Cloud.cs CNM 190 User Guide

The class is being provided with a dedicated server for multiple uses during the course. Jeremy Sequoia administers this machine, so please email him or bring up any concerns in class rather than contacting inst@eecs. This document is meant to be a quick introduction to getting access to this storage from your computer at home or in the computer labs.

Backups

The server's /home is approximately 1.5TB in RAID-1. This data is backed up offsite nightly, and a small number of historical archives are kept in case data is accidentally deleted. If data is accidentally deleted, let me know *immediately* to avoid it being propagated to the backups.

Access

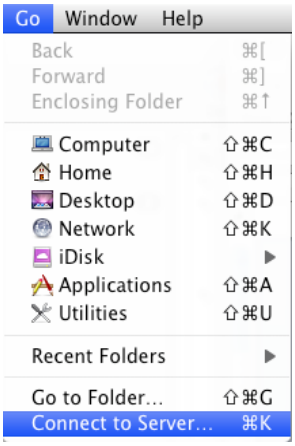
You can access cloud.cs using CIFS, SMB, AFP, or sftp. Each team will have a separate login. The username and password will be given out in class once the final two stories are chosen and teams are formed.

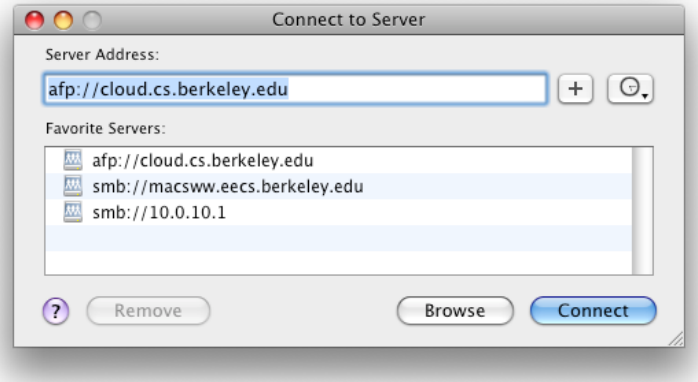
OS X - AFP

The details and images here correspond to Leopard (10.5), but the process is the same on Tiger (10.4) through the most recent OS X release (Mountain Lion (10.8)).

AFP is Apple's "native" network filesystem protocol. To mount the disk, click on the Desktop so "Finder" is the active application. Then choose "Connet to Server" from the "Go" menu.

In the dialog box, enter 'afp://cloud.cs.berkeley.edu'. You can click the '+' icon to add the server to your list of favorites. Click on connect. At the prompt, enter your username and password. The disk will now be mounted at /Volumes/<username> (and appear on your desktop)





If you are on a lab machine (as your cs194 account), you can do this directly from the Terminal (useful if you are ssh'd in for example) by running "mount_cloud.sh" from the command line. Make sure you unmount the server before you leave ("mount_cloud.sh -u" or drag the disk to the trash).

Windows - SFTP

The most secure way to get access to the data from windows is using sftp. putty is a great sftp client that is available for Windows. You can find it by googling for 'putty sftp'. Download the putty-0.60-installer.exe (http://the.earth.li/~sgtatham/putty/latest/x86/putty-0.60-installer.exe) and install it on your system.

Putty does not have a graphical interface for transferring files, so if you prefer that, you should try WinSCP (http://winscp.net).

Windows - SMB

As of this writing, SMB support is being firewalled by EECS, so you can only access the file server via SMB within Soda.

Open up an explorer window and in the address bar, type '\\cloud.cs.berkeley.edu'. You should then be prompted for a login and password. After entering your login and password, you should then see a list of shares, including your folder.

Additionally, you can right click on the folder and select 'map network drive' to map this share to a drive letter.

File Syncing

As the project grows, it will become increasingly difficult to sync files between local users and the file server. Plan ahead. In the past, teams have used unison, rsync, and git. There is no one best solution, so find out who in your team has experience with these products and go from there. If you need help, please don't hesitate to ask!

File System Conventions

It is **very important** to come up with conventions for your file naming and directory hierarchy. Your project will fall apart if everything is in one directory. Create subdirectories for models (and subdirectories of that for each model), shots, sound, scripts, etc. It will be up to the Project Manager (or IT Manager if you choose to split off this role) to lay these guidelines out explicitly in a file in your team's home directory (named NAMING.txt) as part of an upcoming assignment, and we will discuss and refine these in crit.

Web Space

Each team has a directory designated for your use as a team website. The web server has many modern technologies available, but if you need something else, please just ask, and we'll try to make it available for you. The team websites are located at http://cloud.cs.berkeley.edu/~<username> and the files being served for these sites are on cloud in the "public_html" subdirectory of your home directory.

It is up to the team to decide how best to use their site, but it must always contain:

1. Your most recent reel as inline video (ask if you need help).
2. A section dedicated to each team member's individual contributions.

For #2, each team member must maintain a site detailing their individual contributions. This will be useful for demo reel content and will also be used in determining your course grade. Consider this your development journal.

There is no requirement on what technology you decide to use for your site. We have Dreamweaver available on the lab machines if you want to design your sites directly, or you can setup a content management system like Drupal or Wordpress. If nobody on your team has experience setting up or administering systems like this, please talk to us for help.

Advanced Techniques (SSH and Mounting on "The Farm")

OS X is a UNIX system and as such is scriptable using a variety of languages. The machines in the lab have bash, perl, python, and ruby available for your use. An example script is available which will allow you to execute a command on every system (in serial). You can use this command to mount cloud on every machine on the farm.

Because this command, farm_foreach.sh, uses ssh to connect to each machine, it is best that you setup an ssh key for authentication (on your lab account) to avoid needing to enter your password every time. Your key can still be protected by a passphrase that you will use to unlock it and add it to ssh-agent. Your first session would look something like this:

1) Login to the remote system. You can do this from Terminal.app on a Mac or putty on Windows:

```
home $ ssh cs194-aa@ akane.cs.berkeley.edu
```

2) Setup an ssh key for authentication and copy over known_hosts for the farm:

```
~ $ ssh-keygen -C "Jeremy's CNM190 Key"
~ $ cp ~/.ssh/id_rsa.pub ~/.ssh/authorized_keys
~ $ cat ~cnm/resources/farm_hosts >> ~/.ssh/known_hosts
```

The last line copies over public keys for the farm machines to your known_hosts file to save you from typing “yes” many times the first time you run farm_foreach.sh. When you want to run commands on all the farm machines in serial, you can do the following:

1) Cache your ssh credentials with ssh-agent which will allow you to connect again without providing the passphrase each time:

```
~ $ ssh-add
```

2) After that, you can connect to all the farm machines without entering your passphrase. You can then do things easily on all the farm machines:

```
~ $ farm_foreach.sh hostname \; w
~ $ farm_foreach.sh mount_cloud.sh
~ $ farm_foreach.sh mount_cloud.sh -u
```

In order to avoid entering your password for every machine, you should take advantage of the OS X Keychain. mount_cloud.sh first checks the keychain for credentials and only asks for a password if it cannot find one (because it’s not there or access was denied). To add the credentials to the keychain, you should first mount the server within Finder (as on the first page). To let mount_cloud.sh have access to the credentials, run mount_cloud.sh from Terminal.app (with the server not yet mounted) and choose “Always Allow” from the window. This “Always Allow” has some security implications in that any script executed as you could subsequently mount cloud.cs, but that shouldn’t be a big concern.