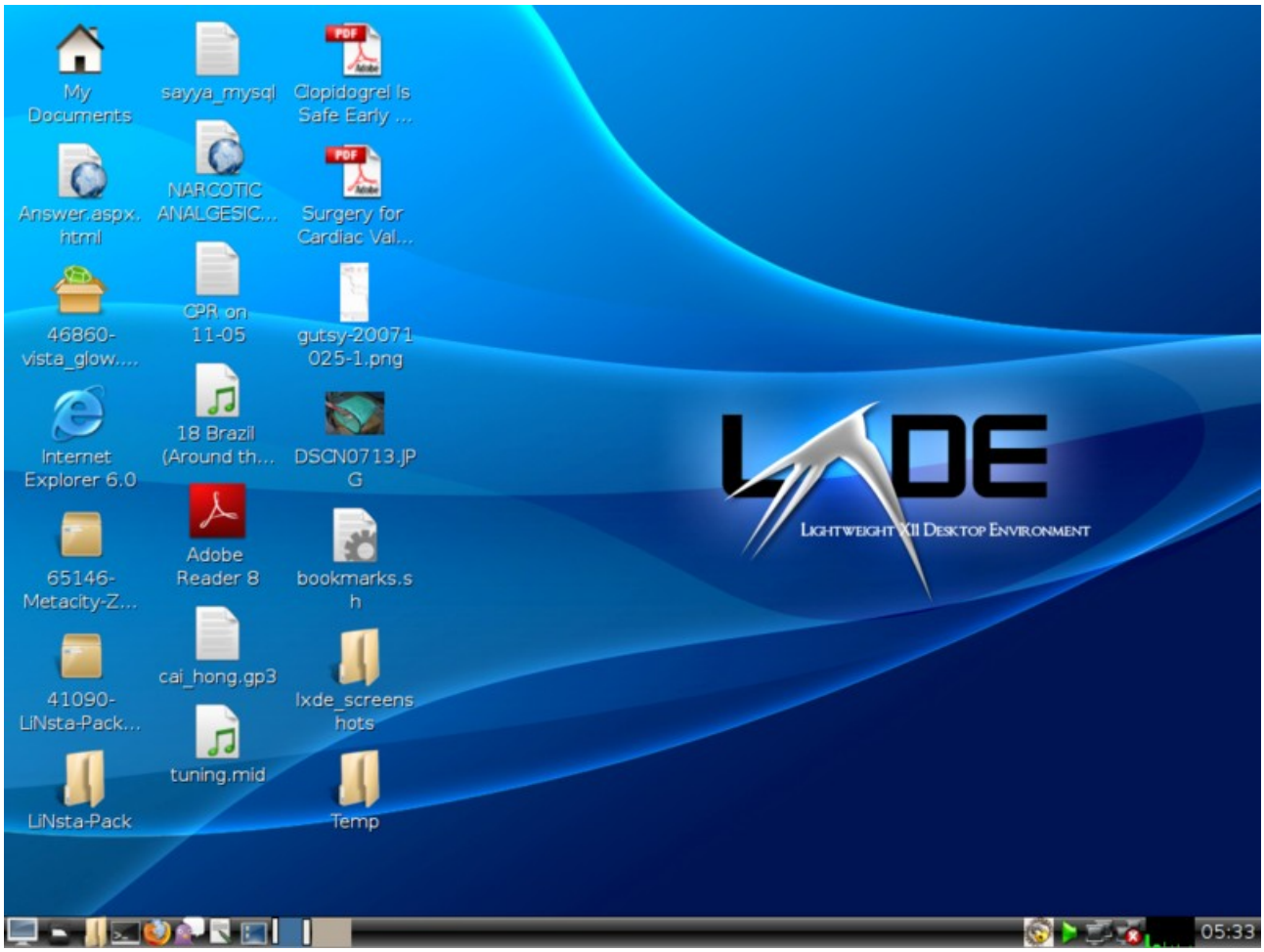


Illustration 15: Le bureau Lenny avec LXDE

Dans sa configuration de base, LXDE ressemble beaucoup à KDE[23] mais il est êtrêmement simple d'en personnaliser l'aspect et de lui donner un look « CrunchBang ». De plus, l'installation de LXDE est aussi simple qu » apt-get install -y lxde »!

L'autre projet qui a attiré mon attention est Bee Desktop[24]. Ce gestionnaire de bureau s'installe à

22 http://fr.wikipedia.org/wiki/LXDE
23 http://www.kde.org/
24http://forum.debian-fr.org/viewtopic.php?t=15964

l'aide d'un script – très richement commenté – qui n'installe que les paquets strictement nécessaires. L'installer vous demande même les applications que vous souhaitez utiliser et le résultat est très propre et parfaitement adapté aux petites config et une très bonne base pour le début d'un environnement léger personnalisé. Une fois le système installé, après le démarrage du PC le système occupe plus ou moins ~60Mo de RAM. Comme ce projet est aussi basé sur OpenBox, il est très facile de lui donner également un look « CrunchBang ».

