Page de man de rsync (1)

Nom

rsync - une alternative rapide et flexible à rcp

Synopsis

```
rsync [OPTION]... SRC [SRC]... [UTIL@]HOTE:DEST

rsync [OPTION]... [UTIL@]HOTE:SRC [DEST]

rsync [OPTION]... SRC [SRC]... DEST

rsync [OPTION]... [UTIL@]HOTE::SRC [DEST]

rsync [OPTION]... SRC [SRC]... [UTIL@]HOTE::DEST

rsync [OPTION]... rsync://[UTIL@]HOTE[:PORT]/SRC [DEST]

rsync [OPTION]... SRC [SRC]... rsync://[UTIL@]HOTE[:PORT]/DEST
```

Description

rsync est un programme très similaire à rcp, mais possède bien plus d'options et utilise le protocole de mise à jour à distance rsync afin d'accélérer significativement le transfert de fichiers lorsque le fichier de destination existe déjà.

Le protocole de mise à jour à distance rsync permet à rsync de ne transférer que la différence entre deux jeux de fichiers à travers le lien de réseau, en utilisant un algorithme efficace de recherche de somme de contrôle qui est décrit dans le document technique fourni avec ce paquetage.

Quelques possibilités supplémentaires offertes par rsync :

o

```
  possibilité de copier des liens, périphériques, propriétaires, groupes et
permissions
```

o

```
  des options exclut et exclut-depuis similaires à GNU tar
```

o

```
  un mode d'exclusion CVS pour ignorer les mêmes fichiers que CVS
```

o

```
  peut utiliser n'importe quel shell (interpréteur de commandes), y compris
rsh et ssh
```

o

```
  ne nécessite pas de privilèges root
```

o

```
  pipelining des transferts de fichiers pour minimiser les coûts de latence
```

o

```
  possibilité d'utilisation de démons rsync anonymes ou avec
authentification (idéal pour le miroitage)
```

Général

Rsync copie des fichiers depuis ou vers un hôte distant, ou localement sur l'hôte actuel (la copie de fichiers entre deux hôtes distants n'est pas prise en charge).

Il y a deux manières pour rsync de contacter le système distant : en utilisant un programme de shell distant comme transport (tel que ssh ou rsh), ou en contactant un démon rsync directement par TCP. Le transport par shell distant est utilisé à chaque fois que le chemin source ou destination contient un séparateur «:» après la spécification de l'hôte. Le contact direct avec un démon rsync est utilisé lorsque le chemin source ou destination contient un séparateur «::» après la spécification de l'hôte, OU lorsqu'une URL rsync: *est spécifiée (voir aussi la section «UTILISER LES FONCTIONNALITÉS D'UN DÉMON RSYNC VIA UNE CONNEXION SHELL DISTANT» pour une exception à cette dernière règle). Un cas spécial : si la source distante est spécifiée, mais pas la destination, les fichiers distants sont listés avec un affichage similaire à «ls -l». Intuitivement, si ni la source ni la destination ne spécifient d'hôte distant, la copie est faite en local (voir aussi l'option –list-only). ===== Mise en place ===== Consultez le ficher README pour les instructions d'installation. Une fois installé vous pouvez utiliser rsync vers toute machine à laquelle vous pouvez accéder via un shell distant (ainsi que celles auxquelles vous pouvez accéder via le protocole rsync). Pour les transferts, le rsync moderne utilise ssh, cependant il peut avoir était configuré pour utiliser par défaut un shell distant différent, tel que rsh ou remsh. Vous pouvez aussi spécifier un autre shell quelconque, soit en utilisant l'option de la ligne de commande -e, ou en utilisant la variable d'environnement RSYNC_RSH. Une alternative couramment utilisée et offrant un degré de sécurité élevé est ssh. Notez que rsync doit être installé sur la machine locale ainsi que sur la machine de destination. ===== Utilisation ===== rsync s'utilise de la même façon que rcp. Vous devez spécifier une source et une destination, l'une des deux pouvant être distante. Le meilleur moyen d'expliquer la syntaxe est peut être avec quelques exemples : rsync -t *.c foo:src/ Ceci transfère tous les fichiers correspondant au motif *.c du répertoire courant vers le répertoire src sur la machine foo. Si un fichier existe déjà sur le système distant, alors le protocole de mise à jour à distance rsync est utilisé pour mettre à jour le fichier en envoyant*

uniquement les différences. Consultez la documentation technique pour plus de détails. rsync -avz foo:src/bar /data/tmp Ceci transfère récursivement tous les fichiers du répertoire src/bar de la machine foo dans le répertoire /data/tmp/bar de la machine locale. Les fichiers sont transférés en mode «archive», ce qui assure la préservation des liens symboliques, périphériques, attributs, permissions, propriétés, etc lors du transfert. De plus les donnés transférées seront compressées. rsync -avz foo:src/bar/ /data/tmp Une barre oblique à la fin du chemin source modifie ce comportement pour transférer tous les fichiers du répertoire src/bar de la machine foo dans /data/tmp/. Une barre oblique à la fin d'un chemin source signifie «copie le contenu de ce répertoire». Sans la barre oblique à la fin cela signifie «copie le répertoire». Néanmoins, dans les deux cas, les attributs du répertoire sont transférés au répertoire sur la machine destination. Autrement dit, chacune de ces deux commandes copie les fichiers de la même manière, y compris les attributs de /dest/foo : rsync -av /src/foo /dest rsync -av /src/foo/ /dest/foo Notez également qu'une barre oblique n'est pas nécessaire si vous spécifiez le répertoire par défaut d'un hôte ou d'un module. Par exemple, ces deux lignes copient le contenu du répertoire distant vers «/dest» : rsync -av hote: /dest rsync -av hote::module /dest Vous pouvez aussi utiliser rsync en mode local uniquement, lorsque ni la source ni la destination n'ont de «:» dans leur nom. Dans ce cas rsync fonctionne comme une commande performante de copie. rsync unhote.mondomaine.com:: Ceci liste tous les modules rsync anonymes, disponibles sur l'hôte unhote.mondomaine.com. (Consultez la section suivante pour plus de détails.) ===== Utilisation avancée ===== La syntaxe pour récupérer plusieurs fichiers depuis un hôte distant requiert l'utilisation d'espaces protégées dans SRC. Quelques exemples : rsync hote::'nommod/rep1/fichier1 nommod/rep2/fichier2' /dest Ceci copie fichier1 et fichier2 vers /dest depuis un démon rsync. Chacun des arguments additionnels doit inclure le même préfixe « nommod» que le premier et doit être précédé d'un espace. Toutes les autres espaces sont considérées comme faisant partie d'un nom de fichier. rsync -av hote:'rep1/fichier1 rep2/fichier2' /dest Ceci copie fichier1 et fichier2 vers /dest à l'aide d'un shell distant. La coupure des mots est faite par le shell distant, par conséquent si cela ne fonctionne pas c'est que le shell n'est pas configuré pour couper les arguments sur les caractères blancs (configuration très rare, mais pas impossible). Si vous avez besoin de transférer un fichier dont le nom contient des caractères blancs, il est nécessaire ou bien de protéger ces caractères de manière à ce que le shell distant les interprète correctement, ou bien d'utiliser des méta-caractères à la place des blancs. Voici deux exemples : rsync -av hote:'nom\ de\ fichier\ avec\ espaces' /dest rsync -av hote:nom?de?fichier?avec?espaces /dest Ce second exemple suppose que votre shell passe les méta-caractères non développés. S'il affiche une erreur du type «no match» ou «pas de correspondance», mettez le nom entre guillemets. ===== Connexion à un démon rsync ===== Il est possible également d'utiliser rsync sans utiliser rsh ou ssh comme agents de transport. Dans ce cas, vous vous connecterez directement à un démon rsync distant, en général, en utilisant le port TCP 873. (Bien sûr, cela suppose qu'un démon soit en cours d'exécution sur le système distant ; pour plus d'information à ce sujet, référez-vous à la section DÉMARRER UN DÉMON RSYNC QUI ACCEPTE LES CONNEXIONS ci-dessus.) Utiliser rsync comme cela est identique à une utilisation via un shell distant excepté que : o vous utilisez soit un double deux-points «::» au lieu de l'unique deux-points pour séparer le nom d'hôte du chemin, soit une URL rsync:. o

```
  le premier mot après les «::» est le nom d'un module.
```

o

```
  le démon distant peut afficher un message du jour lorsque vous vous
connectez.
```

o

```
  si vous ne spécifiez pas de chemin sur le démon distant, alors la liste
```

```
des chemins accessibles sur le démon sera affichée.
```

o

```
  si vous ne spécifiez pas de destination locale, alors une liste des
fichiers spécifiés sur le démon distant est affichée.
```

Un exemple qui copie tous les fichiers vers un module distant nommé «src» :

```
  rsync -av hote::src /dest
```

Certains modules sur le démon distant peuvent nécessiter une authentification. Si c'est le cas, un mot de passe vous sera demandé lors de votre connexion. Vous pouvez éviter la demande de mot de passe en fixant la variable d'environnement RSYNC_PASSWORD à la valeur du mot de passe que vous voulez utiliser ou en utilisant l'option –password-file. Ceci peut être utile en cas d'utilisation de rsync dans des scripts.

AVERTISSEMENT : sur certains systèmes, les variables d'environnement sont visibles par tous les utilisateurs. Sur ces systèmes, l'utilisation de –password-file est recommandée.

Vous pouvez établir la connexion via un proxy web en donnant à la variable d'environnement RSYNC_PROXY une valeur nomdhote:port qui pointe vers votre proxy web. Notez que votre proxy web doit permettre l'utilisation du port 873.

Utiliser les fonctionnalités d'un démon rsync via une connexion shell distante

Il peut parfois être utile d'utiliser différentes fonctionnalités d'un démon rsync (telles que les modules nommés) sans pour autant autoriser de nouvelle socket de connexion à un système (autre que ce qui est déjà requis pour permettre un accès shell distant). Rsync supporte les connexions à un hôte en utilisant un shell distant puis en lançant un serveur «démon» à usage unique qui lit son fichier de configuration dans le répertoire personnel de l'utilisateur distant. Cela peut être utile si vous souhaitez crypter les données de transfert de style démon ; cependant, comme le démon vient d'être démarré par l'utilisateur distant, il se peut que vous ne puissiez pas utiliser les fonctionnalités telles que le chroot ou le changement d'uid par le démon. (Une autre manière de crypter le transfert avec un démon : utiliser ssh pour faire un tunnel vers un port local de la machine distante et ensuite configurer un démon rsync normal sur l'hôte distant qui n'accepte que les connexions depuis «localhost».)

Du point de vue de l'utilisateur, un transfert démon via une connexion shell distant utilise pratiquement la même syntaxe de ligne de commande qu'un transfert avec un démon rsync, excepté que vous devez explicitement spécifier le programme de shell distant sur la ligne de commande via l'option –rsh=COMMANDE. (Spécifier la variable d'environnement RSYNC_RSH n'activera pas cette fonctionnalité.) Par exemple :

```
  rsync -av --rsh=ssh hôte::module /dest
```

Si vous devez spécifier un utilisateur shell distant différent, n'oubliez pas que le préfixe util@ avant

l'hôte spécifie la valeur de l'utilisateur rsync (pour un module qui requière une authentification basée sur l'utilisateur). Cela signifie que vous devez passer l'option «-l util» à ssh lorsque vous spécifiez le shell distant :

```
rsync -av -e "ssh -l util-ssh" util-rsync@hôte::module /dest
```

Le nom «util-ssh» sera utilisé au niveau de ssh, tandis que le nom «util-rsync» sera utilisé pour l'accès au module.

Démarrer un démon rsync qui accepte les connexions

Afin de se connecter à un démon rsync, le système distant doit avoir un démon déjà en cours d'exécution (ou il faut avoir configuré un service comme inetd pour lancer un démon rsync pour les connexions entrantes sur un port particulier). Pour les informations complètes sur la manière de démarrer un démon qui prend en charge les sockets de connexions entrantes, voir la page de man rsyncd.conf(5) – ceci est le fichier de configuration du démon, elle contient tout les détails sur la manière de lancer le démon (y compris les configurations autonome et inetd).

Si vous utilisez un transport par shell distant pour le transfert, il n'est pas nécessaire de démarrer manuellement un démon rsync.

Exemples

Voici quelques exemples de l'utilisation que je fais de rsync.

Pour sauvegarder le répertoire personnel de mon épouse, qui est constitué de gros fichiers MS Word et de répertoires de courrier électronique, j'utilise un job cron qui exécute

```
rsync -Cavz . arvidsjaur:backup
```

chaque nuit à travers un lien PPP vers un répertoire de duplication sur ma machine «arvidsjaur».

Pour synchroniser mon arborescence du code source de samba, j'utilise les cibles de Makefile suivantes :

```
get:
        rsync -avuzb --exclude '*~' samba:samba/ .
put:
        rsync -Cavuzb . samba:samba/
sync: get put
```

ceci me permet de synchroniser avec un répertoire CVS à l'autre bout du lien. J'effectue ensuite des opérations CVS sur la machine distante, ce qui sauve beaucoup de temps car le protocole CVS distant n'est pas très efficace.

Je miroite un répertoire entre mon «ancien» et «nouveau» site ftp avec la command e :

rsync -az -e ssh –delete ~ftp/pub/samba nimbus:“~ftp/pub/tridge”

Ceci est lancé depuis cron plusieurs fois par jour.

