Synchronisation avec rsync

Author : pierrehirel

rsync fait partie des outils indispensables pour synchroniser deux répertoires, i.e. s'assurer que leurs contenus sont identiques, et donc effectuer des sauvegardes ou synchronisations régulières. Cet article présente brièvement quelques scénarios d'utilisation.

1. Présentation de rsync

Sur le principe, rsync a un fonctionnement simple : c'est un outil qui va copier des fichiers d'un endroit (la source) vers un autre (la destination). Très logiquement la syntaxe est :

client:~$ rsync <source> <destination>

La différence avec une simple copie (telle que cp ou scp) c'est que rsync va d'abord comparer les listes de fichiers entre source et destination, et ne copiera que les fichiers qui sont nouveaux et ceux qui sont plus récents.

Il est important dès à présent de comprendre une chose : rsync ne va pas « magiquement » synchroniser deux répertoires. La commande ci-dessus est unidirectionnelle : elle ne fait que copier certains fichiers de la source vers la destination. Si la destination contient des fichiers qui n'existent pas dans la source, ils ne seront pas copiés vers la source. Pour cela, il faudra exécuter rsync une seconde fois en inversant source et destination, comme nous le verrons ci-dessous.

2. Sauvegardes sur un disque dur externe

Passons à la pratique : imaginons que nous ayons un ordinateur (nommé « client ») sur lequel sont enregistrés des documents, dans un dossier « Documents ». Nous souhaitons sauvegarder régulièrement ces documents sur un disque externe, monté sur /media/disque/. Il est possible d'utiliser rsync :

client:~$ rsync -a /home/user/Documents/ /media/disque/Documents/

L'option **-a** (pour « archive ») permet de s'assurer que les liens symboliques, dates de modification, propriétaires et permissions sur les fichiers, sont conservés lors du transfert. Cette option implique aussi que les dossiers seront copiés récursivement, i.e. tous les sous-dossiers et leur contenu seront aussi copiés.

La position des symboles slash (/) a son importance losqu'on utilise rsync. Ainsi la commande ci-dessus indique que le contenu du dossier source /home/utilisateur/Documents/ sera copié dans le dossier destination /media/disque/Documents/, autrement dit les deux dossiers contiendront la même chose après copie. En revanche si le dernier slash est omis de la source :

client:~$ rsync -a /home/user/Documents /media/disque/Documents/

alors rsync créera un sous-dossier /media/disque/Documents/Documents/ et y copiera le contenu de la source, ce qui n'est pas forcément l'effet souhaité. Lorsque l'on souhaite copier des dossiers il faut donc toujours terminer le

chemin avec un slash.

3. Synchronisation avec un disque dur externe

Comme expliqué auparavant, la commande ci-dessus est unidirectionnelle et ne copiera des fichiers que depuis l'ordinateur vers le disque externe. Ceci est suffisant si le disque externe n'est utilisé par personne d'autre et n'est utilisé que pour faire des sauvegardes.

Imaginons maintenant que le disque externe soit utilisé par d'autres personnes, et que les documents qui s'y trouvent puissent être modifiés. L'utilisateur voudra alors s'assurer que les documents qu'il a sur son ordinateur sont les mêmes que ceux qui sont sur le disque. Il devra alors exécuter rsync deux fois, en inversant source et destination, pour synchroniser complètement les deux médias :

```
client:~$ rsync -a /home/user/Documents/ /media/disque/Documents/
client:~$ rsync -a /media/disque/Documents/ /home/user/Documents/
```

Suite à cela, l'ordinateur et le disque externe contiendront les versions les plus récentes de chaque fichier : ils seront bien *synchronisés*.

4. Synchronisation avec un serveur distant

Bien évidemment, rsync peut aussi être utilisé pour synchroniser des fichiers entre un ordinateur client et un serveur distant. Pour cela il utilise par défaut le protocole ssh.

Supposons que le serveur distant dispose bien d'un serveur ssh, et pour corser la chose imaginons qu'il soit accessible par le port 2222 (au lieu de 22, le port ssh par défaut). La commande à appeler ressemblera à ceci :

```
client:~$ rsync -e "ssh -p 2222" -az /home/user/Documents/ user@server.org:/home/user/Documents/
```

L'option **-e** (à utiliser avec des guillemets) indique quel protocole utiliser pour le transfert, ici on indique qu'il faut utiliser ssh avec le port 2222.

L'option **-z** indique à rsync de compresser les données pendant le transfert, ce qui réduit la bande passante utilisée. Ceci n'était pas indispensable lors d'une copie locale (vers un disque externe), mais peut faire gagner du temps lorsque le transfert se fait via Internet. Notez que cette option *ne signifie pas* que les données seront enregistrées sous forme d'archives compressées (en .zip ou .tar.gz) sur la destination : les données ne sont compressées que pendant le transfert, puis décompressées avant d'être enregistrées sur la destination.

Côté serveur, après l'adresse (server.org) il faut ajouter deux-points (:) puis le répertoire de destination (ici /home/user/Documents/). Attention à ne pas oublier les symboles slash pour bien copier les bons répertoires.

Là encore si l'on veut synchroniser les données dans les deux sens il faut appeler rsync une seconde fois, en inversant source et destination, soit :

```
client:~$ rsync -e "ssh -p 2222" -az user@server.org:/home/user/Documents/ /home/user/Documents/
```

5. Se débarrasser des fichiers qui ont été supprimés de la source

Par défaut rsync ne supprime aucun fichier sur la destination, il ne fait que copier les fichiers manquants ou plus récents. Dans certains cas, si l'on supprime des fichiers de la source, on aimerait aussi qu'ils soient supprimés sur la destination. Ceci se fait en utilisant rsync avec l'option **delete-after**, qui indique d'effacer les fichiers superflus après le transfert :

```
client:~$ rsync -a --delete-after <source> <destination>
```

Il existe aussi une option **delete-before** qui indique de supprimer les fichiers avant le transfert. Attention avec ces options car aucune confirmation ne sera demandée avant de supprimer les fichiers.

6. Synchronisations automatiques

Pour automatiser les sauvegardes et synchronisations il est possible de sauvegarder les commandes appropriées à votre usage dans un script bash, et/ou d'utiliser un job cron pour les exécuter à intervalles réguliers.