Learning About UNIX-GNU/Linux

Module 5: Internals and System Administration

The previous 4 modules in this series are intended to be fairly generic. This final module contains some generic information, but some other material is specific to the *UPSCALE* server *Faraday*. Even the "generic" parts sometimes aren't: different flavors of UNIX/Linux often lay out the configuration and maintenance materials discussed below in different ways.

There is a somewhat fuzzy line between the *system* part of Faraday and the *applications* side. Physics Computing Services (PCS) maintains the system side, and undergraduate staff maintains the applications side. Performing backups is also done by PCS.

There is a further document on the "nitty gritty" of maintaining Faraday, which I feel will be of no conceivable interest to anybody except the people who do the tasks discussed there. Thus, it is not included in this series.

- Filesystems
- More About Files and Directories
- System Configuration
- The root User
- Building Programs From Source
- The Red Hat Package Manager `rpm`
- Running Jobs Automatically
- Daemons
- Dealing With Windoze
- Some "Trivial" Maintenance Tasks
    - Mounting and unmounting the CDROM and floppy drives
    - Runaway processes
    - Logging out a user who forgot to log themselves out.
    - Cancelling a print job.
    - Changing a user's password
    - Restarting a service
    - Changing the "message of the day"
    - Rebooting
    - Halting the system
- Exercise 1
- Exercise 2

Filesystems

- Physical disks are partitioned into different *filesystems*.
    - Each filesystem has a maximum size, and a maximum number of files and directories that it can contain.
- The filesystems can be seen with the `df` command.

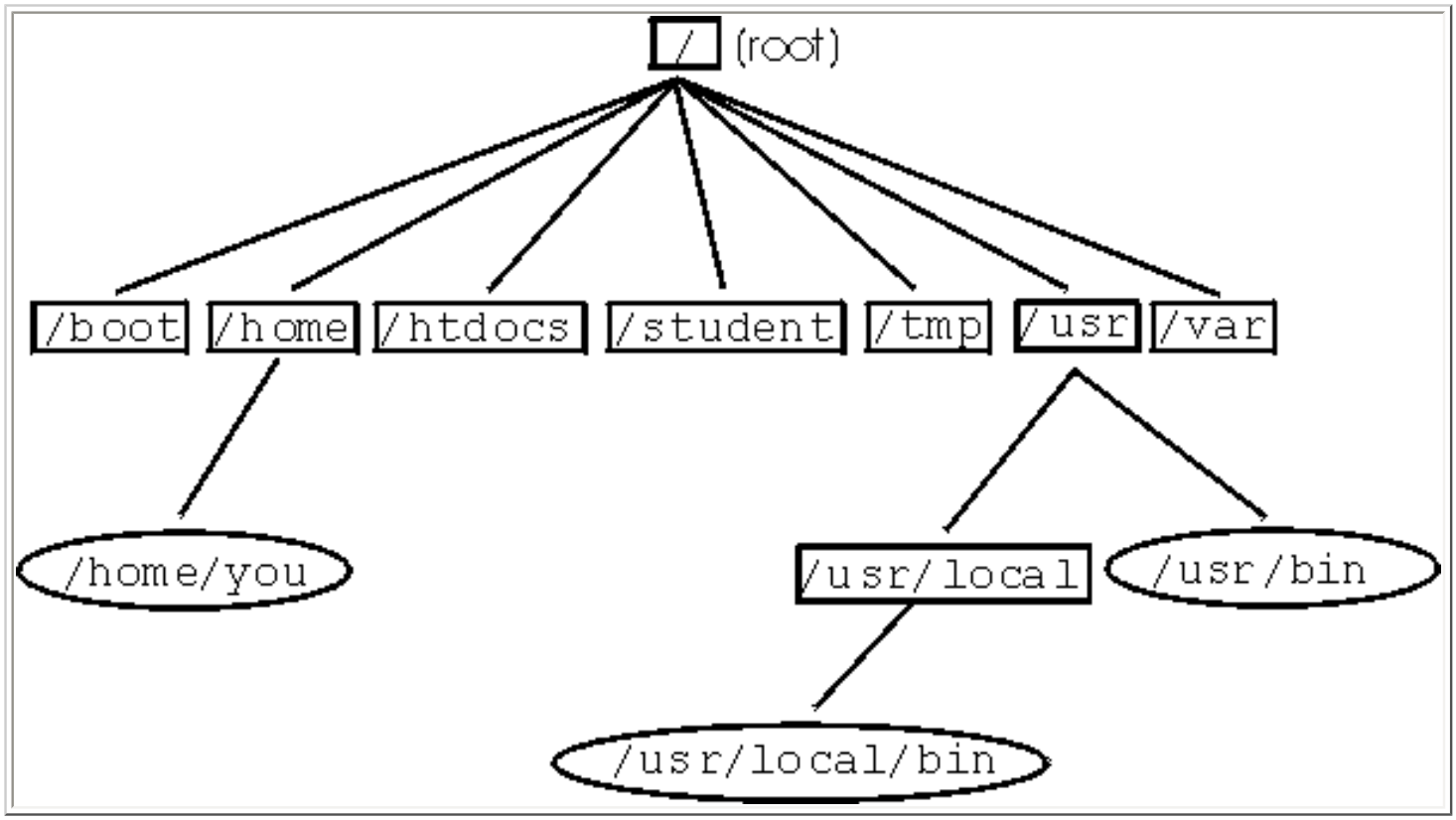
```
[you@faraday you]$ df
Filesystem           1k-blocks      Used Available Use% Mounted on
/dev/md0                497765    135940    336126  29% /
/dev/sda1               132207      7945    117436   7% /boot
/dev/md2               3937132   2854464    882668  77% /home
/dev/md1               4032000    341836   3485344   9% /htdocs
/dev/md5               7060152   3391736   3309776  51% /student
/dev/md4                497765      9370    462696   2% /tmp
/dev/md7               3241440   1240296   1836488  41% /usr
/dev/md6               6048196   2745272   2995692  48% /usr/local
/dev/md3                497765    275153    196913  59% /var
[you@faraday you]$ _
```

