Atelier Linux

Sauvegarde

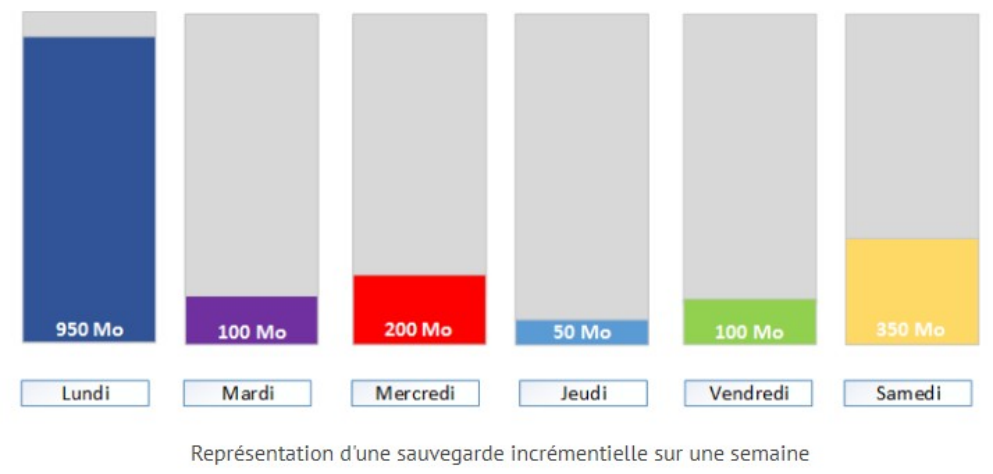
Rsync – Cron

I – Rsync

Rsync est un outil qui va permettre de synchroniser plusieurs dossiers ensemble et qui de plus l'avantage de pouvoir être utilisé à travers SSH.
On va donc pouvoir par exemple l'utiliser pour déployer une application, depuis un serveur local vers un serveur distant.
Autre exemple, on va également pouvoir s'en servir pour réaliser des sauvegardes **complètes, incrémentales et différentielles.**

Petit rappel sur ces deux formes de sauvegarde (source IT connect)

Sauvegarde incrémentale :



Représentation d'une sauvegarde incrémentielle sur une semaine

Sur cet exemple je suis partis d'une sauvegarde sur une semaine.

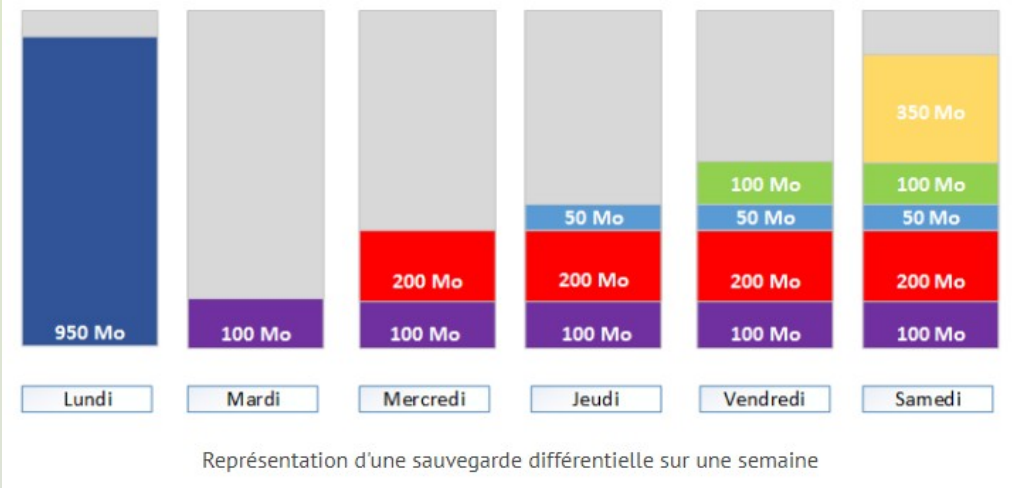
Pour commencer, il va falloir réaliser en premier lieu une sauvegarde complète ici le lundi.

