– General Linux 1 –

(Linux Professional Institute Certification)

111-5 Maintain an effective data backup strategy [3]

a


```
   .~.      by: Grant Parnell (slides 8 to 38)
   /V\     and  Andrew Eager  (slides 39 to ??)
  // \\         geoffrey robertson
  @._.@         geoffrey@zip.com.au
```



```
$Id: gl2.111.5.slides.tex,v 1.1 2003/08/27 05:27:45 geoffr Exp $
```

List of Slides

Maintain an effective data backup strategy

Objective

Candidate should be able to plan a backup strategy and backup filesystems automatically to various media. Tasks include dumping a raw device to a file or vice versa, performing partial and manual backups, verifying the integrity of backup files and partially or fully restoring backups.

Key files, terms and utilities

- `cpio`
- `dd`
- `dump`
- `restore`
- `tar`
-

Backups

Decide what data is important and how long you can do without it.

- Is this used 24 x 7 or just business hours?
- During business hours how long can you do without it? 4 hours, 30 minutes, 5 minutes?
- How up-to-date is it required to get you running in an emergency?
- Are you backing up for archival or high availability or espionage?

Examples of Data

Static

Configurations of running servers. You need these 24x7 but they don't change much.

Databases / Transactions - financial & otherwise

These are updated frequently and need to balance. Associated with these are logs & duplication & other means of rollback & integrity checking. With databases it's often a good idea to dump them in a good portable format, especially if the inbuilt format is not cross platform or cross version compatible.

Example:

```
$ mysqldump mydata >mydata.dump ↩
```

This will give you a text file which can be used on most mysql versions and possibly adapted to other database packages.

Logs

People don’t tend to read them unless something goes wrong in which case they’re valuable. These need to be kept but don’t need to be restored in a hurry.

Home directories

This is a mixed bag of everything but some policies could be instated to make the admin's life easier.

EG Making specific sub-directories for things and assigning them different backup/restore priorities.

Often the existence of a home directory is more important than the rest of the contents as it may make a user unable to login without it.

Code repositories

Programmers should be accustomed to doing regular backups anyway, they often need to revert to an old version to figure out what they broke.

Any tools used such as CVS that have a central repository should be backed up almost as often as programmers commit code, at least once a day but they could probably cope with it being missing for half a day.

High availability - read only

Websites frequently used by your clients.

They can contain dynamic data but customers don’t update it.

This sort of scenario lends itself to frequent replication to a backup server.

High availability - interactive

Taking a website again, this one might allow the customer to do such things as place orders. The website maintains some state information to allow building of an order. This is the most difficult, the state information can be stored in a replicated database. In the event of web server failure the other one comes into play and the customer may have to login again but the information is kept. (Otherwise complex designs and expensive hardware can be used to seamlessly migrate the state to the other webserver).

Important Linux directories

/var/spool/mail - daily backup
/var/lib/mysql - databases - backup the dumps, and possibly the binary.
/var/log ? - from "don't care" to "backup daily"
/etc - backup config changes
/home - be selective, but if you can't, backup daily.
/home/<user>/mail - contains the user's mail folders (may also be 'Mail' or 'Maildir')
/home/<user>/.ssh - If you login using ssh keys only, this is a must have.
/usr/local - locally installed apps & data
Application specifics

Backup & Restore methods

Copy the files to another directory

This is the poor mans backup and does not offer much peace of mind. It does protect against accidental deletion & corruption by users. One advantage is that it can be very quick for things such as log files. You can also keep multiple copies, one for every day of the week for example. See `/etc/logrotate.conf`.

Backup to a standby partition

This has about the same level of peace of mind as the above. The backup partition can be left un-mounted after the backup. The backup is slower than the above but the restore operation can be quick. See also ”Broken Mirror” method below.

Backup to tape

This is probably the most common backup used in the commercial world. It's easy to backup the lot every day provided you have the tape capacity. If you don't, you become more selective as to what to backup. There's a variety of software to do this but there's 3 main basic systems. Tar, cpio and dump. Often commercial software uses these basic systems and provide for labelling & indexing as well as multi-server capability from a simple GUI. The reason for using the basic systems is you can restore from them if you have to.

Backup to standby disk

This can offer peace of mind and a fairly cheap backup for people that don't require 24x7 service. Basically a removable drive bay houses another hard disk of similar capacity and the entire system is backed up. This can be done partition by partition or file by file using dd, cpio or rsync. Additional steps can be taken to ensure that the backup is also bootable. The backup drive should be removed once done and treated like a tape. The disadvantage here is that you most likely will need to power down the system twice for one backup. Alternately, if you have an external USB or fire-wire storage medium it becomes possible to do this without downtime.

Backup to CDROM/DVD

Under Linux (as far as I know) there's no software to directly write data without creating an image first. This means there must be sufficient space available. It would be possible to create a bootable CD with restore software and a compressed filesystem but I haven't seen this. It may be OK if you don't have a large filesystem or you have a DVD writer or you're not backing up everything.

RAID System

Not strictly a backup but a RAID system can protect against hard drive failure by providing redundancy. Data is written simultaneously to 2 or more hard drives and can include parity information. It does not protect against corrupt databases and people removing files. It will corrupt & remove files equally well on all disks. Linux can do RAID in software very well but the ideal is a hardware solution involving hot swapable disks so they can be replaced while the system is fully running. A RAID system can mean the difference between going on-site at 3am and saying ”Oh dear, we’ll replace that first thing in the morning”. Just ensure that you do have a replacement readily available and do not have to wait a week.

RAID Tape array

In a similar manner to RAID 5 disks, data is written in parallel to 5 tape drives which increases throughput and data integrity.

