Sincronizando carpetas con rsync

Los principios básicos de rsync

`rsync` es una herramienta de copiado muy versátil que viene incluida en casi todas las versiones de Linux. Puede ser utilizada como una herramienta avanzada de copiado, permitiéndonos copiar archivos localmente o remotamente. Además puede ser utilizada como una herramienta para respaldos, soporta la creación de respaldos incrementales.

`rsync` cuenta con un famoso famoso algoritmo delta-transfer que nos permite transferir archivos nuevos, así como los cambios recientes a archivos existentes, mientras ignora archivos que no han sido modificados. Adicionalmente a esto, el comportamiento de `rsync` puede ser personalizado, ayudándonos a automatizar los respaldos, así mismo puede ser utilizado como un servicio para convertir la computadora en un servidor y permitir que otros clientes de `rsync` se conecten a ella.

Además del copiado de archivos y carpetas locales, `rsync` nos permite copiar utilizando como medio SSH (Secure Shell), RSH (Remote Shell) y puede ejecutarse como servicio en una computadora y permitir a otras computadoras conectarse, cuando `rsync` es usado como servicio escucha el puerto 873.

Cuando utilizamos `rsync` como servicio o cuando utilizamos RSH, los datos enviados entre computadoras no viaja cifrado, por consiguiente, si estás transfiriendo archivos entre dos computadoras en la misma red local, esto es útil, pero esto no debe ser utilizado para transferir archivos en redes inseguras, como lo es Internet. Para este propósito SSH es lo mejor a usar.

Está es la principal razón por la que yo favorezco el uso de SSH para mis transferencias, además, gracias a que es seguro, muchos servidores tienen el servicio de SSH disponible. Pero el uso de `rsync` como servicio para transferencias en conexiones rápidas, como es usualmente el caso en redes locales, es útil. No tengo el servicio de RSH funcionando en mis computadoras por lo que podrías encontrar que favorezco SSH en los ejemplos. Los ejemplos utilizan SSH como el medio de transporte para las transferencias entre computadoras remotas, pero en una publicación separada cubro el uso de rsync como servicio.

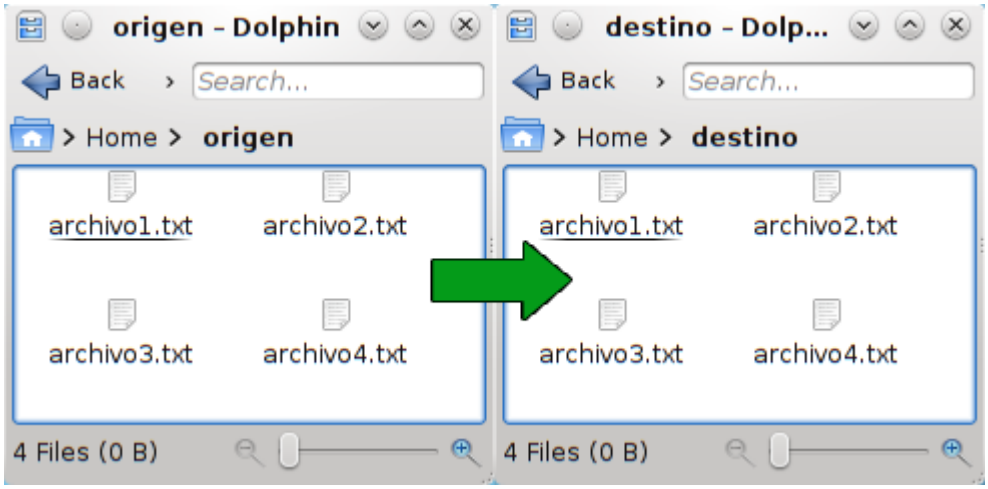
Copiando archivos y carpetas locales

Para copiar los contenidos de una carpeta en otra, reemplazando los archivos de la carpeta de destino, podemos utilizar:

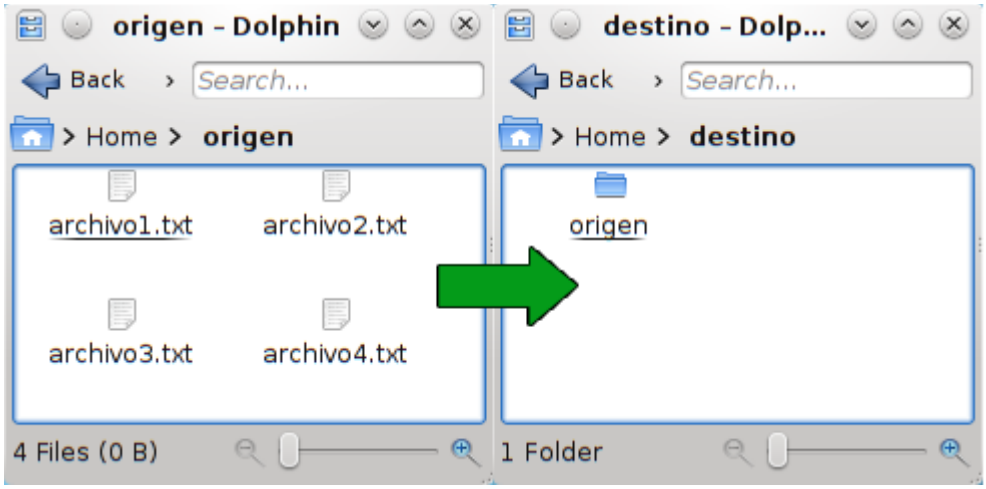
```
rsync -rtv carpeta_origen/ carpeta_destino/
```

Nota que en la carpeta_origen agregué una diagonal al final, hacer esto previene que una nueva carpeta sea creada, si no agregamos la diagonal, entonces una nueva carpeta llamada carpeta_origen será creada en carpeta_destino. Por consiguiente, si deseas copiar los contenidos de una carpeta llamada Imágenes dentro de una carpeta llamada Imágenes que ya existe, necesitas agregar la diagonal al final, de otra manera, una carpeta llamada Imágenes será creada dentro de la carpeta llamada Imágenes que especificamos como destino.

```
rsync -rtv origen/ destino/
```




```
rsync -rtv origen destino/
```



El parámetro `-r` significa que hará el copiado recursivo de carpetas, es decir, creará toda la estructura de las carpetas que hay dentro de carpeta_origen en carpeta_destino, y copiará todos los archivos que es encuentren dentro de estas.

El parámetro `-t` hace que `rsync` preserve los tiempos de modificación de los archivos que copia desde la carpeta origen.

El parámetro `-v` significa que la información imprimida durante la ejecución del programa será mucho más detallada, podemos utilizar esto para ver el progreso del comando.

Estos son los parámetros que utilizo frecuentemente, ya que usualmente estoy respaldando archivos personales y estos no contienen cosas como vínculos simbólicos, pero otro parámetro muy útil que se puede utilizar con `rsync` es el parámetro -a.

```
rsync -av origen/ destino/
```

El parámetro `-a` también hace la copia recursiva y preserva los tiempos de modificación, pero adicionalmente copia los vínculos simbólicos que encuentra como vínculos simbólicos, preserva los permisos, preserva la información de el dueño y el grupo del archivo, y preserva los archivos de dispositivo y los archivos especiales. Esto es útil si se está copiando completo el directorio personal de un usuario, o si se están copiando carpetas del sistema a otro lugar.

Lidiando con los espacios en blanco y demás caracteres raros

Podemos escapar espacios y caracteres raros al igual que en Bash, utilizando \ antes de cualquier espacio en blanco y carácter raro. Adicionalmente, podemos utilizar comillas sencillas para encerrar la cadena de texto:

```
rsync -rtv or\{ig\ en/ des\ ti\{no/
rsync -rtv 'or{ig en/' 'des ti{no/'
```

Actualizar los contenidos de una carpeta

Para ahorrar ancho de banda y tiempo, podemos evitar copiar archivos que ya tenemos en la carpeta de destino que no has sido modificados en la carpeta de origen. Para hacer esto agregamos el parámetro -u a `rsync`, esto actualizará los contenidos de la carpeta destino en base a la carpeta de origen, aquí es donde el algoritmo delta-transfer entra. Para actualizar los contenidos de una carpeta basados en los contenidos de otra utilizamos:

```
rsync -rtvu carpeta_origen/ carpeta_destino/
```

Por default, `rsync` toma la fecha de la última modificación del archivo y el tamaño de este para decidir que archivos necesitan ser transferidos y que archivos pueden ser ignorados, pero podemos utilizar en lugar de este método un hash para decidir si el archivo es diferente o no.