Résumé des options

Voici un court résumé des options disponibles dans rsync. Veuillez vous référer à la description détaillée ci dessous pour une description complète.

```
 -v, --verbose               plus loquace

 -q, --quiet                 moins loquace
 -c, --checksum              utilise la somme de contrôle, pas la date ni la
taille
 -a, --archive               mode archivage; identique à -rlptgoD (pas -H)
 -r, --recursive             visite récursive des répertoires
 -R, --relative              utilise des noms de chemins relatifs
     --no-relative           désactive --relative
     --no-implied-dirs       ne transmet pas les répertoires implicites de -
R
 -b, --backup                effectue des sauvegardes (cf. --suffix et --
backup-dir)
     --backup-dir=RÉP        effectue des sauvegardes dans le répertoire RÉP
     --suffix=SUFFIXE        suffixe de sauvegarde («~» par défaut sauf si -
-backup-dir)
 -u, --update                saute les fichiers plus récents chez le
destinataire
     --inplace               mise à jour de fichiers sur place
 -d, --dirs                  ne transfert pas les répertoires récursivement
 -l, --links                 copie les liens symboliques comme liens
symboliques
 -L, --copy-links            transforme les liens symboliques par les
fichiers référencés
     --copy-unsafe-links     ne transforme que les liens «non-sûrs»
     --safe-links            ignore les liens extérieurs à l'arborescence
 -H, --hard-links            préserve les liens matériels
 -K, --keep-dirlinks         traite les liens de répertoire comme des
répertoires
 -p, --perms                 préserve les permissions
 -o, --owner                 préserve le propriétaire (root uniquement)
 -g, --group                 préserve le groupe
 -D, --devices               préserve les périphériques (root uniquement)
 -t, --times                 préserve les dates
 -O, --omit-dir-times        omet les répertoires lors de la préservation de
date
 -S, --sparse                traite les fichiers à trous efficacement
 -n, --dry-run               montre ce qui aurait été transféré
 -W, --whole-file            copie les fichiers entiers (jamais l'algorithme
rsync)
     --no-whole-file         toujours utiliser l'algorithme rsync
incrémental
```

```
 -x, --one-file-system       ne traverse pas les limites du système de
fichiers
 -B, --block-size=TAILLE     force la taille de bloc de la somme de contrôle
 -e, --rsh=COMMANDE          spécifie un shell distant
     --rsync-path=PROGRAMME  spécifie le nom de l'exécutable rsync chez le
récepteur
     --existing              met à jour uniquement les fichiers qui existent
déjà
     --ignore-existing       ignore les fichiers qui existent déjà
     --remove-sent-files     les fichiers envoyés sont supprimés de chez
l'émetteur
     --del                   synonyme pour --delete-during
     --delete                efface les fichiers qui n'existent pas chez
l'émetteur
     --delete-before         efface avant le transfert (par défaut)
     --delete-during         efface au cours du transfert, pas avant
     --delete-after          efface après transfert, pas avant
     --delete-excluded       efface également les fichiers exclus côté
réception
     --ignore-errors         efface même s'il y a eu des erreurs E/S
     --force                 force la suppression de répertoires même non-
vides
     --max-delete=NUM        n'efface pas plus de NUM fichiers
     --max-size=TAILLE       ne transfert le fichiers plus gros que TAILLE
     --partial               conserve les fichiers partiellement transférés
     --partial-dir=RÉP       place les fichiers partiellement transférés
dans RÉP
     --delay-updates         ne remplace les fichiers mis à jour qu'à la fin
     --numeric-ids           ne remplace pas les uid/gid par des noms
d'utilisateur/groupe
     --timeout=DURÉE         fixe la durée d'attente E/S en secondes
 -I, --ignore-times          ne saute pas les fichiers similaires par la
taille et la date
     --size-only             saute les fichiers qui sont similaires par la
date
     --modify-window=NUM     compare les dates avec une précision moins fine
 -T  --temp-dir=RÉP          crée des fichiers temporaires dans le
répertoire RÉP
 -y, --fuzzy                 se base sur des fichiers similaires si le
fichier manque
     --compare-dest=RÉP      compare les fichiers transmis également à RÉP
     --copy-dest=RÉP         ...et inclut les fichiers non modifiés
     --link-dest=RÉP         crée un lien matériel vers les fichiers de RÉP
si non modifiés
 -z, --compress              transfert en compressant les données
 -C, --cvs-exclude           ignore automatiquement des fichiers, comme le
ferait CVS
 -f, --filter=RÈGLE          ajoute une règle de filtrage de fichier
 -F                          identique à --filter='dir-merge /.rsync-filter'
                             deuxième fois : --filter='- .rsync-filter'
     --exclude=MOTIF         exclut les fichiers correspondant au MOTIF
```

```
     --exclude-from=FICHIER  lit des motifs d'exclusion depuis FICHIER
     --include=MOTIF         n'exclut pas les fichiers correspondant au
MOTIF
     --include-from=FICHIER  lit des motifs d'inclusion depuis FICHIER
     --files-from=FICHIER    lit des fichiers à transférer depuis FICHIER
 -0, --from0                 tous les fichiers des *-from/filtres sont
séparés par 0
     --address=ADRESSE       se lie à l'adresse pour la connexion sortante
vers le démon
     --port=PORT             spécifie un autre numéro de port rsyncd
     --blocking-io           utilise des E/S bloquantes pour le shell
distant
     --no-blocking-io        désactive les E/S bloquantes
     --stats                 affiche quelques statistiques de transfert de
fichiers
     --progress              montre l'avancement pendant le transfert
 -P                          équivalent à --partial --progress
 -i, --itemize-changes       affiche un résumé des changements pour chaque
mise à jour
     --log-format=FORMAT     affiche les noms fichiers selon le format
spécifié
     --password-file=FICHIER lit le mot de passe depuis FICHIER
     --list-only             liste les fichiers au lieu de les copier
     --bwlimit=KBPS          limite la bande passante E/S, En ko par seconde
     --write-batch=FICHIER   enregistre les modifications dans FICHIER
     --only-write-batch=FICHIER comme --write-batch sans mettre à jour la
destination
     --read-batch=FICHIER    exécute le fichier de modification FICHIER
     --protocol=NUM          force l'utilisation d'un protocole rsync
version NUM
     --checksum-seed=NUM     spécifie la graine pour la somme de contrôle%%
 -4, --ipv4                  préfère IPv4
 -6, --ipv6                  préfère IPv6
     --version               affiche le numéro de version
 -h, --help                  affiche cet écran d'aide
```

Rsync peut également être exécuté comme démon, auquel cas il accepte les options suivantes :

```
     --daemon                exécute rsync en tant que démon
     --no-detach             ne se détache pas du parent
     --address=ADRESSE       se lie à l'adresse spécifiée
     --config=FICHIER        spécifie un autre fichier rsyncd.conf
     --bwlimit=KBPS          limite la bande passante I/O, ko par seconde
     --port=PORT             spécifie un autre numéro de port rsyncd
 -v, --verbose               plus loquace
 -4, --ipv4                  préfère IPv4
 -6, --ipv6                  préfère IPv6
 -h, --help                  affiche cet écran d'aide
```

Options

rsync utilise le paquetage «longopt» de GNU. De nombreuses options de la ligne de commande ont deux variantes, une courte et une longue. Celles ci sont exposées ci dessous, séparées par une virgule. Certaines options n'existent que dans la variante longue. Le «=» pour les options qui prennent un paramètre est optionnel ; des blancs peuvent être utilisés à la place.

-h, –help

```
  Affiche une page d'aide décrivant succinctement les options disponibles
dans rsync.
```

–version

```
  Affiche le numéro de version de rsync et quitte.
```

-v, –verbose

```
  Cette option augmente la quantité d'information que vous obtenez lors du
transfert. Par défaut, rsync travaille silencieusement. Avec un -v, rsync
vous indique quels sont les fichiers actuellement transmis et affiche un
bref résumé à la fin. Avec deux drapeaux -v, rsync vous informe des fichiers
ignorés et affiche un résumé avec légèrement plus d'information à la fin.
Plus de deux drapeaux -v ne devraient être utilisés que pour déboguer rsync.
  Remarquez que les noms des fichiers transmis qui sont affichés le sont
selon la valeur par défaut de --log-format qui est «%n%L», ce qui ne
présente que le nom du fichier et, si c'est un lien, vers où il pointe. Avec
un seul -v, il ne sera pas fait mention si un fichier a eu ses attributs
modifiés. Si vous demandez une liste des attributs modifiés (soit avec --
itemize-changes ou bien en ajoutant «%i» à --log-format), alors l'affichage
fera mention de tous les attributs modifiés pour une quelconque raison.
Référez-vous à l'option --log-format pour plus d'information.
```

-q, –quiet

```
  Cette option diminue la quantité d'information affichée lors du transfert,
les messages du serveur distant notamment sont supprimés. Cette option est
utile lorsque vous appelez rsync à partir de cron.
```

-I, –ignore-times

```
  Normalement rsync ignore tous les fichiers qui ont la même taille et ont
une horodate identique. Cette option arrête ce comportement.
```

–size-only

```
  Normalement rsync ignore tous les fichiers qui ont la même taille et une
horodate identique. Avec l'option --size-only, les fichiers seront ignorés
s'ils ont la même taille, indépendamment de l'horodate. Ceci est utile
```

```
lorsque l'on commence à se servir de rsync après avoir utilisé un autre
outil de miroitage qui peut ne pas préserver les horodates exactement.
```

–modify-window

```
  Lors de la comparaison de deux horodates, rsync les considère égales si
elles ne diffèrent pas plus que la fenêtre temporelle. Celle-ci est
normalement nulle (c'est-à-dire une comparaison exacte), mais cela peut être
utile de la fixer à une valeur plus grande dans certaines situations. En
particulier, lors de transferts impliquant un système de fichiers FAT (qui
représente les dates avec une résolution de deux secondes), --modify-
window=1 est alors utile (permettant aux horodates d'avoir jusqu'à une
seconde de différence).
```

-c, –checksum

```
  Ceci force l'expéditeur à faire une somme de contrôle 128-bit MD4 de tous
les fichiers avant le transfert. La somme de contrôle est ensuite
explicitement vérifiée à la réception et tous les fichiers du même nom qui
existent déjà et ont la même somme de contrôle et la même taille sur le
système de réception sont ignorés. Cette option peut être assez lente.
```

-a, –archive

```
  Ceci est équivalent à -rlptgoD. C'est un moyen rapide de dire que vous
voulez la récursion et tout préserver pratiquement tout. La seule exception
est que si --files-from a été spécifié alors -r n'est pas utilisée.
  Notez toutefois que -a ne préserve pas les liens matériels, parce que
trouver les fichiers multiplement liés est coûteux en ressources. Vous devez
spécifier séparément -H.
```

-r, –recursive

```
  Ceci dit à rsync de copier les répertoires récursivement. Référez-vous
également à --dirs (-d).
```

-R, –relative

```
  Utilise des chemins relatifs. Ceci signifie que les chemins complets
spécifiés sur la ligne de commande sont envoyés au serveur plutôt que juste
la dernière partie des noms de fichiers. Ceci est particulièrement utile
lorsque vous voulez envoyer plusieurs répertoires différents en même temps.
Par exemple, si vous avez utilisé la commande
```

```
      rsync /foo/bar/foo.c remote:/tmp/
```

```
  alors ceci aurait créé un fichier appelé foo.c dans /tmp/ sur la machine
distante. Si vous aviez utilisé plutôt
```

```
      rsync -R /foo/bar/foo.c remote:/tmp/
```

```
  alors un fichier appelé /tmp/foo/bar/foo.c aurait été créé sur la machine
distante. Le nom de chemin complet est préservé. Pour limiter la taille du
chemin transmis, vous pouvez faire
```

```
      cd /foo
      rsync -R bar/foo.c remote:/tmp/
```

```
  Cela va créer /tmp/bar/foo.c sur la machine distante.
```

–no-relative

```
  Désactive l'option --relative. Cela n'est nécessaire que si vous souhaitez
utiliser --files-from sans --relative qui est automatiquement activée.
```

–no-implied-dirs

```
  Lorsque cette option est combinée à --relative, les répertoires implicites
des chemins ne sont pas dupliqués lors du transfert. Outre le fait que le
transfert est plus optimal, cela permet aussi d'avoir des liens symboliques
d'un côté sans que ce ne soit le cas de l'autre coté. Par exemple, si vous
transférez le fichier "/path/foo/file" avec -R, par défaut rsync s'assurera
que "/path" et "/path/foo" chez le destinataire correspondent exactement aux
répertoires ou au liens sur la source. En utilisant --no-implied-dirs ces
deux répertoires implicites seront omis, c'est-à-dire que si "/path" est un
répertoire réel sur une machine et un lien symbolique sur l'autre, rsync ne
le modifiera pas.
```

-b, –backup

```
  Avec cette option les fichiers de destination préexistants sont renommés
lors du transfert de chaque fichier. Vous pouvez contrôler le répertoire de
sauvegarde et le suffixe de sauvegarde en utilisant respectivement les
options --backup-dir et --suffix. Remarquez que si vous ne spécifiez pas
l'option --backup-dir alors l'option --omit-dir-times sera activée.
```

–backup-dir=REP

```
  En combinaison avec l'option --backup, ceci dit à rsync de garder toutes
les sauvegardes dans le répertoire spécifié. Ceci est très utile pour des
sauvegardes incrémentales. Vous pouvez spécifier en plus un suffixe de
sauvegarde en utilisant l'option --suffix (autrement, les fichiers sont
sauvegardés dans le répertoire spécifié avec leur noms d'origine).
```

–suffix=SUFFIXE

```
  Cette option permet de modifier le suffixe de sauvegarde utilisé par
l'option -b. Par défaut c'est un «~» à moins que --backup-dir n'ait été
spécifiée, dans quel cas le suffixe par défaut est vide.
```

-u, –update

```
  Ceci force rsync à ignorer tous les fichiers pour lesquels le fichier de
destination existe déjà et avec une date postérieure à celle du fichier de
source. (Si un fichier destination existant a une date égale à celle du
fichier source, il ne sera mis à jour que si les tailles diffèrent.)
  Dans l'implémentation courante de --update, une différence de format du
fichier entre le récepteur et l'émetteur est toujours considérée comme
suffisamment importante pour le mettre à jour, quelque soient les dates des
fichiers. Autrement dit, si la source est un répertoire ou un lien
symbolique tandis que la destination possède un fichier, alors un transfert
sera lancé, peu importe les dates. Ce comportement est susceptible d'être
modifié dans le futur (si vous avez un point de vue, vous pouvez en faire
part, en anglais, à la liste de diffusion).
```

