Data Integrity:
Backups and RAID

Track SA-E

AfCHIX workshop

Blantyre, Malawi

(Original slides by Phil Regnauld)

AfNOG

Introduction

“Keeping your data safe and reliable™”

Backups

- Types of backups
- Strategy
- Tools

RAID (Redundant Array of Independent Disks)

- Types of RAID
- What type to Use
- Disk Issues
- Hardware vs. Software RAID

Backup

- **What is backup?**
  - backup is part of a larger domain called data security:
    - integrity, protection: cryptography
    - availability, redundancy: mirroring / RAID
- **Why Backup?**
  - Software and Hardware failures are a common thing in the computer world. Any number of occurrences can cause loss of valuable data.

Backup

Types of failures

- Power failures (software/hardware failure)
- Natural disasters (fire, flood)
- Security incidents (theft)
- Hardware Failures (disk crash)
- User error (rm -rf)
- Social issues (stolen data)

A common backup strategy

“Do nothing”

Not a computer program, but it is the most widely used backup strategy. There are no initial costs. There is no backup schedule to follow. Just say no. If something happens to your data, grin and bear it!

If your time and your data is worth little to nothing, then “Do nothing”' is the most suitable backup program for your computer. But beware, UNIX is a useful tool, you may find that within six months you have a collection of files that are valuable to you.

“Do nothing’” is the correct backup method for /usr/obj, /usr/src and other directory trees that can be exactly recreated by your computer – but if in doubt, BACK IT UP!

Creating a backup strategy

This is driven by many variables. Such as:

- How long can you be offline before your org disappears?
- Do you have legal responsibilities.
- Levels of backup planned:
  - Daily
  - Weekly
  - Monthly
  - Quarterly
  - Semi-annually
  - Annually
- How long must you keep the data?
- How do you restore the data?
- Does your restore need to be “bare metal” or just data?
- Bare metal, fast restore, long-term storage = more $$

UNIX Backup Tools

Open Source options

- dd
- dump
- tar
- rsync (Apple's Time Machine uses this)
- Amanda
- Bacula (heavily used, very popular)

dd

- The lowest level type of backup
- Bit-for-bit copy
- For example:

```
dd if=/dev/ad0s1a of=/backup/root
```

- Exact copy, but not efficient
  - if you only use 100 MB on a 1 GB partition, you still end up with a backup of 1 GB
  - compression helps, but you still spend time copying unused space
- Best for doing system recovery, or…
- Copying media (CD-ROMs, DVDs, etc.)

Dump

- The traditional UNIX® backup programs
  - dump and restore.
  - Works at inode level
  - Takes backups of entire file systems, but only the used space
  - It is unable to backup only part of a file system
- It does not backup across mount points (directory tree that spans more than one file system)

- **Note**: If you use dump on your / partition, you would not back up /home, /usr or or any other mounted FS. You must explicitly run dump for each FS.

Dump

- Dump can backup to several media
  - local file
  - remote file
  - tape
- Dump can take incremental dumps
  - only files that have changed are backup up

Remote dump

It is possible to run dump over ssh for a secure transport:

```
# /sbin/dump -0uan -f - /usr | gzip -2    |      \
ssh targetuser@targetmachine.example.com |      \

dd of=/backups/dump-usr.gz
```

Anyone asking, “Where’s the `if` parameter for dd?

Tar

- tar(1) (Tape Archive) dates back to Version 6 of AT&T UNIX (circa 1975). tar operates in cooperation with the file system; tar writes files and directories to tape or to a file.

- Just like with dump, one can use ssh to backup across the network:

```
# tar -cfz - /
   | (ssh remote;
      cat >/backups/backup-0425.tgz)
```

Examples using tar

- Let's take a backup of /etc where most configuration files reside, and place it in /home/backups:

```
# mkdir /home/backups
# tar -cvf /home/backups/etc.tar /etc
```

Note: The -c option to tar tells it to create an archive, -v specifies verbose output and -f specifies the file to be either written to or read from

- You'll see quite a lot of output as tar creates the archive at this point.