- A `-h` option to `df` makes its output more human-readable.

```
[you@faraday you]$ df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/md0              486M  133M  328M  29% /
/dev/sda1             129M  7.8M  114M   7% /boot
/dev/md2              3.8G  2.8G  861M  77% /home
/dev/md1              3.8G  334M  3.3G   9% /htdocs
/dev/md5              6.7G  3.3G  3.1G  51% /student
/dev/md4              486M  9.2M  451M   2% /tmp
/dev/md7              3.1G  1.2G  1.7G  41% /usr
/dev/md6              5.8G  2.7G  2.8G  48% /usr/local
/dev/md3              486M  269M  192M  59% /var
[you@faraday you]$ _
```

  - The first column is the physical device, a specified part of a particular hard disk.
    - On Faraday, each "device" is actually specified parts of two different disks. The two disks mirror each other in what is called a "RAID1" configuration.
      - *RAID* stands for "Redundant Array of Inexpensive Disks."
      - The idea is that if one disk fails, the other has a perfect copy of the contents.
  - The last column is the name of the filesystem as seen by users.
  - Different systems will have their filesystems laid out differently.
  - The `/` directory is called the *root* of the filesystem.
  - `/boot` contains the Linux kernel and various files related to the kernel.
  - `/home` contains the home directories for non-student and non-TA users.
  - `/htdocs` contains the documents we serve via the web.
  - `/student` contains the home directories for students, the directory where their commands are located `/student/sbin`, and a library directory for students `/student/slib`.
  - `/tmp` is a directory that is world readable and writable, and is used for temporary storage. We have it as a separate filesystem so that if it becomes full it doesn't clog other filesystems.
  - `/usr` is a number of bins, libraries etc. for users.
  - `/usr/local` is where local enhancements are placed whenever possible.
  - `/var` has various log files and system utilities. It is also where jobs are spooled for the printers and where user mail is kept.
  - The top level of every filesystem has a directory named `lost+found` which is used by the system for maintenance.
  - The top level of the `/student` and `/var` filesystems each contain a file `aquota.user` that puts quotas on maximum storage and number of files that may be stored by student users.
- The file `/etc/fstab` is the table describing the different filesystems.

The figure illustrates some directories on *Faraday*. Directories that are the top level of a filesystem are in rectangles, other directories are in ellipses. The string *you* indicates, as always, your login name.