Ensuite, l'incrémentielle du mardi va se baser sur la précédente qui sera cette fois la complète en sauvegardant uniquement les nouveaux fichiers créés ou modifiés entre-temps.

Mercredi, l'incrémentielle va se baser sur la précédente qui sera cette fois le mardi en sauvegardant uniquement les nouveaux fichiers créés ou modifiés entre-temps.
Et ainsi de suite jusqu'à la prochaine sauvegarde complète.

Si on souhaite récupérer l'ensemble de la sauvegarde de la semaine, il faudra restaurer tous les éléments du lundi au samedi ce qui prendra un certain temps, mais cela reste la méthode qui consomme le moins d'espace avec un total de 1750 Mo.

Sauvegarde différentielle :



Représentation d'une sauvegarde différentielle sur une semaine

Le lundi, on démarre avec une sauvegarde complète. Et c'est à partir de cette sauvegarde que les différentielles se baseront chaque jour.

Mardi la différentielle va sauvegarder uniquement les nouveaux fichiers créés ou modifiés entre-temps d'après la complète. Soit n.

Mercredi la différentielle va sauvegarder uniquement les nouveaux fichiers créés ou modifiés entre-temps d'après la complète. Soit n+1.
Et ainsi de suite jusqu'à la prochaine sauvegarde complète.

On constate en effet que la sauvegarde du jeudi contient en plus les données du mardi et du mercredi. Étant donné que les différentielles se basent sur la complète et non pas les précédentes, elles ajoutent tous les fichiers ajoutés ou modifiés depuis le lundi.

Si on souhaite récupérer l'ensemble de la sauvegarde de la semaine, il faudra restaurer simplement le lundi et le samedi qui correspondent à l'ensemble de la semaine. La restauration prendra donc moins de temps, mais c'est la méthode qui consomme le plus d'espace avec un total de 2950 Mo.

Rsync va donc être capable de reconnaître les différences et de ne télécharger que les nouveaux fichiers ou ceux qui ont été modifiés.

S'il on se réfère au manuel disponible sur cette page :

https://linux.die.net/man/1/rsync

On remarque de suite qu'il s'agit d'une application très puissante mais dont la maîtrise de toutes ses possibilités demande beaucoup de temps d'apprentissage.

