UNIVERSITY OF TECHNOLOGY SYDNEY

Faculty of Engineering & Information Technology

31338 Network Servers
32520 Systems Administration

Week 7
Filesystems and backups
(Connect to Lab 7)

Professor Jian Zhang

School of Electrical & Data Engineering

University of Technology, Sydney

Filesystems (1)

- What we have learnt in week 2
  - Partitions: `/dev/sdb1, /dev/sdb5 (SATA devices)`
  - File system: ext4, swap
  - Create portions and filesystem, and mount, Steps: `1)fdisk, 2) mkfs, 3)mkswap, 4)mount`
  - Modify `/etc/fstab` filesystem table, system recovery if this file is broken
  - Retrieve filesystem statistics: `df, du, lsblk, blkid`
- **What is new in this week?**
  - Enforcing disk quotas
  - Using file permissions including Linux Access Control Lists (ACLs)
  - Changing file ownership
  - Creating/changing links
  - Familiarity with *Filesystem Hierarchy Standard* (FHS)
  - Locating files
  - Backup

Filesystems (2)

- A filesystem is a data structure used for storing files on a disk partition
  - Directory structure indexing performance and file volume
  - older Linux machines use ext2
  - formerly common: ext3 (ext2 + journal)
  - ext4 now the default as an enhanced ext3
    - ext4 is backward-compatible with ext3 and ext2
    - Support large file – up to 16 TB
    - Provide journaling function: tracking data for recover
  - **xfs** replaces the ext4 as the default file system since CentOS 7
    - support for accessing non-Linux partitions (e.g. NTFS – journalling file system, VFAT* – Windows PC or USB, HFS – Mac OS, ISO-9660 – CD/DVD, others)
  - Swap (RAM extension to HD) is technically not a filesystem, but usually treated in a similar way, create virtual memory using space on a physical device

File Allocation Table (FAT) VFAT is an extension of the FAT file system and was introduced with Windows 95

Creating partitions and filesystems

- Create/edit partition table
  - `fdisk` **command** – It is used for MBR** partition tables.
    - each partition has a partition ID/type (really just a "hint" to the OS)
      - **0x83** = Linux data (can be ext2/ext3/reiserfs/jfs/xfs)
      - **0x82** = Linux swap
      - 0x07 = NTFS, 0x0C = Win95 FAT32, 0xAF = HFS, etc.
  - `gdisk` command – It is used for GPT* indexing method
  - `parted` command – GNU program, `gparted` GUI program (install package)
- `mkfs` **command** – make filesystem (cf. formatting disk)
  - `mkfs -t ext4 /dev/sda3`
    - really just invokes `mkfs.ext4` command
  - Typical options include:
    - `-t` = type of filesystem
    - `-c` = bad block check
- `mkswap` **command** – make swap space
  - `mkswap /dev/sda4`
  - `swapon /dev/sda4`

* GPT – GUID (globally unique identifier) Partition Table. ** MBR – Mater boot record.

Lab 7a Task 3: tmpfs

- **Memory** based file system, don't need do mkfs
- Saved in virtual memory: Real Memory + swap (from hard disk, slower)
- **Default size**: half of Real Memory (*df –h*, and *top*)
- When Real Memory run out, using swap.
- **Swap** is not compulsory if Real Memory is large enough

tmpfs: Temporary File System

Mounting filesystems in lab 7

UNIVERSITY OF TECHNOLOGY SYDNEY
Faculty of Engineering & Information Technology

- `/etc/fstab` is filesystem table (what can be mounted)
  - The filesystems listed in /etc/fstab gets mounted during booting process.
  - `/etc/fstab` is mount table (what is currently mounted)
- Example fstab entries:

```
# Device     Mount point            FS type      FS options      F P
/dev/sda2    /                      ext3         defaults        1 1
/dev/sda1    /boot                  ext3         defaults        1 2
/dev/sda5    /home                  ext3         defaults        1 2
/dev/sda3    swap                   swap         defaults        0 0
/dev/sdb     /media/cdrom           iso9660      defaults        0 0
```

