Drop the Dropbox

Sauf à avoir des périodes d'hibernation de trois ou quatre ans, vous avez certainement entendu parler de Dropbox. C'est un outil parfait pour synchroniser un dossier de votre ordinateur avec un serveur de Dropbox mais également avec tous vos autres PC. Bien entendu, cela se fait de manière transparente.

Le problème est que Dropbox n'est pas un produit libre et que, finalement, vous stockez des informations plus ou moins confidentielles sur les serveurs d'une entreprise. Autre restriction, seule 2 Go de stockage sont alloués gratuitement, ensuite, il faut payer et le prix du Go n'est pas vraiment bon marché.

Enfin, il y a ce vicieux administrateur réseau qui vient de couper toutes les communications vers le site Dropbox pour des raisons de sécurité et autres considérations de bande passante.

Mais vous avez un avantage, vous avez un serveur Debian, et nous allons mettre en œuvre une solution libre qui devrait couvrir un certain nombre de vos demandes.

Le depôt

Dans un premier temps, nous allons créer le dépôt sur le serveur en utilisant git, à savoir, selon wikipedia : « Git est un logiciel de gestion de version décentralisée. C'est un logiciel libre créé par *Linus Torvalds*, le créateur du noyau Linux, et distribué sous la GNU GPL version 2 ». Effectivement, voir le nom de *Linus Torvalds* nous laisse penser que nous sommes en droit de nous dire que c'est un produit sérieux ;)

Comme souvent, pour installer un logiciel sur Debian, nous allons utiliser `apt-get` :

```
apt-get install git-core
```

Maintenant, il faut initialiser votre dépôt git, mais avant, créons un utilisateur qui sera propriétaire de ce dépôt. À vous d'adapter les nom et répertoire :

```
adduser --home /home/mabox/ --shell / ↵
   bin/bash mabox
```

Maintenant, il faut se logger sous le profil de l'utilisateur mabox et créer ce fameux dépôt git qui contiendra tous les documents que vous aurez synchronisés :

```
su - mabox
git init --bare mondepot.git
```

Il est important d'écrire `su - mabox` et non `su mabox` car la commande qui suit doit être exécutée dans le répertoire */home/mabox/*. D'ailleurs, pour vérifier que tout s'est bien passé, il suffit de lister les éléments du répertoire home de mabox :

```
ls /home/mabox/
```

Si, en retour, vous voyez « mondepot.git », c'est parfait, votre dépôt est bien créé au bon endroit. Pour la partie serveur c'est terminé, du moins pour le moment, nous y reviendrons un peu plus tard. En quelques commandes, vous voilà l'égal d'un administrateur système Dropbox.

Le client

Nous allons utiliser SparkleShare, un outil libre qui progresse assez rapidement et qui est disponible pour Linux, Mac OS X et bientôt Windows, Android et iOS. Bref, de quoi couvrir pas mal de systèmes qui vous entourent.

SparkleShare est donc compatible avec Debian. Vous disposez certainement d'une autre machine que votre serveur supportant git. Installer SparkleShare sur le serveur est possible mais ne présente pas grand intérêt car il va synchroniser un dossier d'un emplacement de son disque vers un autre emplacement.

Avant d'installer SparkleShare, il serait bon de récupérer le logiciel à installer. Cette fois, nous allons compiler à la main.

Pour récupérer SparkleShare, nous aurons besoin de git, c'est déjà l'outil que nous avons utilisé pour installer le serveur. Donc, même combat, nous allons utiliser `apt-get` :

```
apt-get install git
```

Plaçons-nous ensuite dans le répertoire */usr/src* pour récupérer SparkleShare. Comme nous l'avons vu auparavant, nous allons utiliser git pour télécharger la dernière version de l'application :

```
cd /usr/src
git clone https://github.com/ ↵
    hbons/SparkleShare.git
```

Normalement un répertoire *SparkleShare* a dû être créé dans */usr/src*. Jusque-là, il serait étonnant que vous ayez une erreur sauf problème de téléchargement. Avant de passer à la compilation, nous allons installer les outils de compilation et autres dépendances obligatoires pour l'installation de SparkleShare. Cette fois encore, nous utiliserons les dépôts Debian et `apt-get` :

```
apt-get install git build- ↵
    essential intltool ↵
  mono-devel gtk-sharp2 nant ↵
  python-nautilus libtool ↵
  libwebkit-cil-dev ↵
  libnotify-cil-dev
```

Cela fait peu de paquets mais vous constaterez vite que beaucoup de dépendances s'ajouteront. La compilation devrait bien se passer, vous ne devriez pas avoir d'erreur.