–inplace

```
  Cela change le comportement par défaut de rsync qui crée une nouvelle
copie du fichier puis le déplace vers son véritable nom. Avec cette option
rsync écrira directement par dessus le fichier, ce qui implique que
l'algorithme de rsync ne pourra pas réduire la transmission réseau autant
qu'il n'en est capable (étant donné qu'il ne tente pas encore de trier la
correspondance de données). Une exception à cela est lorsque cette option
est combinée à --backup, dans quel cas rsync sera suffisamment rusé pour
utiliser le fichier de sauvegarde comme base pour le transfert.
  Cette option est utile pour le transfert de fichier volumineux avec des
changements regroupés ou des données ajoutées, ou alors sur les systèmes qui
ont des performances limitées par les accès disques plutôt que réseaux.
  Cette option implique --partial (puisque qu'un transfert interrompu
n'efface pas le fichier), mais elle est incompatible avec --partial-dir et -
-delay-updates. Avant la version 2.6.4 de rsync --inplace était également
incompatible avec --compare-dest et --link-dest.
  AVERTISSEMENT : Les données du fichier seront dans un état inconsistant le
temps du transfert (et éventuellement après si le transfert a été
interrompu), par conséquent cette option n'est pas recommandée pour la mise
à jour de fichier en cours d'utilisation. Notez également que rsync sera
incapable de mettre à jour un fichier sur place s'il n'a pas d'accès en
écriture pour l'utilisateur destinataire.
```

-d, –dirs

```
  Indique à l'émetteur d'inclure tous les répertoires rencontrés.
Contrairement à --recursive, le contenu des répertoires n'est pas copié à
moins que le répertoire ait été explicitement spécifié sur la ligne de
commande par «.» ou par un nom suivi de «/» (par exemple «foo/»). Sans cette
option ou --recursive, rsync omettra tous les répertoires rencontrés (en
affichant un message indiquant l'omission).
```

-l, –links

```
  Lorsque des liens symboliques sont rencontrés, ils sont recréés à la
destination.
```

-L, –copy-links

```
  Lorsque des liens symboliques sont rencontrés, le fichier vers lequel ils
pointent est copié plutôt que le lien symbolique. Dans les anciennes
versions de rsync, cette option avait pour effet de bord de forcer le coté
récepteur à suivre les liens symboliques, tels que les liens symboliques
vers des répertoires. Dans les versions récentes de rsync, il est nécessaire
de spécifier --keep-dirlinks (-K) pour obtenir cette fonctionnalité
supplémentaire. Il y a une exception lorsque le rsync du coté récepteur est
trop vieux pour comprendre -K -- dans ce cas, l'option -L entraînera l'effet
de bord comme dans les anciennes versions.
```

–copy-unsafe-links

```
  Ceci dit à rsync de copier le fichier référencé par les liens symboliques
qui pointent en dehors de l'arborescence source. Les liens symboliques
absolus sont également traités comme des fichiers ordinaires, ainsi que les
liens symboliques dans l'arborescence source elle même lorsque --relative
est utilisée.
```

–safe-links

```
  Ceci indique à rsync d'ignorer tous les liens symboliques qui pointent
vers l'extérieur de l'arborescence copiée. Tous les liens symboliques
absolus sont ignorés également. Utiliser cette option en même temps que --
relative peut donner des résultats inattendus.
```

-H, –hard-links

```
  Ceci indique à rsync de recréer les liens matériels sur le système distant
à l'identique du système local. Sans cette option, les liens matériels sont
traités comme des fichiers réguliers.
  Notez que rsync ne peut détecter les liens matériels que si les deux
parties du lien sont dans la liste de fichiers envoyés.
  Cette option peut être assez lente, ne l'utilisez que si vous en avez
vraiment besoin.
```

-K, –keep-dirlinks

```
  Coté réception, si un lien symbolique pointe vers un répertoire, il sera
traité comme correspondant à un répertoire chez l'émetteur.
```

-W, –whole-file

```
  Avec cette option, l'algorithme rsync incrémental n'est pas utilisé ; au
lieu de cela, le fichier entier est envoyé tel quel. Le transfert peut être
plus rapide grâce à cette option lorsque la bande passante entre les
```

```
machines source et cible est plus grande que la bande passante vers le
disque (en particulier lorsque le «disque» est en fait un système de
fichiers sur réseau NFS). Cette option est utilisée par défaut lorsque la
source et la cible sont sur la même machine.
```

–no-whole-file

```
  Désactive --whole-file, utile lorsqu'elle est activée par défaut.
```

-p, –perms

```
  Cette option entraîne la mise à jour des permissions distantes pour
qu'elles soient identiques aux permissions locales.
  Sans cette option, tous les fichiers existants (y compris les fichiers mis
à jour) gardent leur permissions actuelles, tandis que les nouveaux fichiers
reçoivent des permissions basées sur les permissions du fichier source mais
masquées par le masque utilisateur du récepteur (ce comportement est
identique aux autres utilitaires de copie de fichier, tel que cp).
```

-o, –owner

```
  Cette option dit à rsync de mettre le propriétaire du fichier de
destination identique à celui du fichier source. Sur la plupart des
systèmes, uniquement le super-utilisateur peut fixer le propriétaire des
fichiers. Par défaut, la conservation est effectuée selon le nom mais
lorsque cela ne fonctionne pas, rsync utilise comme alternative le numéro
ID. Référez-vous à l'option --numeric-ids pour plus de détail.
```

-g, –group

```
  Cette option dit à rsync de mettre le groupe du fichier de destination
identique à celui du fichier source. Si le programme récepteur n'est pas
exécuté en tant que super-utilisateur, uniquement les groupes dont le
récepteur est un membre seront préservés. Par défaut, la conservation est
effectuée selon le nom mais lorsque cela ne fonctionne pas, rsync utilise
comme alternative le numéro ID. Référez-vous à l'option --numeric-ids pour
plus de détail.
```

-D, –devices

```
  Cette option entraîne le transfert d'informations sur les périphériques
caractères et blocs vers le système distant pour recréer ces périphériques.
Cette option n'est disponible que pour le super-utilisateur.
```

-t, –times

```
  Ceci dit à rsync de transférer les dates de modifications avec les
fichiers et de les mettre à jour sur le système distant. Notez que si cette
option n'est pas utilisée, l'optimisation qui consiste à exclure les
fichiers qui n'ont pas été modifiés ne peut pas être efficace ; en d'autres
```

```
termes, un -t ou -a absent va faire que le prochain transfert se ferra comme
s'il utilisait -I, ce qui implique que tous les fichiers seront mis à jour
(même si l'algorithme rsync rendra l'opération relativement efficace si les
fichiers n'ont pas été modifiés, vous avez tout intérêt à utiliser -t).
```

-O, –omit-dir-times

```
  Ceci indique à rsync d'omettre les répertoires lorsque les dates de
modification sont préservées (cf. --times). Si les répertoires du
destinataire sont partagés via NFS, il est recommandé d'utiliser -O. Cette
option est automatiquement sélectionnée si vous utilisez --backup sans --
backup-dir.
```

-n, –dry-run

```
  Ceci dit à rsync de ne faire aucun transfert, mais de juste rapporter les
actions qu'il aurait faites.
```

-S, –sparse

```
  Essaie de traiter les fichiers à trous efficacement, de sorte qu'ils
prennent moins de place sur la destination. [NDT : sparse file = fichier à
trous]
  NOTE : N'utilisez pas cette option lorsque la destination est un système
de fichiers «tmpfs» Solaris. Il semble qu'il ne traite pas les déplacements
au-dessus de zones vides et cela entraîne des corruptions de fichiers.
```

-x, –one-file-system

```
  Ceci dit à rsync de ne pas traverser les limites du système de fichiers
lors d'un parcours récursif. Ceci est utile pour transférer le contenu d'un
système de fichiers exclusivement.
```

–existing

```
  Ceci dit à rsync de ne créer aucun nouveau fichier -- uniquement mettre à
jour les fichiers qui existent déjà sur la destination.
```

–ignore-existing

```
  Ceci dit à rsync de ne pas mettre à jour les fichiers déjà existants chez
le destinataire.
```

–remove-sent-files

```
  Ceci dit à rsync d'effacer de l'émetteur les fichiers ou les liens
symboliques transmis créés ou mis à jour chez le destinataire. Les
répertoires ainsi que les périphériques ne sont jamais effacés, ni les
fichiers et les liens symboliques dont les attributs sont changés.
```

–delete

```
  Ceci dit à rsync d'effacer tous les fichiers superflus côté réception
(ceux qui ne sont pas du côté envoi); uniquement pour les répertoires
synchronisés. Vous devez explicitement demander à rsync de transmettre le
contenu du répertoire (par ex : «dir» ou «dir/»), sans utiliser de méta-
caractères pour indiquer le contenu du répertoire (par ex : «dir/*»). En
effet, les méta-caractères sont interprétés par le shell, ce qui implique
que rsync reçoit une requête pour transmettre une liste de fichiers et non
pas un répertoire entier. Les fichiers qui sont exclus du transfert ne
seront pas effacés sauf si vous utilisez --delete-excluded ou que vous
indiquez dans les règles de ne faire de correspondance que sur le côté
émetteur (cf. les modificateurs d'inclusion/exclusion de la section RÈGLES
DE FILTRE).
  Cette option n'a pas d'effet si le parcours récursif des répertoires n'est
pas sélectionné.
  Cette option peut être dangereuse si elle n'est pas utilisée correctement
! C'est une très bonne idée d'exécuter rsync avec l'option --dry-run (-n)
pour voir quels seraient les fichiers effacés et s'assurer qu'aucun fichier
important n'est listé.
  Si le côté envoi détecte des erreurs d'entrée/sortie (E/S), alors
l'effacement des fichiers côté destination est automatiquement désactivé.
Ceci prévient des échecs temporaires de système de fichiers (comme les
erreurs NFS) du côté envoi causant une destruction massive de fichiers côté
destination. Vous pouvez passer outre ceci avec l'option --ignore-errors.
  L'option --delete peut être combinée avec l'une des options --delete-QUAND
ainsi qu'avec --delete-excluded. Si aucune des options --delete-QUAND n'est
spécifiée, actuellement, rsync choisira l'algorithme --delete-before.
Cependant, il se peut que les versions futures choisissent par défaut
l'algorithme --delete-during. Voir aussi --delete-after.
```

–delete-before

```
  Avec cette option la suppression des fichiers du côté récepteur est
effectuée lorsque le transfert commence. Ceci est l'option par défaut avec -
-delete ou --delete-excluded si aucune des options --delete-QUAND n'est
spécifiée.
  La suppression avant le transfert est utile si l'espace libre sur le
système de fichiers est faible et que les fichiers superflus auraient
empêché le transfert de se terminer correctement. Cependant, cela introduit
un délai avant le démarrage du transfert, ce qui peut causer une erreur de
temporisation sur le transfert (si --timeout a été spécifiée).
```

–delete-during, –del

```
  Avec cette option, la suppression des fichiers du côté récepteur est
effectuée incrémentalement au cours du transfert. Ceci est une méthode plus
rapide que celles avant ou après le transfert, mais elle n'est supportée que
depuis la version 2.6.4 de rsync. Référez-vous à --delete (qui est
sélectionnée implicitement) pour plus d'information.
```

–delete-after

```
  Avec cette option, la suppression des fichiers du côté récepteur est
effectuée une fois le transfert terminé. Ceci est utile si au cours du
transfert vous envoyez de nouveaux fichiers de règle à fusionner par
répertoire et que vous souhaitez que les exclusions qu'ils contiennent
prennent effet avant la phase de suppression. Référez-vous à --delete (qui
est sélectionnée implicitement) pour plus d'information.
```

–delete-excluded

```
  En plus d'effacer les fichiers côté récepteur qui ne sont pas côté
émetteur, ceci dit à rsync d'effacer également tous les fichiers côté
réception qui sont exclus (voir --exclude). Référez-vous à la section RÈGLES
DE FILTRE pour un moyen d'obtenir ce comportement individuellement à chaque
exclusion du côté récepteur, et pour un moyen de protéger les fichiers de --
delete-excluded. Référez-vous à --delete (qui est sélectionnée
implicitement) pour plus d'information.
```

–ignore-errors

```
  Dit à --delete de continuer et d'effacer les fichiers même lorsqu'il y a
des erreurs E/S.
```

–force

```
  Cette option dit à rsync d'effacer les répertoires même s'ils ne sont pas
vides lorsqu'ils sont remplacés par autre chose qu'un répertoire. Ceci ne
s'applique pas à l'option --delete car les suppressions sont faites en
profondeur d'abord. Requiert l'option --recursive (implicitement
sélectionnée par -a) pour avoir un effet.
```

–max-delete=NUM

```
  Ceci dit à rsync de ne pas effacer plus de NUM fichiers ou répertoires
(NUM doit être supérieur ou égal à un). Ceci est utile lorsque l'on miroite
de très grandes arborescences pour prévenir des désastres.
```

–max-size=TAILLE

```
  Ceci dit à rsync de ne pas transférer de fichier dont la taille est
supérieure à TAILLE. La valeur TAILLE peut être suivie d'une lettre pour
indiquer une unité (K, M ou G) et peut être une valeur fractionnelle (par ex
: "--max-size=1.5m").
```

-B , –block-size=TAILLE

```
  Ceci contrôle la taille de bloc utilisée dans l'algorithme rsync. Elle est
normalement calculée à partir de la taille de chaque fichier à mettre à
jour. Reportez-vous à la documentation technique pour plus de détails.
```

