Rich's lesson module checklist

Last updated 11/26/2018

- ❑ Zoom recording named and published for previous lesson

- ❑ Slides, Project, Lab X1 and Lab X2 posted
- ❑ Print out agenda slide and annotate page numbers

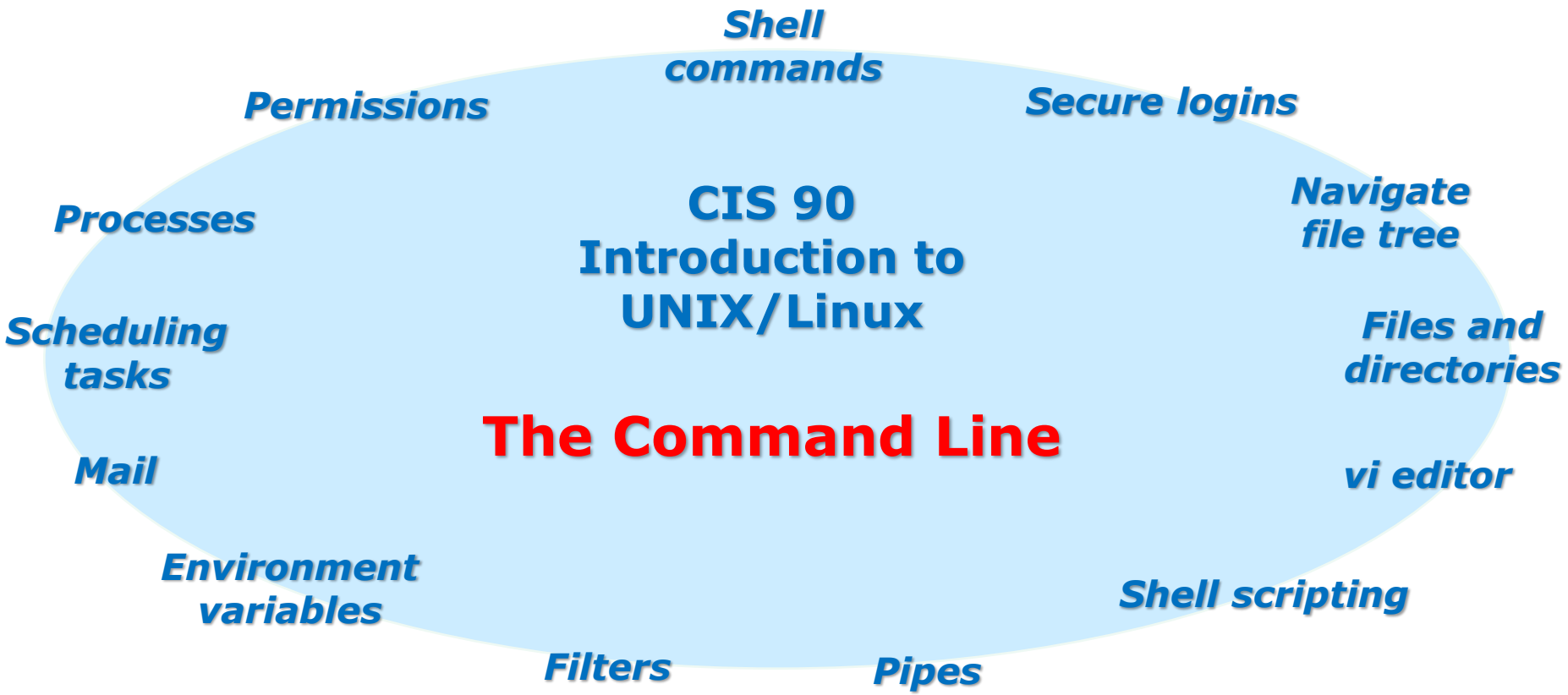
- ❑ Flash cards
- ❑ 1st minute quiz
- ❑ Web Calendar updated

- ❑ Lock turnin directory at midnight (scripts/schedule-submit-locks)
- ❑ CUPS & printer demo equipment (optional)
- ❑ Lab X1 and X2 posted
- ❑ Code samples in depot/scripts directory

- ❑ Backup slides, CCC info, handouts on flash drive
- ❑ Spare 9v battery for mic
- ❑ Key card for classroom door

- ❑ https://zoom.us

- ❑ Putty + Slides + Chrome
- ❑ Enable/Disable attendee sharing
  ^ > Advanced Sharing Options > Only Host
- ❑ Enable/Disable attended annotations
  Share > More > Disable Attendee Sharing



Student Learner Outcomes

1. Navigate and manage the UNIX/Linux file system by viewing, copying, moving, renaming, creating, and removing files and directories.
2. Use the UNIX features of file redirection and pipelines to control the flow of data to and from various commands.
3. With the aid of online manual pages, execute UNIX system commands from either a keyboard or a shell script using correct command syntax.

