# Unix System Backups

Chris Wilson
Aptivate Ltd, UK

AfNOG 2012

Download this presentation at:
http://www.ws.afnog.org/afnog2012/sse/backup

## Opening Questions

- How much data do you have?
- When did you last back it up?
- How much would it cost to back it all up, once?
  - Amazon costs $0.10 per GB per month
  - Hard disks cost $0.10 per GB for ~3 years?
- What is it worth?
  - What would happen if you lost it all?

## Why bother with backups?

- Recover from user error
  - Deleted files
  - Overwritten files
  - Corrupted filed
- Recover from a major disaster
  - Loss of an entire disk, system, office or data centre
  - Fire, theft, flooding, nuclear war

## User error

- Perhaps it happened a long time ago!
    - As much history as possible
- Perhaps it only just happened
    - As close as possible to real time
- Most likely to be used
    - Needs to be fast, easy and cheap to use
- Most likely to be small
    - Needs to be fast and easy to extract single files and directories

## Major disaster

- You'll know very quickly when it happens
  - History is not very important
- You've lost a lot of data
  - Needs to be as close to real-time as possible
- Downtime is extremely expensive
  - Needs to be fast to restore everything
- You can't rely on anything you own, or nearby
  - Keep it as far away as possible

## Different requirements

- Recovery from user error (*random access*):
  - Random access and fast seek time
  - Lots of history → small incremental writes or CDP
  - Onsite hard disks are a good choice
- Recovery from a disaster (*bulk*):
  - Fast bulk access (terabytes at a time)
  - Fast restore of entire system to latest snapshot
  - Frequent snapshot updates (→ not increments)
  - Offsite hot/cold spare systems are a good choice

## Complicating factors

- Security
  - Why are backups a security risk?
- Cost
  - Tapes are expensive and unreliable
  - Disks are expensive and unreliable
  - Amazon S3 is really expensive, but reliable
  - Actually taking backups can be expensive in time
    - When did you last back up **your** data? And **how**?
- Consistency
  - Databases and virtual machine images are hard!

## The plan

- Make a disaster recovery plan!
  - Plans are worthless. Planning is essential.
    *Dwight D. Eisenhower, general and US President*
  - By failing to prepare, you are preparing to fail.
    *Benjamin Franklin*
- Define (un)acceptable loss
- Back up everything
- Organise everything (for recovery)
- Monitor everything
- Document what you have done

## Backing up everything

- Files on file servers
- Files on desktops
- Files on laptops
- Databases and virtual machines
  - How important is consistency?
  - Can you stop the world? For how long?
- External systems
- Hardware (desktops, servers, networks)
- People

## Types of backups

- Full
- Differential
  - Everything since the last full backup
- Incremental
  - Everything since the last incremental

## Backing up files and systems

- Main types of backup software:
  - Snapshots
  - Continuous Data Protection (lsync, Box Backup, DropBox)
    - Closest to real-time, little or no history
  - File copiers (rsync)
    - Easy to restore
  - File archivers (tar, zip, duplicity, amanda)
    - Keep lots of history, restore to point-in-time, slow restore
  - System imagers (dump, Ghost, Ghost 4 Linux, Acronis)
    - Huge images, restore to point-in-time, restore all or nothing

# Software options

| Name | Type | Client | Server | Simple | History | Encrypted |
|---|---|---|---|---|---|---|
| snapshot | Local | Some Unix | - | High | Yes | No |
| rsync | Copier | Unix, Windows | Unix | Med | Not really | No |
| rsync s3fs | Copier | Unix with FUSE | S3 | Med | Not stable | Yes |
| rsync and snapshot | Copier | Unix, Windows | Some Unix | Med | Yes | No |
| rdiff-backup | Copier | Unix, Windows | Unix, Windows | Med | Yes | No |
| duplicity | Archiver | Unix | Any | Low | Yes | Yes |
| amanda | Archiver | Unix, Windows | Any | Med | Yes | Maybe |
| bacula | Archiver | Unix, Windows | Any | Med | Yes | Maybe |
| dump | Imager | Some Unix | Any | Med | Yes | Maybe |
| Ghost etc. | Imager | Unix, Windows | Any | High | Not really | Maybe |

# FreeBSD UFS snapshots (1)

- Create a snapshot:
  - ***sudo mkdir /var/snapshot***
  - ***sudo mount -u -o snapshot /var/snapshot/snap-120508-1400 /var***
- Mount it:
  - ***sudo mdconfig -a -t vnode -f /var/snapshot/snap-120508-1400***
    - Outputs the name of the device, e.g. *md0*
  - ***sudo mount -r /dev/md0 /mnt***
- What would you expect to see in */mnt*?