-e, –rsh=COMMANDE

```
  Cette option vous permet de choisir un shell distant alternatif pour la
communication entre les copies locales et distantes de rsync. Par défaut,
rsync va utiliser ssh, mais vous pouvez apprécier l'utilisation de rsh dans
un réseau local.
  Si cette option est utilisée en conjonction de [util@]hote::module/chemin,
alors le shell distant COMMANDE sera utilisé pour lancer un démon rsync sur
la machine hôte, et toutes les données seront transmises via la connexion à
ce shell distant plutôt que directement via une connexion socket avec le
démon rsync de l'hôte distant. Reportez-vous à la section «UTILISER LES
FONCTIONNALITÉS D'UN DÉMON RSYNC VIA UNE CONNEXION SHELL DISTANT» ci-dessus
pour plus d'information.
  Il est possible d'ajouter des arguments à COMMANDE, il suffit de faire en
sorte que COMMANDE soit présentée comme un seul argument à rsync. Par
exemple :
```

1. e “ssh -p 2234”

```
  (Notez que les utilisateurs de ssh peuvent aussi personnaliser en fonction
du site les options de connexion à l'aide du fichier .ssh/config.)
  Vous pouvez aussi choisir le shell distant en utilisant la variable
d'environnement RSYNC_RSH, qui accepte les mêmes valeurs que -e.
  Voir aussi l'option --blocking-io qui est affectée par cette option.
```

–rsync-path=PROGRAMME

```
  Utilisez ceci pour spécifier le programme à lancer sur la machine distante
pour démarrer rsync. Utile si ce chemin n'est pas dans le $PATH de votre
shell distant (par ex : --rsync-path=/usr/local/bin/rsync). Notez que
PROGRAMME est exécuté à l'aide du shell, par conséquent ce peut être
n'importe quel programme, script ou suite de commande que vous souhaitez
tant que l'entrée et la sortie standard que rsync utilise pour communiquer
ne sont pas corrompus.
  Un exemple quelque peu pernicieux est de modifier le répertoire par défaut
de la machine distante pour l'utiliser avec l'option --relative. Par exemple
:
```

```
      rsync -avR --rsync-path="cd /a/b && rsync" hote:c/d /e/
```

-C, –cvs-exclude

```
  Ceci est un raccourci utile pour exclure une grande quantité de fichiers
que vous n'avez souvent pas envie de transférer entre deux systèmes. Il
utilise l'algorithme que CVS utilise pour déterminer si un fichier doit être
ignoré.
  La liste d'exclusion est initialisée à :
```

```
          RCS SCCS CVS CVS.adm RCSLOG cvslog.* tags TAGS .make.state
.nse_depinfo *~ #* .#* ,* _$* *$ *.old *.bak *.BAK *.orig *.rej .del-* *.a
```

```
*.olb *.o *.obj *.so *.exe *.Z *.elc *.ln core .svn/
```

```
  ensuite les fichiers listés dans $HOME/.cvsignore sont ajoutés à la liste
ainsi que tous les fichiers listés (séparés par des blancs) dans la variable
d'environnement CVSIGNORE.
  Finalement, tous les fichiers qui sont dans le même répertoire qu'un
.cvsignore et qui correspondent à l'un des motifs listé dans celui-ci sont
ignorés. Contrairement aux filtres et aux fichiers d'exclusion de rsync, les
motifs sont séparés par des blancs. Consultez le manuel cvs(1) pour plus
d'information.
  Si vous combinez -C à vos propres règles --filter, vous devriez garder à
l'esprit que ces exclusions CVS sont ajoutées à la fin de vos propres
règles, quelle que soit la position de -C dans la liste d'arguments. Cela
leur donne une plus faible priorité que les règles que vous spécifiez
explicitement. Pour contrôler quand exactement ces exclusions CVS sont
insérées dans vos règles de filtre, vous devez omettre -C de la ligne
d'arguments et utiliser à la place la combinaison de --filter=:C et --
filter=-C (soit sur la ligne de commande, soit en plaçant ":C" et "-C" dans
l'un de vos fichiers de règles). La première des options active
l'utilisation du fichier .cvsignore. La seconde importe en une fois toutes
les exclusions CVS mentionnées précédemment.
```

-f, –filter=RÈGLE

```
  Cette option vous permet d'ajouter des règles pour exclure de manière fine
certains fichiers depuis une liste de fichier à transmettre. Cette option
est particulièrement utile en combinaison à un transfert récursif.
  Il est possible des spécifier plusieurs fois l'option --filter sur la
ligne de commande, vous permettant ainsi de construire à vos souhaits une
liste de fichiers à exclure.
  Voir la section RÈGLES DE FILTRE pour plus d'information sur cette option.
```

-F

```
  L'option t-F est un raccourci pour ajouter deux règles --filter à votre
commande. La première occurrence est un raccourci pour cette règle :
```

1. -filter=': /.rsync-filter'

```
  Ceci indique à rsync de rechercher récursivement les fichiers .rsync-
filter éparpillés dans la hiérarchie et d'utiliser leurs règles pour filtrer
les fichiers lors du transfert. Si -F est répété, c'est alors un raccourci
pour la règle suivante :
```

1. -filter='- .rsync-filter'

```
  Ceci filtre les fichiers .rsync-filter eux-mêmes lors du transfert.
  Voir la section RÈGLES DE FILTRE pour plus d'information sur le
fonctionnement de cette option.
```

–exclude=MOTIF

```
  Cette option est une forme simplifiée de l'option --filter avec une règle
d'exclusion et qui n'accepte pas la syntaxe complète des règles de filtre.
  Voir la section RÈGLES DE FILTRE pour plus d'information sur cette option.
```

–exclude-from=FICHIER

```
  Cette option est similaire à l'option --exclude, mais ajoute les motifs
d'exclusion listés dans le fichier FICHIER à la liste d'exclusion. Les
lignes vides dans FICHIER ou les lignes commençant par «;» ou par «#» sont
ignorées. Si FICHIER est - alors la liste sera lue depuis l'entrée standard.
```

–include=MOTIF

```
  Cette option est une forme simplifiée de l'option --filter avec une règle
d'inclusion et qui n'accepte pas la syntaxe complète des règles de filtre.
  Voir la section RÈGLES DE FILTRE pour plus d'information sur cette option.
```

–include-from=FICHIER

```
  Ceci spécifie une liste de motifs d'inclusion à lire depuis un fichier. Si
FICHIER est - alors la liste sera lue depuis l'entrée standard.
```

–files-from=FICHIER

```
  Cette option vous permet de spécifier la liste exacte des fichiers à
transférer (obtenue depuis le FICHIER spécifié ou depuis l'entrée standard
si le nom est «-»). Elle modifie également le comportement par défaut de
rsync pour rendre le transfert de seulement les fichiers et le répertoires
spécifiés plus simple :
```

```
      o
          L'option --relative (-R) est activée, ce qui préserve
l'information de chemin spécifié pour chaque entrée du fichier (utilisez --
no-relative si vous souhaitez la désactiver).
      o
          L'option --dirs (-d) est activée, ce qui créera les répertoires
spécifiés dans la liste chez le récepteur au lieu d'afficher un
avertissement et de les sauter.
      o
          Le comportement de l'option --archive (-a) ne contient pas --
recursive (-r), vous devez donc la spécifier explicitement si vous la
souhaitez.
```

```
  Les noms de fichiers qui sont lus depuis FICHIER sont tous relatifs au
répertoire source -- les «\|» en début de nom sont supprimés et les «..» ne
sont pas autorisés à remonter plus haut que le répertoire source. Par
exemple, la commande suivante :
```

```
      rsync -a --files-from=/tmp/foo /usr hotedst:/backup
```

Si /tmp/foo contient la chaîne «bin» (ou même «/bin»), le répertoire /usr/bin sera créé en tant que /backup/bin sur la machine hotedst. S'il contient «bin/» (remarquez la barre oblique), le contenu immédiat du répertoire sera aussi transmis (sans qu'il y ait besoin de le mentionner explicitement dans le fichier, depuis la version 2.6.4). Dans les deux cas, si l'option -r est activée, alors toute la hiérarchie du répertoire sera transférée (n'oubliez pas que -r doit être explicitement spécifiée car, avec --files-from, elle n'est pas impliquée par -a). Gardez également en tête que l'effet de l'option --relative (activée par défaut) est de dupliquer uniquement l'information de chemin présente dans le fichier ; elle ne force pas la duplication du chemin source (/usr dans le cas présenté).

De plus, le fichier de --files-from peut être lu depuis l'hôte distant au lieu de l'hôte local, il faut pour cela spécifier «nom-de-l-hôte:» avant le nom du fichier (le nom de l'hôte doit correspondre à l'une des deux bouts de la transmission). Un préfixe «:» est un raccourci pour indiquer d'utiliser l'hôte distant. Par exemple :

```
rsync -a --files-from=:/chemin/file-list src:/ /tmp/copy
```

Ceci copiera tous les fichiers spécifiés dans le fichier /chemin/file-list situé sur l'hôte distant «src».

-0, –from0

Ceci indique à rsync que les noms de règles/fichiers lus depuis un fichier sont séparés par un caractère nul («\0»), pas un NL, CR ni CR+LF. Ceci affecte --exclude-from, --include-from, --files-from, ainsi que tous les fichiers inclus depuis une règle --filter. Par contre, ceci n'affecte pas --cvs-exclude (puisque tous les noms lus depuis un .cvsignore sont séparés par un blanc).

-T, –temp-dir=RÉP

Cette option dit à rsync d'utiliser RÉP comme répertoire de dépôt pour la création de copies temporaires des fichiers transférés sur le côté réception. Le comportement par défaut est de créer les fichiers temporaires dans le répertoire de réception.

-y, –fuzzy

Cette option indique à rsync de rechercher un fichier de base pour tout fichier destination manquant. L'algorithme actuel recherche dans le même répertoire que le fichier destination un fichier qui ait soit la même taille et la même horodate, ou un fichier au nom similaire. Si un tel fichier est trouvé alors rsync utilisera ce fichier de base flou pour tenter d'accélérer le transfert.

Notez que l'utilisation de --delete peut supprimer les fichiers flous potentiels, par conséquent il vaut mieux utiliser --delete-after ou bien spécifier des exclusions de fichier qui évitent ces suppressions.

–compare-dest=RÉP

```
  Cette option ordonne à rsync d'utiliser REP sur la machine de destination
comme un répertoire supplémentaire avec lequel il faut comparer les fichiers
de destination (pour les fichiers qui ne sont pas présents dans le
répertoire de destination). Si un fichier est trouvé dans RÉP et qu'il est
identique au fichier émis, alors le fichier ne sera PAS transmis vers le
répertoire de destination. Cela est utile pour créer une sauvegarde
incrémentale, qui ne contient que les fichiers modifiés par rapport à la
sauvegarde précédente.
  Depuis la version 2.6.4, il est possible de spécifier plusieurs
répertoires --compare-dest, rsync recherchera alors un fichier identique
dans l'ordre de la liste spécifiée. Si un fichier identique est trouvé mais
que ses attributs diffèrent, alors une copie locale est faite et les
attributs sont mis à jour. Lorsqu'aucun fichier identique n'est trouvé, un
des fichiers des différents RÉP est utilisé comme base pour accélérer le
transfert.
  Si REP est un chemin relatif, il est relatif au répertoire de destination.
Voir aussi --copy-dest et --link-dest.
```

–copy-dest=REP

```
  Cette option a le même comportement que --compare-dest, mais rsync copiera
également les fichiers inchangés trouvés dans REP vers le répertoire de
destination en utilisant la copie locale. Ceci est utile pour des transferts
vers un nouveau répertoire en laissant l'ancien intact, procédant à un
remplacement-éclair de l'ancien par le nouveau une fois le transfert
terminé.
  Il est possible de spécifier plusieurs répertoires --compare-dest, rsync
recherchera alors un fichier identique dans l'ordre de la liste spécifiée.
Si aucun fichier identique n'est trouvé, un des fichiers des différents RÉP
est utilisé comme base pour accélérer le transfert.
  Si RÉP est un chemin relatif, il est relatif au répertoire de destination.
Voir aussi --compare-dest et --link-dest.
```

–link-dest=RÉP

```
  Cette option a le même comportement que --copy-dest, mais, à la place
d'une copie, un lien matériel est effectué entre les fichiers inchangés. Les
fichiers doivent être identiques même au niveau des attributs (par ex.
permissions, droits...) afin que les fichiers puissent être liés. Un exemple
:
```

```
      rsync -av --link-dest=$PWD/rép_précédent hôte:rép_src/ nouv_rép/
```

```
  Depuis la version 2.6.4, il est possible de spécifier plusieurs
répertoires --compare-dest, rsync recherchera alors un fichier identique
dans l'ordre de la liste spécifiée. Si un fichier identique est trouvé mais
que ses attributs diffèrent alors une copie locale est faite et les
attributs sont mis à jour. Lorsqu'aucun fichier identique n'est trouvé, un
```

des fichiers des différents RÉP est utilisé comme base pour accélérer le transfert.
  Si RÉP est un chemin relatif, il est relatif au répertoire de destination. Voir aussi --compare-dest et --copy-dest.
  Notez qu'un bogue était présent dans les versions de rsync antérieures à 2.6.1, ce qui pouvait empêcher --link-dest de fonctionner correctement pour les utilisateurs non-root lorsque -o était spécifiée (ou implicitement activée par -a). Vous pouvez contourner le problème en évitant d'utiliser -o lors d'envois vers un vieux rsync.

-z, –compress

  Avec cette option, rsync compresse toutes les données des fichiers qu'il envoie à la machine de destination. Cette option est utile avec des connexions réseau lentes.
  Notez que cette option permet d'obtenir des taux de compression meilleurs que ce qui peut être obtenu avec des shells distants qui compressent, ou un transport qui compresse, car cela prend en compte l'information implicite envoyée pour les blocs de données qui correspondent entre eux.

–numeric-ids

  Avec cette option, rsync va transférer le numéro identificateur de groupe et d'utilisateur plutôt que d'utiliser les noms de groupe et d'utilisateur en les faisant correspondre des deux cotés.
  Par défaut rsync va utiliser le nom d'utilisateur et de groupe pour déterminer quel propriété donner aux fichiers. Le numéro d'utilisateur spécial 0 et le numéro de groupe spécial 0 ne sont jamais en correspondance avec les noms d'utilisateurs/groupes même si l'option --numeric-ids n'est pas spécifiée.
  Si un utilisateur ou un groupe n'a pas de nom sur l'émetteur ou n'est pas présent sur le récepteur, alors le numéro identificateur est utilisé à la place. Voir aussi les remarques sur les paramètres d'«utilisation de chroot» dans la page man de rsyncd.conf pour plus d'information à propos de l'influence des paramètres de chroot sur la capacité de rsync à rechercher les noms d'utilisateur et de groupe, et sur les possibilités d'éviter les problèmes.