F = backup using dump utility (1 = yes, out of date), P = order of FS checking at boot time 1, after boot time 2 or 0 means ignore

- Mounting:

```
mount /media/cdrom
```
(if in fstab)

```
mount -t vfat /dev/sdb1 /media/usbdisk
```
(not in fstab)

- Unmounting:

```
umount /media/cdrom
```
(works regardless of fstab)

Newer /etc/fstab formats

- In the previous slide, `/etc/fstab` used partition names (devices)
- Two strategies to help avoid getting your partitions mixed up:
  - ext2/3/4 allows you to assign a LABEL to each partition
  - each ext2/3/4 partition has a unique identifier, its UUID

- `/etc/fstab` may use LABELs or UUIDs to refer to partitions, e.g.

```
# Device     Mount point             FS type          FS options       F P
LABEL=/      /                       ext3             defaults         1 1
LABEL=/boot  /boot                   ext3             defaults         1 2
```

```
For example, "LABEL=/" = "/dev/sda2"
```

- or:

```
# Device                Mount point  FS type          FS options       F P
UUID=abcd1234-56ef      /            ext3             defaults         1 1
UUID=9876fedc-54ba      /boot        ext3             defaults         1 2
```

```
For example, "UUID=abcd1234-56ef" = "/dev/sda2"
```

F = backup using dump utility (1 = yes, out of date), P = order of FS checking at boot (/ root first)
0 means ignore

Monitoring disk usage

- Monitor disk usage by partition: **df**

  df -k **#**Display all file systems and their disk usage in 1k block size (*Size Used Avail Use% Mounted on*)

  ```
  /dev/loop0 18761008 15246876 2554440 86% /
  /dev/sda3 174766076 164417964 10348112 95% /host
  ```

  df -h

  ```
  /dev/loop0 18G 15G 2.5G 86% /
  /dev/sda3 167G 157G 9.9G 95% /host
  ```

- Monitor disk usage by directory: **du**

  du /home/brookes **#** To find out the disk usage summary of a /home/brookes directory tree and each of its sub directories

  du -sh /home/brookes **#** To get the summary of a grand total disk usage size of a directory use the option "**-s**" as follows, e.g.,:

  ```
  685M    /home/brookes
  ```
  #show the subdirectory usage, -s provide summary

  du -sh *.txt #grand total disk usage for all .txt file

  -s: summarize **#**display only a total for each argument.

  -h: human-readable **#**print sizes in human readable format, e.g. 1k, 245M, 2G

- Quota report – when disk quotas are being used:

  repquota -a **#** Report on all filesystems indicated in **/etc/mtab** with quotas.

Note: /etc/mtab is a list of currently mounted filesystems.

Enforcing disk quotas (1)

- Set limits on user/group
  - What limit: maximum size, #inode
  - For example, set 500MB for user's `/home` directory
  - Require separate partition (`/dev/sdb1, 2…`) on `/home`, that is disk quotas are applied in partition only.
  - Steps:
    - Create a separate partition: e.g., `/dev/sdb1`
    - Mount `/dev/sdb1` to `/home.` `Each user has own home director under /home`
    - Set limit on `/dev/sdb1` or `/home` for user peter
    - Generate files in `/home/peter` to watch if the file size getting close to limit

A disk quota is a **limit set by a system administrator that restricts certain aspects of file system usage on modern operating systems**. With disk quota, we can set up the maximum disk space a user can use, for example, user Peter can use up to 500Mb space on partition 1.

