System Administration

Backups

Why Backup?

- Problems will occur
  - Hardware failure
  - Accidental deletion
  - Unwanted changes

Backup Philosophies

- At mimimum back up what you can not replicate
  - Your documents, important configuration files
- You can back up entire system
  - But the process to restore the system may be easier if you just install it from scratch and restore files
- Backup to something other than your machine
  - Put it on tape or other removable media
  - Back up to a remote machine

Test Your Backups

- You want to be sure you can restore your files ***BEFORE*** you have to.
- The last time you want to discover that your backup system isn't working is when you need to use it.

Redundancy

- Consider points of failure in your backup strategy
- If you archive everything to a remote hard drive, and that drive fails, what do you do?
- If you make backups onto tapes and keep multiple copies in multiple locations, and then your one tape drive fails, what do you do?

Automate, Automate, Automate

- If it is a hassle to do it, you won't do it
- Let the machine do the boring and repetitive work

Backup Types

- Full, incremental and differential

Full Backups

- Backs up a copy of every file you want backed up
- Advantages:
  - Everything is there in one backup
- Disadvantages
  - Full backups take the longest and require the most storage space

Incremental Backups

- Start with a full backup. Every next backup you only back up those files that have changed since the last backup
- Advantages:
  - After the full backup, each incremental backup tends to be smaller, takes less space and less time to create

Incremental Backups

- Disadvatages:
  - You are depending on an increased number of backups
  - For example: if you do a full backup on Sunday and an incremental every day, and you need to restore files on a Friday, you now depend on backups from Sunday, Monday, Tuesday, Wednesday, Thursday and Friday
  - More potential points of failure

Differential Backups

- Start with a full backup. Each backup, backup everything that has changed since the last full backup
- Advantages:
  - Restoring a particular backup depends only on the last full backup and a particular differential backup

Differential Backups

- Disadvantages:
  - The farther you get away from your last full backup, the larger your backups tend to be, and the longer they tend to take

Backup hardware

- Tapes, hard drives, DVDs?
- DVDs can store perhaps 4-8 GB of data
  - which is not a lot by today's standards
  - shelf life of burnable media is not really known

Backup Hardware

- Hard drives have a low initial investment, but a higher longer term cost
  - It it relatively inexpensive to buy a hard drive, but if you need more space, you need to buy another drive
- Tape drives have a high initial investment but a smaller long term cost
  - Tape drives are ~10x the cost of a hard drive, but tapes are cheaper than additional drives
  - Tapes also can survive falls off of shelves

Tar

- Tape ARchiver, a standard Unix utility
- Can archive to files as well as tapes

Tar Usage

- `tar -cvf <filename> <stuff to back up>`

  -c create

  -v be verbose

  -f create this tar file
- <filename> can be a .tar file, or a tape device
- `<stuff to back up>` can be individual files, or directories

Tar Usage

- What you put in <stuff to back up> determines how files are restored

-
  ```
  cd /home
  tar -cvf filename.tar myhomedir
  ```

  will give you myhomedir/file1
  myhomedir/dir1

Tar Usage

- `tar -cvf filename.tar /home/myhomedir`

  will give you home/myhomedir/file1

  home/myhomedir/dir1

Tar Usage

- `cd /home/myhomedir`

  ```
  tar -cvf filename.tar *
  ```

  will give you file1

  dir1

Extracting from Tar

- `tar -xvpf filename.tar`
  - -x extract
  - -v be verbose
  - -p restore permissions
  - -f read from this file
- restores everything to the current directory

Seeing what is in a tarball

- `tar -tvf` *`file.tar`*

  allows you to see what is in a tar file

Exercises

- create a tarball of everything in `/home/yourusername`, storing the tarball in `/tmp/my.tar`

```
cd
tar cvf /tmp/my.tar ./
```

Exercises

- Create the directory "/tmp/restore". Unpack /tmp/my.tar there

```
cd /tmp
mkdir restore
cd restore
tar xvf /tmp/my.tar
```

Rsync

- Remote SYNC
- Synchronize two directories, potentially over a network

Rsync usage

-
  ```
  rsync -av source dest
  ```

  -a archive (recursion, save permssions, etc)

  -v verbose

-
  ```
  rsync -e ssh -av source user@remote.machine:dest
  ```

  send over a network to a remote machine, using ssh

Rsync source specification

- if source ends in a slash, it will copy the contents of that directory, but not the name of that directory
- if source contains files a, b and c

```
rsync -av source/ dest
```

- will give you

```
dest/a
dest/b
dest/c
```

Rsync Source Specification

- If it does not end in a slash, the directory name is used

```
rsync -av source dest
```

- gives

dest/source/a

dest/source/b

dest/source/c

rsync over the network

- Again, either your source or your destination can be preceded with `username@remotemachine:`
- Recent enough versions of rsync will use ssh
  - older versions used rsh by default
  - you can specify `-e ssh`
- With ssh keys, this can be automated

Exercises

- Create the directory /tmp/backup/. Use rsync to back up your home directory to /tmp/backup in such a manner that /tmp/backup looks like

  /tmp/backup/home/linuxed/...

```
mkdir /tmp/backup
rsync -av   /home/linuxed    /tmp/backup
ls /tmp/backup
```

Exercises

- Now erase everything in /tmp/backup and use rsync to back up your home directory to /tmp/backup so that the files in /home/linuxed are right inside of /tmp/backup

/home/linuxed/file1 -> /tmp/backup/file1

```
rm -rf /tmp/backup/*
rsync -av   /home/linuxed/    /tmp/backup
ls /tmp/backup
```

Exercises

- Use rsync to back up your home directory to /tmp/backup as the user `linuxed` on the machine `localhost` over ssh in such a manner that /tmp/backup looks like

  /tmp/backup/home/myusername/...

```
rm -rf /tmp/backup

rsync -e ssh  -av /home/linuxed linuxed@localhost:/tmp/backup

ls /tmp/backup
```

RAID

- Quickes backup is to use a **R**edundant **A**rray of **I**nexpensive **D**isks
- Disks are cheap – buy lots, buy often
- Extra disks essentially copy the data of the other disks
- Disk *containers* look like one drive/partition to your OS

RAID

- Multiple levels – RAID 0, RAID 1, RAID 5, etc
- Can rebuild while system is running
- Use hot-swappable drives and keep back-ups around
- Hardware RAID – extra controller handles all the IO
- Software RAID – operating system handles RAID tasks

RAID 1

- *Mirror* one disk onto another disk
- Lose one disk, the other takes over while you rebuild
- Least efficient space usage
- Good idea for system disks

RAID 5

- Use several disks with *parity* and data bits *striped* across all disks
- Striping = different chunks of related data on separate disks
- Parity = number of 1s in a byte
- Advantages: More efficient use of disk space

RAID 5 Example

Four disks, with the following bytes on three:

D1: 00000111 D2: 0000101 D3: 00000000

Parity byte = D1 XOR D2 XOR D3 = 00000010

Lose disk 2, rebuild it with D1, D3 and Parity byte

D2 = D1 XOR D3 XOR Parity Byte

Other Options

- Open and Free source tools like AMANDA, Bacula
- Commercial products
- You may have an existing backup solution
- The most important thing is to do it, test it, automate it