$Id: lab0u-intro-unix.mm,v 1.79 2015-12-07 14:37:46-08 - - $
PWD: /afs/cats.ucsc.edu/courses/cmps012b-wm/Labs-cmps012m/lab0u-intro-unix
URL: http://www2.ucsc.edu/courses/cmps012b-wm/:/Labs-cmps012m/lab0u-intro-unix/

1. Overview

This lab will introduce you to Unix (Linux) and basic commands used to navigate around the file system. You will learn how to submit assignments using the **submit** command on Unix. File and directory security and organization will also be covered, specifically for the AFS file system. Interacting with Unix is done by typing commands at the shell prompt. We will discuss only the **bash** shell.

2. Directories and ACLs

Organize your work by creating a separate directory for each course you take, and a separate subdirectory under that for each assignment or lab that you work on. In the following commentary, what the computer types out is shown in Courier plain type, and what the user types in is shown in Courier bold type.

```
bash-01$ cd
bash-02$ mkdir private
```

The command **cd** without arguments sets the current directory to **$HOME**, which can also be referred to by a tilde (**~**). The **mkdir** command creates a new directory. An error will be printed if you already have the directory.

```
bash-03$ fs setacl private $USER all -clear
bash-04$ fs listacl private
Access list for private is
Normal rights:
  foobar rlidwka
```

This has set an access control list (ACL) on the private directory, which limits access to the user **foobar**. The environment variable **$USER** always contains your user name. Any directories created under this directory will inherit the ACL from its parent. An ACL may be placed only on a directory, never on an individual file. The letters **rlidwka** show which access rights are available to which user :

r	(read)	allows read access to the files in the directory.
l	(list)	allows **ls**(1) to list the names of the files in the directory.
i	(insert)	allows new files to be inserted into the directory.
d	(delete)	allows files to be deleted from the directory.
w	(write)	allows writing (updating) to files in the directory.
k	(lock)	allows files in the directory to be locked.
a	(admin)	allows administrative (**fs setacl**) access to the directory.

If an access right is given as **system:authuser rl**, it means that any authorized user may read files and list directories. This is usually what is placed on course directories, such as the one you are looking at now.

If you are working from off campus, you will need to connect to the servers **unix.ucsc.edu**. Read *"Unix Timeshare : How to Connect"* :

http://its.ucsc.edu/unix-timeshare/tutorials/how-to-connect.html

When you are in the Unix lab, you do not need to use the servers.

3. Lab exercises

For this lab, you will be asked to create files and submit them using the **submit** command.

(01) You should have a separate directory for each course and a subdirectory of that for each lab or programming project. Create one for lab0. The shell variable **$HOME** and the tilde (**~**) both refer to your home directory.

```
bash-05$ cd ~/private
bash-06$ mkdir -p cmps012b/lab0
bash-07$ cd cmps012b/lab0
```

The **pwd**(1) command shows you which directory is your current directory.

```
bash-08$ pwd
```

The **ls**(1) command shows you the contents of a directory. The **-l** option produces output in long format and **-a** shows hidden files as well.

```
bash-09$ ls -la
bash-10$ pwd >01_private-lab0
bash-11$ ls -la >>01_private-lab0
bash-12$ cat 01_private-lab0
bash-13$ submit cmps012b-wm.w16 lab0 01_private-lab0
```

(02) Create a file called 02_date by redirecting the output of the **date**(1) command. Then submit it.

```
bash-14$ date
bash-15$ date >02_date
bash-16$ cat 02_date
bash-17$ submit cmps012b-wm.w16 lab0 02_date
```

Note that the first command prints to the terminal, while the second one redirects the output to a file.

(03) Verify that you have properly set security on your private file hierarchy. Also check your disk quota.

```
bash-18$ fs listacl ~/private
bash-19$ fs listquota
bash-20$ fs listacl ~/private >03_acl
bash-21$ fs listquota >>03_acl
bash-22$ cat 03_acl
bash-23$ submit cmps012b-wm.w16 lab0 03_acl
```

Note that **>** redirects output to a file, but **>>** appends output to an existing file. You can abbreviate the first operand of **fs** : **fs sa**, **fs la**, and **fs lq**.

(04) Verify that you are using **bash** and not **tcsh**.

```
bash-24$ echo $0
tcsh
bash-25$ bash
bash-26$ echo $0
-bash
bash-27$ echo $0 >04_which_shell
bash-28$ submit cmps012b-wm.w16 lab0 04_which_shell
```

So far, there have been no differences between the two shells, but later on in the course, there will be some little difficulties with **tcsh**. If the first **echo**

command displays **bash**, then you are already using **bash**. If you wish to change your shell to **bash** permanently, type the command **chsh** and follow instructions. Whether or not you change your shell is up to you, but there are some advantages to using **bash** as opposed to **tcsh**.