Passons dans le vif du sujet : la configuration de la compilation, c'est à cette étape que tous les prérequis de la compilation seront testés. Pour la suite, nous allons nous logger sous le profil de l'utilisateur root pour éviter les problèmes de droit :

```
su root
cd SparkleShare/
./autogen.sh --prefix=/usr ↵
  --sysconfdir=/etc
```

Si la dernière commande ne vous retourne pas d'erreur, c'est que la compilation se présente bien. Il reste juste à compiler puis installer le tout dans les bons répertoires :

```
make
```

Avouez que c'était plutôt simple. Avant de passer à l'étape suivante, vérifiez bien que la dernière ligne de la commande `make` est la suivante :

```
Compilation succeeded - x ↵
  warning(s)
```

Si tout est bon, nous pouvons passer à l'installation :

```
make install
```

La seule erreur que vous risquez d'avoir porte sur un problème de droit ; vérifiez juste que vous êtes effectivement connecté sous le profil de l'utilisateur root (99% des problèmes). Voilà, votre client SparkleShare est désormais installé, bien entendu il y a encore du travail !

Cryptage

La sécurité est un élément essentiel lors du transfert de données sur un réseau. Évidemment, faire transiter en clair des fichiers n'est peut-être pas la chose à faire. SparkleShare utilise SSH pour synchroniser vos PC avec votre dépôt.

Dans un premier temps, il faut installer un client SSH sur votre PC :

```
apt-get install openssh- ↵
  client
```

Un client SSH sert à se connecter à un serveur SSH. C'est une phrase qui semble évidente mais qui pour-

Figure 1. *Sur le site de SparkleShare, les versions Windows, Android et iOS sont déjà prévues*

Figure 2. *Il faut pas mal de paquetage pour compiler SparkleShare*

tant, nous oblige à nous reconnecter à notre serveur (là où nous avons installé le dépôt « mabox »). Lancez cette commande pour installer ssh (si ce paquetage est déjà installé, la commande ne s'exécutera pas) :

```
apt-get install ssh
```

L'infrastructure SSH est en place, retournez sur votre PC pour générer une clé qui vous permettra de nous connecter de manière sécurisée à votre serveur et cela, sans demander de mot de passe.

Connectez-vous sous votre profil d'utilisateur. Ensuite, tout se passe encore en ligne de commande :

```
ssh-keygen
```

Vous êtes invité à répondre à quelques questions, tout est à valider : il ne faut absolument rien changer sinon la configuration que nous allons appliquer à SparkleShare ne fonctionnera pas.

Maintenant que nous avons généré une clé, il faut la copier sur notre serveur pour de futures connexions sans mot de passe :

```
ssh-copy-id mabox@ip_serveur
```

Vérifions que tout s'est bien passé, connectons-nous en SSH sur notre serveur, si aucun login ou mot de passe n'est demandé, c'est gagné :

```
ssh mabox@ip_serveur
```

Pour la « connectique » c'est terminé, voyons maintenant comment démarrer SparkleShare.

Configuration de SparkleShare

Ne soyez pas sous le profil de l'utilisateur root pour démarrer SparkleShare ; de toute manière, vous seriez rejeté par l'application. Lancer l'interface de configuration s'effectue par cette commande :

```
sparkleshare start
```

Une interface s'ouvre et vous demande votre nom et votre mail.

Ensuite, quelques écrans vous présentent SparkleShare, vous pouvez sauter cette présentation ou faire défiler les écrans (il y en a très peu). Arrive enfin le dernier écran, cliquez sur *Terminer*. SparkleShare est installé ! Mais nous ne sommes pas encore connectés à notre serveur.

En haut à droite, prés de l'heure, vous devriez voir une icône en forme de dossier avec une étoile à l'intérieur : il s'agit de SparkleShare. Faites un clic droit dessus et sélectionnez *Add hosted Project*.

C'est à travers cette interface que nous renseignons notre serveur. D'autres choix sont possibles, mais nous avons pris le parti de maitriser totalement les données à transférer. Nous vous invitons donc à compléter cet écran de la manière suivante :

- Where's your project hosted : Selectionner « On my own server »
- Address : *mabox@ip_serveur*
- Remote Path : */home/mabox/mondepot.git*

Enfin, cliquez sur *Ajouter* en bas à droite : votre dépôt est désormais ajouté. Le lien entre votre PC et votre serveur se fera automatiquement et chaque fois que vous placerez un élément dans le répertoire *SparkleShare/mondepot/*, il sera aussitôt synchronisé. L'inverse est également vrai, si un autre PC verse un document, il est envoyé sur le serveur git puis votre PC le récupère.

