

# Rsync

Need to move/copy a lot of files between 2 directories locally or across the network? Use rsync.

## Basic usage

If you want to a directory with files only in it, we use a simple command like this one: `rsync -v /home/user/target /tmp/destination` with `-v` flag for nice progress, but when we want to copy a directory with a subdirectory we need to use the `-r` flag and take account in how we want to move the directory, whether we want to only copy the files in a directory to another directory, or we want to copy the whole directory itself to another path. Here is an example:

```
> ls
file1 file2 file3 dir1 dir2 dir3

> pwd
/home/user/target

# Running rsync on the directory
> rsync -rv /home/user/target /tmp/destination

> ls /tmp/destination
target

# As you can see, running rsync on the directory copies the whole directory, now let's see what happens when
we add a trailing / at the end of target directory
> rsync -rv /home/user/target/ /tmp/destination

> ls /tmp/destination
file1 file2 file3 dir1 dir2 dir3

# Now we see that only the files in the directory have been moved, this is an important difference and
depending on what you want to do you can add a trailing / or no.
```

# Syncing directory

If you also want to sync files in the directory or create complete backups of what's in them you don't have to do anything that much different than now because rsync syncs files by default, if you want to do a complete sync and remove the files that were deleted in the original directory we add the `--delete` flag that will delete files in the destination directory if they exist but aren't present in the target directory.

“ example

```
rsync -rv --delete /home/user/target /tmp/destination
```

# Backup

If you want to do backup, that's where rsync needs quite a few flags to be powerful enough to do it, and with them you can do a complete backup of the system from the root of your filesystem and have an effective backup.

- `-a` the flag that enables a lot of other "archiving" flags that will help with preserving file permissions, file owners, ...
- `-h` to preserve the hard links
- `-A` additional perms
- `-X` extended attributes

“ example

```
rsync -ahAXv --delete /home/user/target /tmp/destination
```

# Remote server access

To run rsync with a remote server as target or destination it needs to be installed on both local machine and the remote server, also make sure to have something like ssh also installed for easy access.

To just copy from local machine to remote server we need to add and user and server address to the destination path like this: `user@remote:/tmp/destination`. As such an example command can look like this: `rsync -vP /home/user/target user@remote:/tmp/destination`. Same is done when you want to get files from the remote server, just reverse the order like so: `rsync -vP user@remote:/home/user/target /tmp/destination`.

The `-P` is important here in the command to make sure to optimize traffic for network copying of files, and to make sure if there happens to be a downtime or losing of connection we don't have to restart from the beginning, it's recommended to use it every time we are getting files to, or from a network remote.

## Adding new files to remote

One of most common uses of rsync for me is adding new files onto a remote server in a specific directory, here is an example on how to do it with rsync.

We need to make sure that the folder structure is the same on both ends so rsync will know where to put the new files on the remote.

“ example folder structure on the remote and local system

```
remote
├─ dir
│ └─ file
├─ file
└─ test
```

```
local
├─ dir
│ └─ file2
├─ dir2
│ └─ file
└─ file2
```

As you can see from the example, we have some new files and directories on the local system, when compared to the remote one, but rsync is more than smart enough to automatically fix that up for us with no issues, here is an example command I use quite a lot personally.

```
rsync -rvP /home/user/local/ user@remote:/home/user/remote
```

This command will strictly only get the files from the local directory and move them to the remote directory, placing them in correct places so that it will end up looking like this:

```
remote
├─ dir
│  └─ file
│  └─ file2
├─ dir2
│  └─ file
├─ file
├─ file2
└─ test
```

---

Revision #3

Created 6 March 2024 10:47:52 by CronyAkatsuki

Updated 6 March 2024 11:35:15 by CronyAkatsuki