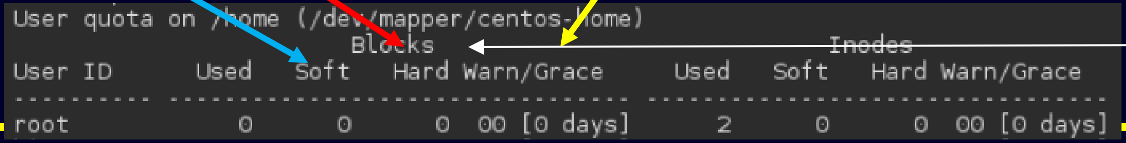
Enforcing disk quotas (2)

UNIVERSITY OF TECHNOLOGY SYDNEY
Faculty of Engineering & Information Technology

- Basic process of setting up disk quotas:
  - Add `usrquota` and/or `grpquota` option to `/etc/fstab`

```
# Device        Mount point       FS type        FS options                     F P
/dev/sda1       /boot             ext3           defaults                       1 2
/dev/sda5       /home             ext3           defaults,usrquota,grpquota     1 2
```

    - Note the difference between `usrquota and userquota ! When` editing fstab
  - Remount filesystem will make quota option take effect
  - Run `quotacheck` to generate initial quota file
  - Run `quotaon` to turn on quotas
  - Use `edquota` to edit a particular user's quota: `edquota -u peter`
- `edquota` modify in file:
  - hard: no further disk space can be used once the limit is reached
  - soft: can be exceeded for a certain amount of time
  - grace period: the amount of time a soft limit can be exceeded



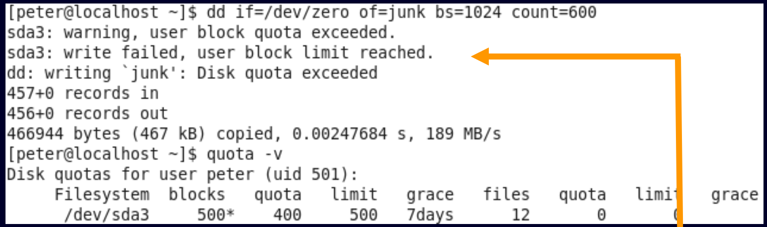
```
User quota on /home (/dev/mapper/centos-home)
                    Blocks                                Inodes
User ID       Used     Soft     Hard  Warn/Grace     Used     Soft     Hard  Warn/Grace
---------- ---------------------------------------- ----------------------------------------
root             0        0        0  00 [0 days]       2        0        0  00 [0 days]
```

blocks → current disk usage.

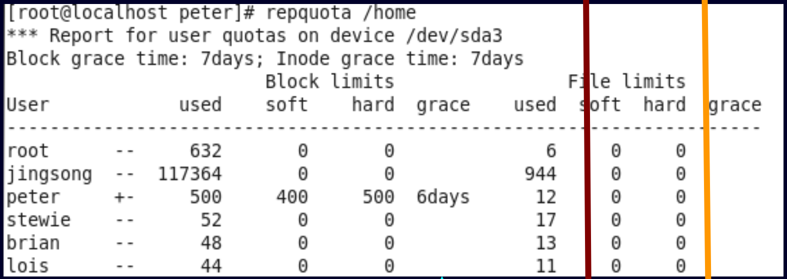
Faculty of Engineering & Information Technology