# FreeBSD UFS snapshots (2)

- Try it out!
- Unmount and release it:
  - ***sudo umount /mnt***
  - ***sudo mdconfig -d -u 0*** (for md0)
  - ***sudo rm /var/snapshot/snap-120508-1400***
  - The snapshot file is read-only, so it will ask you to "override" that to delete it; just enter "***y***"

## Pros and cons of snapshots

- Pros:
  - Very fast and efficient
  - Completely consistent view of filesystem, databases
- Cons:
  - Maximum of 20 per filesystem
  - Only on FreeBSD UFS and ZFS, Linux ZFS
  - No protection from disk corruption or crash

## rsync (1)

- Simple local file mirroring:
  - ***sudo rsync -avP /etc /var/tmp/etc-backup***
  - What would you expect to see in */var/tmp/etc-backup*?
- Simple file mirroring to another computer using ssh:
  - ***sudo rsync -avP /etc afnog@vmYY.sse.ws.afnog.org:vmXX***
  - Copies your */etc* to a subdirectory called vmXX on another computer vmYY, logging in as user afnog
  - You'll need to accept their host key and enter the password for their *afnog* user
  - Get their permission before logging into their computer!

## rsync (2)

- Generate an SSH key to replace the password
  - *sudo ssh-keygen*
  - Press Enter to accept the default location, and Enter twice to set no passphrase on the key
- Copy the SSH key onto your friend's computer:
  - *sudo cat /root/.ssh/id_rsa.pub | ssh afnog@vmYY.sse.ws.afnog.org tee -a .ssh/authorized_keys*
- Try it again:
  - *sudo rsync -avP /etc afnog@vmYY.sse.ws.afnog.org:vmXX*

# rsync (3)

- Log into *afnog@vmYY.sse.ws.afnog.org* (your friend's computer)
- What do you notice?
  - What does your backup look like?
  - How would you restore the files?
  - Is this a security risk? How?
- Can improve security of passwordless keys:
  - Restrict the commands that can be run
  - Restrict the IP addresses that can use the key
  - Chroot the backup user to protect the host

## rsync (4)

- To secure the *ssh* key:
  - On the destination side (your friend's server), edit the *.ssh/authorized_keys* file
  - Add the following text before “ssh-rsa”, on the same line:
  - ***command="rsync --server -av",no-port-forwarding,no-X11-forwarding,no-agent-forwarding*** ssh-rsa …
  - When you connect using ssh, you should now get just a flashing cursor instead of a prompt, and not be able to execute any commands.
  - Add ***from=vmXX.sse.ws.afnog.org*** to restrict IP address

# Pros and cons of *rsync*

- Pros:
  - Efficient use of network bandwidth
  - Very easy to restore files
- Cons:
  - Where's the history?
  - How do you verify your backup? Without using rsync?
  - Lots of small files are inefficient to store
  - No compression or encryption
  - Heavy disk I/O (scanning directories) impacts system

## tar (1)

- Simple archiving:
  - *tar czf etc-vmXX-120502.tgz /etc*
  - *scp etc-vmXX-120502.tgz backup@196.200.219.208*
- What does your backup look like?
- How do you restore it?
  - *tar xzf etc-vmXX-120502.tgz*
- What does it all mean?
  - "c" for Create, "t" for lisT, "x" for eXtract
  - "z" for compression, "j" for more compression
  - "v" for verbose (list files during operation)

## tar (2)

- How big is your backup file?
- How big are the files that you backed up?
- What if you wanted to store history?
  - Every 15 minutes for a year?
  - With 1 GB of files?
  - With 100 GB of files?

## tar for differential backups

- Create a directory for timestamps
  - *sudo mkdir /etc/backup*
- Run a full backup (weekly)
  - *sudo touch /etc/backup/daily*
  - *tar czf etc-weekly.tgz /etc*
- Run a daily differential backup
  - *tar czf etc-daily-diff.tgz --newer-than /etc/backup/weekly /etc*
- How would you restore?

## Pros and cons of *tar*

- Pros:
  - Ancient, reliable
  - Single file archives
- Cons:
  - Everything is manual: scheduling, encryption, shipping
  - Whole files are backed up
  - Difficult to use tapes efficiently
  - Slow to restore files from a large archive
  - Inefficient use of disk and network bandwidth
  - How do you restore a file to a specific date?