Examples using tar

- Now we check whether our archive has actually been created

```
# cd /home/backups
# ls
```

- This now show us a new file in this directory `etc.tar`
- If we now wanted to view the contents of this backup we can run

```
# tar -tvf etc.tar
```

Examples using tar

- This will show you the contents of the etc directory as you backed it up.

- To actually restore and and unpack the contents that were backup up previously:

```
# cd /home/backups
# tar -xvf etc.tar
```

Examples using tar

- Notice that the restore actually creates a new directory etc where you are located – not in `/etc`!

- This is because tar by default removes the leading '/' from the directories it has backed up in order not to overwrite the original files on your system when you choose to do a restore (a security consideration)

Rsync

- Another very powerful tool is rsync
  http://samba.anu.edu.au/rsync/

- Rsync is very efficient: it only transfers files that have changed, and for those files, only the *parts* of the files that have changed
  - This is very efficient for large trees with many files, some of them large
- Great for replicating a server off-site, or for doing quick backups for a migration.

Rsync

- Combined with the `--link-dest` option, it allows to do snapshot-like backups.
- `--link-dest` takes the newest backup, and makes links (which take 0 space) to the files that have not changed, and replicates those that have changed
- Allows for backup.0, backup.1, backup.2, backup.3, where backup.X is a COMPLETE copy of the replicated source, but the disk space used is ONLY the difference.

Rsync – example script

- On remote backup host:

```
# rm -rf /backups/etc.2
# mv /backups/etc.1 /backups/etc.2
# mv /backups/etc.0 /backups/etc.1
# mv /backups/etc /backups/etc.0
```

- On machine to be backed up:

```
# rsync -avHS \
 --link-dest=etc.0 \
 /etc/ host:/backups/etc/
```

- This will backup only changed files from /etc/ to host:/etc/. Unchanged files are linked from etc.0

Other tools

- Rdiff-backup
  http://www.nongnu.org/rdiff-backup/
- Unison
  http://www.cis.upenn.edu/~bcpierce/unison/
- Rnapshot
  http://www.rsnapshot.org/

Other possible Backup methods

- Disk duplication
  - Using the dd command mentioned earlier, it is possible to duplicate your entire disk block by block on another disk. However the source and destination disk should be identical in size or the destination must be bigger than the source.

- Another way of doing this is using RAID1 mirroring and hot swappable disks:
  - make sure the RAID volume is rebuilt (OK)
  - remove one of the two disks (call it “backup”)
  - replace “backup” with a fresh disk, let the RAID rebuild
  - take “backup” home

  Remember: RAID or mirroring is not backup. An “`rm -rf /`” on your RAID set works very well!

Other possible Backup methods

- Disk duplication (2)
  - instead of mirroring the two disks, make two filesystems, and use rsync to copy every night from disk 1 to disk 2
  - in case of user error (rm -rf), you can recover from disk 2, without having to pull the backup tapes out of the safe

  NOTE: IT DOES NOT HELP IF THE SERVER IS STOLEN OR THERE IS A FIRE, IF BOTH DISKS ARE IN THE MACHINE!

Networked backup systems

- There are a number of networked backup systems out there for backing up many servers to one or more backup servers, using tape drives or disk storage.

- In the Open Source world, two backup systems stand out:

  - AMANDA - http://www.amanda.org/
  - BACULA - http://www.bacula.org/

Amanda

- Advanced Maryland Automatic Network Disk Archiver
  - Has been around for many years
  - Networked backup
  - Support incremental backups to disk, tape
  - Can backup to a holding disk, flush to tape later
  - Encrypted data flows and backup data
  - Tape library / loader control and labeling
  - Windows backup using a windows client
  - All source code for Amanda is open source