–timeout=DURÉE

  Cette option vous permet de fixer le temps d'attente avant échec d'entrée/sortie en secondes. Si aucune donnée n'est transférée pendant la durée spécifiée alors rsync se termine. Vaut 0 par défaut, ce qui signifie une attente infinie.

–address=ADRESSE

  Par défaut rsync va s'attacher à l'adresse joker lors d'une connexion vers un démon rsync. L'option --address vous permet de spécifier une adresse IP (ou un nom d'hôte) particulière vers laquelle se connecter. Voir aussi cette

```
option dans la section du mode --daemon.
```

–port=PORT

```
  Ceci spécifie un numéro de port TCP alternatif à utiliser au lieu du port
par défaut 873. Cela n'est vraiment utile que lorsque vous utilisez une
syntaxe avec un double deux-points «::» pour vous connecter à un démon rsync
(puisque la syntaxe à l'aide d'une URL peut spécifier le port directement
dans l'URL). Voir aussi cette option dans la section du mode --daemon.
```

–blocking-io

```
  Ceci dit à rsync d'utiliser des entrées/sorties (E/S) bloquantes lorsqu'il
démarre un transport via shell distant. Si le shell est rsh ou remsh, cette
option est utilisée par défaut, sinon des E/S non-bloquantes sont utilisées.
(Notez que ssh préfère les E/S non-bloquantes.)
```

–no-blocking-io

```
  Désactive --blocking-io, à utiliser lorsqu'elle est activée par défaut.
```

-i, –itemize-changes

```
  Demande une liste simple des changements effectués sur chacun des
fichiers, y comprit les modifications d'attribut. C'est équivalent à
l'option --log-format='%i %n%L'.
  Le «%i» génère une sortie obscure de 9 caractères. Le format général de
cette chaîne est UXcstpoga, où U est remplacé par le type de mise à jour
effectuée, X remplacé par le type de fichier, et les autres lettres
représentant des attributs qui seront affichés s'ils ont été modifiés.
  Les différents types de mises à jour qui remplacent U sont :
```

```
      o
          Un < signifie que le fichier est transféré vers la machine
distante (envoi).
      o
          Un > signifie que le fichier est transféré vers la machine locale
(réception).
      o
          Un c signifie que l'entité a été créée ou modifiée (tel que la
création d'un répertoire ou la modification d'un lien symbolique, etc.).
      o
          Un h signifie que l'entité est un lien matériel vers un fichier
(requiert --hard-links).
      o
          Un . signifie que l'entité n'a pas été mise à jour (bien que
certains de ces attributs aient pu être modifiés).
```

```
  Les types de fichiers qui remplacent le X sont : f pour un fichier, d pour
un répertoire, L pour un lien symbolique, et D pour un périphérique.
```

Les autres lettres de la chaîne de caractères sont les lettres qui seront affichées si l'attribut qui leur est associé est mis à jour, s'il n'y a pas de changement ce sera un «.». Il y a trois exceptions à cela : (1) une entité qui vient d'être créée a toutes les lettres remplacées par «+», (2) une entité identique remplace les points par des espaces, et (3) un attribut inconnu remplace la lettre par un «?» (cela peut arriver lors de la connexion à un vieux rsync).

Les attributs associés à chaque lettre sont :

- Un c indique que la somme de contrôle du fichier est différente et qu'il sera mis à jour lors du transfert (requiert --checksum).
- Un s indique que la taille du fichier est différente et qu'il sera mis à jour lors du transfert.
- Un t indique que la date de modification est différente et qu'elle sera changée à la valeur de celle de l'émetteur (requiert --times). Un T indique que la date sera celle du transfert, ce qui est toujours le cas lorsqu'un lien symbolique est transféré, et cela arrive aussi lorsqu'un fichier ou un périphérique est transféré sans --times.
- Un p indique que les permissions sont différentes et qu'elles seront changées à la valeur de celles de l'émetteur (requiert --perms).
- Un o indique que le propriétaire est différent et qu'il sera changé à la valeur de celle de l'émetteur (requiert --owner et les privilèges root).
- Un g indique que le groupe est différent et qu'il sera changé à la valeur de celle de l'émetteur (requiert --group et les permissions de changer de groupe).
- Le a est réservé pour une version future qui supporte les attributs étendus, tels que les ACLs.

Un autre affichage est possible : quand des fichiers sont supprimés, le «%i» affichera «*deleting» pour chaque entité supprimée (si la connexion est faite avec un rsync suffisamment récent qui journalise les suppressions plutôt que de les indiquer par un message texte).

–log-format=FORMAT

Ceci vous permet de spécifier exactement ce que le client rsync affiche sur la sortie standard pour chaque fichier [NDT : to log = journaliser]. Le format de journalisation est une chaîne de caractère contenant des séquences d'échappement d'un caractère préfixées par un caractère pourcent «%». La liste des différents caractères d'échappement possibles est disponible dans la partie sur le format de journalisation de la page de manuel de rsyncd.conf. (Notez que cette option ne modifie pas l'aspect du journal du démon rsync.)

```
  Spécifier cette option forcera rsync à mentionner chaque fichier,
répertoire, etc. modifié de manière significative (fichier transféré, lien
symbolique/périphérique recréé, ou répertoire modifié). Si une séquence
d'échappement des modifications d'attribut (%i) est inclue dans la chaîne de
caractères, alors la journalisation des noms mentionnera toute entité qui a
été modifiée d'une quelconque façon (tant que le côté récepteur est un rsync
version 2.6.4 ou supérieure). Voir l'option --itemize-changes pour une
description de l'affichage de «%i».
  L'option --verbose implique le format «%n%L», mais vous pouvez utiliser --
log-format sans --verbose, ou redéfinir le format de l'affichage par fichier
à l'aide de cette option.
  Rsync affiche la chaîne de journalisation avant le transfert du fichier à
moins qu'une des statistiques de transfert ne soit demandée. Dans ce cas, la
journalisation est faite après la fin du transfert. Lorsque la
journalisation se fait après et que --progress a été spécifié, rsync
affichera aussi le nom du fichier transféré avant l'information d'avancement
(qui, bien sûr, sera ensuite suivie de l'affichage de journalisation).
```

–stats

```
  Demande à rsync d'afficher un jeu de statistiques verbeux sur le transfert
de fichiers, ce qui vous permet de quantifier l'efficacité de l'algorithme
rsync pour vos données.
```

–partial

```
  Par défaut, rsync va effacer tous les fichiers transférés partiellement si
le transfert est interrompu. Dans certaines circonstances il est préférable
de conserver les fichiers partiellement transférés. L'option --partial
demande à rsync de conserver le ficher incomplet ce qui devrait accélerer un
transfert ultérieur de la fin du fichier.
```

–partial-dir=REP

```
  Une meilleure manière que l'option --partial pour conserver les fichiers
partiels est de spécifier un REP qui sera utilisé pour contenir les données
partielles (au lieu de les écrire dans le fichier destination). Lors du
transfert suivant, rsync utilisera le fichier de ce répertoire comme donnée
pour accélérer la reprise du transfert et puis l'effacera une fois le
transfert achevé. Notez que si --whole-file est spécifié (même
implicitement), tout fichier présent dans le répertoire partiel
correspondant à un fichier mis à jour sera purement et simplement supprimé
(puisque dans ce cas rsync envoie les fichiers sans utiliser l'algorithme
incrémental).
  Rsync créera REP s'il est manquant (uniquement le répertoire feuille).
Cela rend facile l'utilisation d'un chemin relatif (tel que «--partial-
dir=.rsync-partial»), ce qui fait que rsync crée le répertoire partiel dans
le répertoire du fichier de destination si nécessaire, et l'efface une fois
que le fichier partiel est supprimé.
  Si la valeur du répertoire partiel n'est pas un chemin absolu, rsync
```

ajoutera aussi une option --exclude à la fin des exclusions existantes avec comme valeur ce répertoire. Ceci évite que les répertoires partiels ne soient transmis et évite les suppressions par inadvertance des répertoires partiels chez le destinataire. Par exemple, l'option --partial-dir précédemment mentionnée ajoutera une règle «--exclude=.rsync-partial/» à la fin des autres règles de filtre. Remarquez que si vous spécifiez vos propres règles de filtre, il se peut que vous deviez manuellement insérer cette règle d'exclusion quelque part plus tôt dans la liste afin qu'elle ait une priorité suffisante pour être efficace (par ex : si vous avez à la fin une règle --exclude='*', la règle automatique ne sera jamais considérée).

IMPORTANT : Le répertoire partiel ne doit jamais être autorisé en écriture aux autres utilisateurs, cela introduirait un risque de sécurité. Par exemple, ÉVITEZ «/tmp».

Vous pouvez également spécifier la valeur partial-dir dans la variable d'environnement RSYNC_PARTIAL_DIR. Le fait que cette variable soit définie ne force pas l'activation de --partial, mais ceci indique où seront placées les fichiers partiels si --partial est activée. Par exemple, au lieu d'utiliser --partial-dir=.rsync-tmp avec l'option --progress, vous pouvez définir RSYNC_PARTIAL_DIR=.rsync-tmp dans votre environnement et puis juste utiliser l'option -P pour activer l'utilisation du répertoire .rsync-tmp pour les transferts partiels. Les seuls cas où --partial n'utilise pas cette variable d'environnement sont (1) quand --inplace est spécifiée (car --inplace rentre en conflit avec --partial-dir), et (2) quand --delay-updates est spécifiée (voir ci-dessous).

Dans le contexte de configuration du démon, pour la spécification du «refus d'option», --partial-dir n'implique pas --partial. De cette manière, refuser l'option --partial peut être utile lors de transferts partiels pour empêcher d'écrire par dessus des fichiers de destination, tout en autorisant quand même la version plus sûre --partial-dir.

–delay-updates

Cette option indique à rsync de placer les versions temporaires des fichiers mis à jour dans un répertoire séparé jusqu'à la fin du transfert, ces fichiers seront ensuite renommés avec leur véritable nom. L'objectif est de rendre la mise à jour un petit plus atomique. Par défaut, les fichiers sont placés dans le sous-répertoire «.~tmp~» de chaque répertoire de destination, mais vous pouvez le redéfinir avec l'option --partial-dir. (Notez que RSYNC_PARTIAL_DIR n'a aucun effet sur cette valeur, de même, lorsque --partial-dir est implicitement utilisée pour la spécification du «refus d'option», elle n'a pas d'effet sur ce répertoire.) Cette option rentre en conflit avec --inplace.

Cette option utilise plus de mémoire du côté récepteur (un bit par fichier transféré) et aussi requiert un espace disque suffisant chez le récepteur pour pouvoir contenir les copies supplémentaires de tous les fichiers mis à jour. Il faut aussi éviter d'utiliser un chemin absolu pour --partial-dir à moins d'être sûr de ne jamais avoir deux fichiers ayant le même nom (car, si le chemin est absolu, tous le fichiers mis à jour seront placé dans le même répertoire).

Voir aussi le script perl «atomic-rsync» dans le sous-répertoire «support» pour un algorithme de mis à jour encore plus atomique (il utilise --link-

```
dest avec une hiérarchie de fichier parallèle).
```

–progress

```
  Cette option demande à rsync d'afficher des informations montrant la
progression des transferts. Ceci donne à un utilisateur qui s'ennuie quelque
chose à regarder. Implique l'option --verbose lorsqu'elle n'a pas déjà été
spécifiée.
  Au cours du transfert d'un fichier, les données ressemblent à :
```

```
        782448  63%  110.64kB/s    0:00:04
```

```
  Ceci indique la taille courante du fichier, le pourcentage du transfert
effectué, la vitesse de transmission actuelle (prenant en compte à la fois
les données transmises et les données réutilisées localement), et une
estimation du temps restant avant la fin du transfert.
  Une fois qu'un fichier est entièrement transmit, les données ressemblent à
:
```

```
       1238099 100%  146.38kB/s    0:00:08  (5, 57.1% of 396)
```

```
  Ceci indique la taille finale du fichier, qu'il a été complété (100%), la
vitesse moyenne de transmission du fichier, la durée prise par le transfert
du fichier, et entre parenthèses, un résumé du transfert global. Ces trois
nombres vous indiquent combien de fichiers ont été mis à jour jusqu'à
présent, et quel pourcentage du nombre total de fichiers a été parcouru.
```

-P

```
  L'option -P est équivalente à --partial --progress. Elle a pour but de
rendre plus simple l'activation de ces deux options, couramment appelées
pour de longs transferts qui peuvent subir des coupures.
```

–password-file

```
  Cette option vous permet de fournir un mot de passe dans un fichier pour
accéder à un démon rsync distant. Notez que cette option est utile
uniquement pour accéder à un démon rsync en utilisant le transport interne,
pas lors de l'utilisation d'un shell distant comme transport. Le fichier ne
doit pas être lisible par tout le monde. Il doit contenir juste le mot de
passe sur une ligne seule.
```

–list-only

```
  Cette option liste les fichiers sources au lieu de les transférer. Cette
option est implicite s'il n'y a pas de destination de spécifiée, par
conséquent il n'est généralement pas nécessaire de l'utiliser explicitement.
Cependant, elle peut être utile pour les utilisateurs qui souhaitent éviter
les options «-r --exclude='/*/*'» que rsync utilise normalement lors de
listages non-récursifs pour assurer la compatibilité, ou pour lister les
```

fichiers qui sont impliqués lors d'une copie locale (puisque le répertoire destination n'est pas optionnel pour la copie locale, vous devez spécifier cette option explicitement et aussi indiquer la destination).