## dump

- Try to dump /etc:
  - *sudo dump 0Luf - /etc > etc-120507.0.dump*
  - dump: /etc: unknown file system
- So what can we dump?
  - *sudo dump 0Luf - /var > var-120507.0.dump*
- How big is the backup? The source data?
- Add a file, remove a file, run an incremental dump:
  - *sudo dump 1Luf - /var > var-120501.1.dump*
- How big is it? How long does it take?

## undump

- How to restore files from a *dump*?
  - ***restore -if /var/tmp/usr-120507-full.dump***
- How to restore an entire *dump*?
  - ***newfs -U /dev/ad0s1d*** (for example)
  - ***mount /dev/ad0s1d /mnt/target***
  - ***cd /mnt/target***
  - ***restore -rf /var/tmp/usr-120507-full.dump***
  - Important: you need space in /tmp to be able to restore!
- How to list files in a *dump*?
  - ***restore -tf /var/tmp/usr-120507-full.dump***

## Pros and cons of *dump*

- Pros:
  - Works with FreeBSD snapshots for consistent view
  - Fast restores of individual files or whole filesystems
- Cons:
  - What about Linux and Windows systems?
  - Can only dump whole filesystems
    - ***chflags nodump /var/tmp***
    - ***ls -ldo /var/tmp***
  - Whole files are backed up
  - Needs a lot of free disk space to store dump files
  - How do you restore a file to a specific date?

## Installing Amanda server

- Install the package and create directories:
    - *sudo -E pkg_add -r amanda-server aespipe*
    - *sudo mkdir -p /var/amanda /usr/local/etc/amanda*
    - *sudo chown amanda /var/amanda /usr/local/etc/amanda*
    - *sudo -u amanda /usr/local/bin/bash*
    - [amanda] $ *mkdir -p /var/amanda/vtapes/slot{1..25}*
    - [amanda] $ *mkdir -p /var/amanda/holding*
    - [amanda] $ *mkdir -p /usr/local/etc/amanda/MyConfig/{curinfo,log,index}*

## Configuring Amanda server (1)

- Copy the sample configuration file:
  - ***sudo cp /usr/local/share/amanda/example/amanda.conf /usr/local/etc/amanda/MyConfig/***
- Edit */usr/local/etc/amanda/MyConfig/amanda.conf*:
  - mailto "***your-email@example.com***"
  - infofile "/usr/local/etc/amanda/***MyConfig***/curinfo"
  - logdir "/usr/local/etc/amanda/***MyConfig/log***"
  - indexdir "/usr/local/etc/amanda/***MyConfig***/index"
  - tapedev "***chg-disk:/var/amanda/vtapes***"
  - holdingdisk hd1 { … directory "***/var/amanda/holding***" }

## Configuring Amanda server (2)

- Still in */usr/local/etc/amanda/MyConfig/amanda.conf*, uncomment and change:
  - autolabel "***MyConfig***-%%%" empty
  - labelstr "^***MyConfig***-[0-9][0-9]*$"
  - define dumptype server-encrypt-fast {
    - server_encrypt "/usr/***local/***sbin/amcrypt"
  - define dumptype global {
    - ***auth "ssh"***
    - ***ssh_keys "/var/db/amanda/.ssh/id_rsa"***

## Configuring Amanda server (3)

- Generate an SSH key for the amanda user:
    - ***sudo -u amanda ssh-keygen -t rsa -C "SSH Key for Amanda Backups"***

## Configuring an Amanda client (1)

- Install the client software:
  - ***sudo -E pkg_add -r amanda-server aespipe***
- Create directories for file list storage:
  - ***sudo mkdir /usr/local/var/amanda/gnutar-lists***
  - ***sudo chown amanda /usr/local/var/amanda/gnutar-lists***
- Copy the SSH key from the master:
  - ***sudo -u amanda cp /var/db/amanda/.ssh/id_rsa.pub /var/db/amanda/.ssh/authorized_keys***
- Unlock the Amanda account:
  - ***sudo chsh -s /bin/sh amanda***

## Configuring am Amanda client (2)

- Check that SSH login works (from the server):
  - ***sudo -u amanda ssh amanda@localhost echo It works***
  - Should output: "It works"
- Secure the SSH key:
  - Edit */var/db/amanda/.ssh/authorized_keys*
  - Add the following text before "ssh-rsa":
  - ***command="/usr/local/libexec/amanda/amandad -auth=ssh amdump",no-port-forwarding,no-X11-forwarding,no-agent-forwarding*** ssh-rsa …