Bacula

Written by the people who invented AutoCAD

- Extremely popular and well-tested. Claims to be the most popular Open Source, Enterprise-level backup package around.
- Impressive documentation (400- pages!), including a developer's guide and tutorial
- Support incremental backups to disk, tape
- Complete SQL backend (MySQL, PgSQL, SQLite)
- Encrypted data flows using TLS (standard!)
- Tape library / loader control and labelling
- Native Windows client
- Good documented scenarios for specific backup cases, including complete “bare metal” restore

Bacula: Supported OS's

Operating Systems	Version	Client Daemon	Director Daemon	Storage Daemon
GNU/Linux	All	X	X	X
FreeBSD	≥ 5.0	X	X	X
Solaris	≥ 8	X	X	X
OpenSolaris		X	X	X
MS Windows 32bit	Win98/Me	X		
	WinNT/2K	X	*	*
	XP	X	*	*
	2008/Vista	X	*	*
MS Windows 64bit	2008/Vista	X	*	*
MacOS X/Darwin		X	*	*
OpenBSD		X	*	
NetBSD		X	*	
Irix		*		
True64		*		
AIX	≥ 4.3	*		
BSDI		*		
HPUX		*		

The "big three"

1. Windows
2. Mac OS X
3. UNIX/Linux

Are fully supported in Bacula. Additional, typical "enterprise" OS versions are supported as well (HP/UX, AIX, Solaris, etc.)

Reminder: Backup security

1. Take the disks / tapes / CDs off site!
   -> it does not help if there is a fire or if tapes are stolen

2. Consider encrypting the data on the disks / tapes / CDs
   -> what happens if the tapes are stolen? what happens when you throw them out?

RAID

Redundant **A**rray of **I**ndependent **D**isks

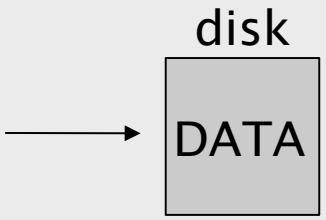
Redundant **A**rray of **I**nexpensive **D**isks

RAID 0
RAID 1
RAID 3
RAID 5
RAID 6
RAID 1+0 or 10

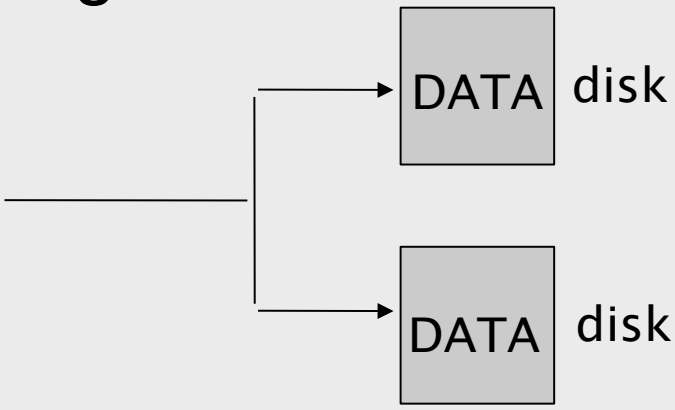
Types of redundancy

There are different levels of redundancy:

- **none** – if a disk crashes, data is lost

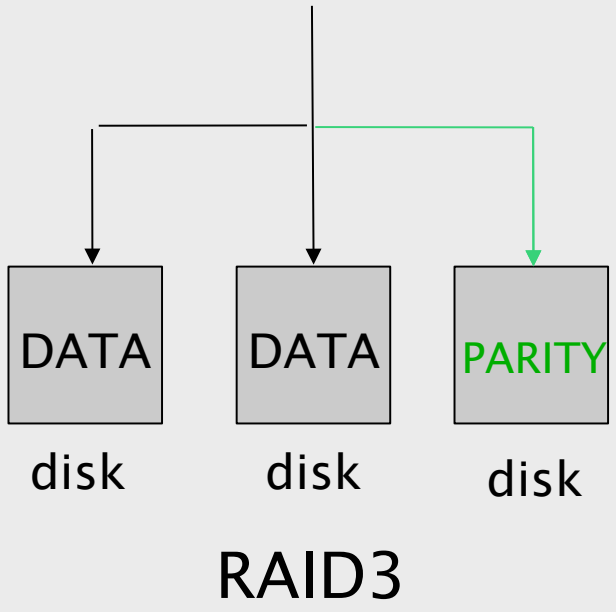


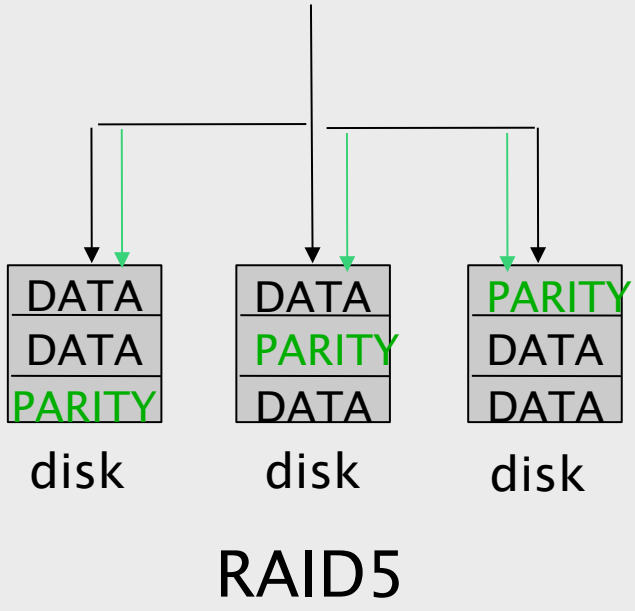
- **RAID1** – 2 disks are mirrored, data is written to both disks at any time. One disk can be lost without losing data.



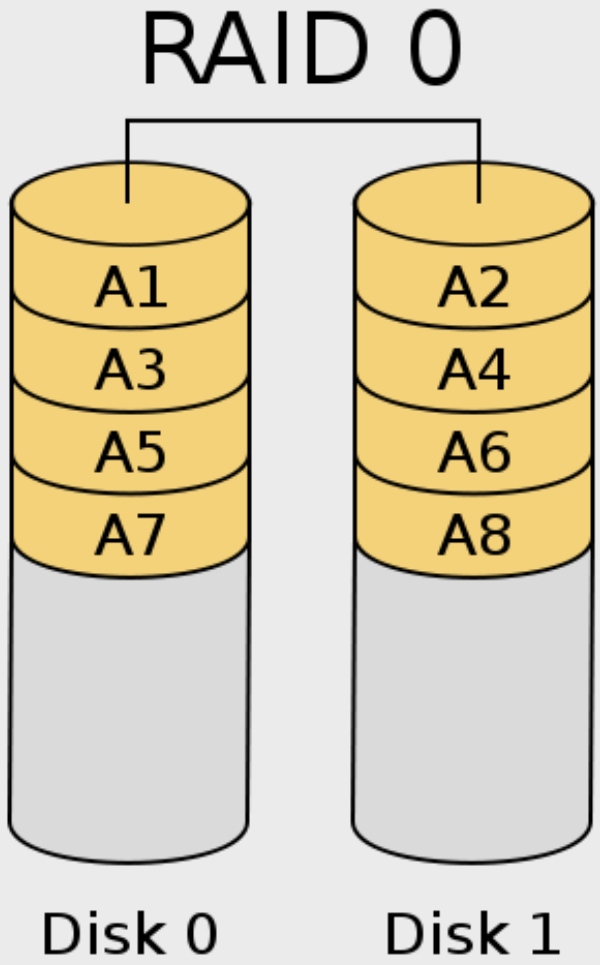
Types of redundancy

- RAID3, RAID5 – data is distributed across several disks, data parity, used to rebuild a defective drive, is either placed on a dedicated drive (RAID3) or across all drives (RAID5):





