

# Incremental backup with Rsync

Eric Blair

10 January 2007

I think part of why I'm into the whole computing thing is the stability it affords.

There's this funny dichotomy in the world that half the world thinks that statement is ironically funny—computers? stable?—and the other half knows exactly what I mean.

I've mentioned before that my dear family moved a whole lot in my life, once because the rent across the street was cheaper than the rent at the place we were living. We didn't even get a truck, but just walked everything over box by box. Other moves across political boundaries involved more of an upheaval. Since 1999, I've had eight addresses, but only one home directory.

When I go, all that'll be left of me is my home directory.

I'd be depressed by this, but it's probably more than most of our predecessors have had. I would tell my eventual progeny, or maybe my neighbor's progeny, to copy my home directory into a folder in their home directory, and after I am no more they can carry that around with them like the photo/text album that it is. [Mr DCH of Philadelphia, PA, be sure to tell Mr LH about all this when he's old enough to read.] Generations from now, the kids will have a folder of parent's work, which contains a folder of grandparent's work, which contains a great grandparent's folder.

Beyond your private data, there are of course the attempts to archive and sort public data for the long haul. The Rosetta Project hopes to make a permanent record of the world's languages before they all turn into English; here's a URL<sup>1</sup> that they provided for their online archive two years ago. The National Archives of Australia<sup>2</sup>, among others, is encoding its digital data into formats that are more likely to be legible by future readers.

<sup>1</sup><http://www.rosettaproject.org:8080/live/search/languagesearch>

<sup>2</sup><http://www.computerworld.com.au/index.php?id=954149621&eid=-6787>



Figure 1: The transition from analog media (as pictured by the wave form at left) to digital media (represented by the 1-0 at right) has both solved and created problems for data storage.

There's still a dichotomy between the world's public data, which is constantly merging into bigger and more unified databases<sup>3</sup>, and the data on our hard drives, which remains our personal problem. I often wonder if that dichotomy will remain in the future, or all our data will someday be in one huge database with lots of switches that grant permissions for your data only to you and the NSA. But your guess is as good as mine on that one.

In the mean time, making sure your data is safe is your problem.

Any medium of any sort, even stone, will eventually deteriorate<sup>4</sup>. But if you always remember to copy your data to a new location when new technology comes out, then you can basically overcome this. I've owned or had access to around ten computers since 1999, but by taking care to have exactly one canonical and up-to-date home directory at all times, I've survived the obsolescence and accidental dropping of a lot of hardware.

The other problem is you: you are an idiot, and will find ways to break your own data. The only hope is that you have the wisdom to back up your data regularly. Me, I have my data backed up onto five (5) different locations.

The half-assed option is to keep a backup somewhere on your working computer. This will only save you from your own dumbness, and not the day that you accidentally drop your entire laptop down the sewer grate.

**How to back up your data properly** A hundred bucks will get you an external 40GB or 60GB hard drive. That may seem like a lot for a device that has no chance of getting you laid, but you'll be kissing it and crying tears of joy when your primary hard drive suddenly starts making chunkchunk noises a year or two from now. And it's nice for carrying data between home and work.

On to the backup method. Windows users, you might get lucky and find that your system supports hard links, or you may not; using Windows is exciting that way. [There may be reasons for why Windows shortcuts aren't just plain old links<sup>5</sup>, but the fact remains that they aren't, and you can't play along with the below on the VFAT filesystem, which doesn't allow links.]

Everybody else, the backup process consists of a *two line script*. [McUsers, if your version of `cp` doesn't do links, see the note below.] The idea is perfected by the RIBS (rsync incremental backup system), but I felt their script to be a bit of overkill for something that can be executed with two commands.

We need to keep not just the latest good version of your home directory, but the last twenty or so. That way, if you don't realize that you accidentally deleted all your family photos until a month after the error, you can still go back in your backup archives and pull them out. The problem, of course, is that you need a way to make twenty backups without taking up twenty times the space.

First the super-short scripts that solve the problem, then an explanation of how they work. Here is the local version:

```
DEST=/mnt/
```

---

<sup>3</sup><http://arstechnica.com/news.ars/post/20061229-8522.html>

<sup>4</sup><http://www.straightdope.com/columns/020816.html>

<sup>5</sup><http://www.microsoft.com/technet/technetmag/issues/2006/09/WindowsConfidential/>

```
rsync -aP --delete --exclude ".mozilla/" \
      --exclude ".opera/" $HOME/ $DEST/current/
cp -al $DEST/current /$DEST/`date +%d-%b-%y`
```

And here's a remote version, in case you have a gigabyte or two at a remote computer:

```
AC=my_login@backuphost.edu
DEST=/home/me/backupdir
rsync -aPe ssh --delete --exclude ".mozilla/" \
      --exclude "tmp/" $HOME/ $AC:$DEST/current/
ssh $AC "cd $DEST; cp -al current `date +%d-%b-%y`"
```

The first line or two just set(s) variables, which makes it easier for you to do multiple backups by just changing these lines.

[Below, a commenter notes that you should take care to not put spaces before or after the = (because I used to have spaces here—oops).]

The next line uses the delightful `rsync` program, which checks every file for changes before copying. Online, that means less bandwidth and wait; both online and locally, this will have a major payoff below. Notice the `--exclude` lines that indicate directories to not copy; feel free to add as many of these as necessary, or RTFM for other means of specifying what `rsync` should ignore.

The last line, `cp -al` (that's an ell, not a one), copies an entire directory, but using hard links. What's a hard link? It's basically a second name for the same file. The actual file is located on the hard drive somewhere, at some illegible address like `0x6344ab32b`. To give it a name, the disk has a table of contents that says that the file, say `current/letters`, is located at that address. But why not have multiple table of contents entries pointing to the same position? You could have `january06/letters` point to exactly the same file, which means that what looks like two copies of the file in the ToC is taking up as much space as one copy. Notice that the copy and the original are identical—both are single entries pointing to the same data. If you delete the original, the copy would persist. [It's up to the file system's internals to keep count of how many names a file has and delete the file when the last name is deleted.]

So the first time you run this script, you will have in your `$DEST` directory a `current` directory and a directory named for the current date, and both will be identical. In a few days, when you get around to running the script again, `rsync` will modify only those files in the current directory that have changed in your home directory. That is, `rsync` will erase the duplicate entry for the file (but not the file itself) and write a brand new edition of the file divorced from the original. Since `rsync` won't touch the files that haven't changed, they will continue to take up (approximately) zero additional space.

Thus, the magic of `rsync` and hard linking will produce separate subdirectories that each look like a complete backup, but which only take up as much space as a single backup plus a series of incremental updates.

**Does your cp not link?** The GNU edition of `cp` will do the trick above to regenerate a directory with hard links, but some versions of `cp`, like the one that ships with McIntosh computers, are less extensive in their features.

In that case, try `pax`, which is a program comparable to `tar`, that makes backups. We don't care about its archiving, but it will do the hard-linking, albeit with slightly less grace. Again, the `-l` is an ell for link.

```
mkdir $DEST/`date +%d-%b-%y`  
pax -r -w -l $DEST/current $DEST/`date +%d-%b-%y`
```