Top

More About Files and Directories

- Above we said that everything is a file. That includes directories.
- Each file is assigned an *inode number* by the kernel.
  - Attributes in a *file table* in the kernel include its name, permissions, ownership, time of last modification, time of last access, and whether it is a file, directory or some other type of entity.
- A `-i` flag to `ls` shows the inode number of each entry.

```
[you@faraday you]$ lc
Directories:
some_directory

Files:
empty_file     some_file
[you@faraday you]$ ls -i
 258939 empty_file   258941 some_directory   258940 some_file
[you@faraday you]$ _
```

- A `-i` flag to df shows the number of inodes instead of the amount of space:

```
[you@faraday you]$ df -i
Filesystem            Inodes   IUsed   IFree IUse% Mounted on
/dev/md0                125k     20k    106k   16% /
/dev/sda1                33k      32     33k    1% /boot
/dev/md2                489k     90k    399k   19% /home
/dev/md1                501k     12k    489k    3% /htdocs
/dev/md5                877k    112k    765k   13% /student
/dev/md4                125k      95    125k    1% /tmp
/dev/md7                402k     71k    331k   18% /usr
/dev/md6                750k     39k    711k    6% /usr/local
/dev/md3                125k    1.5k    124k    2% /var
[you@faraday you]$ _
```

- You can form a *link* to a file, which associates a second name with the same inode, with `ln`

```
[you@faraday you]$ ln some_file link_to_file
[you@faraday you]$ ls -i
 258939 empty_file     258941 some_directory
 258940 link_to_file   258940 some_file
[you@faraday you]$ _
```

  - Note that inode 258940 is now associated with both `some_file` and `link_to_file`.
  - The syntax is: `ln existing_name new_name`
    - Note that the syntax is the same as for the copy command `cp`.
  - You can link from a file in any directory to another file in another directory provided both directories are in the same filesystem.
    - You can not link across filesystems.
  - The `ls -l` command lists the number of links to the inode in the second column.

```
[you@faraday you]$ ls -l some_file
-rw-r-----    2 you      users          32 Apr 29 12:49 some_file
[you@faraday you]$ _
```

  - If you remove one of the names associated with a particular inode, the other names are preserved.
    - The file is not removed from the filesystem until the "link count" goes to zero.
  - All permissions and ownerships of a link are identical to the file that you have linked to.
  - You may not link to a directory.
- You can form a *symbolic link* to a file by giving a `-s` flag to `ln`.

```
[you@faraday you]$ ln -s some_file symln
[you@faraday you]$ ls -i
 258939 empty_file     258941 some_directory   258961 symln
 258940 link_to_file   258940 some_file
[you@faraday you]$ _
```

  - The symbolic link has its own inode number.
  - The link count for `some_file` and `link_to_file` has not changed.
  - `ls -l` now displays that `symln` is a symbolic link:

```
[you@faraday you]$ ls -l symln
lrwxrwxrwx    1 you      users           9 Apr 29 13:24 symln -> some_file
[you@faraday you]$ _
```

    - The `l` that begins the line indicates a symbolic link.
    - The listing indicates world permissions for everything, but in fact the permissions of the file itself are

preserved.
        - The final column shows explicitly where the symlink points.
    - You can form a symbolic link to a directory.
        - To remove a symbolic link to a directory, use `rm`, not `rmdir`.
    - You can form symbolic links across filesystems.
    - If you remove the file or directory that a symlink points to, it still points to that name. If you then re-create the file or directory, the symlink will point to the new version.

Top

System Configuration

- Most of the material in this section is PCS's territory, but is probably useful for you to know.
- Virtually all system configuration files and directories are in the directory `/etc`
    - Many of the files and directories in `/etc` are symlinks to other files and directories.
- Modern UNIX/Linux systems have multiple *runlevels*.
    - Level 5 is the default for Faraday. It is full multiuser with X running.
    - Level 1 is single user mode. For use by experts when something has gone terribly wrong.
- The file `/etc/inittab` controls which runlevels correspond to which services. It is used by the "master" process `init`.
    - `init` is the first process to be run when the system boots.
    - It always has process identification number 1.
- The directory `/etc/rc.d/` has control over many of the services that may or may not be started for each runlevel.
    - The `.d` suffix is often used, as here, to indicate a directory.
    - Different flavors of UNIX/Linux sometimes set things up differently. Different releases of the same flavor sometimes change things too. We are describing fairly recent RedHat Linux distributions.
    - The directory has a number of files with `rc` in their name, each of which are run when the system boots.
    - The directory `init.d` contains the master copies of the shell scripts that are capable of starting and stopping processes.
    - The directory `rc5.d` contains processes that will be run under runlevel 5.
        - Each file in the directory is a symlink to a file in `init.d`.
        - If the name of the symlink begins with `S`, then the process is run.
        - If the name of the symlink begins with `K`, then that process is not run.
        - The numbers following the leading `S` in the name are the order in which each process is started. This insures that everything begins in the correct order.
    - The program `setup` is a GUI-interface to mark which services should be started.
- There are many others files and directories in `/etc` that control other aspects of the system.