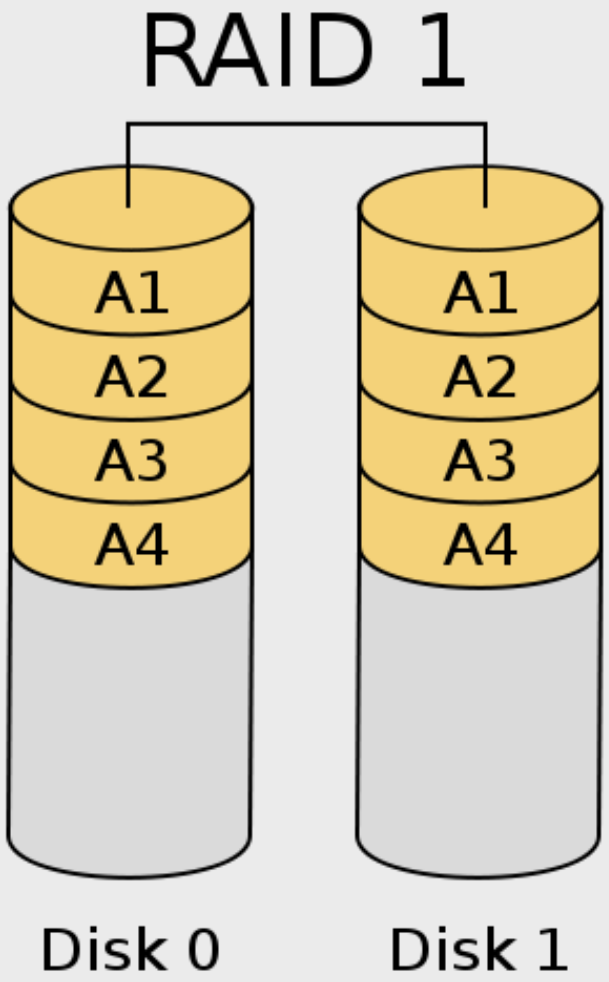
RAID 0



Striping

Not technically RAID, but a RAID card is used to implement. Data is striped between disks. Improves I/O in most cases.