–bwlimit=KBPS

Cette option vous permet de spécifier un taux de transfert maximum en kilo-octets par secondes. Cette option est plus efficace lorsqu'utilisée avec de gros fichiers (plusieurs mega-octets et plus). Étant donnée la nature des transferts rsync, des blocs de données sont envoyés, puis si rsync détermine que le transfert a été trop rapide, il va attendre avant d'envoyer le bloc de données suivant. Le résultat est un taux de transfert moyen égal à la limite spécifiée. Une valeur de zéro signifie qu'il n'y a pas de limitation.

–write-batch=FICHIER

Génère un fichier qui pourra être appliqué à une autre destination identique à l'aide de --read-batch. Voir aussi la section MODE TRAITEMENT PAR LOTS pour plus de détails, ainsi que l'option --only-write-batch.

–only-write-batch=FICHIER

Fonctionne de la même manière que --write-batch, à l'exception qu'aucune mise à jour ne sera effectuée chez le récepteur lors de la création du fichier de traitement par lot. Cela vous permet d'envoyer les modifications sur le récepteur par d'autres moyens, et appliquer les changements via --read-batch.

Remarquez que vous êtes libre d'enregistrer ce fichier de traitement directement sur un média portable : si ce média est remplit avant le fin du transfert, vous pouvez juste appliquer ce transfert partiel sur la destination et répéter le processus jusqu'à ce que la totalité des changements a été appliquée (tant que ce n'est pas gênant qu'il y ait un fichier partiellement mis à jour pendant toute la durée du cycle).

Notez également que cela n'économise la bande passante que lorsque les changements sont envoyés vers un système distant car les données de mis à jour sont directement déviées vers le fichier de traitement par lot au lieu d'être transmises sur le réseau au récepteur (lors d'une réception, l'émetteur est distant et donc il ne peut pas écrire le fichier de traitement par lot).

–read-batch=FICHIER

Applique tout les changements enregistrés dans FICHIER, un fichier précédemment généré à l'aide de --write-batch. Si FICHIER est «-» alors le lot à traiter sera lu depuis l'entrée standard. Voir la section MODE TRAITEMENT PAR LOTS pour plus de détails.

–protocol=NUM

```
  Force l'utilisation d'un protocole de version antérieure. Cela est utile
pour créer un fichier de traitement par lot qui soit compatible avec une
version antérieure de rsync. Par exemple, si rsync 2.6.4 est utilisé avec
l'option --write-batch mais que ce sera rsync 2.6.3 qui sera utilisé pour
exécuter l'option --read-batch, alors vous devrez utiliser «--protocol=28»
lors de la création du fichier de traitement par lot pour forcer
l'utilisation d'un protocole antérieur dans le fichier de traitement par lot
(si vous ne pouvez pas mettre à jour rsync sur la seconde machine).
```

-4, –ipv4 ou -6, –ipv6

```
  Indique à rsync de préférer IPv4/IPv6 lors de la création de sockets. Ceci
n'a d'influence uniquement sur les sockets directement contrôlée par rsync,
telles que la socket sortante lors d'une connexion directe vers un démon
rsync. Voir aussi ces options dans la section du mode --daemon.
```

–checksum-seed=NUM

```
  Initialise la graine de la somme de contrôle MD4 à NUM. Cette graine de 4
octets est utilisée dans le calcul de la somme de contrôle MD4 pour chaque
bloc et chaque fichier. Par défaut cette graine est générée par le démon et
initialisée au time() courant. Cette option permet de choisir une graine
spécifique, ce qui peut être utile pour les applications qui nécessitent une
somme de contrôle de bloc ou de fichier reproductible, ou alors lorsque
l'utilisateur souhaite une graine de somme de contrôle plus aléatoire. Notez
que définir NUM à 0 forcera rsync à utiliser la graine par défaut, time().
```

Options du démon

Les options permises lorsque rsync est lancé comme démon sont les suivantes :

–daemon

```
  Ceci dit à rsync de fonctionner en tant que démon. On y accède en
utilisant la syntaxe host::module ou rsync://host/module.
  Si l'entrée standard est une socket, rsync va supposer qu'il est lancé par
inetd, sinon il va se détacher du terminal courant et va devenir un démon
tournant en arrière plan. Le démon lit le fichier de configuration
(rsyncd.conf) à chaque connexion faite par un client et répondre aux
requêtes en conséquence. Consultez la page de manuel rsyncd.conf(5) pour
plus de détails.
```

–address

```
  Par défaut, rsync écoutera sur l'adresse joker (wildcard address)
lorsqu'il est exécuté en tant que démon (avec l'option --daemon). L'option -
-address vous permet de spécifier une adresse IP particulière (ou un nom
d'hôte) à laquelle se connecter. Ceci permet de définir des hôtes virtuels
```

(virtual hosting) en conjonction avec l'option --config. Voir aussi l'option globale «address» de la page de manuel rsyncd.conf.

–bwlimit=KBPS

Cette option vous permet de spécifier un taux de transfert maximum en kilo-octets par secondes pour les données transmises par le démon. Le client peu spécifier une plus petite valeur --bwlimit, par contre sa valeur sera tronquée si elle dépasse la valeur du démon. Pour de plus amples détails, voir ci-dessus la définition de cette option pour le client.

–config=FICHIER

Spécifie un fichier de configuration alternatif. Ceci n'a de sens que si l'option --daemon est spécifiée. Par défaut le fichier est /etc/rsyncd.conf, à moins que le démon ne soit exécuté via un shell distant et que l'utilisateur distant ne soit pas root, auquel cas le fichier par défaut sera rsyncd.conf du répertoire courant (en général $HOME).

–no-detach

Lorsque rsync fonctionne en tant que démon, cette option lui demande de ne pas se détacher pour devenir un processus fonctionnant en arrière-plan. Cette option est nécessaire lorsque rsync est exécuté en tant que service avec Cygwin, et peut également être nécessaire lorsque rsync est supervisé par un programme comme daemontools ou le Contrôleur de Ressources Système de AIX. --no-detach est également recommandé lorsque rsync est exécuté dans un débogueur. Cette option n'a pas d'effet si rsync est exécuté depuis inetd ou sshd.

–port=PORT

Spécifie un numéro de port TCP alternatif à utiliser au lieu du port par défaut 873. Voir aussi l'option globale «port» de la page de manuel rsyncd.conf.

-v, –verbose

Cette option augmente le niveau d'information journalisé par le démon lors de la phase de démarrage. Un fois le client connecté, le niveau d'information rapporté par le client sera fonction des options passées par le client et l'option «max verbosity» dans la section de configuration des modules.

-4, –ipv4 or -6, –ipv6

Indique à rsync de préférer IPv4/IPv6 lors de la création de sockets d'entrée utilisées par le démon pour attendre les connexions. Il se peut qu'une de ces options soit nécessaire pour contourner un bogue d'IPv6 dans d'ancienne version du noyau Linux (si vous obtenez une erreur «address

```
already in use» alors que personne d'autre n'utilise le port, essayez de
spécifier --ipv6 ou --ipv4 au démarrage du démon).
```

-h, –help

```
  Lorsqu'elle est spécifiée après --daemon, cette option affiche une page
d'aide décrivant les options disponibles au démarrage du démon rsync.
```

Règles de filtre

Les règles de filtre permettent une sélection flexible des fichiers à transférer (inclus) et des fichiers à ignorer (exclus). Les règles peuvent définir directement des motifs d'inclusion/exclusion; elles peuvent aussi définir un moyen d'acquérir plus de motifs d'inclusion/exclusion (c-à-d de les lire depuis un fichier).

Au fur et à mesure que la liste des fichiers/répertoires à transférer est construite, rsync compare chaque nom avec la liste des motifs à inclure/exclure. À la première occurrence d'un motif qui correspond, si c'est un motif d'exclusion alors le fichier est ignoré ; si c'est un motif d'inclusion ou si aucun motif ne correspond alors le fichier est transmit.

Rsync construit la liste de règles de filtre dans le même ordre que spécifié sur la ligne de commande. Les règles de filtre ont la syntaxe suivante :

```
  RÈGLE [MOTIF_OU_FICHIER]
  RÈGLE,MODIFICATEURS [MOTIF_OU_FICHIER]
```

Vous avez le choix d'utiliser des noms de RÈGLE longs ou courts, comme décrit ci-dessous. Si vous utilisez une règle avec un nom court, la «,» séparant la RÈGLE du MODIFICATEURS est optionnelle. Le MOTIF ou FICHIER qui suit (lorsqu'il est présent) doit être séparé par un espace ou un tiret bas (_). Les préfixes de règle disponibles sont :

```
  exclude, - spécifie un motif d'exclusion.
  include, + spécifie un motif d'inclusion.
  merge, . spécifie un fichier à fusionner qui contient d'autres règles.
  dir-merge, : spécifie un fichier à fusionner par-répertoire.
  hide, H spécifie un motif pour cacher les fichiers lors du transfert.
  show, S les fichiers correspondants à ce motif ne sont pas cachés.
  protect, P spécifie un motif pour protégé les fichiers de la suppression.
  risk, R les fichiers correspondants à ce motif ne sont pas protégés.
  clear, ! efface la liste d'inclusion/exclusion courante (ne prend pas
d'argument).
```

Quand les règles sont lues depuis un fichier, les lignes vides sont ignorées, ainsi que les lignes de commentaire (celles commençant pas un «#»).

Remarquez que les options –include/–exclude en ligne de commande ne comprennent pas l'ensemble complet de règle décrit ci-dessus. Elles ne permettent uniquement la spécification de motifs d'inclusion/exclusion plus un «!» pour effacer la liste (ainsi que les commentaires lorsque les règles sont lues depuis un fichier). Si un motif ne commence ni par «- » (moins, espace), ni par «+ » (plus,

espace), alors la règle sera interprétée comme si un «+ » (pour une option d'inclusion) ou un «- » (pour une option d'exclusion) était préfixée à la chaîne de caractères. Par contre, pour l'option –filter, il doit toujours y avoir un nom de règle court ou long en début de règle.

Remarquez aussi que les options –filter, –include, et –exclude ne prennent chacune qu'une seule règle. Pour en spécifier plus d'une, répétez l'option sur la ligne de commande, utilisez la syntaxe de fusion de fichier de l'option –filter, ou utilisez les options –include-from/ –exclude-from.

Motifs d'inclusion et d'exclusion

Vous pouvez inclure et exclure des fichiers en spécifiant des motifs à l'aide des règles de filtre «+», «-», etc. (tel que présenté dans la section RÈGLES DE FILTRE ci-dessus). Chacune des règles peut définir un motif qui sera comparé à chacun des fichiers transférables. Les motifs peuvent prendre plusieurs formes :

o

```
  si le motif commence par un /, il est vérifié pour correspondance à une
place donnée dans la arborescence de fichiers, sinon il est vérifié avec la
fin du nom de fichier. Cela est équivalent à un ^ comme premier caractère
dans une expression régulière. Ainsi, «/foo» correspond avec un fichier
nommé «foo» à la base de l'arborescence de transfert (lors d'une règle
globale) ou dans le répertoire du fichier fusionné (lors d'une règle par
répertoire). Un simple «foo» correspond avec tous les fichiers appelés «foo»
n'importe où dans l'arborescence parce que l'algorithme est appliqué
récursivement de haut en bas ; il se comporte comme si chaque composante de
chemin était à son tour la fin du nom de fichier. De même, un «sub/foo» non
préfixé correspond à n'importe où dans l'arborescence où il y a un «foo»
dans un répertoire «sub». Voir la section ANCRER DES MOTIFS
D'INCLUSION/EXCLUSION pour les détails sur la manière de spécifier des
motifs correspondant à la racine du transfert.
```

o

```
  si le motif finit par un /, il va uniquement correspondre à un répertoire,
pas à un fichier, ni un lien ni un périphérique.
```

o

```
  si le motif contient un méta-caractère parmi les caractères *?[, la
correspondance d'expression est appliquée en utilisant les règles de
correspondance du shell. Sinon une simple correspondance de chaîne de
caractères est utilisée.
```

o

```
  le motif double astérisque «**» correspond à tous les caractères, y
comprit les barres obliques, tandis que le motif astérisque «*» s'arrête aux
barres obliques.
```

o

```
  si le motif contient un / (à l'exclusion d'un / final) ou un «**» alors il
est comparé au nom de fichier complet, y compris les répertoires précédant
le nom. Si le motif ne contient pas de / ni de «**» alors il n'est comparé
qu'à la dernière composante du nom de fichier. Encore une fois, souvenez-
vous que l'algorithme est appliqué récursivement. Ainsi, «nom complet» peut
être une portion quelconque d'un chemin.
```

Notez que, lors de l'utilisation de l'option –recursive (-r) (qui est implicite avec -a), chaque sous-composant de chaque chemin est visité récursivement, donc les motifs d'inclusion/exclusion sont appliqués récursivement au nom complet de chaque sous-composant (par exemple, pour inclure «/foo/bar/baz» les sous-composants «/foo» et «/foo/bar» ne doivent pas être exclus). En fait, les motifs d'exclusion court-circuitent l'étape de descente de répertoire lorsque rsync recherche les fichiers à transmettre. Si un motif exclut un répertoire parent en particulier, cela peut rendre l'inclusion d'un sous-répertoire inopérante car rsync ne descendra pas dans cette hiérarchie exclue par le motif. Ceci est particulièrement important lors de l'utilisation d'une règle «*». Par exemple, :

```
  + /un/chemin/ce-fichier-ne-sera-pas-trouvé
  + /fichier-inclut
  - *
```

Cela ne fonctionne pas car le répertoire parent «un» est exclu par la règle «*», donc rsync ne découvrira jamais les fichiers présents dans les répertoires «un» ou «un/chemin». Une solution est de demander d'inclure tous les répertoires en utilisant la règle «+ */» (placez la quelque part avant la règle «- *»). Une autre solution est d'ajouter des règles d'inclusion spécifiques pour tous les répertoires parents qui nécessitent d'être visités. Par exemple, ce jeu de règles fonctionne correctement :

```
  + /un/
  + /un/chemin/
  + /un/chemin/ce-fichier-est-trouvé
  + /fichier-également-inclut
  - *
```

Voici quelques exemples d'exclusion/inclusion :

o

```
  «- *.o» exclut tous les fichiers correspondant à *.o
```

o

```
  «- /foo» exclut un fichier du répertoire de base nommé foo
```

o

```
  «- foo/» exclut tous les répertoires nommés foo
```

o

```
  «- /foo/*/bar» exclut tous les fichiers nommés bar dans tous les
répertoires situés deux niveaux au-dessous du répertoire nommé foo dans le
répertoire de base.
```

o

```
  «- /foo/**/bar» exclut tous les fichiers nommés bar dans tous les
répertoires situés deux niveaux ou plus au-dessous du répertoire nommé foo
dans le répertoire de base.
```

o

```
  La combinaison de «+ */», «+ *.c», et «- *» inclut tous les répertoires et
fichiers sources C et rien d'autre
```

o

```
  La combinaison de «+ foo/», «+ foo/bar.c», et «- *» inclut uniquement le
répertoire foo et foo/bar.c (le répertoire foo doit être inclus
explicitement sinon il est exclu par le «*»)
```

Règles de filtre de fusion de fichier

Il est possible de fusionner des fichiers entiers dans vos règles de filtre en spécifiant une règle de filtre de type merge (.) ou dir-merge (:) (comme présenté dans la section RÈGLES DE FILTRE ci-dessus).