Para hacer esto necesitamos usar el parámetro `-c`, que realizará un checksum en los archivos a ser transferidos. Esto ignorará cualquier archivo donde el checksum coincide.

```
rsync -rvuc carpeta_origen/ carpeta_destino/
```

Sincronizando dos carpetas con rsync

Para mantener dos carpetas en sincronía, no solo necesitamos agregar los nuevos archivos en la carpeta de origen a la carpeta de destino, también necesitamos remover los archivos que fueron borrados de la carpeta origen en la carpeta destino. `rsync` nos permite hacer esto con el parámetro --delete, esto usado en conjunto con el parámetro previo -u que actualiza los archivos modificados nos permite mantener dos carpetas en sincronía ahorrando ancho de banda.

```
rsync -rtvu --delete carpeta_origen/ carpeta_destino/
```

El proceso de borrado se puede llevar antes, durante y después de la transferencia, podemos controlar en que momento ocurre utilizando alguno de los parámetros siguientes:

- `rsync` puede buscar los archivos faltantes y borrarlos antes de de el proceso de transferencia, este es el comportamiento por default y puede ser ajustado con `--delete-before`

- `rsync` puede buscar los archivos faltantes y borrarlos después que la transferencia se ha completado, con el parámetro `--delete-after`
- `rsync` puede borrar los archivos durante la transferencia, cuando se encuentra un archivo faltante, se borra en ese momento, activamos este comportamiento con `--delete-during`
- `rsync` puede hacer la transferencia y encontrar los archivos faltantes durante este proceso, esperar hasta que ha terminado y borrar los archivos que encontró después, esto puede lograrse con `--delete-delay`

Por ejemplo:

```
rsync -rtvu --delete-delay carpeta_origen/ carpeta_destino/
```

Comprimiendo los archivos antes de transferirlos

Para ahorrar algo de ancho de banda, y usualmente ahorrar algo de tiempo también, podemos comprimir la información que está siendo transferida, podemos agregar el parámetro `-z` a `rsync` para lograr esto.

```
rsync -rvz carpeta_origen/ carpeta_destino/
```

Si estamos transfiriendo una gran cantidad de archivos sobre una conexión rápida, `rsync` puede ser más lento con el parámetro `-z` que sin el, ya que tomará más tiempo comprimir cada archivo que el tiempo que se tomaría simplemente hacer la transferencia de los archivos a la carpeta_destino. Usa este parámetro si tienes una conexión con velocidad limitada entre dos computadoras, o si necesitas ahorrar ancho de banda.

Transfiriendo archivos entre dos sistemas remotos

`rsync` puede copiar archivos y sincronizar una carpeta local con una carpeta remota en un sistema que cuente con acceso vía SSH, vía RSH, o que esté ejecutando `rsync` como un servicio. Los ejemplos aquí utilizan SSH para la transferencia de archivos, pero los mismos principios aplican si quieres hacer esto con `rsync` funcionando como servicio en la computadora remota, lee Ejecutando rsync como servicio cuando SSH no está disponible más abajo para mayor información sobre esto.

Para transferir archivos entre la computadora local y una computadora remota, necesitamos especificar la dirección de la computadora remota, puede ser un nombre de dominio, una dirección IP, o el nombre de un servidor que hayamos guardado previamente en nuestro archivo de configuración de SSH (información sobre como hacer esto puede verse en Definiendo servidores de SSH), seguido de dos puntos, y la carpeta que queremos usar para la transferencia. Nota que `rsync` no puede transferir archivos entre dos sistemas remotos, solo una carpeta local o una carpeta remota pueden ser utilizadas en conjunto con una carpeta local. Para hacer esto utilizamos:

Carpeta local a carpeta remota, utilizando un dominio, una dirección de IP o un servidor definido en el archivo de configuración de SSH:

```
rsync -rtvz carpeta_origen/
usuario@dominio:/ruta/a/carpeta_destino/
```

rsync -rtvz carpeta_origen/ usuario@xxx.xxx.xxx.xxx/ruta/a/carpeta_destino/
rsync -rtvz carpeta_origen/ nombre_servidor:/ruta/a/carpeta_destino/

Carpeta remota a carpeta local, utilizando un dominio, una dirección de IP o un servidor definido en el archivo de configuración de SSH:

```
rsync -rtvz usuario@dominio:/ruta/a/carpeta_origen/
carpeta_destino/
```

rsync -rtvz usuario@xxx.xxx.xxx.xxx:/ruta/a/carpeta_origen/ carpeta_destino/
rsync -rtvz nombre_servidor:/ruta/a/carpeta_origen/ carpeta_destino/

Excluyendo archivos y directorios

Hay muchos casos en los que necesitamos excluir ciertos archivos y directorios de `rsync`, un caso común es cuando sincronizamos un proyecto local con un repositorio remoto o incluso con un sitio vivo, en este caso podríamos querer excluir algunos directorios de desarrollo y algunos archivos ocultos de ser transferidos al sitio vivo. Excluir archivos puede ser hecho con `--exclude` seguido del directorio o el archivo que queremos excluir. La carpeta de origen o la carpeta de destino pueden ser un directorio local o un directorio remoto como se explico en la sección previa.

```
rsync -rtv --exclude 'directorio' carpeta_origen/ carpeta_destino/
rsync -rtv --exclude 'archivo.txt' carpeta_origen/
carpeta_destino/
rsync -rtv --exclude 'direccion/a/directorio' carpeta_origen/
carpeta_destino/
rsync -rtv --exclude 'direccion/a/archivo.txt' carpeta_origen/
carpeta_destino/
```

Las rutas son relativas a la carpeta desde la que estamos llamando `rsync`, salvo que comience con una diagonal, en cuyo caso la ruta sería absoluta.