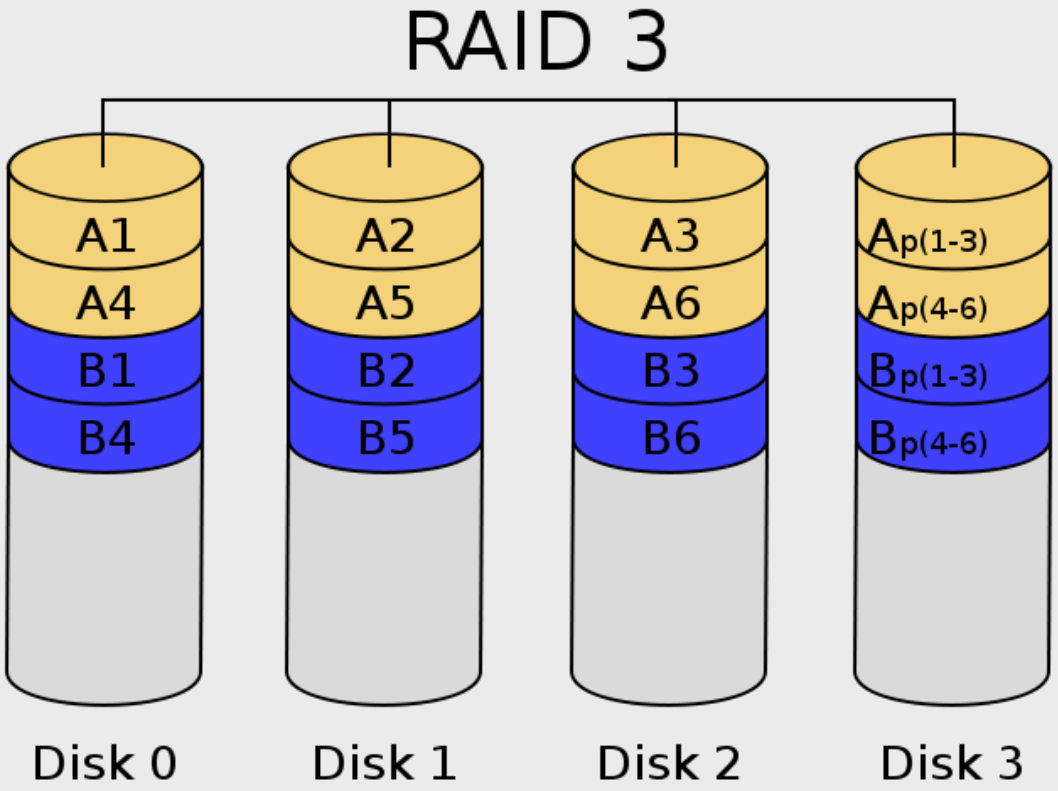
RAID 1



Mirroring

Disks are mirrored, data is written to both disks at any time. One disk can be lost without losing data.

RAID 3



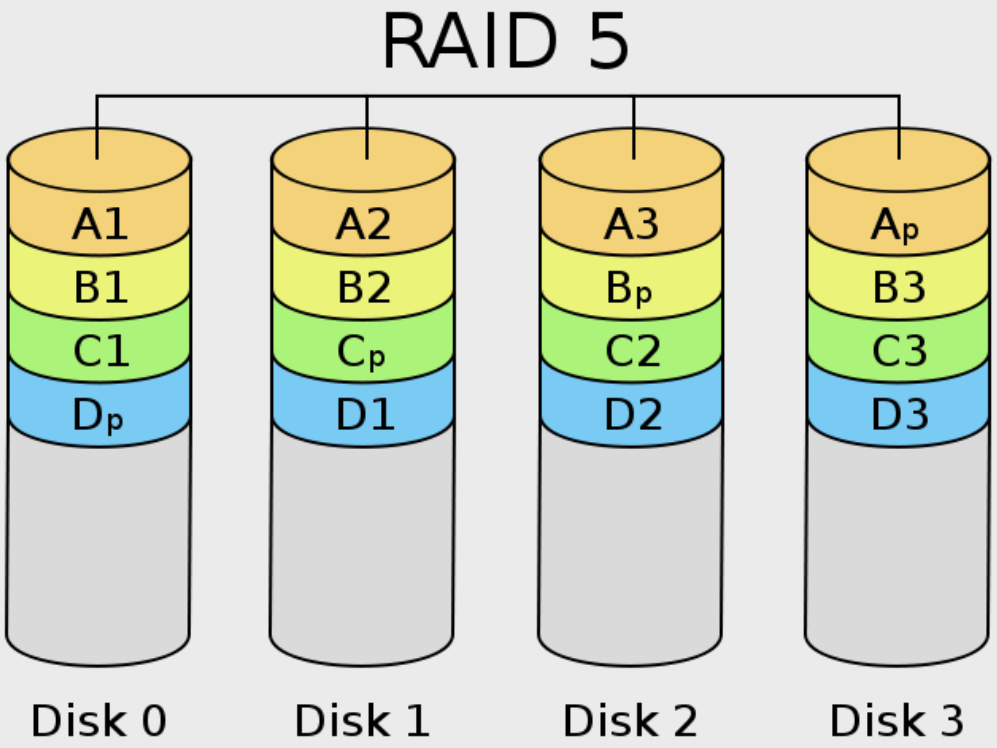
Striping +Dedicated Parity

Data is written across multiple disks (striping). A dedicated disk is used for parity.

Recovering from remaining disks plus parity disk. Lost parity disk = lost RAID array.

Fast I/O.