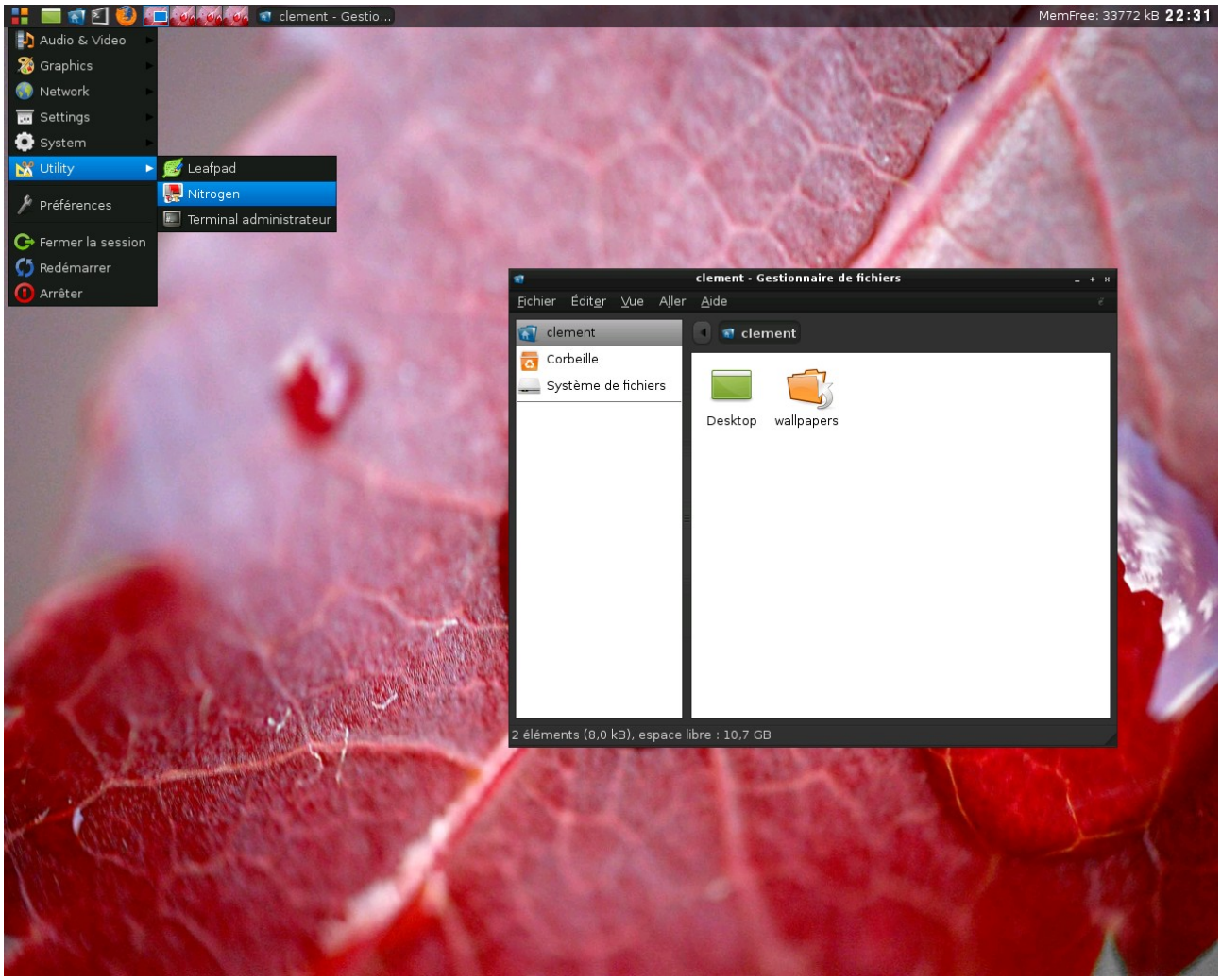


Illustration 16: Le bureau "Bee" sous Lenny

5) Cet article explique...

Cet article présente un méthodologie pour tenter de migrer une distribution basée sous Ubuntu vers une distribution basée sous Debian. Il est aussi l'opportunité de s'initiser aux outils de gestions de prgogrammes sous la plateforme Ubuntu/Debian. Il s'accompagne également d'une brève introduction à la création de scripts « bash »

6) Ce qu'il faut savoir...

Pour aborder cet article, il faut disposer des droits administrateurs et savoir installer des logiciels sous Debian depuis la ligne de commande. Une connexion à Internet est requise

7) Concernant l'auteur

Responsable informatique de l'éditeur logiciel silog.fr

Installé à Caen sur un grand coup de coeur pour la ville et sa région

Diplômé d'informatique et électromécanique de l'U.T.C

Membre de Calvix.org

Table des matières