Top

The root User

- The user named *root* has absolute power over the system.
    - Virtually all checks of permissions etc. are not performed for *root*.
    - This coupled with the "the user is always right" philosophy of UNIX/Linux means you can completely vaporize a system with a single typo.
- The best way to become *root* is from the system console.
    - You should log everything you do as *root* except for the totally trivial.
    - When in doubt, *nothing* is trivial: log it.
    - The log for Faraday is on the bottom shelf of the South wall of MP121C.

- If you are already logged in you can become root by:

```
[you@faraday you]$ /bin/su -
Password: _
```

  - If you are logged in to an X-terminal or emulator **secure the keyboard before executing `/bin/su`**, as discussed in Module 4 here. You can unsecure the keyboard after you have typed the password and pressed `Enter`.
  - Typing `/bin/su` instead of just `su` insures that you get the correct program.
  - Your `PATH` variable will now include two system bins: `/sbin` and `/usr/sbin`.
    - I adopted the name `sbin` for the name of the bin directory for students before the system bins `/sbin` and `/usr/sbin` became part of typical UNIX/Linux distributions. I haven't felt the need to change my naming convention.
  - Omitting the trailing hyphen gives you root privileges but with the same environment, (`$PATH`, present working directory, etc.) that you had when invoked the program.
- The root user always has a sharp sign # as the final component of the shell prompt.

```
[root@faraday some_directory]# _
```

  - If UNIX/Linux were not always *terse*, the shell prompt for root might be:

```
Be careful you turkey! You are root! You can do damage! # _
```

- The root user is identified by a *user identification number* (uid) of 0 in the password file.
- Systems maintained by PCS have another login in the password file with uid = 0, named *pcs*.
  - The *pcs* user has the same privileges as *root* .
  - The *pcs* password is not the same as the root password.
  - Even I do not know what that password is. Nor should I.

Top

Building Programs From Source

- For a source distribution, on Faraday we usually keep the source in a sub-directory of `/usr/local/src/`
  - Although in the examples below we shall be root and install the source and the built program in system areas, in practice I usually build a source distribution with my non-root login in my non-system areas first and test it there.
- The standard mechanism for distributing source is a "tar ball":
- `tar` is a *tape archive* program.
  - A `-f` option accepts a file name as its argument, which is used in place of the tape drive.
  - A `-x` flag extracts the contents of the file.
  - A `-z` flag uncompresses the tar ball.
    - Some versions of `tar` do not support a `-z` option. In this case, one uses the `zcat` command which uncompresses a file and sends the output to `stdout`. Then you can pipe the output to `tar`.
  - A `-t` flag lists the titles of the files and directories in the archive.
  - A `-c` flag creates a tar ball from the files given as arguments to `tar`.
- By convention, a distribution's tar ball will be named: `foo-XXX.tar.gz`:
  - `foo` is the name of the program.
  - `XXX` is the revision number, e.g. `2.3.0-1.71`.
  - The `tar` suffix identifies that it is a tar file.
  - The `gz` suffix indicates it is compressed, so `tar` will require a `-z` flag.
- A well-behaved distribution will unpack into a sub-directory `foo-XXX` of the present working directory. Thus, to unpack the distribution:

- Check that it is well-behaved:

```
[root@faraday some_directory]# cd /usr/local/src
[root@faraday src]# mkdir foo
[root@faraday src]# cd foo
[root@faraday foo]# cp <path to tarball> .
[root@faraday foo]# tar -tzf foo-XXX.tar.gz | head
foo-XXX/
foo-XXX/README
foo-XXX/Makefile
foo-XXX/foo.c
foo-XXX/foo.h
foo-XXX/docs/
foo-XXX/docs/foo.ps
foo-XXX/docs/foo.1
foo-XXX/config/
foo-XXX/config/sample.cfg
[root@faraday foo]# _
```

  - For versions of `tar` that do not support the `-z` option, use:

```
[root@faraday foo]# zcat foo-XXX.tar.gz |
> tar -tf -
```

