Let that rsync in

Tom Ryder
tom@sanctum.geek.nz
https://sanctum.geek.nz/

Copying files locally

- Fundamental feature of operating systems
- Use the `cp(1)` command—in Unix since Bell Labs days:

```
$ cp original copy
```

- All the bytes are read and written, start to finish
  - Whether from/to RAM, or from/to disk

Pretty straightforward!

Pretty straightforward!

…right?

Options to GNU `cp(1)`: 1/6

`-v`: Be **verbose**: show the user what's going on.

```
$ cp -v original copy
'original' -> 'copy'
```

Options to GNU `cp(1)`: 2/6

`-p`: Preserve **permissions** and **modification times**.

```
$ cp original copy
$ ls -fl original copy
-rw-rw-rw- 1 tejr tejr 2 1991-12-26 01:05:22 original
-rw-r--r-- 1 tejr tejr 2 2021-03-09 22:47:24 copy
```

```
$ cp -p original copy
$ ls -fl original copy
-rw-rw-rw- 1 tejr tejr 2 1991-12-26 01:05:22 original
-rw-rw-rw- 1 tejr tejr 2 1991-12-26 01:05:22 copy
```

Options to GNU `cp(1)`: 3/6

`-r`: Copy files **recursively**: copy the whole tree

```
$ mkdir newdir
$ cp dir* file* newdir
cp: -r not specified; omitting directory 'dir1'
cp: -r not specified; omitting directory 'dir2'
$ ls newdir
file1  file2  file3
$ cp -r dir* file* newdir
$ ls newdir
dir1/  dir2/  file1  file2  file3
```

Options to GNU `cp(1)`: 4/6

`-P`: Don’t dereference **symbolic links**

```
$ touch file
$ ln -s file link
$ cp file cp-without-P
$ cp -P file cp-with-P
$ ls -l
total 0
lrwxrwxrwx 1 tejr ... cp-with-P -> file
-rw-r--r-- 1 tejr ... cp-without-P
-rw-r--r-- 1 tejr ... file
lrwxrwxrwx 1 tejr ... link -> file
```

Options to GNU `cp(1)`: 5/6

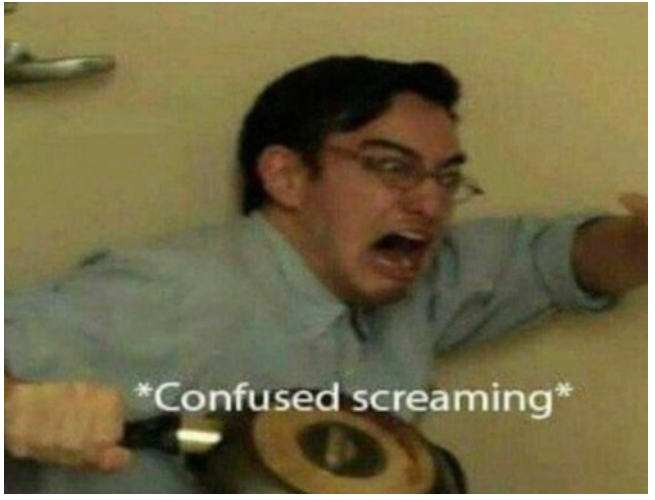
`-s`: Make **symbolic links** instead of copying

```
$ touch file
$ cp file copy
$ cp -s file softlink
$ ls -l
total 0
-rw-r--r-- 1 tejr ... copy
-rw-r--r-- 1 tejr ... file
lrwxrwxrwx 1 tejr ... softlink -> file
```

Options to GNU `cp(1)`: 6/6

- `l`: Make hard **links** instead of copying

```
$ touch file
$ cp file copy
$ cp -l file hardlink
$ ls -il
total 0
938223 -rw-r--r-- 1 tejr ... copy
938220 -rw-r--r-- 2 tejr ... file
938220 -rw-r--r-- 2 tejr ... hardlink
```