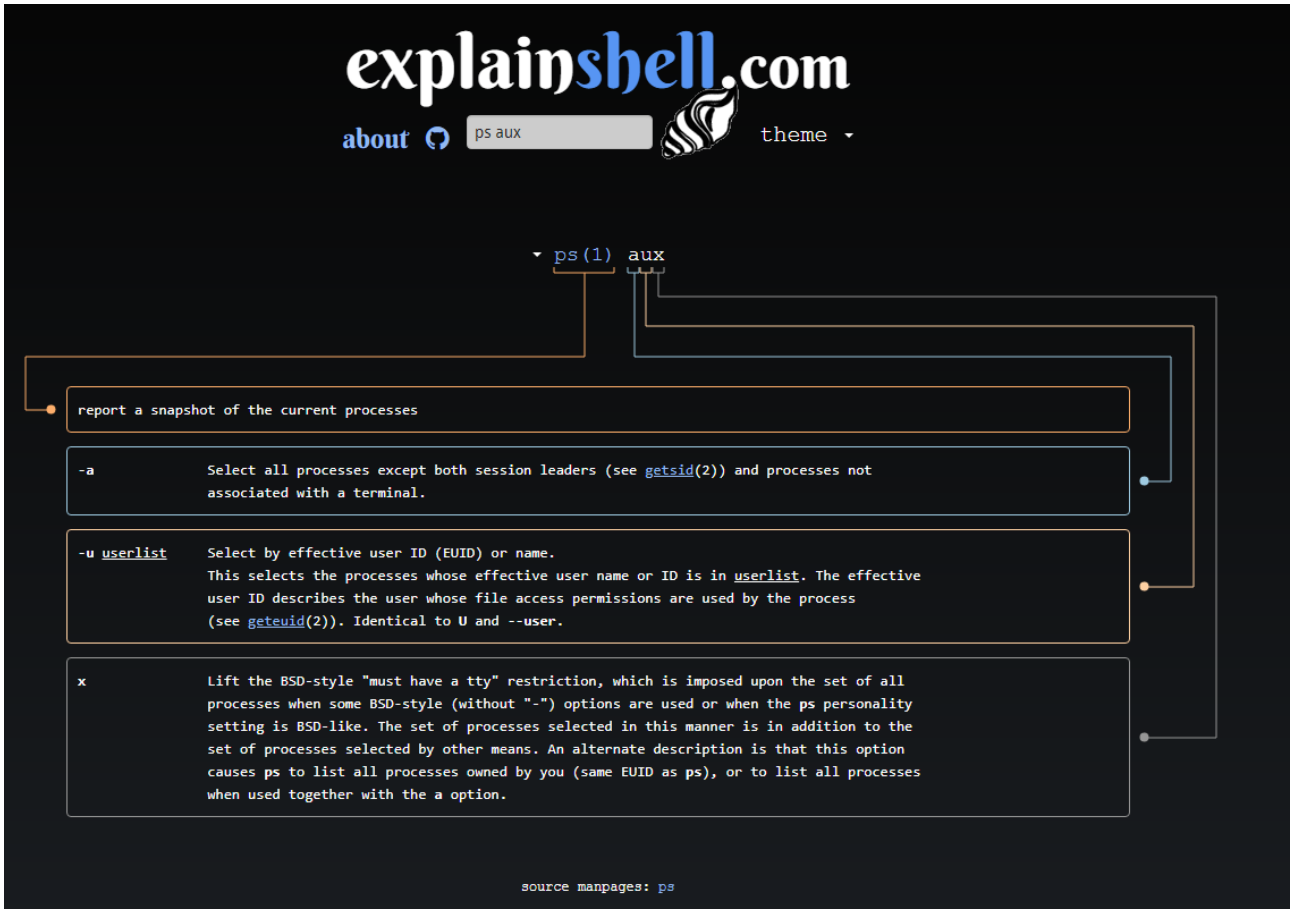
Toutefois, la plupart des usages relatifs à la synchronisation et aux tâches de sauvegardes se contentent d'options qui se comptent en nombre plus restreint.

Comme souvent sous Linux, il ne faudra pas hésiter à se référer au manuel (commande MAN ou –help) pour en apprendre davantage sur les options disponibles.

Aussi, on peut également utiliser ce très bon site pour connaître la finalité d'une commande et des options associées :

https://explainshell.com/

Exemple pour la commande « **ps aux** » :



Pour ce petit atelier, vous aurez besoin de deux machines virtuelles sur le même réseau.

Vous prendrez deux Debian par exemple.

Dans mon cas, je vais utiliser **SRV-DEB1** et **SRV-DEB2**.

SSH est bien entendu activé et les deux machines parviennent à communiquer entre elles.

1 – sur SRV-DEB1, créer un dossier « source » et des fichiers test1 ⟶ test6.

```
[root@srv-deb1 ~]$ls -lR
.:
total 4
drwxr-xr-x 2 root root 4096 mars   9 20:44 source

./source:
total 0
-rw-r--r-- 1 root root 0 mars   9 20:44 test1
-rw-r--r-- 1 root root 0 mars   9 20:44 test2
-rw-r--r-- 1 root root 0 mars   9 20:44 test3
-rw-r--r-- 1 root root 0 mars   9 20:44 test4
-rw-r--r-- 1 root root 0 mars   9 20:44 test5
-rw-r--r-- 1 root root 0 mars   9 20:44 test6
[root@srv-deb1 ~]$
```