Pour chacun de vos dépôts, il faudra posséder de la sorte. Pourquoi chacun de vos dépôts ? Tout simplement parce que vous pourriez vous connecter à plusieurs serveurs : un personnel mais pourquoi pas, un autre professionnel que vous partageriez avec vos collègues, un autre pour mettre des documents communs entre vous et d'autres personnes. Bref, les possibilités sont infinies, à vous de faire le tri et d'évaluer ce que SparkleShare peut vous offrir.

Pour que SparkleShare fonctionne, il faut le démarrer :

```
cp /usr/share/applications/ ↵
   sparkleshare.desktop     ↵
   ~/.config/autostart
```

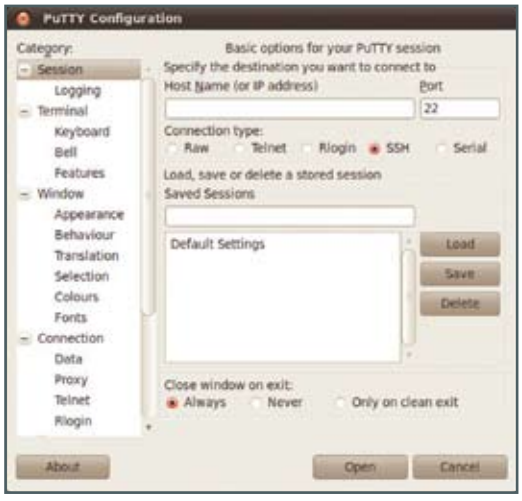

Figure 3. *Vous pouvez utiliser Putty comme client SSH*

Figure 4. *Premier écran de configuration de SparkleShare*

Vous pouvez redémarrer votre PC pour vous assurer que SparkleShare se lance effectivement sous votre session. C'est terminé, tout est interconnecté, synchronisé, dupliqué, bref ça fonctionne, en quelques lignes vous êtes devenu Dropbox à vous tout seul ;)

Et maintenant...

Imaginez maintenant que vous êtes à l'autre bout du monde, au Nakatomi Plaza par exemple à Los Angeles. Vous avez un besoin urgent de récupérer un document qui se trouve sur votre SparkleShare. Le problème est que vous n'avez qu'une connexion Internet sur un PC qui n'est pas le vôtre. Pas de problème, nous allons installer une interface web pour avoir accès au fichier qui se trouve sur le serveur.

Pour cela, nous utiliserons Gitweb, une application qui répond parfaitement à nos attentes :

```
apt-get install git-web
```

Cette commande permet également d'installer un serveur web (apache2) automatiquement. Il ne reste plus qu'à configurer le tout pour activer l'interface web.

Première étape, création des répertoires nécessaires à l'interface et copie de fichiers pour que Apache puisse interpréter correctement les pages de gitweb. Pour cela, en tant que root, lancez ces commandes :

```
su root
mkdir /home/mabox/gitweb
cp /usr/share/gitweb/* /home/ ↵
   mabox/gitweb
cp /usr/lib/cgi-bin/gitweb. ↵
   cgi /home/mabox/gitweb
```

Ensuite, il faut configurer le serveur web apache2 pour qu'il pointe bien vers notre gitweb. en créant un fichier */etc/apache2/conf.d/mabox* et en le complétant ainsi :

```
Alias /gitweb /home/mabox/ ↵
   gitweb
<Directory /home/mabox/ ↵
   gitweb>
   Allow from all
   AllowOverride all
   Order allow,deny
```