Lab 7b Task 1

- Before Tasks 1: mount /home partition
  - Backup /home: *cp –a /home /tmp/myhome*
  - Mount /home:
    - ***mount /dev/sda3 /home***
    - **In /etc/fstab file: /dev/sda3 /home ext4 defaults 0 0**
  - Restore /home: *cp -a /tmp/myhome/* /home*
- Steps:
  - *cp /etc/fstab /etc/fstab.bak*
  - In /etc/fstab file: /dev/sda3 /home ext4 defaults,usrquota 0 0
  - *mount -o remount /home*, check it by *df -h*
  - *quotacheck -cugm /home*
  - *quotaon -auv*
  - *edquota -u peter*, and put 400(500) for soft(hard)
  - Login with peter by "*su peter*" and then "*cd /home/peter*"
  - *dd if=/dev/zero of=junk bs=1024 count=600*
  - *quota -v* ***(unsuccessful!)***
  - Login with root, and run *repquota /dev/sda3* or *repquota /home*, you can see…



```
[peter@localhost ~]$ dd if=/dev/zero of=junk bs=1024 count=600
sda3: warning, user block quota exceeded.
sda3: write failed, user block limit reached.
dd: writing `junk': Disk quota exceeded
457+0 records in
456+0 records out
466944 bytes (467 kB) copied, 0.00247684 s, 189 MB/s
[peter@localhost ~]$ quota -v
Disk quotas for user peter (uid 501):
     Filesystem  blocks   quota   limit   grace   files   quota   limit   grace
      /dev/sda3     500*    400     500   7days      12       0       0
```



```
[root@localhost peter]# repquota /home
*** Report for user quotas on device /dev/sda3
Block grace time: 7days; Inode grace time: 7days
                        Block limits                File limits
User            used    soft    hard  grace    used  soft  hard  grace
----------------------------------------------------------------------
root      --     632       0       0              6     0     0
jingsong  --  117364       0       0            944     0     0
peter     +-     500     400     500  6days      12     0     0
stewie    --      52       0       0             17     0     0
brian     --      48       0       0             13     0     0
lois      --      44       0       0             11     0     0
```

Note: use man command to check the quotacheck. Quotaon etc

File permissions (1)

- 10-character code, e.g.
  - `drwx r-x --x` (4,3,3)=10 characters
  - First char is file type (`-` = file, `d` = dir, `l` = symlink, etc)
  - Next are permissions for users, group, others
  - Q: r, w, x for file vs. directory
  - Q: suppose directory /tmp: drwxr-xrwx, /tmp/file: -rw-r-----, can peter (other) delete /tmp/file?
- `chmod` command
  - `chmod {ugoa*}+/-/={r/w/x} file e.g. u+x; g+x?`
  - For octal, r = 4, w = 2, x = 1: e.g. 755 = `rwxr-xr-x`, 640 = `rw-r-----`
  - Special bits – setuid (4000), setgid (2000), sticky (1000)
    - setuid file = will execute as if run by owner of file
    - setgid file = will run with group permission set to group of file
    - setgid dir = files created in dir owned by dir group, not creator's group
    - sticky dir = only user who create file can delete it (e.g., `/tmp`), shared

* A combination of the letters ugoa controls which users' access to the file will be changed: the user who owns it (u), other users in the file's group (g), other users not in the file's group (o), or all users (a).

File permissions (2)

- **`umask`** = default file creation mode
  - sort of opposite of **`chmod`**
    - umask is subtracted from 777 for new directories (dirs), and from 666 for files
    - **`umask xxxx`** # temporary setting
    - insert in **`/etc/profile`**, **`/etc/bashrc`** for all users # persistent setting
    - Insert in **`~/.bash_profile`**, **`~/.bashrc`** for individual user # persistent setting
  - e.g.
    - **`umask 022`** = files created with **`-rw-r--r-(644)`**, dirs with **`drwxr-xr-x`** (755)
- ACLs = Access Control Lists
  - more sophisticated control than traditional Unix perms: **`setfacl`**
    - e.g., allow specific other users specific access levels
- **`chattr`** = change file attributes
  - some filesystems support extra attributes, e.g., immutable, compressed, append-only, secure deletion, no tail merging, etc.
  - **`chattr -a file1.txt #`** can only be opened in append mode for writing

Default permissions are 666 for files and 777 for directories.

File ownership

- `chown` = change owner (and group too)

```
chown brookes /tmp/somefile
chown brookes:staff /tmp/somefile
chown -R brookes:staff /tmp/somedir
```

#change owner to brookes, group of staff for all files and directories in /tmp/somedir

-R: operate on files and directories recursively.