Ensuite, si rsync n'est pas déjà installé sur votre machine, installez-le :

```
[root@srv-deb1 BACKUP]$apt install rsync
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les NOUVEAUX paquets suivants seront installés :
  rsync
0 mis à jour, 1 nouvellement installés, 0 à enlever et 0 non mis à jour.
Il est nécessaire de prendre 397 ko dans les archives.
Après cette opération, 746 ko d'espace disque supplémentaires seront utilisés.
Réception de :1 http://deb.debian.org/debian buster/main amd64 rsync amd64 3.1.3-6 [397 kB]
397 ko réceptionnés en 2s (166 ko/s)
Sélection du paquet rsync précédemment désélectionné.
(Lecture de la base de données... 32185 fichiers et répertoires déjà installés.)
Progression : [ 20%] [#####################.....................................................................................]
Progression : [ 40%] [##########################################.....................................................................]
Progression : [ 60%] [###############################################################.....................................]
Progression : [ 80%] [###########################################################################################.....]
Traitement des actions différées (« triggers ») pour man-db (2.8.5-2) ...
Traitement des actions différées (« triggers ») pour systemd (241-7~deb10u6) ...
[root@srv-deb1 BACKUP]$
```

Dans la plupart des cas, la commande rsync est d'abord suivie de l'option **-a**.

L'option -a est en fait un alias regroupant plusieurs options :

```
-a, --archive               archive mode; equals -rlptgoD (no -H,-A,-X)
```

La seconde option peut être **-v** qui signifie « **verbose** ».
Cette option va donc nous permettre d'avoir les informations sur ce que la commande rsync va faire.

Une fois que les options sont choisies, il convient d'indiquer la source puis la destination.

Ainsi, si nous voulons copier le contenu du dossier « source » dans le dossier « backup », voici ce que serait notre première commande :

```
[root@srv-deb1 ~]$rsync -av source/ ./backup
```

```
[root@srv-deb1 ~]$rsync -av source/ ./backup
sending incremental file list
created directory ./backup
./
test1
test2
test3
test4
test5
test6

sent 366 bytes  received 164 bytes  1,060.00 bytes/sec
total size is 0  speedup is 0.00
[root@srv-deb1 ~]$
```

Résultat :

```
[root@srv-deb1 ~]$ls -lR backup/
backup/:
total 0
-rw-r--r-- 1 root root 0 mars   9 20:44 test1
-rw-r--r-- 1 root root 0 mars   9 20:44 test2
-rw-r--r-- 1 root root 0 mars   9 20:44 test3
-rw-r--r-- 1 root root 0 mars   9 20:44 test4
-rw-r--r-- 1 root root 0 mars   9 20:44 test5
-rw-r--r-- 1 root root 0 mars   9 20:44 test6
[root@srv-deb1 ~]$
```

Les fichiers ont bien été copiés.
Dans l'alias -a figure l'option -r pour la récursivité.

Ainsi, le dossier « backup » a automatiquement été crée et si nous avions eu des sous-répertoires, ceux-ci auraient également été copiés.

Il s'agit donc là d'une opération de « **sauvegarde complète** ».

Voyons ce qu'il se passe si l'on relance cette commande :

```
[root@srv-deb1 ~]$rsync -av ./source/ ./backup
```

```
[root@srv-deb1 ~]$rsync -av ./source/ ./backup
sending incremental file list
./
test1
test2
test3
test4
test5
test6
REP/

sent 422 bytes  received 141 bytes  1,126.00 bytes/sec
total size is 0  speedup is 0.00
[root@srv-deb1 ~]$
```

Comme attendu, il y a beaucoup moins d'informations (422 octets envoyés, 141 reçus), tout simplement parce que rsync a comparé les deux dossiers et ne voyant **aucune différence**, il n'a rien fait.

Supprimons donc un fichier dans le dossier « backup », par exemple « test5 » et relançons la commande :

```
[root@srv-deb1 ~]$rm backup/test5
[root@srv-deb1 ~]$rsync -av ./source/ ./backup
```

```
[root@srv-deb1 ~]$rsync -av ./source/ ./backup
sending incremental file list
./
test5

sent 224 bytes  received 39 bytes  526.00 bytes/sec
total size is 0  speedup is 0.00
[root@srv-deb1 ~]$
```

Cette fois, rsync – en comparant les deux dossiers – a pu voir que le fichier « test5 » a disparu et a donc de nouveau copié le fichier « test5 » depuis le dossier « source ».
Les deux dossiers sont donc synchronisés.