Il y a deux sortes de fusion de fichier – une seule fois («.») et par répertoire («:»). Lors de la fusion une seule fois, le fichier est lu une seule fois, et ses règles sont incorporées dans la liste de filtre à la place de la règle «.». Lors de la fusion par-répertoire, rsync recherchera dans chaque répertoire qu'il traverse un fichier avec le nom donné. Si le fichier existe, il fusionnera le contenu dans la liste de filtre héritée en cours. Ces fichiers de règles par-répertoire doivent être créés du côté émetteur car c'est ce côté qui est parcouru pour trouver les fichiers à transmettre. Ces fichiers de règles peuvent également avoir besoin d'être transmis chez le récepteur si vous souhaitez qu'ils influent également sur les fichiers ne devant pas être effacés (voir RÈGLES PAR RÉPERTOIRE ET SUPPRESSION ci-dessous).

Quelques exemples :

```
  merge /etc/rsync/default.rules
  . /etc/rsync/default.rules
  dir-merge .filtre-par-répertoire
  dir-merge,n- .exclusions-non-hérités-par-répertoire
  :n- .exclusions-non-hérités-par-répertoire
```

Les modificateurs suivants sont acceptés après une règle merge ou dir-merge :

o

```
  Un - spécifie que le fichier ne doit consister qu'en des motifs
d'exclusion, sans autres types de règles que les commentaires.
```

o

```
  Un + spécifie que le fichier ne doit consister qu'en des motifs
d'inclusion, sans autres types de règles que les commentaires.
```

o

```
  Un C est une manière de spécifier que le fichier doit être lu d'une
manière compatible avec CVS. Cela active «n», «w», et «-», et autorise aussi
le caractère d'effacement de liste (!). Si aucun fichier n'est donné,
«.cvsignore» est lu.
```

o

```
  Un e exclura le nom du fichier fusionné du transfert ; par exemple «dir-
merge,e .rules» est identique à «dir-merge .rules» et «- .rules».
```

o

```
  Un n spécifie que les règles ne sont pas héritées par les sous-
répertoires.
```

o

```
  Un w spécifie que les règles sont séparés par des caractères blancs au
lieu d'être séparés par des retours à la ligne. [NDT : w pour whitespace].
En plus, ceci désactive les commentaires. Note : l'espace qui sépare le
préfixe de la règle est traité spécialement, donc «- foo + bar» est
interprété comme deux règles (en considérant que l'interprétation de préfixe
n'a pas été désactivée).
```

o

```
  Vous pouvez aussi spécifier les modificateurs pour les règles «+» et «-»
(ci-dessous) afin que les règles qui sont lues depuis le fichier aient par
défaut ce modificateur déjà positionné. Par exemple, «merge,-/ .excl»
traitera le contenu de .excl comme des exclusions en chemin absolu, tandis
que «dir-merge,s .filt» et «:sC» ne verrons chacun leurs règles par-
répertoire appliquées que du côté émetteur.
```

Les modificateurs suivants sont acceptés après un «+» ou un «-» :

o

```
  Un «/» spécifie que les inclusions/exclusions doivent être traitées comme
des chemins absolus, relatifs à la racine du système de fichiers. Par
```

```
exemple, «-/ /etc/passwd» exclura le fichier passwd à chaque transfert du
répertoire «/etc».
```

o

```
  Un «!» spécifie que l'inclusion/exclusion prend effet si le motif ne
correspond pas. Par exemple, «-! */ » exclura tout ce qui n'est pas un
répertoire.
```

o

```
  Un C spécifie que toutes les règles d'exclusion CVS doivent être insérées
comme exclusion là où se trouve le «-C». Aucun argument ne doit suivre.
```

o

```
  Un s spécifie que la règle ne s'applique qu'au côté émetteur (s comme
sender). Lorsqu'une règle affecte le côté émetteur, elle empêche certains
fichiers d'être transférés. Par défaut, une règle affecte les deux côtés à
moins que --delete-excluded ne soit spécifiée, auquel cas par défaut les
règles n'affectent que l'émetteur. Voir aussi les règles hide (H) et show
(S), qui sont une autre manière de spécifier des inclusions/exclusions
uniquement pour l'émetteur.
```

o

```
  Un r spécifie que la règle ne s'applique qu'au côté récepteur (r comme
receiver). Lorsqu'une règle affecte le côté récepteur, elle empêche les
fichiers d'être supprimés. Voir le modificateur s pour plus d'information.
Voir aussi les règles protect (P) et risk (R), qui sont une autre manière de
spécifier des inclusions/exclusions uniquement pour le récepteur.
```

Les règles par-répertoire sont héritées dans tous les sous-répertoires du répertoire ou le fichier fusionné a été trouvé à moins que le modificateur «n» n'est été utilisé. Chacune des règles des sous-répertoires sont préfixées aux règles par-répertoires héritées de leurs parents, ce qui donne une priorité plus élevée aux nouvelles règles par rapport aux règles héritées. Tout l'ensemble des règles de fusion de répertoire est groupé à l'emplacement où le fichier fusionné était spécifié, donc il est possible de remplacer une règle de fusion de répertoire par une règle spécifiée avant dans la liste des règles globales. Lorsque la règle d'effacement de liste (!) est lue depuis un fichier par-répertoire, elle n'efface que les règles héritées du fichier actuellement fusionné.

Un autre moyen d'éviter une simple règle d'un fichier de fusion de répertoire d'être héritée est de l'ancrer à l'aide d'une barre oblique au début. Les règles ancrées dans un fichier fusionné par-répertoire sont relatives au répertoire du fichier fusionné, donc un motif «/foo» ne correspondra qu'avec le fichier «foo» dans le répertoire où le fichier du filtre de fusion de répertoire a été trouvé.

Voici un exemple de fichier filtre que vous pouvez spécifier via –filter=“. fichier”:

```
  merge /home/util/.filtre-global
  - *.gz
  dir-merge .règles
```

```
+ *.[ch]
- *.o
```

Cela fusionnera le contenu du fichier /home/util/.filtre-global au début de la liste et indique que le nom de fichier «.règles» correspond à des fichiers filtres par-répertoire. Toutes les règles lues avant le début du balayage du répertoire suivent les règles globales ancrées (c-à-d qu'une barre oblique correspond à la racine du transfert).

Si un fichier fusionné par-répertoire est spécifié avec un chemin qui est un répertoire parent du premier répertoire transféré, rsync balayera tous les répertoires parents depuis ce point de départ jusqu'au répertoire de transfert pour trouver le fichier par-répertoire indiqué. Par exemples, voici un filtre habituel (voir -F) :

1. -filter=': /.rsync-filter'

Cette règle indique à rsync de balayer tous les répertoires depuis la racine jusqu'au répertoire parent du transfert pour trouver .rsync-filter avant de débuter le balayage normal des fichiers contenus dans les répertoires. (Note : pour un démon rsync, la racine est toujours identique au «chemin» du module).)

Quelques exemples de ce pré-balayage pour trouver des fichiers par-répertoire :

```
rsync -avF /src/chemin/ /dest/rép
rsync -av --filter=': ../../.rsync-filter' /src/chemin/ /dest/rép
rsync -av --filter=': .rsync-filter' /src/chemin/ /dest/rép
```

Les deux premières commandes ci-dessus rechercherons «.rsync-filter» dans «/ » et «/src » avant que le balayage normal ne commence à chercher des fichiers dans «/src/chemin/» et ses sous-répertoires. La dernière commande évite le balayage des répertoires parents et ne cherchera les fichiers «.rsync-filter» que dans les répertoires qui font partie du transfert.

Si vous souhaitez inclure le contenu d'un «.cvsignore» dans vos motifs, il est avantageux d'utiliser la règle «:C», elle crée une fusion par répertoire du fichier .cvsignore mais parsé de manière compatible avec CVS. Vous pouvez l'utiliser pour indiquer où l'option d'inclusion par répertoire du fichier .cvsignore –cvs-exclude (-C) sera placée parmi les règles en indiquant «:C» là où vous le souhaitez. Sans cela, rsync ajoutera la fusion par-répertoire du fichier .cvsignore après toutes les autres règles (en lui donnant une priorité inférieure aux règles sur la ligne de commande). Par exemple :

```
cat <<EOT | rsync -avC --filter='. -' a/ b
+ foo.o
:C
- *.old
EOT
rsync -avC --include=foo.o -f :C --exclude='*.old' a/ b
```

Les deux commandes ci-dessus sont identiques. Chacune fusionnera toutes les règles .cvsignore (et qui sont par répertoire) au milieu de la liste plutôt qu'à la fin. Cela permet aux règles spécifiques au répertoire d'être prioritaires sur les règles déclarées après le :C, au lieu d'être soumises l'ensemble des règles. Pour affecter les autres règles d'exclusion CVS (c'est-à-dire la liste par défaut d'exclusion, le contenu de $HOME/.cvsignore, et la valeur de $CVSIGNORE) vous devez omettre l'option de ligne de commande -C et à la place insérer une règle «-C» dans les règles de filtres ; par exemple «–filter=-

C».

Règle de filtre d'effacement de la liste

Il est possible d'effacer la liste courante d'inclusion/exclusion en ajoutant la règles de filtre «!» (comme introduit dans la section RÈGLES DE FILTRE ci-dessus). La liste «courante» est soit la liste globale de règles (si la règle est lue lorsque les options de filtre sont parsées), soit un ensemble de règles par-répertoire qui sont héritées dans leurs sous-listes respectives (cela permet à un sous-répertoire d'effacer les règles de son parent).

Motifs d'ancrage d'inclusion/exclusion

Comme décrit précédemment, les motifs globaux d'inclusion/exclusion sont ancrés «à la racine du transfert» (en opposition aux motifs par-répertoire qui sont ancrés au répertoire du fichier fusionné). En imaginant le transfert comme un sous-arbre de noms qui sont envoyés de l'émetteur au récepteur, la racine du transfert est là où l'arbre commence a être dupliqué dans le répertoire de destination. Cette racine fixe à partir d'où les motifs commençant par un / correspondent.

Étant donné que les correspondances sont relatives à la racine du transfert, la modification de la barre oblique en début de chemin de source ou l'usage de l'option –relative affecte le chemin utilisé pour les correspondances (en plus des modifications concernant l'arbre de fichier à dupliquer sur l'hôte destination). Les exemples suivants démontrent cela.

Supposons que l'on souhaite faire correspondre deux fichiers sources, l'un avec comme chemin absolu «/home/me/foo/bar», l'autre avec comme chemin «/home/you/bar/baz». Voici les différentes commandes possibles pour un transfert avec deux sources :

```
Commande : rsync -a /home/me /home/you /dest
Motif +/- : /me/foo/bar
Motif +/- : /you/bar/baz
Fichier cible : /dest/me/foo/bar
Fichier cible : /dest/you/bar/baz
```

```
Commande : rsync -a /home/me/ /home/you/ /dest
Motif +/- : /foo/bar (noter le «me» manquant)
Motif +/- : /bar/baz (noter le «you» manquant)
Fichier cible : /dest/foo/bar
Fichier cible : /dest/bar/baz
```

```
Commande : rsync -a --relative /home/me/ /home/you /dest
Motif +/- : /home/me/foo/bar (noter le chemin absolu)
Motif +/- : /home/you/bar/baz (idem)
Fichier cible : /dest/home/me/foo/bar
Fichier cible : /dest/home/you/bar/baz
```

```
Commande : cd /home; rsync -a --relative me/foo you/ /dest
Motif +/- : /me/foo/bar (commence depuis un chemin donné)
```

```
Motif +/- : /you/bar/baz (idem)
Fichier cible : /dest/me/foo/bar
Fichier cible : /dest/you/bar/baz
```

La manière la plus simple de voir quel nom vous devriez filtrer est de simplement regarder la sortie en utilisant –verbose et de rajouter un / au début de chaque nom (utilisez l'option –dry-run si vous n'êtes pas prêt à copier des fichiers).