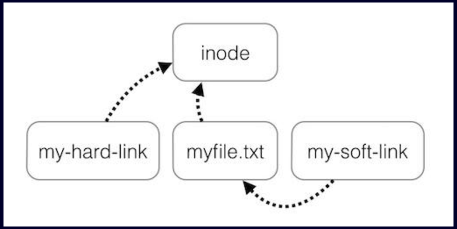
- `chgrp` = change group

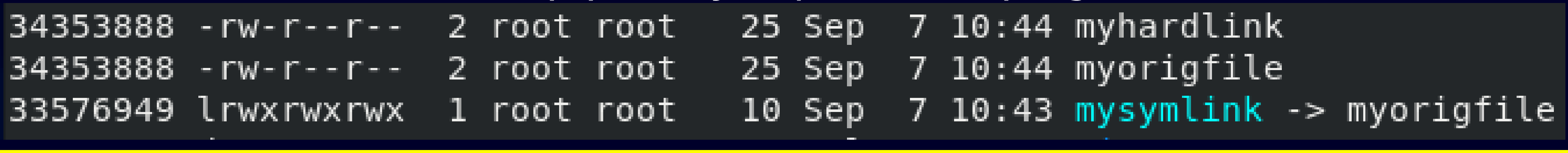
```
chgrp staff /tmp/somefile
```

- View current ownership with `ls -l` command

Symbolic and hard links – Linux (1)

- Links allow references to files by multiple names
  - Hard links: It creates two different directory entries that point to the same file (the same inode) -- **same reference count**
    - inode vs. filename: inode saves date, permission, owner, but no filename
    - Cannot hard-link across partitions
    - To delete a file with a hard link, must delete both references
    - `ln myorigfile myhardlink` #must be the owner of the file
    - `ls -il` will show that the *inode numbers* for both files are the same
  - Symbolic link (soft link): It creates a "pointer" to real file/directory
    - Can symlink across partitions, *different inode number*
    - If you delete where the symlink points to, any attempt to access the symlink will fail because the referenced file no longer exists
    - `ln -s myorigfile mysymlink` #-s option means symbolic
    - `ls -l` will show that mysymlink just points to myorigfile



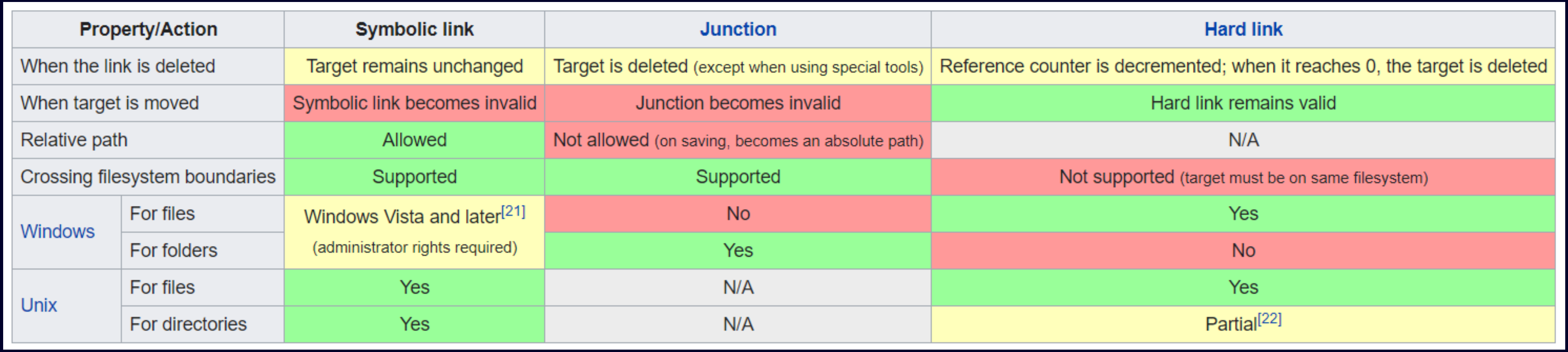


```
34353888 -rw-r--r--  2 root root   25 Sep  7 10:44 myhardlink
34353888 -rw-r--r--  2 root root   25 Sep  7 10:44 myorigfile
33576949 lrwxrwxrwx  1 root root   10 Sep  7 10:43 mysymlink -> myorigfile
```