On remarque également qu'il ne copie que le fichier « test5 » et non les autres.
En ce sens, on parlerait donc de copie « **incrémentale** ».

Allons un peu plus loin et **modifions** le fichier « test1 » du dossier « source », ce qui nous permettra de voir si la modification d'un fichier est aussi pris en compte par rsync dans le processus de synchronisation.

Éditez le fichier « test5 » :

```
[root@srv-deb1 ~]$echo "IT for coffee" > ./source/test1
```

Puis relancez – toujours – la même commande :

```
[root@srv-deb1 ~]$rsync -av ./source/ ./backup
sending incremental file list
test1

sent 243 bytes  received 36 bytes  558.00 bytes/sec
total size is 14  speedup is 0.05
[root@srv-deb1 ~]$
```

De nouveau, rsync a seulement opéré sur le fichier « test1 » après un travail de comparaison entre les deux fichiers (dans « source » et « backup »).

rsync travaille donc bien sur la date de modification pour prendre une décision de copie vers la destination.

Vous pouvez également faire le même test, cette fois en modifiant les **droits** du fichier « test1 » et constater ce qu'il se passe à la sortie de la commande rsync.

2 – détails sur l'alias -a :

-a, --archive archive mode; equals -rlptgoD (no -H,-A,-X)

-r = recursive (comme nous l'avons déjà vu)
-l = permet de copier le lien symbolique s'il y en avait dans un dossier ou pour un fichiers
-p = permet de copier les permissions (voir votre dernier test)
-t = permet le transfert de date de modification. Ceci est très important car c'est grâce à cette option qu'il est possible de synchroniser un fichier qui a été modifié.
-g = permet de copier les groupes (propriétaires d'un fichier / dossier)
-o = permet de copier les owners (utilisateurs propriétaires d'un fichier / dossier)

Petit test (modifier l'appartenance du fichier test1 au groupe et à l'utilisateur « tipiak ») puis nouvelle synchronisation avec rsync :

```
[root@srv-deb1 ~]$chown tipiak:tipiak source/test1
[root@srv-deb1 ~]$rsync -av ./source/ ./backup
sending incremental file list

sent 209 bytes  received 20 bytes  458.00 bytes/sec
total size is 14  speedup is 0.06
[root@srv-deb1 ~]$ls -l source/
total 8
drwxr-xr-x 2 root   root   4096 mars   9 21:04 REP
-rw-r--r-- 1 tipiak tipiak   14 mars   9 21:20 test1
-rw-r--r-- 1 root   root      0 mars   9 20:44 test2
-rw-r--r-- 1 root   root      0 mars   9 20:44 test3
-rw-r--r-- 1 root   root      0 mars   9 20:44 test4
-rw-r--r-- 1 root   root      0 mars   9 21:14 test5
-rw-r--r-- 1 root   root      0 mars   9 20:44 test6
[root@srv-deb1 ~]$ls -l backup/
total 12
drwxr-xr-x 2 root   root   4096 mars   9 21:04 REP
drwxr-xr-x 3 root   root   4096 mars   9 21:04 source
-rw-r--r-- 1 tipiak tipiak   14 mars   9 21:20 test1
-rw-r--r-- 1 root   root      0 mars   9 20:44 test2
-rw-r--r-- 1 root   root      0 mars   9 20:44 test3
-rw-r--r-- 1 root   root      0 mars   9 20:44 test4
-rw-r--r-- 1 root   root      0 mars   9 21:14 test5
-rw-r--r-- 1 root   root      0 mars   9 20:44 test6
[root@srv-deb1 ~]$
```

Les permissions ont bien été copiées.

3 – revue d'autres options :

a) option -h :

Cette option permet d'afficher les informations affichées par rsync dans un format dit « humainement compréhensible » (rappelez-vous la commande « **df -h** » par exemple).

```
[root@srv-deb1 ~]$rsync -avh ./source/ ./backup
sending incremental file list

sent 202 bytes  received 13 bytes  430.00 bytes/sec
total size is 14  speedup is 0.07
[root@srv-deb1 ~]$
```

b) option -p :

Cette option permet d'afficher la progression du processus de synchronisation entre les deux dossiers.
Très utile lorsque la copie concerne de gros volumes de données, cela permet d'en voir la progression.

