Yet Another Rsync Backup Utility

Edward Grace

February 25, 2010

Abstract

Yarbu is a simple but powerful automated backup utility designed for efficient hands-free backup of servers, desktops and laptop computers across a wide variety of operating systems and hardware. It works well with clients that have an intermittent network connection as well as machines that are connected permanently, makes use of an encrypted connection for the backup process and is straightforward for users to modify for their precise needs.

Contents

This documentation concerns version 1.4.8 of yarbu and is available as a pdf[1], self-contained HTML archive[2] or online[3].

1 Suitability

If you agree with the following statements, YARBU is probably suitable for you; the sub items elucidate the reasoning.

- I am nervous about trusting my backup to a disk image, essentially a large file, that holds all my backups since any minor glitch in that could silently destroy or corrupt my backup.
  - Yes Microsoft Backup / Time Machine I am looking at you! My philosophy is that backup should be a simple and transparent process; it should be the antithesis of sophistication since it is so important! If you find that your backup failed in an obscure way and you cannot recover the files, what does it matter how fancy the user interface is?
- I am responsible for just my machine or the machines of other people that trust me to log in to the client machines.
  - Since the backups will result in locally unencrypted copies of the data on the backup server any users of the backup server can have access to the data on the clients.
- I do not want to bother installing software on the client machines.
  - By default the backup server needs just `ssh`and `rsync`available on the client. Most operating systems have this by default. See section 5.4 for Windows specifics.
- I have access to a machine that can be solely tasked with the role of backup server and this machine can have Linux installed on it.
  - It is important that the backup server be logically isolated from the rest of the clients. The default Linux file systems (ext3) appear to be the most accommodating regarding file names.
- The machine that I wish to backup either has a static DNS name, static IP address or a dynamic DNS name (for example DynDNS [4]).

[1] yarbu.pdf
[2] yarbu-html.tar.gz
[3] http://yarbu.sourceforge.net/
[4] http://www.dyndns.com

  - Since the backup server contacts the client, the client must be locatable by DNS or by fixed IP address. This is necessary because the client is assumed to be dumb, needing no backup software.

- The backup server is able to send me email.
  - This is essential for all backup solutions. Without it you will be unaware of success or failure.

2 Rationale

Robust backup and restoration of user data is *exceptionally* important as anyone that has lost their work will tell you. It is possible that you have been using computers for years without ever entering the situation where you lose precious data through accidental deletion, malicious action or hardware failure. Then again, if that were the case you would probably not be reading this. If you have been the victim of data loss, my commiserations. Now, never let it happen again!

In a heterogeneous computing environment there may be wide combinations of hardware and operating systems as well as laptops, desktops and servers – all of which require backing up. In order to best serve the needs of all these various clients there should be a minimal set of requirements for installation on the clients. Preferably the clients should not need any software installing at all! Without this assumption of a dumb client the maintenance and management of such a wide variety of platforms would quickly become a nightmare – simple is best!

3 Overview

This project is, as the name suggests, another backup / restore utility based around the well-known and powerful `rsync`[5]. It is designed to act as a server on a dedicated backup machine that will periodically backup a set of remote clients in a robust, transparent manner.

As depicted in figure 1 the backup server contacts various clients. These clients can be hetrogeneous and can be servers for other machines. The example below indicates a PC (Linux) client, a SUN Solaris client and a Linux machine acting as a SAMBA server for other Windows PCs all being backed up by a central backup server running yarbu.

3.1 Background

The concept of rotating snapshot like backups is nothing new. Much of this script is inspired by some of the work done by Mike Rubel[6] and others. A list of alternative scripts and projects can be found in the section

