CMPS 111 Winter 2016: Assignment 0

Assigned: January 4

Due: Monday, January 11 at 0100h

Goals

The goal of this assignment is to get your FreeBSD environment set up and running, and ready for the four real assignments in the class. To do this, you'll need to accomplish the following:

- Install a virtual machine (VM) running FreeBSD
- Make a copy of your git repository on the VM
- Create a directory asgn0 at the root of your repository
- In asgn0, write the cat program and the Makefile necessary to build it.

Files for this assignment, as for every other assignment in the class, will be submitted by committing them to your `git` repository and pushing the changes to the `git` server.

Details

Installing a FreeBSD VM

You're going to run your own copy of FreeBSD, so you'll need to download an install disk image and use it to create a virtual machine in VMware or VirtualBox. Please use this FreeBSD version 10.2 disk image rather one from `FreeBSD.org` to ensure that you're using the exact same version as everyone else in the class.

Creating the VM

Once you've got all of the pieces, you'll need to create the virtual machine itself. Follow these steps:

1. Decompress the `.iso` file you downloaded above.
2. In VMware, select “File → New...”
3. In the resulting dialog box, select the `.iso` file. Then, say you want to customize your VM.
4. In the VM settings window, choose Processor & Memory. *Try to choose 2 processors*, if the selection is available. Set memory size to 4 GB (4096 MB). Apply the changes and then Show All (you should do this for all selections).
5. Still in the VM settings window, choose Hard Disk. You want at least 6 GB of hard disk space.
6. Now check Network Adapter. It should be “Share with Mac/PC”.
7. Close the settings window, and run the VM to start the installation process.

Booting from the ISO file and installing FreeBSD

You're now booting from the ISO file that you downloaded. You'll go through the FreeBSD install script, making selections to configure your operating system. In general, <ENTER> is used to confirm a selection (moving to the next screen), and the arrow keys and space bar are used to navigate to an item and select / unselect it.

1. Hit <ENTER> to boot from VMware on `usbus0`
2. Hit <ENTER> to use the default keyboard config
3. Choose a hostname. This can be anything you want, but should start with a letter and contain only letters, numbers, dashes, and underscores. Hostnames are case-insensitive.
4. Select the distributions you want: only install `lib32` and ports (the ones with next to them are selected). *Do not install system source code*!
5. Next, ensure that you're using Auto (ZFS) for partitioning.
6. For ZFS configuration, swap size should be 512MB, but the other defaults are fine.
7. For the Pool Type / Disks option, select “Stripe – No Redundancy”.
8. When asked which disk to use, there should only be a single choice (the VMware disk). Use it.
9. Yes, you want to destroy the contents of your virtual disk. After you hit <ENTER>, the system will copy data from the (virtual) CD to your (virtual) disk. This may take a few minutes—be patient!
10. Select a root password when asked. Write it down! There's no sysadmin besides you who can help if you forget it.
11. For network configuration, you do want IPv4 and DHCP, but you don't want IPv6.
12. Your search domain should be empty (erase localdomain).
13. For time zone, select NO because you don't know if you're using UTC. Then, select the proper time zone (America ➝ United States ➝ Pacific Time)
14. For system configuration, enable `sshd`, `dumpdev`, and `ntpd`.
15. Yes, you'd like to add a user. For username, use your CruzID because it'll simplify things. Login group should be wheel (and no other groups). For shell, pick any of `sh`, `csh`, or `tcsh`. Pick defaults for everything else, and pick a good password. Again, write it down!
16. Exit to shell for final configuration. The one thing you want to do is edit:

    ```
    vi /etc/syslog.conf
    ```

    and remove the # from the line that says:

    ```
    # console.info /var/log/console.log
    ```

17. Save the file and exit the editor. Then, execute this command:

    ```
    touch /var/log/console.log
    ```

18. Exit the shell (type `exit`) and let the VM (not your computer!) reboot.

Setting up FreeBSD

At this point, your system should boot up just fine. First, find your VM's IP address by typing:

```
ifconfig em0
```

and looking for an IP address that looks like `172.12.116.nnn`. This is always the IP address that you can use to connect to your VM via `ssh`. At this point, you may prefer to use a terminal program to connect to your VM rather than typing in the VMware window. You can open as many connections to your VM as you like (well, OK, there's a limit, but it's large).

Next, you need some extra software on your VM. This is pretty straightforward using the FreeBSD package installer. Here are the packages you should install. You need to do the installation running as root, which you can do either by logging in on the console as root or by becoming root using the `su root` command.

Required packages

```
pkg install git
```

This installs the `git` system, which you'll obviously need.

```
pkg install compat6x-amd64
```

This package is needed by the VMware tools that you'll install next. Next, you need to install the VMware tools for FreeBSD. Under the VMware menu item, select Install VMware tools. Then, as root, run the following commands:

```
cd /root
mount -t cd9660 /dev/cd0 /mnt
tar xzf /mnt/vmware-freebsd-tools.tar.gz
umount /mnt
cd vmware-tools-distrib
./vmware-install.pl
```

The default answer to all of the questions should be fine: you can hit <ENTER> whenever it asks a question.

Optional packages

```
pkg install sudo
```

This makes it easier to run things as root. Use `visudo` to edit the `config` file, and uncomment the line following

```
## Same thing without a password
```

by removing the # from the front of the line.

```
pkg install emacs-nox11
```

Which is good if you like `emacs`. `vi` is already installed for you.

```
pkg install bash
```

Which is good if you like `bash`. `tcsh` is already installed for you, and is the default shell. You're welcome to install other packages if you like.

Setting up your repository and `git`

Next, set up your repository. You'll write all of your code here, and use `git` to track changes and submit them for grading. Please read the git shortcuts page for information on how to set up your `git` account. Once that's done, clone your repository from the CS111 `git` server to your FreeBSD VM, preferably logged in as the user you created above (not root).

At this point, you're ready to start writing code!

The `cat` program

The only code you have to write for this assignment is to implement the basic `cat` program, without support for any flags. That means your code needs to copy data from each of the files specified on the command line to standard output. For example,

```
cat file1 file2 file3
```

will copy all of the data from `file1`, `file2`, and `file3` to standard output, in that order. Note that the data might be binary; your code should work in that case. Process files one at a time; if `cat` can't open a file to read it, the program should stop and print an error message to standard error. This may mean that some of the files were copied, and others not—this is fine, and is what the real `cat` program does.

Your program must be written in C (not `C++`, not Java), and must be compiled by a `Makefile` that creates an executable called `cat`. Your code and `Makefile` should be in a directory `asgn0` at the top level of your repository.

Hints

This is a straightforward assignment, designed to get you familiar with the tools you'll need for the remaining assignments. Most of your time will be spent setting up the VM, which isn't difficult, but can take some time (a few hours).

The `cat` program is very short—fewer than 50 lines of code, most of which are error handling. You'll want to use the `open()`, `close()`, `read()`, and `write()` system calls (see the `man` pages for them) as well as `perror()`.

Read the `git` shortcuts page to become familiar with `git`. You'll be using it a lot.

Get in the habit of committing to `git` frequently. Your commits will be stored locally until you push them to the server, which you can do any time you want. Note that, if you commit your changes but don't push them to the server, your grader can't see them. While we're going to use your commit times to determine if your assignment is on time, you need to push your changes to the server within an hour of the actual due date.

Grading

Unlike later assignments, this assignment is pass / no pass. You must pass it in order to pass the class. It won't otherwise impact your final grade, though you'll have a lot of difficulty with later assignments if you have a lot of difficulty with this one.