    - The final hyphen **-** in the above is a synonym for `stdin`.
  - Note that all the files are unpacked into `foo-XXX/` as they should.
  - If the files are unpacked into the present working directory, any earlier files of the same name will be over-written. This is a bad thing if later you want to revert to a previous release.
- Now extract the files with:

```
[root@faraday foo]# tar -xzf foo-XXX.tar.gz
[root@faraday foo]# cd foo-XXX
[root@faraday foo-XXX]# _
```

  or:

```
[root@faraday foo]# zcat foo-XXX.tar.gz |
> tar -xf -
[root@faraday foo]# cd foo-XXX
[root@faraday foo-XXX]# _
```

- The program `make` uses a text file named `Makefile` to automate the building and installation of a program.
- For more complex cases, the source distribution ships with a shell script named `configure` which examines the system to produce a custom `Makefile`.
  - If such a file exists in the distribution, use it with:

```
[root@faraday foo-XXX]# ./configure > config.out
[root@faraday foo-XXX]# _
```

  - Make sure everything worked OK with:

```
[root@faraday foo-XXX]# more config.out
```

    - *Perl* programs often do it a little differently. They ship a configuration file written in Perl instead of the shell named `Makefile.PL`. You produce a `Makefile` with:

```
[root@faraday foo-XXX]# perl Makefile.PL
```

- Build the program with:

```
[root@faraday foo-XXX]# make
```

  - Usually there are other "targets" in the `Makefile` to test the built program or install it:

```
[root@faraday foo-XXX]# make test
```

```
[root@faraday foo-XXX]# make install
```

    - You should thoroughly test the program before installing it, even if the `test` target does not exist in the `Makefile`.
  - Usually there is a target to clean up the files produced in building the program. After installation you can use it like this:

```
[root@faraday foo-XXX]# make clean
```

Top

The Red Hat Package Manager `rpm`

- `rpm` is used to install, update, and uninstall programs either in source or binary form.
  - It can also be used to query whether a package has been installed and which version is installed:

```
[you@faraday you]$ rpm -q bar
bar-5.1.3-2
[you@faraday you]$ _
```

    - Version 5.1.3-2 of package `bar` is installed.
    - You do not need to be root to query installed packages.
  - You can also query all installed packages and use grep to find what you want:

```
[you@faraday you]$ rpm -qa | grep bar
bar-5.1.3-2
[you@faraday you]$ _
```

  - If you have a program, such as `/bin/cut`, and wish to find out which package owns it:

```
[you@faraday you]$ rpm -qf /bin/cut
textutils-2.0.11-7
[you@faraday you]$ _
```

- `rpm` is shipped with many distributions of GNU/Linux, and can be compiled for most flavors of UNIX/Linux.
- An rpm file is typically named: `foo-XXX.i386.rpm`
  - `foo` is the name of the package.
    - Often there are multiple programs, man pages, etc. contained in the package.
  - `XXX` is the revision number, e.g. `2.3.0-1.71`.
  - `i386` indicates it is for an Intel-based computer.

        - Other architectures can include `alpha` or `sparc` for machines using those cpus.
- On Faraday we store installed rpm files in `/usr/local/RPMS/`
    - The `/usr/local/RPMS/Security_7.1` directory contains security patches for our RedHat 7.1 installation.
- Packages often depend on other packages. These dependencies are always checked by `rpm`.
- You install a package with:

```
[root@faraday foo-XXX]# rpm -ivh foo-XXX.i386.rpm
```

    - You must be root to install, uninstall or freshen a package.
- You can uninstall a package with:

```
[root@faraday foo-XXX]# rpm -e foo
```

    - Note that we do not give the revision number.
- You can "freshen" (i.e. upgrade) an *existing* installed package with:

```
[root@faraday foo-XXX]# rpm -Fvh foo-XXX.i386.rpm
```

    - XXX is the revision number contained in the name of the package file, not the existing installed revision number.

Top

Running Jobs Automatically

- If authorised, you can repeatedly run jobs automatically at specified times with *cron*.
- If authorised, you can run a job at a specified time with `at`.
- As *root*, you can view the root user's file that runs under *cron* with:

```
[root@faraday root]# crontab -l
```

- On Faraday, the "master" copy of root's crontab file is `~root/crontab`.

Top

Daemons

- UNIX/Linux systems typically have a number of service programs for email, networks, etc. that are running continuously.
    - Such program are called *daemons*.
    - The name of the program often ends with the letter *d*, such as `crond`.
    - `ps -eaf` shows all programs running on the computer, including the daemons.
    - Below we describe only a few daemons.
- The Dynamic Host Control Protocol daemon `dhcpd` controls other devices on the network.
    - Used heavily by servers such as Faraday.
        - Not used much or at all by workstations.
    - It can assign IP numbers "on the fly."
    - IP numbers are assigned by PCS.
        - Do not choose one arbitrarily: get PCS to assign one.
        - Our IP numbers are of the form `128.100.86.XX`, where XX is two digits.