Cette option -p est également un alias qui regroupe :

- - progress
- - partial

L'option « partial » est très utile car elle permet de **reprendre** le processus de synchronisation là où il a été interrompu (problème de réseau par exemple).

c) option -z :

Avant d'envoyer les données, rsync va les compresser afin notamment de moins solliciter la bande passante disponible sur le réseau.
Bien entendu, l'inconvénient est que cela nécessite des ressources CPU côté machine émettrice (compression) et côté machine réceptrice (décompression).
Le choix de cette option doit donc toujours être mesuré par rapport à ces deux contraintes.

d) option - - dry-run :

Cette option peut être utile dans certaines situations car cela permet de demander à rsync de nous dire ce qu'il va faire, mais sans réaliser l'opération en elle-même.

Supprimons test3 ⟶ test6 sur le dossier « backup » et exécutons la commande avec cette option.

```
[root@srv-deb1 ~]$rm backup/test[3-6]
```

```
[root@srv-deb1 ~]$rm backup/test[3-6]
[root@srv-deb1 ~]$cd backup/
[root@srv-deb1 backup]$ls
test1  test2
```

Exécutons ensuite la commande rsync avec l'option - - **dry-run**.

[root@srv-deb1 ~]$rsync -avhp --dry-run ./source/ ./backup/

```
[root@srv-deb1 ~]$rsync -avhp --dry-run ./source/ ./backup/
sending incremental file list
./
test3
test4
test5
test6
REP/

sent 224 bytes  received 35 bytes  518.00 bytes/sec
total size is 14  speedup is 0.05 (DRY RUN)
[root@srv-deb1 ~]$
[root@srv-deb1 ~]$
[root@srv-deb1 ~]$ls -l backup/
total 4
-rw-r--r-- 1 tipiak tipiak 14 mars   9 21:20 test1
-rw-r--r-- 1 root   root    0 mars   9 20:44 test2
[root@srv-deb1 ~]$
```

En sortie de commande, les fichiers manquants n'ont pas été copiés, comme attendu avec l'option - - dry-run.

L'alias pour - - dry-run est **-n**

e) option - - delete :

Jusqu'à présent, nous n'avons opéré de modification que sur le dossier de destination « backup ». Que se passe-t-il s'il on supprime un fichier dans le dossier « source » - par exemple test2 – et que l'on resynchronise les deux dossiers ?

[root@srv-deb1 ~]$rm ./source/test2
[root@srv-deb1 ~]$rsync -av ./source/ ./backup/

```
[root@srv-deb1 ~]$rm ./source/test2
[root@srv-deb1 ~]$rsync -av ./source/ ./backup/
sending incremental file list
./

sent 196 bytes  received 20 bytes  432.00 bytes/sec
total size is 14  speedup is 0.06
[root@srv-deb1 ~]$ls -l ./backup/
total 8
drwxr-xr-x 2 root   root   4096 mars   9 21:04 REP
-rw-r--r-- 1 tipiak tipiak   14 mars   9 21:20 test1
-rw-r--r-- 1 root   root      0 mars   9 20:44 test2
-rw-r--r-- 1 root   root      0 mars   9 20:44 test3
-rw-r--r-- 1 root   root      0 mars   9 20:44 test4
-rw-r--r-- 1 root   root      0 mars   9 21:14 test5
-rw-r--r-- 1 root   root      0 mars   9 20:44 test6
[root@srv-deb1 ~]$
```

Le fichier « test2 » est toujours présent dans le dossier « backup ».

Il ne s'agit donc pour le moment pas d'une véritable synchronisation dans le sens où seules les modifications dites « positives » sur le dossier « source » seront prises en compte par rsync.
Du moins en l'absence de l'option adéquate, en l'occurrence - - **delete**.

```
[root@srv-deb1 ~]$rsync -av --delete ./source/ ./backup/
sending incremental file list
deleting test2

sent 189 bytes  received 22 bytes  422.00 bytes/sec
total size is 14  speedup is 0.07
[root@srv-deb1 ~]$
```

En ajoutant cette option, rsync a bien synchronisé le dossier « backup » en supprimant test2.