Complicating factors

- Copying local files **isn't** simple. Issues:
  - Modification times
  - Access times (on some filesystems, not on others…)
  - Directories
  - Permissions
  - Symbolic links
  - Hard links
  - Device files (`/dev`)
  - Virtual filesystems (`/proc`, `/sys`…)
  - Access Control Lists
  - …

```
CP(1)                                User Commands                                CP(1)

NAME
       cp - copy files and directories

SYNOPSIS
       cp [OPTION]... [-T] SOURCE DEST
       cp [OPTION]... SOURCE... DIRECTORY
       cp [OPTION]... -t DIRECTORY SOURCE...

DESCRIPTION
       Copy SOURCE to DEST, or multiple SOURCE(s) to DIRECTORY.

       Mandatory arguments to long options are mandatory for short options too.

       -a, --archive
              same as -dR --preserve=all

       --attributes-only
              don't copy the file data, just the attributes

       --backup[=CONTROL]
              make a backup of each existing destination file

       -b     like --backup but does not accept an argument

       --copy-contents
              copy contents of special files when recursive

       -d     same as --no-dereference --preserve=links

       -f, --force
              if  an  existing destination file cannot be opened, remove it and try again (this option is ig-
              nored when the -n option is also used)

       -i, --interactive
              prompt before overwrite (overrides a previous -n option)

       -H     follow command-line symbolic links in SOURCE

       -l, --link
              hard link files instead of copying

       -L, --dereference
              always follow symbolic links in SOURCE

       -n, --no-clobber
              do not overwrite an existing file (overrides a previous -i option)

       -P, --no-dereference
              never follow symbolic links in SOURCE

       -p     same as --preserve=mode,ownership,timestamps

       --preserve[=ATTR_LIST]
              preserve the specified attributes (default: mode,ownership,timestamps), if possible  additional
              attributes: context, links, xattr, all

       --no-preserve=ATTR_LIST
              don't preserve the specified attributes

       --parents
              use full source file name under DIRECTORY

       -R, -r, --recursive
              copy directories recursively

       --reflink[=WHEN]
:
```

And to make matters worse…

- We need to copy files onto *other systems*
- Gigantic files need to go over slow links
  - Even if they only change a *little bit*…
- Links are unreliable—did it *really* copy?
- Case sensitivity of filesystems varies
- User and groups don’t match
- Timezones don’t match
- *Operating systems don’t match…*

What to do?

- Better tools are needed to work with remote files
- Nobody uses `cpio(1)`
  - Which is a shame, because it's decent, and standardised too…
- Piping `tar(1)` archives *sort-of* works, but it's a bit weird:

```
$ tar cf - source | ssh remote tar xf - -C copy
```

- OpenSSH's `scp(1)` is just `cp(1)` with SSH tacked on.
  - And it's likely to be deprecated…

Tridge fixes it (as usual)

- Samba developer
- Kernel developer
- Australian, but give him a pass
- In part for his PhD thesis, implemented **rsync**, an efficient tool to transfer and synchronise files
  - Released in 1996

Installation

Keep it simple. Observe **Tesla's First Law**:

```
<tesla_> bro i just type `apt install $x`
and it works lol
```

```
$ sudo apt install rsync
$ sudo yum install rsync
```

rsync features 1/6—Local copy

- Out of the box, works just like `cp(1)` on local files:

  ```
  $ rsync original copy
  ```

  Does what you'd expect
- Feature parity with `cp(1)`, mostly:
  - Verbose: `$ rsync -v original copy`
  - Recurse through directories: `$ rsync -r original copy`
  - Keep permissions: `$ rsync -p original copy`
  - Keep timestamps: `$ rsync -t original copy`

rsync features 2/6—Remote copy

- Runs over SSH:

  ```
  $ rsync original user@remote:copy
  ```

  Behaves like:

  ```
  $ scp original user@remote:copy
  ```

- Add SSH keys for greater convenience and even automation

rsync features 3/6—Compression

```
$ rsync -rv bigtxtdir user@remote:
...
sent 87,250,453 bytes  received 38,429 bytes  7,590,337.57
bytes/sec
total size is 87,096,690  speedup is 1.00

$ rsync -rvz bigtxtdir user@remote:
...
sent 29,597,390 bytes  received 38,529 bytes  2,043,856.48
bytes/sec
total size is 87,096,980  speedup is 2.94
```

rsync features 4/6—Small changes

Copying a 100 megabyte file with one single-byte change:

```
$ time scp rand user@remote:rand
real:0m8.483s user:0m2.883s sys:0m1.730s
$ time scp rand-tweaked user@remote:rand
real:0m11.219s user:0m2.709s sys:0m1.772s

$ time rsync -a rand sanctum:randomfile
real:0m6.656s user:0m3.230s sys:0m1.653s
$ time rsync -a rand-tweaked sanctum:rand
real:0m2.352s user:0m1.052s sys:0m0.119s
$ time rsync -a rand-tweaked sanctum:rand
real:0m0.460s user:0m0.080s sys:0m0.016s
```

rsync features 5/6—Skip unchanged

Great for backups:

```
$ rsync -av /var/log/mail.log* user@remote:
sending incremental file list
mail.log
mail.log.1
mail.log.2.gz
```