        - This is sometimes called the "86 subnet."
        - It is also sometimes called *pin*, for *Physics Instructional Network*.
    - On Faraday, we use `dhcpd` to assign fixed IP numbers to X-terminals etc.
        - The configuration file is `/etc/dhcp.conf`
        - Every ethernet card has a unique hexidecimal addres, its *MAC* address.
        - We use `dhcpd` to assign names and IP numbers to devices based on their MAC address.
        - We also use `dhcpd` to specify any files to download to the device.
- `xinetd` provides other services such as `tftp`.
    - The "trivial file transfer protocol" `tftp` downloads X-terminal boot files onto the device.
    - For our X-terminals then, `dhcpd` tell them at boot time what their address is and what file they need to download. Then `tftp` does the actual download.
    - The configuration is done with the file `/etc/xinetd.conf` and the files in the directory `/etc/xinetd.d/`
    - Some other UNIX/Linux flavors deliver `tftp` and similar services with a program called `bootp`.
- To cause a daemon, such as `xinetd` to restart with a modified configuration:

```
[root@faraday root]# service xinetd restart
Shutting down xinetd:                      [ OK ]
Starting xinetd:                           [ OK ]
[root@faraday root]# _
```

    - Any program listed in `/etc/init.d` can use the `service` shell script to restart.
    - Not all systems have this facility and not all daemons are listed in `/etc/init.d`. In these cases, if you send a signal to the daemon it will re-read its configuration. For `dhcdpd` the signal is called `SIGTERM`. Determine the pid of the daemon with `ps` and then:

```
[root@faraday root]# kill -SIGTERM <pid>
[root@faraday root]# _
```

        - Actually `dhcpd` can also be restarted with `service`.
        - The man page for the daemon should tell you which signal to send.
- `httpd` is the web server daemon.
    - Faraday uses a heavily customised version of the Apache server daemon..
    - Configuration is in the file `/usr/local/apache/conf/httpd.conf`.
    - Apache provides a utility similar to `service` to control the daemon, called `apachectl`. It may be used to restart the daemon with:

```
[root@faraday root]# cd /usr/local/apache/bin
[root@faraday bin]# ./apachectl graceful
apachectl: httpd  gracefully restarted
[root@faraday bin]# _
```

        - Historically, I believe that `apachectl` precedes `service`.

Dealing With Windoze

- `mcopy` can copy MS-DOS formatted text files from and to UNIX formatted text files.
    - Why MicroSoft later changed the existing simple UNIX format for text files to a slightly more complicated format is a mystery.
- `vi` and `emacs` can edit a MS-DOS formatted file, and when it is saved, it is saved in that format.
- *Samba* allows Windoze users to connect to directories on a UNIX/Linux system.

- Two daemons are associated with Samba:
  - `smbd` - the Samba server itself.
  - `nmbd` - the NetBIOS name server.
- The configuration file is `/etc/samba/smb.conf`
- Connection requires a valid username and password on Faraday.
- On Faraday, the user can connect to three directories:
  - `/tmp` - the temporary storage filesystem.
  - `/usr/local/pcbin` - a collection of Windoze programs and libraries.
  - Their home directory.
  - In Windoze-speak, these three directories are connected by Mapping a Network Drive.
- The user can print to any of the printers that are spooled by Faraday.
  - For student accounts, print accounting is done by the print spooler.
  - When students log in from, say, an X-terminal the system tells them if their print account is bankrupt.
  - When they connect from a PC via Samba no such statements are generated.