Symbolic and hard links - Windows (2)

UNIVERSITY OF TECHNOLOGY SYDNEY
Faculty of Engineering & Information Technology

- Create Links
  - `mklink /H myhardlink myorigfile` #hard link
  - `mklink mysymlink myorigfile` #soft link for file
  - `mklink /D mysymlink myorigfolder` # soft link for folder
  - Shortcut (.lnk)
    - Handled **only by Windows Explorer** (not by the filesystem).
    - Always stores an absolute path to the target.
    - If you move the file to another drive or rename drive letters, the shortcut breaks (unless Explorer's heuristic "find target" manages to guess it).



Property/Action		Symbolic link	Junction	Hard link
When the link is deleted		Target remains unchanged	Target is deleted (except when using special tools)	Reference counter is decremented; when it reaches 0, the target is deleted
When target is moved		Symbolic link becomes invalid	Junction becomes invalid	Hard link remains valid
Relative path		Allowed	Not allowed (on saving, becomes an absolute path)	N/A
Crossing filesystem boundaries		Supported	Supported	Not supported (target must be on same filesystem)
Windows	For files	Windows Vista and later[21] (administrator rights required)	No	Yes
	For folders		Yes	No
Unix	For files	Yes	N/A	Yes
	For directories	Yes	N/A	Partial[22]

https://en.wikipedia.org/wiki/Symbolic_link

FHS

- Filesystem Hierarchy Standard, version 3.0, June 2015
  - formerly FSSTND (Filesystem Standard), from 1994
  - where file system on a UNIX system should be (not just Linux)
  - `man 7 hier`
  - e.g., of some directories:

	Shareable	Unshareable
Static	`/usr` `/opt`	`/etc` `/boot`
Variable	`/home` `/var/mail`	`/var/run` `/var/lock`

Locating files

- FHS says where files should be, but tools show you where particular files are in reality
- `find` command
  - flexible search by filename, owner, size, permissions, etc.
  - `find –name` "query"
- `locate` command
  - based on a database that is updated by `updatedb` **(usually once a day)**
  - less sophisticated than find, but much faster
  - `locate` query
- **`whereis:`**
  - Locates the binary, source, and manual page files for a command
  - **`whereis perl`** **#** List the directories where the perl source files, documentation, and binaries are stored.
- **`which`**
  - searches the executable file associated with a given command.
  - **`which sh # output is /bin/sh`**

Backups

- Operational goals:
  - Restore entire filesystems after crash (rare)
  - Restore individual files after users accidentally delete them (more common)
- Backup tools
  - tar*, cpio

cpio stands for "**copy in, copy out**". It is used for processing the archive files like

- Backup strategy
  - what to backup, when
- Backup automation
  - cron, at

*The Linux 'tar' stands for tape archive, which is used to create Archive and extract the Archive files.

Backup tools

- tar = Tape ARchive (none, dash -, --)
  - Support Unix-style option (-), BSD-style (none), and GNU style (--)
  - Doesn't have to read/write to tape – other media is fine
  - Backup: `tar cf files.tar /usr` # create archive containing /usr
  - List contents: `tar tvf files.tar` **#** t tells tar to list content, v tells tar to operate verbosely, f for filename
  - Restore: `tar xf files.tar` # x tells tar to extract files from an archive
  - Common options: `a, c, v, t, z, j, x, u, C, r`
- cpio = copy in and out (earlier)
  - Backup: `find /usr -print | cpio -o< namelist > /dev/rmt0`
  - List contents: `cpio -it < /dev/rmt0`
  - Restore: `cpio -i < /dev/rmt0` **Extract files from rmt0**
- dd = low-level tool for copying data byte-by-byte
  - e.g. copying from tape/CD to a file in "raw" format