~ one hour later… ~

```
$ rsync -av /var/log/mail.log* user@remote:
sending incremental file list
mail.log
```

rsync features 6/6—Exclusions

- Exclude files:

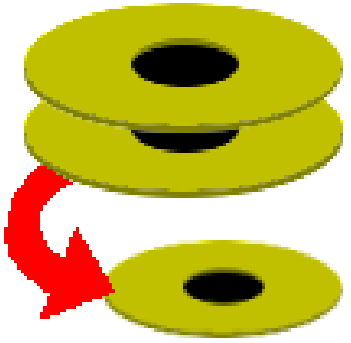
```
$ rsync -av /home/user \
--exclude=/secrets \
user@remote:
```

- Keep exclusions list in a separate file:

```
$ rsync -av /home/user \
--exclude-from=/home/user/secrets \
user@remote:
```

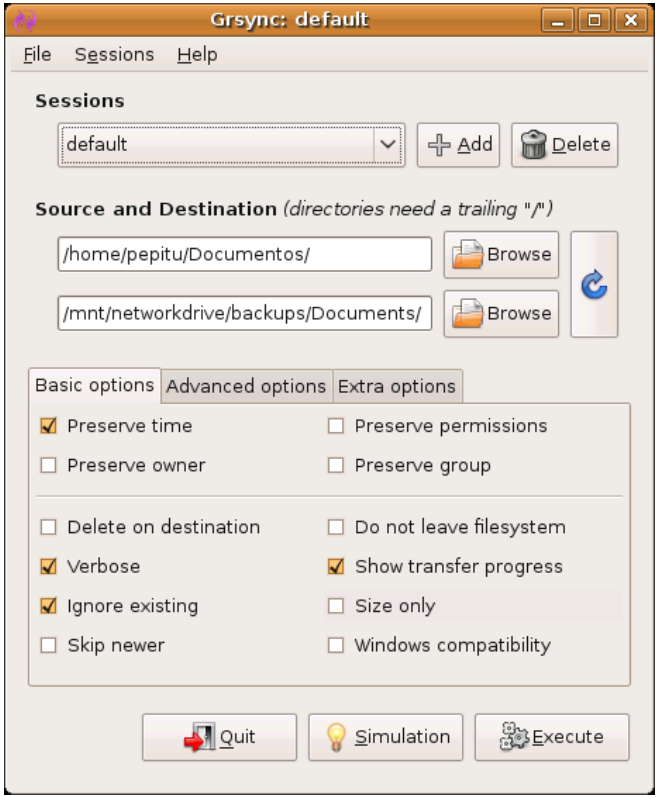
Applications 1/3—Dirvish

- **Dirvish**: Preserve hard links to make complete images of your backups…
  - …but store each version of each file *only once*
  - Uses rsync's `--link-dest` feature
- Tom presented on this in 2019



Applications 2/3—Grsync

- **Grsync:** GUI frontend for rsync
  - Doesn’t have the full feature set, but a good way to get a feel for what it can do
  - If you’re curious about rsync but don’t like the command line, start here



Applications 3/3—Unison

- **Unison:** Bi-directional file sync
  - Keep one or more sets of the same files in sync
  - Deal with conflicts as they arise
  - Uses rsync-the-algorithm, not rsync-the-tool
  - With a little setup, essentially a **free software Dropbox!**

Questions?

- **Rsync:** https://rsync.samba.org/
- **Dirvish:** https://dirvish.org/
- **Grsync:** http://www.opbyte.it/grsync/
- **Unison:** https://www.cis.upenn.edu/~bcpierce/unison/

Email: tom@sanctum.geek.nz
Website: https://sanctum.geek.nz/
Twitter: @tejrnz
Fediverse: @tejr@mastodon.sdf.org