Top

Some "Trivial" Maintenance Tasks

- Mounting and unmounting the CDROM and floppy drives.
  - The directory `/mnt`, part of the root filesystem, contains two empty directories `cdrom/` and `floppy/`. The root user can *mount* the CDROM and floppy drives to these "mount points" which makes their contents available.
    - Similar "mount points" exist for all file systems except the root one.
      - For example, there is a directory in the root file system named `home/`, and the `/home` file system is mounted on that mount point at boot time.
      - The mount point does not have to be empty, but when another file system is mounted to it, the contents of the mount point are not accessible.
  - Recall that the UNIX/Linux philosophy includes the idea that *everything* is a file. That can include the CDROM and floppy drives.
  - You mount the drive with:

```
[root@faraday root]# mount /mnt/cdrom
[root@faraday root]# _
```

or

```
[root@faraday root]# mount /mnt/floppy
[root@faraday root]# _
```

    - Some systems automatically mount a CDROM drive if a disk is inserted.
  - If either of these are mounted they will be shown by `df`

```
[root@faraday root]# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/md0              486M  133M  328M  29% /
/dev/sda1             129M  7.8M  114M   7% /boot
/dev/md2              3.8G  2.8G  858M  77% /home
/dev/md1              3.8G  334M  3.3G   9% /htdocs
/dev/md5              6.7G  3.3G  3.1G  51% /student
/dev/md4              486M  9.2M  451M   2% /tmp
/dev/md7              3.1G  1.2G  1.7G  41% /usr
/dev/md6              5.8G  2.6G  2.9G  47% /usr/local
/dev/md3              486M  269M  192M  59% /var
/dev/scd0             257M  258M     0 100% /mnt/cdrom
[root@faraday root]# _
```

- You can then change to the mounted directories with:

```
[root@faraday root]# cd /mnt/cdrom
[root@faraday cdrom]# _
```

or

```
[root@faraday root]# cd /mnt/floppy
[root@faraday floppy]# _
```

  - All normal file and directory commands work as expected in these directories.
- You can unmount the device with:

```
[root@faraday root]# umount /mnt/cdrom
[root@faraday root]# _
```

or

```
[root@faraday root]# umount /mnt/floppy
[root@faraday root]# _
```

  - There is only one letter *n* in the `umount` command.
  - The command will fail if *any* user's present working directory is in the mounted directory's hierarchy.
  - As root you can `umount` any mounted filesystem. Don't!

- Runaway processes
  - Runaways are all too common.
  - To find them use `top`, which lists the programs that are the top consumers of resources.
  - To kill them, note the *process identification number* (PID) given by `top`.
  - Execute `kill -9 <PID>` where you will substitute for `<PID>`

```
[root@faraday root]# kill -9 123456
[root@faraday root]# _
```

    - The `-9` indicates that you are sending a *KILL* signal to the process.
    - You may instead use `kill -SIGKILL <PID>`, where again you substitute for `<PID>`.
- Logging out a user who forgot to log themselves out
  - Another all too common occurrence in our environment.
  - You will send a *KILL* signal to their login shell.
    - `ps -fu <USERNAME>`, where you will substitute for `<USERNAME>`, is a good way to determine the login shell.
    - The `<USERNAME>` is the login.

        - The login shell will have a hyphen prepended to it.
        - The students' login shell is `sh`, not `bash`.

```
[root@faraday root]# ps -fu bozo
UID        PID  PPID  C STIME TTY          TIME CMD
bozo     24971 24091  0 14:57 ?        00:00:00 /bin/bash -log
bozo     25005 24971  0 14:57 ?        00:00:00 /student/sbin/
bozo     25019 25005  0 14:57 ?        00:00:00 xclock -geomet
bozo     25021 25005  0 14:57 ttyp0    00:00:00 -sh
bozo     25084     1  0 14:57 ttyp0    00:00:00 xterm_tekmgr
bozo     25090 25021  0 14:57 ttyp0    00:00:00 main
bozo     25091 25090  0 14:57 ttyp0    00:00:00 bash -i
bozo     25102 25091  5 14:58 ttyp0    00:01:28 /usr/lib/netsca
bozo     25108 25102  0 14:58 ttyp0    00:00:00 (dns helper)
[root@faraday root]# kill -9 25021
[root@faraday root]# _
```

- Cancelling a print job.
    - Sometimes garbage gets sent to the printer, causing it to choke.
    - You can determine the *request id* assigned by the print spooler with `lpq -a`.
    - Root can use `cancel` to cancel any print job.
        - `cancel` was discussed in Module 3: here.
- Changing a user's password
    - Passwords are stored in an encrypted form.
        - The encryption is essentially "one way": getting the password back from the encrypted version is not feasible.
    - Our users often forget their password.
        - 1st and 2nd year student accounts:
            - Begin with the letter `x`.
            - Can not change their password.
            - Almost never have to have their password changed, since we keep a record of them.
            - Where the record is kept is discussed in another document.
        - Other users can and do change their passwords, and then forget what it is.
            - Change their password with `passwd <USERNAME>` where you will substitute for `<USERNAME>`
            - The `<USERNAME>` is the login.
            - I often use `ChangeMe` as the new password.
            - The traditional password in this circumstance is `stoopid`.

```
[root@faraday root]# passwd bozo
Enter new password: _
```