(05) The **getent**(1) command shows you information about your own username.

```
bash-29$ getent passwd $USER
bash-30$ getent passwd $USER >05_getent
bash-31$ submit cmps012b-wm.w16 lab0 05_getent
```

Its seven fields show the username, what used to be the password field (but is no longer used), your numeric userid, your numeric group number (which is the same for everyone), your actual name, your home directory, and your default shell.

(06) The file **~/.bashrc** in your home directory controls how **bash** starts up. It is not created by default, so using your favorite editor, create a file called .bashrc in your home directory with the following line :

```
alias sub12b="submit cmps012b-wm.w16"
```

Every time you start bash from the command line, all commands in this file are executed. To source it the first time

```
bash-32$ source ~/.bashrc
```

This is done automatically every time you start **bash**.

```
bash-33$ submit cmps012b-wm.w16 lab0 ~/.bashrc
```

You do not have to source this file the next time you log in.

(07) Also create a file **~/.bash_profile** with the line

```
source $HOME/.bashrc
```

This will cause your **.bashrc** to be sourced whether you start it as a login shell or subshell.

```
bash-34$ sub12b lab0 .bashrc .bash_profile
```

You can now abbreviate the **submit** command.

(08) To verify what you have submitted, use the command

```
bash-35$ ls -la /afs/cats.ucsc.edu/class/cmps012b-wm.w16/lab0/$USER
```

You may submit files as many times as you like before the due date. When you do submit files, they are prefixed with sequence numbers. So, for example, if you submit **foo** four times, you will see **1_foo**, **2_foo**, **3_foo**, **4_foo**, in the submit directory. Always verify what you have submitted. If you are not sure of what you submitted, submit again. Type this **ls** command again and redirect output into a file called **08_verifying**.

```
bash-36$ submit cmps012b-wm.w16 lab0 08_verifying
```

(09) Write a program in Java which prints the message **Goodbye, Java** to the standard error (**System.err**) and then uses **System.exit** to exit with exit status 1.

```
bash-37$ javac goodbye.java
bash-38$ java goodbye
Goodbye, Java.
bash-39$ echo $?
1
bash-40$ submit cmps012b-wm.w16 lab0 goodbye.java
```

(10) Create a shell script that compiles, runs, and prints the exit status of this program:

```
bash-41$ cat mk.goodbye
#!/bin/bash
javac goodbye.java
java goodbye
echo $?
bash-42$ chmod +x mk.goodbye
bash-43$ submit cmps012b-wm.w16 lab0 mk.goodbye
```

This final program is a shell script which automates a small task. The **x**-bit must be turned on for it to work.

This lab has asked you to submit 10 files. Verify that they are all submitted.

4. Naming files

(1) Filenames should be spelled using only lower case letters, digits, periods, and underscores or minus signs. Upper case letters in filenames are generally to be avoided, except for special names such as **Makefile** and **README**, which must be spelled exactly that way. Commands like **ls**(1) sort filenames lexicographically and thus list capitalized names before lower case names. The following non-alphanumeric characters generally to not cause problems when used in filenames:

```
% + , - . : = @ _
```

(2) Shell metacharacters are prohibited in filenames. A slash (**/**) is a directory separator, so Unix will not let you use it in a filename, even if you quote it. Following is a list of shell metacharacters:

```
! " # $ & ' ( ) * / ; < > ? [ \ ] ^ ` { | } ~
```

(3) The tilde (**~**) only has special meaning when it is the first character in a filename, in which case it causes username interpolation.

(4) The plus (**+**) and minus (**-**) characters should never appear as the first character of a filename, since they also typically introduce command-line options.

(5) Be careful about using dot (**.**) as the first character because that makes files "hidden". The term "dotfile" is often used to refer to such a file. Control files such as **.bashrc** usually begin with a dot.

(6) And never use space or tabs in filenames!

5. Learning an editor

If you are already familiar with editing files on Unix, you may ignore this part. If not, try one of the following:

(1) The following command will give you a tour of **vim**:

```
bash-44$ vimtutor
```

(2) The following command will give you an introduction to **emacs**:

```
bash-45$ emacs &
```

Then select from the menu **Help** → **Emacs tutorial**.

(3) A beginner might prefer to use `pico` or `nano`, which are extremely simple editors, but are not used by professionals.

(4) Never under any circumstances use M*cr*$*ft W*rd to create a program file.

(5) Never cut and paste anything from a PDF into a text file. If you cut and paste, do so from a text file.

With apologies to Marcus Porcius Cato Maior (DXX–DCV AUC):
"Praeterea, censeo Microflaccidem esse delendam."

6. Grading guidelines

The subdirectory `.score` under this directory contains instructions to the graders. This directory will not be shown on the web, and with `ls`(1), only with the `-a` option.

7. Submit checklist

Carefully review the submit checklist:
`/afs/cats.ucsc.edu/courses/cmps012b-wm/Syllabus/submit-checklist/`
`http://www2.ucsc.edu/courses/cmps012b-wm/://Syllabus/submit-checklist/`