

CSE 399-004 Unix/Linux Skills

Spring 2006

Administrative

- Final exam
 - Due on Friday, 3/3 (note: this was changed)
- Homework 6
 - Due today
- Class on Wednesday, but no lab on Friday

Unix/Linux Skills - Marc Corliss

CVS

- Stands for concurrent versions system
- Records the history of your source files
 - Bugs can arise w/ new versions of software, w/ cvs can get back old versions
- Allows multiple people to simultaneously work on same project (and same files)
 - Each person works with their own copy of the files
 - When an individual is done editing they tell cvs to merge their changes
 - People can even edit the same files
 - However, if the edit the same line, there will be a conflict

Unix/Linux Skills - Marc Corliss

Running CVS

- CVS is a command-line tool
 - To run cvs, type "cvs ..." at prompt
- The "cvs" command by itself does nothing
 - Need to use a sub-command
 - For example: "cvs import" or "cvs checkout" (more on this later)
 - To see all sub-commands type "cvs --help-commands"
 - Can also man cvs

Unix/Linux Skills - Marc Corliss

Repository

- CVS uses a central repository to keep track of all projects
 - Repository is some directory usually in one of the users home
 - To use cvs, need to set environment variable CVSROOT
 - Which is simply the path to the repository
- To add a project to the repository use "cvs import"
 - Move to the directory of the project and type:
\$ `cvs import <projectname> sample init`
 - <projectname> is the name you will use to refer to the project
 - "Sample" and "init" are not important, can substitute whatever string you want

Unix/Linux Skills - Marc Corliss

Permissions

- The permissions of the repository matter
- You do not need to change the permissions of the files
 - These should be read only
- You may want to change the permissions of the directories
 - This determines who can work on the project
 - Whoever has write permission to a directory can use cvs for that project
 - You change permissions just the same way as you change any other directory in Unix/Linux

Unix/Linux Skills - Marc Corliss

Checking Out A Project

- Once a project is added to the repository, can check it out
 - Immediately after adding project, should use checked out copy
- To check out a project do the following:
`$ cvs checkout <projectname>`
 - Will add a directory called <projectname> with all of its contents to the current directory
 - In each directory is a directory called CVS (meta-info. used by CVS)

Unix/Linux Skills - Marc Corliss

Committing Changes

- At some point after making changes, need to commit those changes
 - Can do this using "cvs commit":
`$ cvs commit -m "Made some changes"`
 - Updates the central repository
 - -m flag specifies a message to accompany this commit
 - Should make the message as descriptive as possible
 - If you omit -m option, then will open an editor (vi by default) and ask you to enter the message
 - Can set the CVS_EDITOR environment variable to change CVS's default editor
 - By default "cvs commit" commits all changed files
 - Works recursively, like most CVS commands
 - You can also specify which files you want to commit:
`cvs commit -m "Made a change to foo.c" foo.c`

Unix/Linux Skills - Marc Corliss

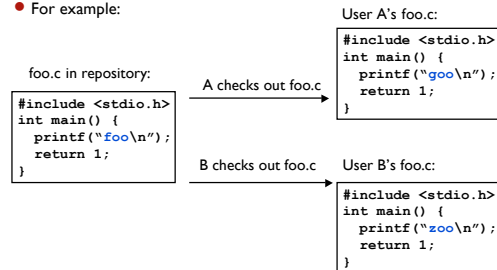
Working with Multiple Users

- When working with others, need to update occasionally
 - Probably right before you begin to edit a file
- To update, use "cvs update":
`$ cvs update foo.c`
 - By default cvs update will update all files unless a file is specified
- After finishing editing, should commit right away
 - This reduces the chance for a conflict
 - i.e., two people editing the same line in a file and trying to commit

Unix/Linux Skills - Marc Corliss

Merging Files

- Needs to be some coordination between multiple users
 - For example:



Unix/Linux Skills - Marc Corliss

Conflicts

- User A commits foo.c, then user B tries to as well
 - User B gets an error: must do an update first

```
$ cvs commit foo.c
cvs commit: Up-to-date check failed for 'foo.c'
cvs [commit aborted]: correct above errors first!
```

- When user B updates, gets another error: there's a conflict

- Same line was edited
 - If A and B had edited different lines than update would have succeeded

```
$ cvs update
cvs update: Updating .
RCS file: /home1/m/moorliss/systems1/cvs/tmp/foo.c,v
retrieving revision 1.1.1.1
retrieving revision 1.2
Merging differences between 1.1.1.1 and 1.2 into foo.c
rsmerge: warning: conflicts during merge
cvs update: conflicts found in foo.c
C foo.c
```

Unix/Linux Skills - Marc Corliss

Conflict Shown in File

- CVS shows the conflict in the file:

```
$ cat foo.c
#include <stdio.h>
int main() {
<<<<<< foo.c
    printf("zoo\n");
=====
    printf("goo\n");
>>>>>> 1.2
    return 1;
}
```

- New changes are between "<<<<" and "====",
Old changes are between ">>>>" and "===="

Unix/Linux Skills - Marc Corliss

Getting the Status

- If you want to find out the status of a file, use “cvs status”:

```
$ cvs status foo.c
=====
File: foo.c          Status: Needs Merge
Working revision:   1.15 Mon May 23 15:15:14 2005
...
```

- Can help identify files that others are working on
 - 4 states: “need merge”, “up-to-date”, “locally modified”, “needing patch”

Unix/Linux Skills – Marc Corliss

Adding New Files

- To add new files to the repository use “cvs add”:

```
$ cvs add goo.c
```
- Must also do a commit:

```
$ cvs commit -m “...” goo.c
```
- Unlike most cvs commands, “cvs add” is not recursive
 - To add a directory and its contents must first add directory then the files:

```
$ cvs add dir/
$ cvs add dir/file.txt
```

Unix/Linux Skills – Marc Corliss

Removing Files

- To remove files use “cvs remove”
 - But first must remove file (otherwise, you’ll get an error):

```
$ rm goo.c
$ cvs remove goo.c
```
- Must also do a commit:

```
$ cvs commit
```

Unix/Linux Skills – Marc Corliss

Other Features

- Branch and work on 2 copies of the project
 - Ex: one with lots of features, one without
- Tag a particular version of a file for easier access later
 - Only have to remember tag name not version number or date
- View the log for any file or the entire project
 - Question in the final exam
- Find differences between two version of a file
 - Question in the final exam

Unix/Linux Skills – Marc Corliss

More Information

- We have only covered the basics of CVS
- To find more information:
 - <http://ximbiot.com/cvs/manual/cvs-1.11.21/cvs.html>
 - <http://www.csc.calpoly.edu/~dbutler/tutorials/winter96/cvs/>
 - Or just type “cvs tutorial” in Google
 - There are *many* online references

Unix/Linux Skills – Marc Corliss