        - When you as a regular user try to change your password, you are asked for your old password first.
        - When *root* changes a password, either for himself or for another user, no such confirmation is required.
- Restarting a service.
    - This was discussed above.
    - On Faraday there are a couple of services that are somewhat flakey as of this writing. All others should be deferred to PCS.
        - `lpd`: the daemon for printing.
        - `xinetd`. If X-terminals do not display a login prompt after somebody logs out this is probably the culprit.
            - On April 12, 2002 I made a change to the configuration file `/etc/xinetd.d/tftp` that I am hopeful has fixed the problem with `xinetd`. We shall see.
            - If you are curious, the configuration file is text, so you can see its contents. Note that the change I made is documented and dated in the comments.
    - To restart the print daemon:

```
[root@faraday root]# service lpd restart
```

- To restart `xinetd`:

```
[root@faraday root]# service xinetd restart
```

- Changing the "message of the day"
  - This is the message that is displayed for all logins.
  - The contents are in the file `/etc/motd`
  - Long messages of the day are typically not read.
  - Besides the standard welcome message I typically only add announcements for:
    - When we are going to purge "x accounts" from the system.
    - When there will be service interruptions for significant amounts of time.
- Rebooting
  - Not really "trivial" but pretty simple.
  - Should only be done when Faraday has really gone crazy and PCS is not available.
  - Reboot from the console, not an X-terminal or ssh connection.

```
[root@faraday root]# shutdown -r now
```

  - Depending on how crazy Faraday is, the whole process takes somewhat less time than rebooting a Windoze box.
  - If this doesn't work you can try:
    - Holding down `Ctrl-Alt-F1` to get a non-X login prompt.
    - Hold down `Ctrl-Alt-Delete`.
  - Turning off the power and then turning it back on is not a suitable alternative.
    - Because the file system is always "up in the air" considerable damage can and probably will be done.
    - Repairing the damage is a job for PCS.
    - I have done this to reboot Faraday because nothing else would work. The results were not pretty.
- Halting the system
  - Should only be done when there is a fire or something similar and you have a couple of minutes before the flames get you.

```
[root@faraday root]# shutdown -h now
```

  - When the system says it is halted you may power it down.

Top

Exercise 1

- Create a file with any unique name and any contents that you wish somewhere in your directories.
  - Look at its link count with `ls -l` and its inode number with `ls -i`.
- Go to some other sub-directory of your home directory and create a link to the file you just created.
  - Look at the link count and inode of the new linked file you have just created.
- Change to `/tmp` and create a file in it with any name and contents that you wish.
  - Determine whether or not `/tmp` is part of the same filesystem as your home directory.
- Try to create a link to the new file you just created in `/tmp` from somewhere in your home directories.
  - If this succeeded, look at the contents of the version in your home directories.
- Create a symbolic link to the file in `/tmp` from somewhere in your home directories.
  - Verify the existence of the symlink with `ls -l`.
  - Look at the contents of the version in your home directories.
- Remove the file that you created in `/tmp`.

    - Verify the existence of the symlink with `ls -l`.
    - Try to look at the contents of the version in your home directories.
- In `/tmp` re-create the file you originally created there with the same name as before but with different contents.
- Look at the contents of the symlink in your home directories.
- Clean up by removing the files and links you have created in your home directories and in `/tmp`.

Exercise 2

- A sample tar ball `hello-1.1.tar.gz` has been prepared.
- Download the tar ball into some directory such as `~/src/hello` by clicking here.
- Unpack the tar ball.
- Change into `hello-1.1/` and look at the files and directories. You may look at the contents of any of the files that you wish since they are all text files.
    - The files ending in `.c` are C source files.
    - The file `hello.h` is a C "include" file.
- Do **not** use the file `configure` yet. Instead just type: `make`.
    - Look at the files that exist, noting any new ones.
    - Execute the new compiled binary with: `./hello`
    - Use `strings` on the new binary.
- Try `make test` and `make clean` and see what they do.
- Remove the `Makefile` and execute `./configure` to re-create it.
- If `rpm` is installed on your system:
    - Verify the the program `/bin/cut` is part of the `textutils` package.
    - Read the man page for `rpm` to find out how to find out all the files that are part of that package.