[5] http://samba.anu.edu.au/rsync

[6] http://www.mikerubel.org/computers/rsync%5Fsnapshots/

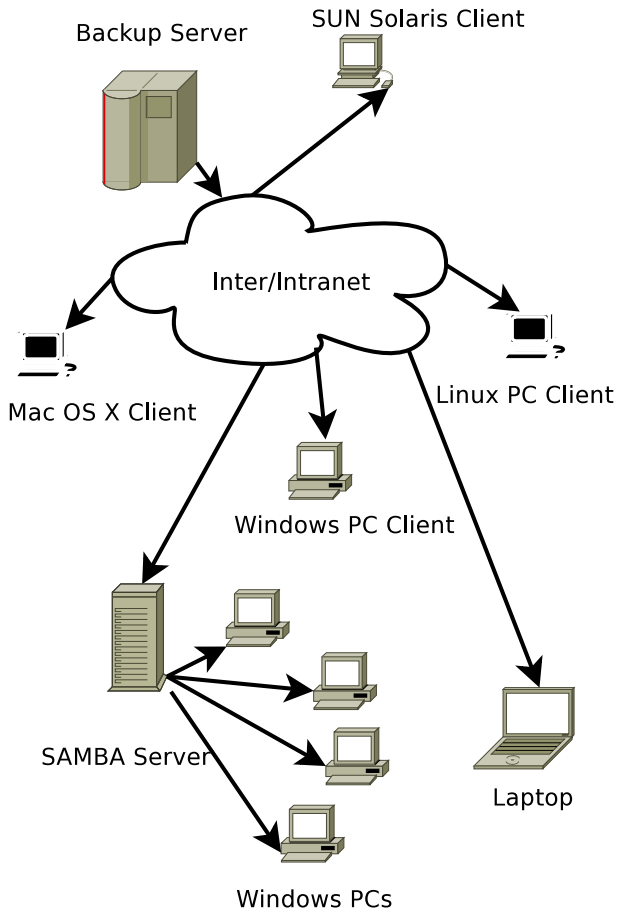

Figure 1: Basic concept. Server regularly contacts clients via intranet or internet and carries out incremental backups.

Contributed codes[7]. For a number of reasons I have found a number of issues that these scripts do not all tackle and ended up rolling my own solution to the problem. If you are using a modern Mac with Snow Leopard then you probably are using Time Machine – I am. That said, it is a Mac only solution and does suffer a few drawbacks which can be irritating. It is my aim for this project to fulfill the following criteria:

1. intelligent backup of often disconnected machines
2. server-side only, client machines should be considered dumb
3. straightforward but powerful config files
4. low client loading, by default we want backups every hour. No one using the client should notice a performance hit when the backup takes place.
5. well validated input and helpful error messages
6. easy for a novice user to implement
7. ideally using this should be as simple as
   (a) install rpm
   (b) run setup utility to populate clients with `ssh`keys and test machines

4 Download

The most recent version of yarbu, 1.4.8 , may be obtained from the sourceforge download a
either as an RPM suitable for RedHat / Mandrake based Linux systems or
as a source archive.

4.1 Concepts

There are naturally a number of different ways of approaching the problem of backing up a directory tree. In general we would like to be able to restore the state of every file within a certain directory on a machine *exactly* as they were at a given time in the past. This means preserving the ownership, time stamps, permissions and other attributes of files or symbolic links and archiving all these files in some easily recoverable place.

Traditional backup systems are based upon tape drives. With the price of hard disk storage plummeting the cost benefit of different storage media has changed dramatically. Since a good automated tape drive costs many thousands of pounds, despite the ultimately lower cost of the media, hard disks still offer a cheaper storage option for medium sized institutions (hundreds of machines) with the bonus of essentially instant access to the required data.

[7] http://www.mikerubel.org/computers/rsync%5Fsnapshots/#Contrib
[8] http://sourceforge.net/project/showfiles.php?group%5Fid=103606

Ideally we wish the backup system to be as divorced from the detail of the clients as possible. We do not want to have to install packages on the client machines since this can become an administrative nightmare. Far better is to treat the client machines as "dumb", unable to communicate with the backup server and demand anything. This allows all the logic to be centred on the server and essentially removes any maintenance requirements from the end user. It keeps 'out of the way' of end users.

5 Configuration scenarios

Configuration for this backup system is based around familiar, simple configuration files which are essentially just a list of parameter, value pairs. It is deliberately designed to be straightforward. If you wish to do weird and wonderful things with your backup strategy, your backup strategy is probably flawed.