    ```
    dd if=/dev/cdrom of=/tmp/mycd.iso bs=2048
    ```

Automation (1) – cron

- Automating backups, or running any job in the future
  - Three tools: cron, anacron and at
- `cron` = run commands at regular intervals
  - The software utility cron is a time-based job scheduler in Unix-like computer operating systems. Use cron to schedule jobs (commands or shell scripts) to run periodically at fixed times. It is a system daemon which can be started up at the beginning, it will read its own conf file /etc/crontab (cron tables)
  - Reads `/etc/crontab`, `/etc/cron.d` and user's own crontab file (`crontab -l` command – also see other `crontab` commands)
    - `/etc/crontab` often runs scripts in `/etc/cron.daily, /etc/cron.weekly, /etc/cron.monthly, /etc/cron.hourly`
  - `/etc/crontab` (not recommend to edit directly): check by `man 5 crontab`

```
02 4 * * * root /bin/somecommand
```

  - minute, hour, day-of-month, month, day-of-week, owner, command
  - From Root, type `crontab -e -u username` to create a user's cron jobs
  - From Users, type `crontab -e` to do the same job as above.
  - User access is controlled with `/etc/cron.allow` and `/etc/cron.deny`

Lab 7c-Implementing backups with cron

UNIVERSITY OF TECHNOLOGY SYDNEY
Faculty of Engineering & Information Technology

- Task 1: cron
  - *crontab* *-e*, # open the crontab with vi
  - Edit /etc/crontab # **Add** **min** **hour** * * * peter touch /tmp/peter
  - Check with *ls –la* to check owner of these files
  - *chkconfig –list crond*

-C Creates an archive by bundling files and directories together.
-V Display verbose info.
–f Specifies the filename of the archive to be created or extracted.
-t Displays or lists the files and directories contained within an archive.

- Task 2: tar* option
  - create tar: *tar -cvpf /tmp/backup-etc.tar /etc*
  - check tar: *tar tvf /tmp/backup-etc.tar*
  - create cpio: *find /opt - print | cpio -ov > /tmp/backup-opt.cpio*
  - check cpio: *cpio -it < /tmp/backup-opt.cpio*
  - restore tar: *tar –xvpf /tmpbackup-etc.tar -C /root* etc/yum.conf
  - check it from *"ls /root"*
- Task 3 refer to task 1
  - mins hour * * 0 xxxxxx (your command)

* https://www.geeksforgeeks.org/tar-command-linux-examples/ *p (preserve permissions)

Automation (2) – anacron, at

- `anacron` = catch up on jobs missed by cron
  - cron runs jobs at specified times, but if the computer is turned off when a cron job is scheduled, it does not run
  - anacron can ensure that regular maintenance tasks are performed at reasonable intervals on workstations, e.g., once 7 days (periodically)
  - On the day when the backup.sh job is supposed to be executed, if the system is down for some reason, anacron will execute the backup.sh script 15 minutes after the system comes back up (without having to wait for another 7 days).
  - configured in `/etc/anacrontab`
  - 7 15 test.daily /bin/sh /home/sathiya/backup.sh
  - period delay job-identifier command
  - Period: number of days, delay: delay in minutes to execute the command after machine starts up, job-identifier: a file containing one line indicating last time when this job was executed, command: Command or shell script that needs to be executed.
- `at` = run a command once in the future
  - `at -f /usr/local/bin/myscript 5pm Friday`
  - `atq` (shows jobs in the at queue)
  - `atrm 3` (removes job 3 from the at queue)
  - Access is controlled with `/etc/at.allow` and `/etc/at.deny`
  - at command can be useful for shutdown system at specified time, taking one time backup, sending email as reminder at specified time etc.