Otra manera de hacer esto es crear un archivo con la lista de archivos y directorios a excluir de `rsync`, así como patrones (todos los archivos que caigan dentro del patrón dado serían excluidos, *.txt excluiría cualquier archivo con esa extensión), uno por linea, y llamar este archivo con `--exclude-from` seguido de el archivo que queremos utilizar para la exclusión de los archivos. Primero creamos y editamos este archivo en nuestro editor de texto favorito, en este ejemplo utilizo gedit, pero puedes usar kate, Vim, nano, o cualquier otro editor de texto:

```
touch excluidos.txt
gedit excluidos.txt
```

En este archivo podemos agregar lo siguiente:

```
directorio
direccion/relativa/a/directorio
archivo.txt
direccion/relativa/a/archivo.txt
/home/juan/directorio
```

```
/home/juan/archivo.txt
*.swp
```

Y después llamamos a `rsync`:

```
rsync -rvz --exclude-from 'excluidos.txt' carpeta_origen/
carpeta_destino/
```

Adicionalmente a borrar archivos que han sido eliminados de la carpeta de origen, como se explicó en Sincronizando dos carpetas con rsync, `rsync` puede borrar del destino archivos que están excluidos de la transferencia, podemos hacer esto con el parámetro `--delete-excluded`, por ejemplo:

```
rsync -rtv --exclude-from 'excluidos.txt' --delete-excluded
carpeta_origen/ carpeta_destino/
```

Este comando hará a `rsync` recursivo, preservará los tiempos de modificación de la carpeta de origen, incrementar la verbosidad, excluir todos los archivos que caigan dentro de los patrones en el archivo excluidos.txt, y borrar todos los archivos excluidos si existen en la carpeta de destino.

Utilizando rsync como un servicio cuando SSH no está disponible

Esto fue movido a su propia sección, Utilizando rsync como servicio

Algunos parámetros adicionales de rsync

`-t` Preserva los tiempos de modificación de los archivos que están siendo transferidos.
`-q` Suprime todos los mensajes que no sean de error, este parámetro es contrario al parámetro -v que muestra mucha más información de la transferencia.
`-d` Copia los archivos de un directorio sin utilizar recursividad para copiar los directorios internos, en decir, solo los archivos son transferidos.
`-l` Copia los symlinks como symlinks
`-L` Copia los archivo a los que un symlink está apuntando cuando encuentre un symlink.
`-W` Copia archivos enteros, ya que cuando utilizamos el algoritmo de delta-transfer solo se copia la parte de un archivo que fue actualizada, algunas veces esto no es deseado.
`--progress` Muestra el progreso de los archivos que están siendo transferidos.
`-h` Muestra la información que provee `rsync` en un formato más legible, las cantidades son dadas en K's, M's, G's, y así sucesivamente.

Notas al pie

La cantidad de opciones que `rsync` nos brinda es inmensa, podemos definir exactamente que archivos queremos transferir, que archivos en específico queremos comprimir, que archivos queremos borrar en la carpeta de destino si estos archivos existen, y podemos lidiar con archivos de sistema también, para más información podemos usar `man rsync` y `man rsyncd.conf`

Deje la información que concierne a respaldos fuera de esta publicación, ya que esto será cubierto, junto con la automatización del proceso de respaldos, en una publicación futura.

Es posible utilizar `rsync` en Windows por medio del uso de cygwin, sin embargo no cuento con una computadora con Windows disponible por el momento (o planeo tenerla en el futuro cercano), así que aunque he hecho esto no puedo publicar al respecto. Sin embargo, si utilizas `rsync` como un servicio en Windows, necesitas agregar la linea "`scrict mode = false`" en rsyncd.conf en lo módulos, esto hará que `rsync` no revise los permisos en el archivo de secretos ya que si lo hiciera fallaría por que no estaría apropiadamente configurados (ya que no funcionan igual que en Linux).

Esta publicación podría ser actualizada si hay algo que corregir o agrego más información si lo veo necesario.