RAID 5



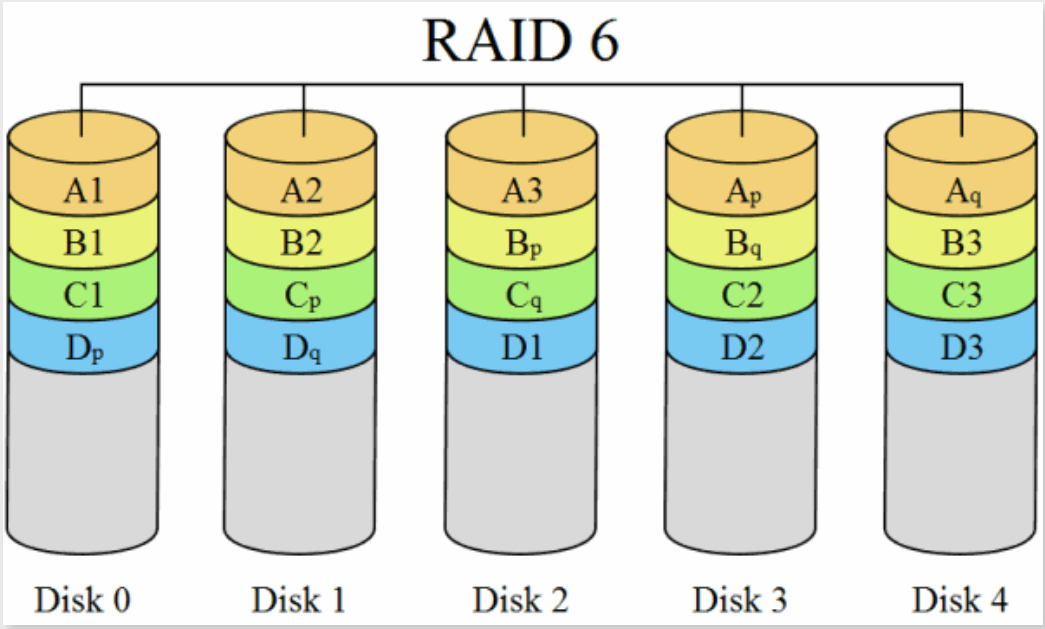
Striping +Distributed Parity

Data is written across multiple disks (striping). Parity is written across all disks.

Most popular type of RAID after RAID 1. Can lose any 1 disk (set of 3)

Has serious subtle issues!