Toutefois, cette commande étant destructive sur le dossier de destination, afin de s'assurer que l'on fait bien les choses, on pourra tout d'abord faire cette même commande mais avec l'option dry-run vue précédemment.

f) option - - exclude :

Dans les cas où l'on souhaite exclure un dossier ou un fichier du processus de synchronisation, on pourra alors utiliser l'option **- - exclude**.

C'est particulièrement utile lorsque l'on utilise l'option - - delete car celle-ci supprime du dossier de destination « backup » **tous** les fichiers/dossiers absents du dossier « source ».

Par exemple, créons un fichier « toto » dans le dossier « backup ».

```
[root@srv-deb1 ~]$touch backup/toto
```

Si l'on relance la commande précédente, alors le fichier « toto » va disparaître car absent du dossier « source ».

Dans ce cas, on ajoutera l'option **- - delete** suivie de - - exclude=[nom du fichier/dossier]

```
[root@srv-deb1 ~]$rsync -av --delete --exclude=toto ./source/ ./backup/
```

```
[root@srv-deb1 ~]$rsync -av --delete --exclude=toto ./source/ ./backup/
sending incremental file list
./

sent 212 bytes  received 20 bytes  464.00 bytes/sec
total size is 14  speedup is 0.06
[root@srv-deb1 ~]$
```

Cela fonctionne également dans les deux sens (exclusion d'un fichier/dossier dans le dossier « source »).

Créons deux nouveaux fichiers dans « source » ⟶ test7 et test8.

Nous continuerons d'exclure le fichier « toto » dans le dossier « backup » mais également exclurons le fichier « test7 » du processus de synchronisation :

```
[root@srv-deb1 ~]$rsync -av --delete --exclude=toto --exclude=test7./source/ ./backup/
```

```
[root@srv-deb1 ~]$rsync -av --delete --exclude=toto --exclude=test7 ./source/ ./backup/
sending incremental file list
./
test8

sent 272 bytes  received 39 bytes  622.00 bytes/sec
total size is 14  speedup is 0.05
[root@srv-deb1 ~]$
```

Seul le fichier « test8 » a été copié, les fichiers « toto » (dossier « backup ») et « test7 » (dossier « source ») exclus du processus de synchronisation.

Cette option est intéressante mais dans le cas où on se retrouve avec beaucoup de fichiers/dossiers à exclure, on peut se retrouver avec une commande qui peut être un peu longue.

On pourra alors utiliser l'option **- - exclude en spécifiant des « jokers » (cf commandes de base 2)**

4 – rsync vers un serveur/hôte distant :

Pour le moment, nous n'avons réalisé de synchronisation qu'entre deux dossiers sur un hôte en local.
Ce qui est tout aussi intéressant, c'est de voir comment cela se passe lorsque l'on veut synchroniser le dossier « source » sur le serveur SRV-DEB1 vers le dossier « distant » sur le serveur « SRV-DEB2 ».

Connectez vous en ssh sur votre second serveur SRV-DEB2 et créez un dossier « distant » (dans le home de root, peu importe).

N'oubliez pas dans ce cas d'autoriser « root » à se connecter en SSH dans le fichier sshd_config. Bien entendu, vous savez déjà bien que ceci n'est pas à faire dans un environnement de production, on privilégiera la connexion en SSH via un utilisateur sans tous les droits sur le système.

Par ailleurs, vérifiez par la même occasion que rsync est **bien installé** sur SRV-DEB2.

Sur SRV-DEB1, la commande rsync n'est pas bien différente et s'apparente beaucoup à celle que vous avez déjà vue → **scp**.

```
[root@srv-deb1 ~]$rsync -av --delete --exclude=toto --exclude=test7 ./source/ root@192.168.1.41:~/distant
```

La partie en gras de cette commande concerne la destination qui cette fois est distante.