Backup Server

All of the methods discussed so far involve direct transfer from server to backup medium. If you have a number of servers it may not be practical to install backup devices on each. Another way is to remotely access the required medium directly (`/dev/rmt0`) but arbitration of access can be an issue. An increasingly popular way is to provide a super-server with a huge amount of disk space capable of holding everything required by the other servers. Transferring the data can happen at any time in either a batch or continuous process. A batch would be say backup a whole directory at once whereas a continuous operation might be transmitting log information or database updates. The backup server itself may then employ any one or more methods to perform backups of itself, possibly based on some statistical analysis. An example of this is a system called ADSM which employs RAID arrays, multiple tape drives, a tape robot with barcode

reader and intelligent software that tells the operators which tapes are to go off-site and which ones it wants back. It essentially is a huge cache that stores frequently changing data locally and stores old data off-site.

Broken Mirror

If you've got about 100Gb of data on a mirrored pair of disks and only have a 10 minute backup window this may be for you. Basically you bring the system down, unhook one of the mirrors and replace it with another set of drives and bring the system up again. Mirroring starts from scratch during quiet time and should be finished before load picks up again. With the drive set you just un-hooked this can then be loaded into the standby server and backed up to tape over the course of many hours. Some high end servers can perform this operation without downtime as the hooking up can be done using inbuilt hardware or such things as dual-port fire-wire drive bays. All that is required in this case is an application shutdown, sync, dismount, remount, application start type operation.

Backup Software

Command line tools

dd Copy and Convert can be used to copy raw disk blocks, even to tape (yuk).

Example:

```
dd if=/dev/hda1 of=/dev/hdb1
```

tar Tape ARchive - you all know how to unpack tgz files, and maybe even create them. Just remove the 'f' option. It also can be an advantage not to use compression as some drives have this built in. Also, a portion of the tape being corrupt can ruin the rest of the data, whereas you can skip corrupt bits and pickup the next file if not compressed.

Example:

```
tar -c /home
cd /tmp; tar -x
```

cpio CP I/O - Similar capabilities of tar but different methodology.

Example:

```
$ find /home | cpio -oB >/dev/tape
$ cd /tmp; cpio -idB </dev/tape
```

`rsync` - remote sync - can sync a directory or whole filesystem by only transferring the changes between them. Be careful about trailing slashes.

Example:

```
$ rsync -a /home /backup/
$ rsync -a -e ssh /home backup@backup:/serverA/
```

Backup Applications

Arkeia - commercial package

BRU - commercial package

Amanda - Open source?

taper - mature open source (ncurses)

kdat - KDE backup tool (alpha software—not my data!)

Thousands more, some are client/server model and can backup multiple operating systems which is great.

See http://www.linuxhelp.com.au/free.shtml for our generic CPIO backup script.

Rotation & off-site strategies

It's no good having a backup if it's sitting next to the computer when there's a fire. You've got to have some off-site backups for really important stuff. On a small scale a friend of mine has a backup of all my music CD's I couldn't live without.

You could use this example strategy with any bulk medium but typically people refer to tapes or a set of tapes and for convenience I'll refer to a tape. If you can fit everything on one tape good for you, life is easy, backup the lot daily. If you don't you'll have to do an incremental backup (ie what's changed) daily and do a whole backup with multiple tapes weekly. Take the weekly backup off-site home from work or over to a trustworthy friend's place. Once a month take a weekly backup to long term storage and keep it for 7 years or something if it's got all your tax info on it. It goes without saying the tapes should be labelled full/incremental and a date, hostname

and what sequence in the set they are. Daily backup tapes may be rotated once a week with a new tape supplied once a week for a specific day of the week. Eg week1 will be all new tapes with one shipped off on Monday morning. week2 it'll be a new tape for Sunday morning, week3 it'll be Saturday mornings tape that's new. Alternately, some people believe the weekly or monthly should be on a fresh tape that's never been used.

With this strategy you get reasonable rotation of the tapes keeping costs down and for archival purposes, if you keep at least a months worth of data on the server you'll be able to go back to any point over the last few years and pull out a file. If you keep at least 3 months on hard disk you'll have 3 copies of this on 3 separate tapes because believe it or not they do fail and it will happen to you. To explain this more fully lets look at the following table and assume we have some wages data every week and the company's just started and there's 4 weeks per month.

server has	weekly tape has	monthly tape has
wk1	wk1	-
wk2	wk1-2	-
wk3	wk1-3	-
wk4	wk1-4,month1	wk1-4,month1
wk5	wk1-5,month1	-
wk6	wk1-6,month1	-
wk7	wk1-7,month1	-
wk8	wk1-8,month1-2	wk1-8,month1-2
wk9	wk1-9,month1-2	-
wk10	wk1-10,month1-2	-
wk11	wk1-11,month1-2	-
wk12	wk1-12,month1-3	wk1-12,month1-3
wk13	wk2-13,month1-3	-
wk14	wk3-14,month1-3	-
wk15	wk4-15,month1-3	-
wk16	wk5-16,month2-4	wk5-16,month2-4
wk17	wk6-17,month2-4	-
wk18	wk7-18,month2-4	-
wk19	wk8-19,month2-4	-
wk20	wk9-20,month3-5	wk9-20,month3-5
wk21	wk10-21,month3-5	-
....		

A complete backup and archive strategy should provide a means of going back to any point in time for critical data. Sometimes keeping the whole lot of data is not required. For example you could drop the weekly information and keep the monthly summary information and do a dedicated monthly backup for this data. The monthly data may be optimised and arranged for searching and an index provided but essentially contain all the information from the weekly data.