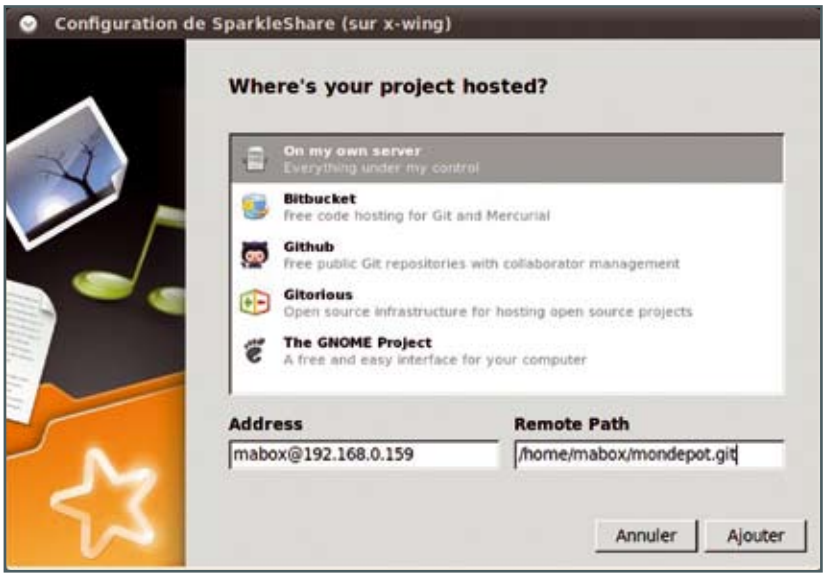

Figure 5. *Ajoutez votre dépôt à SparkleShare*

```
    Options FollowSymLinks ↵
   +ExecCGI
   AddHandler cgi-script .cgi
 <Files gitweb.cgi>
    SetHandler cgi-script
  </Files>
</Directory>
DirectoryIndex gitweb.cgi
SetEnv  GITWEB_CONFIG /etc/ ↵
   gitweb.conf
```

Relancez le serveur pour prendre en compte cette configuration apache. Mais au préalable, nous allons effacer la configuration par défaut de Gitweb pour éviter les interférences :

```
rm /etc/apache2/conf.d/gitweb
```

Il ne reste plus qu'à relancer Apache 2 pour prendre en compte toutes ses modifications :

```
/etc/init.d/apache2 restart
```

Pour la partie apache, tout est configuré, il ne reste plus qu'à paramétrer directement Gitweb. Le fichier de configuration de cette application est */etc/gitweb.conf*. Éditez ce fichier et remplacez le contenu par ceci :

```
$projectroot = "/home/ ↵
   mabox/";
$git_temp = "/tmp";
#$home_link = $my_uri || "/";
$home_text = "indextext. ↵
   html";
$projects_list = ↵
   $projectroot;
$stylesheet = "gitweb.css";
$javascript = "gitweb.js";
$logo = "git-logo.png";
$favicon = "git-favicon.png";
```

Avec ce dernier paramétrage, Gitweb devrait fonctionner, faisons un premier essai avant d'aller plus loin. Comme il s'agit d'une interface web, il faut nécessairement un navigateur web, quel qu'il soit et sur n'importe quelle plate-forme. Ouvrez votre navigateur et entrez cette URL *http://ip_serveur/gitweb*

L'interface n'est pas des plus jolies mais qu'importe, elle fonctionne très bien et c'est cela le principal. Voici quelques indications pour vous y retrouver facilement : cliquez d'abord sur le nom de votre dépôt (ici *mondepot.git*). Vous arrivez face à un autre écran dans lequel vous pourrez voir vos fichiers.

Figure 6. *Cette icône devrait être présente près de l'heure : SparkleShare fonctionne*

Figure 7. *Apache, le serveur web ultra puissant ;)*

Figure 8. *L'interface quelque peu austère de Gitweb*

Figure 9. *Vous allez faire un carton ;)*

En haut de l'écran, six choix sont proposés (summary, shortlog, log, commit, commitdiff, tree), le plus simple est de cliquer sur *tree* car vous verrez directement l'arborescence de votre répertoire SparkleShare. Désormais, depuis l'autre bout du monde, vous pourrez récupérer vos documents.

Bonne nouvelle, vu la sécurité que nous avons mise, le monde entier peut télécharger vos fichiers. Donc, nous allons sécuriser un minimum cette interface web grâce aux htaccess. Il s'agit de fichiers interprétés par Apache et qui permettent de faire des opérations simples ou complexes (réécriture d'URL, redirection, accès sécurisé à un répertoire...). Vous l'aurez compris, c'est exactement ce qu'il nous faut.

Dans un premier temps, créons un fichier qui contiendra la liste des utilisateurs ayant le droit de se connecter à l'interface web :

```
htpasswd -c /home/mabox/ ↵
   htpasswd mon_utilisateur
```

Vous devez indiquer un mot de passe, choisissez quelque chose de solide surtout si votre serveur est sur le net. Tout est en place ou presque, il suffit de créer ce fameux htaccess. Créez le fichier */home/mabox/gitweb/.htaccess* et complétez le ainsi :

```
AuthName "Acces restreint"
AuthType Basic
AuthUserFile "/home/mabox/ ↵
   gitweb/.htaccess"
Require valid-user
```

Il est inutile de redémarrer Apache, cette modification est automatiquement prise en compte. Il suffit de tester que cette authentification fonctionne en se connectant à l'interface web *http://ip_serveur/gitweb* Si le htaccess est bien pris en compte, vous devriez voir une bannière de login. Si tel n'est pas le cas, vérifiez votre fichier */etc/apache2/conf.d/mabox* et notamment, la commande *AllowOverride all*.

Cette fois, nous sommes au bout de cette installation. ■