On spécifie l'utilisateur et l'adresse IP de l'hôte distant puis **:** suivi du dossier de destination de la copie.

```
[root@srv-deb1 ~]$rsync -av --delete --exclude=toto --exclude=test7 ./source/ root@192.168.1.41:~/distant
root@192.168.1.41's password:
sending incremental file list
./
test1
test2
test3
test4
test5
test6
test8
REP/

sent 556 bytes  received 160 bytes  286.40 bytes/sec
total size is 14  speedup is 0.02
[root@srv-deb1 ~]$
```

Toutes les options vues précédemment sont tout autant valables.

Vérification sur SRV-DEB2 :

```
[root@srv-deb2 ~]$ls -l distant/
total 8
drwxr-xr-x 2 root   root   4096 mars   9 21:04 REP
-rw-r--r-- 1 tipiak tipiak   14 mars   9 21:20 test1
-rw-r--r-- 1 root   root      0 mars   9 22:27 test2
-rw-r--r-- 1 root   root      0 mars   9 20:44 test3
-rw-r--r-- 1 root   root      0 mars   9 20:44 test4
-rw-r--r-- 1 root   root      0 mars   9 21:14 test5
-rw-r--r-- 1 root   root      0 mars   9 20:44 test6
-rw-r--r-- 1 root   root      0 mars   9 22:44 test8
[root@srv-deb2 ~]$
```

Les fichiers ont bien été copiés, à l'exception – comme spécifié dans la commande – du fichier test7.

Dans le cas où le port d'écoute de SSH n'est pas celui par défaut (22), alors il est nécessaire d'ajouter l'option **-e**.

Cette option va permettre de spécifier le **shell distant à utiliser**.

Dans le cas où par exemple, le port utilisé pour SSH serait **21222**, alors la commande pourrait être la suivante :

```
[root@srv-deb1 ~]$rsync -av --delete --exclude=toto --exclude=test7 ./source/ -e « ssh -p 21222 » root@192.168.1.41:~/distant
```

Cette option s'ajoute entre la source et la destination.

Voilà tout ce que vous avez à savoir sur Rsync (et c'est déjà largement suffisant pour une utilisation professionnelle dans de nombreux cas comme de la copie simple ou de la sauvegarde).

Comprenez bien que le processus de synchronisation **fonctionne dans les deux sens**.

Depuis le début de cet atelier, nous avons systématiquement pris comme source le dossier « source » sur SRV-DEB1.

Il est tout à fait possible de reproduire ces commandes depuis le dossier « backup » ou « distant » sur SRV-DEB2 vers la destination « source » sur SRV-DEB1.

Dans ce cas là, vous inverserez la source et la destination dans votre commande rsync.

Tout ceci est très bien mais comme vous pouvez déjà vous en douter, nous avons fait toutes ces commandes de façon manuelle.
Dans le cadre d'un programme de sauvegarde incrémentale d'un dossier de partage par exemple, il serait préférable d'automatiser cette tâche de façon périodique.

Pour ce faire, il y a plusieurs moyens :

- créer un service systemd (niveau avancé)

https://hermit-notebook.site/fr/blog/how-to/create-custom-systemd-services/

- ou créer un script

- puis tout simplement utiliser le planificateur de tâches sous Linux, « cron », lequel va également exécuter le script contenant votre commande rsync.

C'est ici à vous de jouer ;-)

II – CRON

Ressources proposées :

Crontab tutoriel :

https://www.linuxtricks.fr/wiki/cron-et-crontab-le-planificateur-de-taches

https://www.malekal.com/cron-planifier-execution-programme-script-linux/

https://blog.dorian-depriester.fr/linux/cron-rsync-lassurance-de-la-sauvegarde-de-vos-donnees

Vidéo Rsync + Cron :

https://www.youtube.com/watch?v=0UFEmZMOTr4

(à partir 4'30 pour cron)

Pour la curiosité exemple de script plutôt costaud (synchro de 5 dossiers + affichage des logs + envoi de mail à la fin du processus)

https://www.wolwx.net/ssh-script-de-backup-rsync-automatise-en-cron/