Règles par répertoire et suppression

Sans option de suppression, les règles par-répertoire ne sont significatives que du côté émetteur, vous pouvez donc exclure les fichiers fusionnés sans affecter le transfert. Pour simplifier, le modificateur «e» ajoute cette exclusion pour vous, comme l'illustrent ces deux commandes équivalentes :

```
rsync -av --filter=': .excl' --exclude=.excl hote:src/dir /dest
rsync -av --filter=':e .excl' hote:src/dir /dest
```

Cependant, si vous souhaitez effectuer des suppressions du côté récepteur ET que vous voulez aussi que certains fichiers soient exclus des suppressions, vous devrez vous assurer que le côté récepteur sache quels fichiers supprimer. Le moyen le plus simple est d'inclure les fichiers fusionnés par-répertoire dans le transfert et d'utiliser –delete-after, car cela assurera que le côté récepteur a bien les mêmes règles d'exclusion que le côté émetteur avant qu'il ne tente de supprimer quoi que ce soit :

```
rsync -avF --delete-after hote:src/dir /dest
```

Cependant si les fichiers fusionnés ne font pas partie du transfert, vous devrez soit spécifier des règles globales d'exclusion (c'est-à-dire spécifiées en ligne de commande), soit maintenir vos propres fichiers fusionnés par-répertoire du côté récepteur. Voici un exemple de la première méthode (en supposant que les fichiers distants .règles s'excluent eux-mêmes) :

rsync -av –filter=': .rules' –filter='. /mes/extra.rules'

1. -delete hote:src/dir /dest

Dans l'exemple ci-dessus le fichier extra.rules peut affecter les deux côtés du transfert, mais (du côté émetteur) les règles sont moins prioritaires que les règles fusionnées depuis les fichiers .rules puisqu'elles sont spécifiées après la règle de fusion par-répertoire.

Dans un dernier exemple, le côté émetteur exclue les fichiers .rsync-filter du transfert, mais nous souhaitons utiliser nos propres fichiers .rsync-filter pour contrôler ce qui sera supprimé chez le récepteur. Pour ce faire, nous devons spécifiquement exclure les fichiers fusionnés par répertoire (afin qu'ils ne soient pas supprimés) et mettre les règles dans les fichiers locaux pour contrôler ce qui ne devra pas être supprimer. Cela pourra ressembler à cette commande :

```
rsync -av --filter=':e /.rsync-filter' --delete \
    hote:src/dir /dest
```

```
  rsync -avFF --delete hote:src/dir /dest
```

Mode traitement par lots

Le mode de traitement par lots est utilisé pour appliquer le même ensemble de modification à plusieurs systèmes identiques. Imaginons que l'on ait un arbre qui soit répliqué sur un certain nombre d'hôtes. Supposons maintenant que des modifications aient été apportées à cet arbre source et que les changements doivent être propagés aux autres hôtes. Afin d'utiliser le mode de traitement par lots, rsync est exécuté avec l'option write-batch pour appliquer les modifications faites à l'arbre source à l'un des arbres destinations. L'option write-batch indique au client rsync de sauvegarder dans un fichier de traitement par lot toutes les informations nécessaires pour reproduire l'opération avec les autres arbres, identiques, de destinations.

Pour appliquer les modifications sauvegardées à un autre arbre de destination, exécutez rsync avec l'option read-batch, en indiquant le nom du fichier de traitement par lot et l'arbre de destination. Rsync met à jour l'arbre de destination en utilisant les informations stockées dans le fichier de traitement par lot.

Par commodité, un second fichier est créé lorsque l'option write-batch est utilisée. Le nom de ce fichier est créé en ajoutant «.sh» au nom du fichier de traitement par lot. Ce fichier .sh contient une ligne de commande qui permet de mettre à jour l'arbre de destination en utilisant le fichier de traitement par lot. Il peut être exécuté en utilisant un shell Bourne (ou équivalent). Il est possible de passer en argument un chemin alternatif vers l'arbre de destination, qui sera alors utilisé à la place du chemin original. C'est utile lorsque le chemin de l'arbre de destination diffère du chemin de l'arbre de destination original.

Générer le fichier de traitement par lot permet d'éviter d'effectuer la sélection de fichier, la somme de contrôle et la génération des blocs de donnée plus d'une fois lors de la mise à jour de plusieurs arbres de destination. Un protocole de transport multicast peut permettre de transférer le fichier de traitement par lot en parallèle vers plusieurs hôtes en une seule fois, au lieu de transférer les mêmes données à chaque hôte individuellement.

Exemples :

```
  $ rsync --write-batch=foo -a hôte:/source/rep/ /dest-a/rep/
  $ scp foo* distant:
  $ ssh distant ./foo.sh /dest-b/rep/
```

```
  $ rsync --write-batch=foo -a /source/rep/ /dest-a/rep/
  $ ssh distant rsync --read-batch=- -a /dest-b/rep/ <foo
```

Dans ces exemples, rsync est utilisé pour mettre à jour /dest-a/rep/ depuis /source/rep/ et les informations pour répéter cette opération sont stockées dans «foo» et «foo.sh». L'hôte «distant» est ensuite mis à jour à l'aide des données dans le répertoire /dest-b/rep/. Les différences entre les deux exemples révèlent certaines des flexibilités qui vous sont offertes pour gérer le traitement par lot :

o

```
  Le premier exemple montre que la copie initiale n'a pas à être locale --
vous pouvez recevoir ou envoyer les données depuis/vers l'hôte distant en
```

```
utilisant soit la syntaxe du shell distant soit celle du démon rsync, à
votre convenance.
```

o

```
  Le premier exemple utilise le fichier créé « foo.sh» pour spécifier les
bonnes options rsync lors de l'exécution de la commande read-batch sur
l'hôte distant.
```

o

```
  Le deuxième exemple lit les données via l'entrée standard, permettant
ainsi d'éviter de copier le fichier de traitement par lot sur la machine
distante. Cet exemple interdit l'utilisation du script foo.sh car on a
besoin d'utiliser l'option --read-batch de manière différente, mais rien
n'empêche bien sûr de modifier le script foo.sh pour l'utiliser (faites
juste attention qu'aucune autre option ne tente d'utiliser l'entrée
standard, telle que l'option «--exclude-from=-».
```

Avertissements :

L'option read-batch s'attend à ce que l'arbre de destination qui est mis à jour soit identique à l'arbre de destination qui a été utilisé pour créer le fichier de traitement par lot. Lorsqu'une différence entre les arbres de destination est détectée, la mise à jour sera soit abandonnée avec un avertissement (si le fichier semble déjà à jour) soit elle sera tentée et alors, si cela ne fonctionne pas, la mise à jour sera annulée, accompagnée d'une erreur. Par conséquent, relancer une opération read-batch si la commande a été interrompue devrait être sûr. Si vous souhaitez forcer la mise à jour quelque soient la date ou la taille du fichier, utilisez l'option -I (lors de la lecture du fichier de traitement par lot). Si une erreur arrive, l'arbre de destination se retrouvera probablement dans un état de mise à jour partielle. Dans ce cas, rsync peut être utilisé de manière régulière (sans le mode de traitement par lot) pour corriger l'arbre destination.

Les versions de rsync utilisées sur toutes les destinations doivent être au moins aussi récentes que celle utilisée pour générer le fichier de traitement par lot. Rsync se terminera avec une erreur si la version du protocole dans le fichier de traitement par lot est trop récente lors de l'application. Voir aussi l'option –protocol pour un moyen de générer un fichier de traitement par lot pour une version de rsync plus ancienne. (Notez que le format de fichier de traitement par lot a changé dans la version 2.6.3, par conséquent mélanger des versions antérieures et postérieures ne fonctionnera pas.)

Lors de la lecture d'un fichier de traitement par lot, rsync force la valeur de certaines options afin de les ajuster vis à vis des options sélectionnées lors création de ce fichier. Certaines options peuvent (et doivent) être différentes. Par exemple –write-batch est changée en –read-batch, –files-from est supprimée et les options –filter/–include/–exclude ne sont plus nécessaires à moins qu'une des options –delete n'ait été aussi spécifiée.

Le code qui crée le fichier BATCH.sh transforme toutes les options de filtres/inclusions/exclusions en une simple liste qui est ajoutée comme un document en ligne à la fin du fichier script. Un utilisateur avancé peut utiliser cette caractéristique pour modifier la liste d'exclusion si un changement de ce qui doit être supprimé par –delete est souhaité. Un utilisateur normal peut ignorer ces subtilités et concidéré le script shell simplement comme un moyen facile de lancer la commande –read-batch appropriée pour les données de traitement par lot.

À l'origine, le mode de traitement par lot était basé sur «rsync+», les nouvelles versions utilisent une implémentation différente.

Liens symboliques

Trois comportements de base sont possibles lorsque rsync rencontre un lien symbolique dans le répertoire source.

Par défaut, les liens symboliques ne sont pas transférés. Un message «skipping non-regular file» (fichier non-régulier ignoré) est émis pour tous les liens symboliques existants.

Si –links est spécifié, alors les liens symboliques sont recrées avec la même cible. Notez que –archive implique –links.

Si –copy-links est spécifié, alors les liens symboliques sont «suivis» en les remplacant par le fichier vers lequel ils pointent.

De plus, rsync distingue les liens symboliques «sûrs» de ceux «non sûrs». Un exemple où ceci peut être utilisé est un miroir de site web qui veut s'assurer que le module rsync copié ne contient pas de liens symboliques vers /etc/passwd dans la section publique du site. L'utilisation de –copy-unsafe-links va copier tous les fichiers pointés par des liens sur la destination. L'utilisation de –safe-links empêche les liens non sûrs d'être copiés. (Notez que vous devez spécifier –links pour que –safe-links prenne effet.)

Les liens symboliques sont considérés non sûrs si ce sont des liens symboliques absolus (qui commencent par /), vides, ou s'ils contiennent suffisamment de «..» pour remonter au-dessus du répertoire copié.

Voici un résumé de l'interprétation des options de liens symboliques. Cette liste est classée par ordre de priorité, si votre combinaison d'options n'est pas mentionnée, utilisez la première ligne qui soient un sous-ensemble complet de vos options :

–copy-links

```
  Transforme tous les liens symboliques en fichiers réguliers (ne laissant
aucun lien symbolique aux autres options).
```

–links –copy-unsafe-links

```
  Transforme tous les liens symboliques non sûrs en fichiers et duplique
tous les liens symboliques sûrs.
```

–copy-unsafe-links

```
  Transforme tous les liens symboliques non sûrs en fichiers, et ignore les
liens symboliques sûrs en affichant un message.
```

–links –safe-links

```
  Duplique les liens symboliques sûrs et ignore ceux non sûrs.
```

–links

```
  Duplique tous les liens symboliques.
```

Diagnostics

rsync produit occasionnellement des messages d'erreur qui peuvent paraître un peu cryptiques. Celui qui semble causer le plus de confusion est «protocol version mismatch - is your shell clean?».

Ce message est en général causé par vos scripts de démarrage ou un utilitaire pour le shell distant produisant des données parasites sur le flux utilisé par rsync pour son transport. Le moyen de diagnostiquer ce problème est d'exécuter votre shell distant comme ceci :

```
  ssh hote-distant /bin/true > out.dat
```

ensuite examinez out.dat. Si tout fonctionne correctement, alors out.dat devrait être un fichier de taille zéro. Si vous obtenez de rsync l'erreur ci-dessus, alors vous allez probablement vous rendre compte que out.dat contient un peu de texte ou de données. Vérifiez le contenu et essayez de trouver ce qui le produit. La cause la plus courante est un script de démarrage du shell (comme .cshrc ou .profile) mal configuré contenant des instructions de sortie pour des logins non-interactifs.

Si vous avez des difficultés pour déboguer les motifs d'inclusion et d'exclusion, alors essayez de spécifier l'option -vv. À ce niveau de verbosité, rsync affichera la raison d'inclusion ou d'exclusion individuellement pour chaque fichier.

Valeurs de retour

0

```
  Succès
```

1

```
  Erreur de syntaxe ou d'utilisation
```

2

```
  Incompatibilité de protocole
```

3

```
  Erreurs lors de la sélection des fichiers et des répertoires
d'entrée/sortie
```

4

```
  Action non supportée : une tentative de manipulation de fichiers 64-bits
sur une plate-forme qui ne les supporte pas ; ou une option qui est
supportée par le client mais pas par le serveur.
```

5

```
  Erreur lors du démarrage du protocole client-serveur
```

6

```
  Démon incapable d'écrire dans le fichier de log
```

10

```
  Erreur dans la socket E/S
```

11

```
  Erreur d'E/S fichier
```

12

```
  Erreur dans le flux de donnée du protocole rsync
```

13

```
  Erreur avec les diagnostics du programme
```

14

```
  Erreur dans le code IPC
```

20

```
  SIGUSR1 ou SIGINT reçu
```

21

```
  Une erreur retournée par waitpid()
```

22

```
  Erreur lors de l'allocation des tampons de mémoire principaux
```

23

```
  Transfert partiel dût à une erreur
```

24

```
  Transfert partiel dût à la disparition d'un fichier source
```

25

```
  La limite --max-delete a été atteinte
```

30

```
  Dépassement du temps d'attente maximal lors d'envoi/réception de données
```

Variables d'environnement

CVSIGNORE

```
  La variable d'environnement CVSIGNORE permet d'ajouter des motifs
d'exclusion à ceux trouvés dans les fichiers .cvsignore. Voir l'option --
cvs-exclude pour plus de détails.
```

RSYNC_RSH

```
  La variable d'environnement RSYNC_RSH vous permet de passer outre la
commande utilisée par défaut comme transport pour rsync. Des options de
ligne de commande sont permises après le nom de la commande, identiquement à
l'option -e.
```

RSYNC_PROXY

```
  La variable d'environnement RSYNC_PROXY vous permet de rediriger votre
client rsync pour utiliser un proxy web lors de la connexion à un démon
rsync. RSYNC_PROXY doit être de la forme nom-hôte:port.
```

RSYNC_PASSWORD

```
  Fixer RSYNC_PASSWORD au mot de passe nécessaire vous permet d'exécuter des
connexions rsync authentifiées à un démon rsync sans intervention de
l'utilisateur. Notez que cela ne fournit pas de mot de passe à un shell de
transport comme ssh.
```

USER ou LOGNAME

```
  Les variables d'environnements USER et LOGNAME sont utilisées pour
déterminer le nom d'utilisateur par défaut envoyé à un démon rsync. Si
aucune n'est positionnée, le nom d'utilisateur par défaut est «nobody».
```

HOME

```
  La variable d'environnement HOME est utilisée pour trouver le fichier
.cvsignore par défaut de l'utilisateur.
```

Voir aussi

- http://www.delafond.org/traducmanfr/man/man1/rsync.1.html