For example, the following configuration file will backup the directory `/home` of a machine called `apples` to a local directory `/backups` on the backup server.

```
SOURCE=apples:/home
TARGET=/backups
```

Aside from configuring root `ssh`access that is all the configuration that is required. If you have downloaded and installed the rpm version of the software, adding the above file as (say) `/etc/yum/conf/apples.conf` is all you need to do to get rolling hourly backups.

The following are some example configurations and use-cases that may pique your imagination.

5.1 SAMBA server

The important aspects of the server to backup up are the configuration files and the user data that is being served by SAMBA to other machines. As such we could have two configuration files, the first protects the data in the home directories that are shared

```
SOURCE=mysambaserver:/export/shares/home
TARGET=/backups
```

and the other configuration protects the SAMBA configuration itself.

```
SOURCE=mysambaserver:/etc/samba
TARGET=/backups
```

5.2 Laptop

With a laptop that could be anywhere in the world, it's IP address could change from day to day. Consequently the server could find it difficult to know where to look to find the client. As a prerequisite one should install

DynDNS or a similar service on to the laptop. When this has been done, the laptop may then be referred to by its name and configuration of the backup server proceeds in the normal manner.

5.3 Mac OS X

For those of you who do not wish to upgrade to Snow Leopard, pay for Time Machine or don't trust placing your entire backup on a disk image or don't like insecure backups over AFP to a simple Netatalk client you may like to make use of yarbu too.

Like Time Machine, we may not wish to backup certain aspects of the system that are Mac specific. For example backing up `.Trash` subdirectories is probably unnecessary as this contains a load of stuff that's usually junk. It should be assumed by the users that when you delete something it's gone and that the Trash is the first and last port of call!

We may also decide to only keep the last three hours of backups, but to retain the default number of daily, weekly and monthly backups. Since the primary user of `daffy-macbook.dnsalias.net` is Mr. D. Duck we also send email notification of the backups to him. To avoid backing up things in `.Trash` we also write an excludes file called `exclude-trashes` containing:

```
.Trash
```

and the configuration file for homes, called `homes.conf` would be:

```
SOURCE=daffy-macbook.dnsalias.net:/Users
TARGET=/backups
EXCLUDES=exclude-trashes
MAXBACKUPS_HOUR=3
EMAIL="Daffy Duck <d.duck@googlemail.com>"
```

Note that, since the machine is a laptop, it is referred to by a dynamic DNS entry `daffy-macbook.dnsalias.net` however email to daffy is sent to his usual Google mail account.

5.4 Windows

By default Windows does not come with useful utilities such as `rsync`or `ssh`. In order to use the yarbu backup server with Windows clients you must therefore install Cygwin on each Windows client you require.

This is a small annoyance which cannot be avoided at the present time unless Microsoft decide to install these tools (Apple do).

The following are brief installation instructions:

- Become the machine administrator.
- Visit the Cygwin home page[9] and click "Install or update now!".

[9] http://www.cygwin.com

- Run the Cygwin `setup.exe`.
- Select all the nominal defaults, and when it comes to the selection of packages make sure that `openssh` and `rsync`are selected for installation.
- Wait for the installation to complete.
- Right click on "My Computer, Properties, Advanced, Environment Variables". Click "New" and add a new entry to the system variables, the name should be `CYGWIN` and the value `ntsec`. Append `;c:\\cygwin\\bin` to the variable `PATH`.
- Start the cygwin Bash shell and enter `ssh-user-config` to generate a default public, private key pair.
- On the Linux backup server now follow the usual instructions for any other client.

Of course, if your machine is also a laptop, follow the advice above regarding a dynamic DNS name.

6 Avoiding monthly and weekly backups

Typically you may wish your backup strategy to be based around short or medium term disaster recovery rather than long term archival.

With this in mind it is often not necessary to have weekly or monthly backups at all, rather a days worth of hourly backups and a few days of daily backups.

To avoid backups you can simply set the `MAXBACKUPS_WEEKLY` and `MAXBACKUPS_MONTHLY` variables to zero. Likewise you may be perfectly happy to have daily, or weekly backups but not want hourly backups - in which case set `MAXBACKUPS_HOURLY` to zero.

7 Traps for the unwary

Be careful about not allowing yourself enough hourly or daily backups. Consider, for example, what happens if the directory you are backing up is empty. This could happen if there is a disk error and the mount point cannot be mounted or, in a fit of stupidity, you delete everything. When the hourly backup occurs, it will carry on as usual - backing up an empty directory - with the oldest hourly backup being lost.