RAID 6



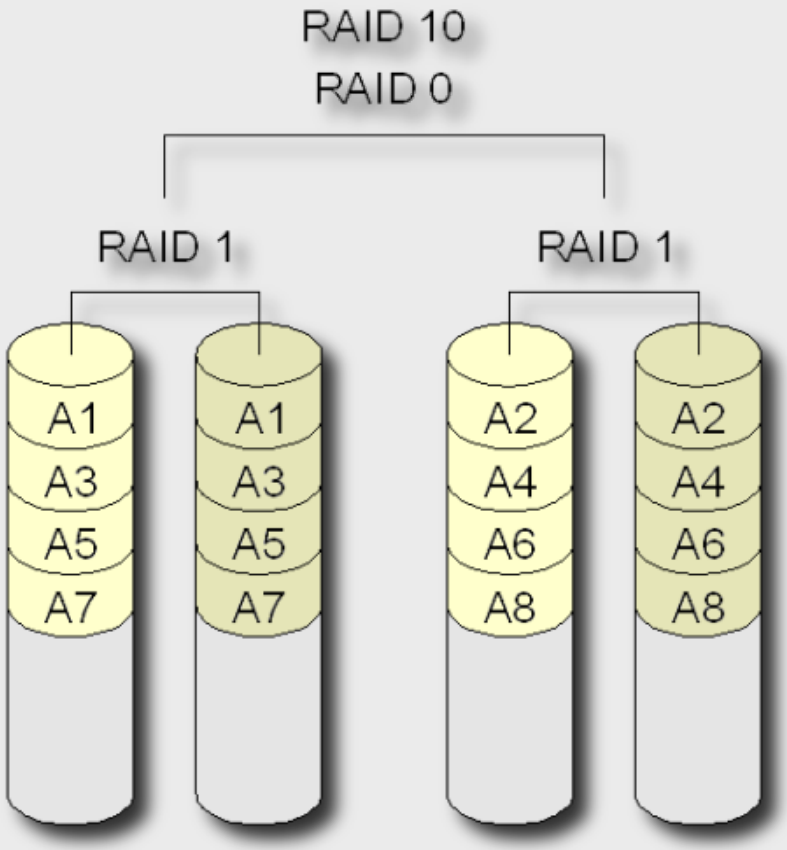
Striping+Double Distributed Parity

Data is written across multiple disks (striping). Parity is written across all disks multiple times.

Fixes issues with RAID 5. Can lose any 2 disk (set of 4).

Fixes issue with 1TB+ drives.

RAID 1+0 or “10”



Mirrored Sets in a Striped Set

Data is mirrored in multiple sets and sets are striped.

Provides performance and fault tolerance. Can lose multiple disks as long as no one mirror loses all disks.

Requires more disks for same storage space.

Referred to as “nested” or “hybrid” RAID.

RAID Controller Failure

What do you do?

Use a hot spare RAID Card.

- Card must be identical.
- Cards must support hot -spare in BIOS.
- Generally connected by on-board data path, or via cable between both cards.

Otherwise, at a minimum, buy 2xRAID card when building your array. If many arrays buy extra cards.

Fun Facts

Due to quantum physics…

- Error rate = 100% for 1TB+ drive writes
- RAID 6+ required to deal with this issue

Enterprise Class Drives

- Built to reduce vibration
- Reduced vibration = More reliable
- Cost a bit more, but essential in critical environments
- **May 2010**
  - **1TB**, 3Gb/s, 7200RPM, 32 MB Cache, 1.2 million hours MTBF SATA drives around USD $150/each.
  - **2TB**, 6Gb/s, 7200RPM, 64 MB Cache, 1.2 million hours MTBF SATA drives around USD $250/each.

http://www.baarf.com/

Dedicated to save us from RAID 5

Five. Five.

AfNOG

Hardware or software ?

- In *general*, hardware RAID is more transparent to the user, and disk replacement is straightforward:
  - remove defective disk
  - install new disk
  - RAID controller detects this and starts rebuilding on new disk
- (Note: real hardware RAID controllers, NOT BIOS RAID such as Promise)

Hardware or software ?

- RAID3 and 5 can be complex to implement in software (in the OS), so hardware might be a better choice
- But what happens if the RAID controller dies? How does one recover if one does not have a spare controller?
- Consider having a spare controller for RAID3/RAID5/RAID6/RAID1+0

(Note: we mean real hardware RAID controllers, *not* BIOS software RAID such as Promise)

Hardware or software ?

- RAID1 is easy to recover from and easier to implement in software (within the OS) – worst case, all one needs is to skip a header at the beginning of each disk.
- FreeBSD and Linux have very good software RAID implementations nowadays
- In FreeBSD, at least 3 implementations:
  - gmirror
  - ccd
  - gvinum (also RAID5, but not recommended)
  - But you want to use ZFS…

ZFS

Incredibly flexible, incredibly powerful file system available natively in FreeBSD.

- For all practical purposes has no file system or file size limits.
- Implements functionality of RAID6 with RAID-Z2 option of double-parity striping in the ZFS volume manage.
- ZFS creates FS over Volume over Hardware.
- ZFS knows about the hardware layer.
- Volume recovery is *fast*! Only recover data, not empty areas of disk (RAID recovers all bits in an array.

References

RAID Overview

http://en.wikipedia.org/wiki/RAID

ZFS

http://en.wikipedia.org/wiki/ZFS

ZFS on FreeBSD

http://wiki.freebsd.org/ZFS

ZFS on Linux (not completed)

http://zfs-on-fuse.blogspot.com/