## Configuring an Amanda client (3)

- On the server, create the file */usr/local/etc/amanda/MyConfig/disklist*:
  - ***localhost /etc high-tar***
- Test the configuration:
  - ***sudo -u amanda amservice localhost ssh noop </dev/null***
  - Should output: " `OPTIONS features= … ;` "
  - ***sudo -u amanda amcheck MyConfig***
  - Look out for lines starting with WARNING: or ERROR:

## Backing up with Amanda (finally!)

- Run a backup manually (from the server):
  - ***sudo -u amanda amdump MyConfig***
  - ***echo $?*** should output “0”
  - Check your email!
- Schedule automatic backups every 10 minutes:
  - Edit */etc/crontab* and add these lines:
  - ****/10 * * * * amanda /usr/local/sbin/amcheck -m MyConfig***
  - ****/10 * * * * amanda /usr/local/sbin/amdump MyConfig***

# Restoring with Amanda

- Install the Amanda client package **on the server**
  - We're using the same machine for both, so already done
- Create */usr/local/etc/amanda/amanda-client.conf*:
  - ***index_server "localhost"***
  - ***auth "local"***
- Run *amrecover*:
  - ***sudo amrecover MyConfig***
  - ***sethost localhost***
  - ***setdisk /etc***
  - ***setdevice***
  - ***lcd /tmp***
  - ***add rc.d***
  - ***extract***
  - ***exit***

# Monitoring Amanda

- Check your tape status:
  - ***sudo -u amanda /usr/local/sbin/amtape MyConfig show***
  - ***sudo du -sh /var/amanda/vtapes/****
- Check your last backup status:
  - ***sudo -u amanda amreport MyConfig***
- Using Nagios to check Amanda:
  - https://gist.github.com/30754 for *check_amanda.pl*

## Pros and cons of Amanda

- Pros:
  - Client-server network backup
  - Server-driven, minimal client configuration
  - Easy restore of individual files to a specified date
  - Works the same way for Windows clients (desktops)
  - VSS backup of open files on Windows clients
- Cons:
  - Difficult to configure!
  - Tape metaphor (virtual tapes) tricky to work with
  - Not designed for policies, e.g. full backup every Friday
  - No verify option!!!

## Using Duplicity (1)

- Install the package:
  - ***sudo -E pkg_add -rv duplicity***
- Run a full backup:
  - ***duplicity full /usr file:///var/tmp/duplicity***
- Run an incremental backup:
  - ***duplicity incremental /usr file:///var/tmp/duplicity***
- Disable password prompt
  - ***PASSPHRASE=afnog duplicity incremental /usr file:///var/tmp/duplicity***

## Using Duplicity (2)

- Backup to a remote machine via SSH:
  - Create an ssh key:
    - *ssh-keygen*
  - Copy it to a friend's computer:
    - *ssh afnog@vmyy.sse.ws.afnog.org tee -a .ssh/authorized_keys < ~/.ssh/id_rsa.pub*
  - Test passwordless login:
    - *ssh afnog@vmyy.sse.ws.afnog.org*
  - Test Duplicity:
    - *PASSPHRASE=afnog duplicity full /usr scp://afnog@vmyy.sse.ws.afnog.org/duplicity-vmXX*

## Using Duplicity (3)

- List files in backup:
  - ***duplicity list-current-files file:///var/tmp/duplicity***
- Restore files:
  - ***duplicity restore file:///var/tmp/duplicity /var/tmp/restored -r include***

## Using Duplicity (4)

- Verify your backup:
  - *PASSPHRASE=afnog duplicity verify file:///var/tmp/duplicity /usr*
- Exclude files:
  - *PASSPHRASE=afnog duplicity full /usr scp://afnog@vmyy.sse.ws.afnog.org/duplicity-vmXX --exclude /usr/home/afnog*
- List backups:
  - *duplicity collection-status file:///var/tmp/duplicity*

## Pros and cons of Duplicity

- Pros:
  - Dead simple to use
  - Secure, high strength encryption
  - Easily restore file to a specific time
  - Supports many backends: disk, FTP, SSH, Amazon S3
- Cons:
  - No tape support
  - Inefficient use of network bandwidth for full backups

## Where to Get Help

- AfNOG Mailing List
  - http://www.afnog.org/mailinglist.html
  - Please subscribe to this list!
- The Aptivate Team!