If you only have four hourly backups, after four hours you will no longer have any hourly backups - the hourly backup queue will have been flushed. If you did not allow daily backups you have now lost all your data! If on the other hand you have daily backups you can still retrieve your (old) data from those.

An important thing to quantify is "*How long will it take for you to notice the problem and attend to it?*". Typically the worst case scenario is three days, the entire weekend plus one for potential bank-holidays.

As an *absolute* minimum therefore you should typically keep three full days of backups.

8 Current limitations and drawbacks

- The machine does need to be connected to the network for at least an hour, to make sure that hourly backups occur.
- There is no client-based threading so a client that takes a long time to back up will stall any other clients that are waiting to be backed up.
- Mac OS X resource forks and extended attributes (Access Control Lists) may not be backed up or restored correctly – this is an advanced topic, if you don't know what a resource fork or ACL is – it probably doesn't matter to you!
- If the backup server dies for no apparent reason there will not necessarily be a warning that the backups have stopped.

8.1 Warning

One should realise that if you are sufficiently motivated, it is possible to do *disastrous* things with this script. Implicit is the assumption that the backup server is "friendly", in other words trusted. Since the backup server requires root access to the clients, compromising the backup server allows the potential compromising of the clients as well. For this reason you should make sure that the backup server does not run anything but the most critical of services. Ideally the backup server should be dedicated to just backups.

9 Future development

It is fairly clear now that the original design aims of making the script bash only and system independent are somwhat conflicting, the differences between Linux and OS X command line tools are significant.

9.1 Disk space

In order to more efficiently use disk space the oldest backups should be removed irrespective of the requirements to keep a given number of various backup levels.

9.2 Encryption and backup browsing

It's my intention to offer integrated EncFS encryption of the filesystem, in this manner there need not be any trust relationship between the client whos data is to be backed up and the server, where the backups reside. For more information on EncFS, see the EncFS website[10].

Rather than backing up the *actual* file system the encrypted version of the filesystem would be backed up. Similarly when accessing the backups only a person with appropriate privaleges can view the unencrypted version of the files.

9.3 Virtual filesystem FUSE integration

A common task when restoring files, is to find the right one you want from some point in the past. Currently this requires looking at each tree, either an hourly or daily backup and drilling down to inspect the file in question. This can be tedious and time consuming.

Instead by using FUSE, browsing the backup repository will always show pseudo directories, for example "Hour Older" and "Hour Newer" which are actually soft links to the same directory as the one you are browsing, only an earlier or later hourly backup.

For the immediate future however the 1.3.x and 1.4.x branches will be developed in bash. This will require writing an appropriate FUSE driver to handle translation between the real file system tree and the virtual one presented to the user. Information about FUSE, for which EncFS is an example can be found on the FUSE[11] website.

9.4 Backup on demand - push backups

Often a client wishes to make ad-hoc backups, or backups from roaming clients such as laptops. For this, a push method of backup will be included such that the client connects to the backup server and requests a backup.

9.5 Live database backup

Certain systems do not keep their on-disk files in a state which, if backed-up while running, would permit faithful restoration. An example of this could be a database server whose database files are actively written to and read from. In order to faithfully carry out a backup without closing down the database server in a live environment one may need to do the following.

1. Backup the on-disk database files and write ahead logs.
2. Poll the database server such that it makes the on-disk representation coherent.
3. Repeat the backup, being that this is incremental it will be very quick – backing up just the changes that occurred between 1 and item 2.

[10] http://arg0.net/encfs
[11] http://fuse.sourceforge.net/

4. Poll the database server to indicate that it can resume normal service.

Clearly the action of backup should be atomic, either the backup succeeds so that the state can be faithfully restored, or it fails. This would require the addition of 'hooks' in to which the user could slot pre- and post conditions for bespoke tasks such as database backup.

10 Biography

The author is based at Imperial College London his research interests include electromagnetism, nonlinear optics, laser physics, computational photonics, optical data storage and nanophotonics. Other interests include computing, finance, cycling and Go.