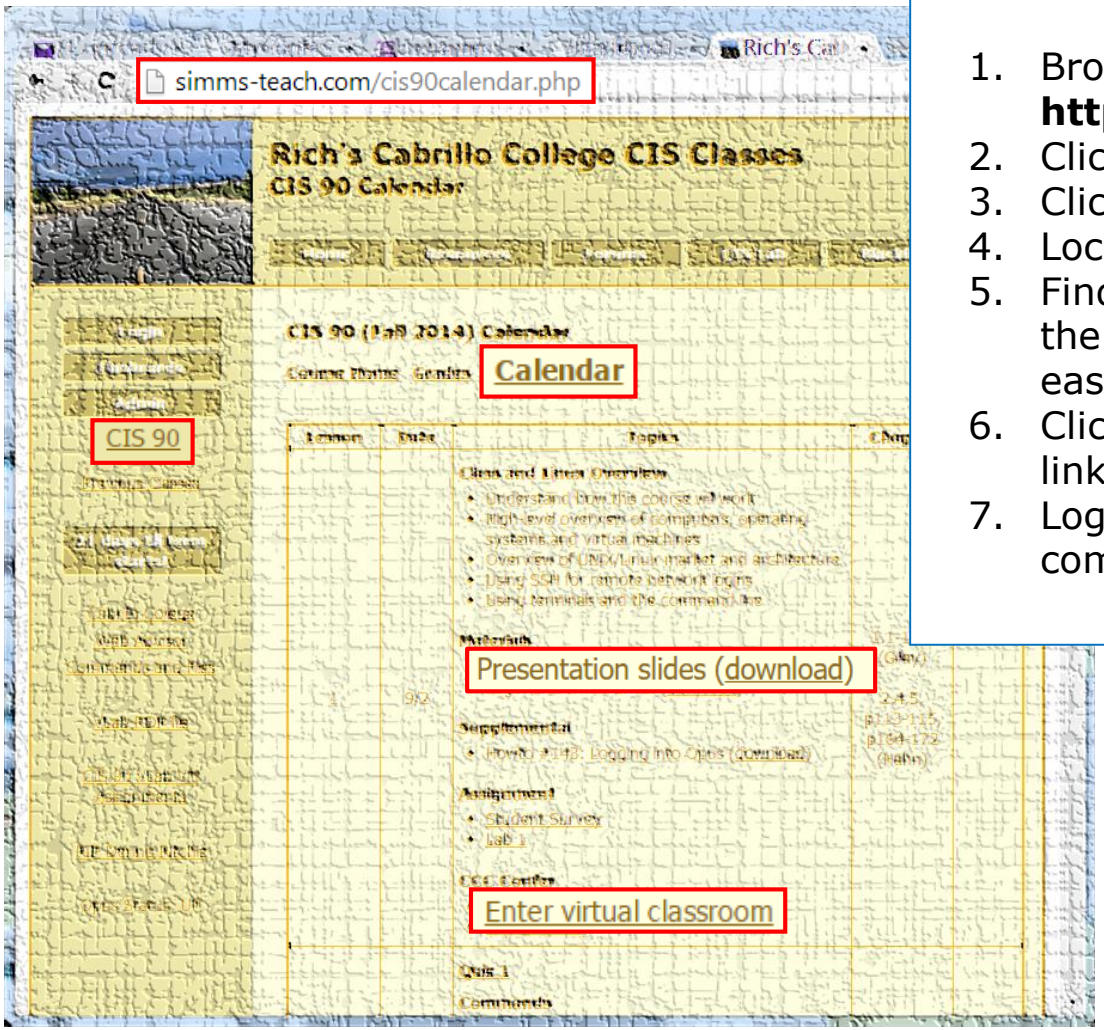
Introductions and Credits

Jim Griffin
- Created this Linux course
- Created Opus and the CIS VLab
- Jim's site: https://web.archive.org/web/20140209023942/http://cabrillo.edu/~jgriffin/

Rich Simms
- HP Alumnus
- Started teaching this course in 2008 when Jim went on sabbatical
- Rich's site: http://simms-teach.com

And thanks to:
- John Govsky for many teaching best practices: e.g. the First Minute quizzes, the online forum, and the point grading system. John's site: http://teacherjohn.com/
- Jaclyn Kostner for many webinar best practices: e.g. mug shot page.

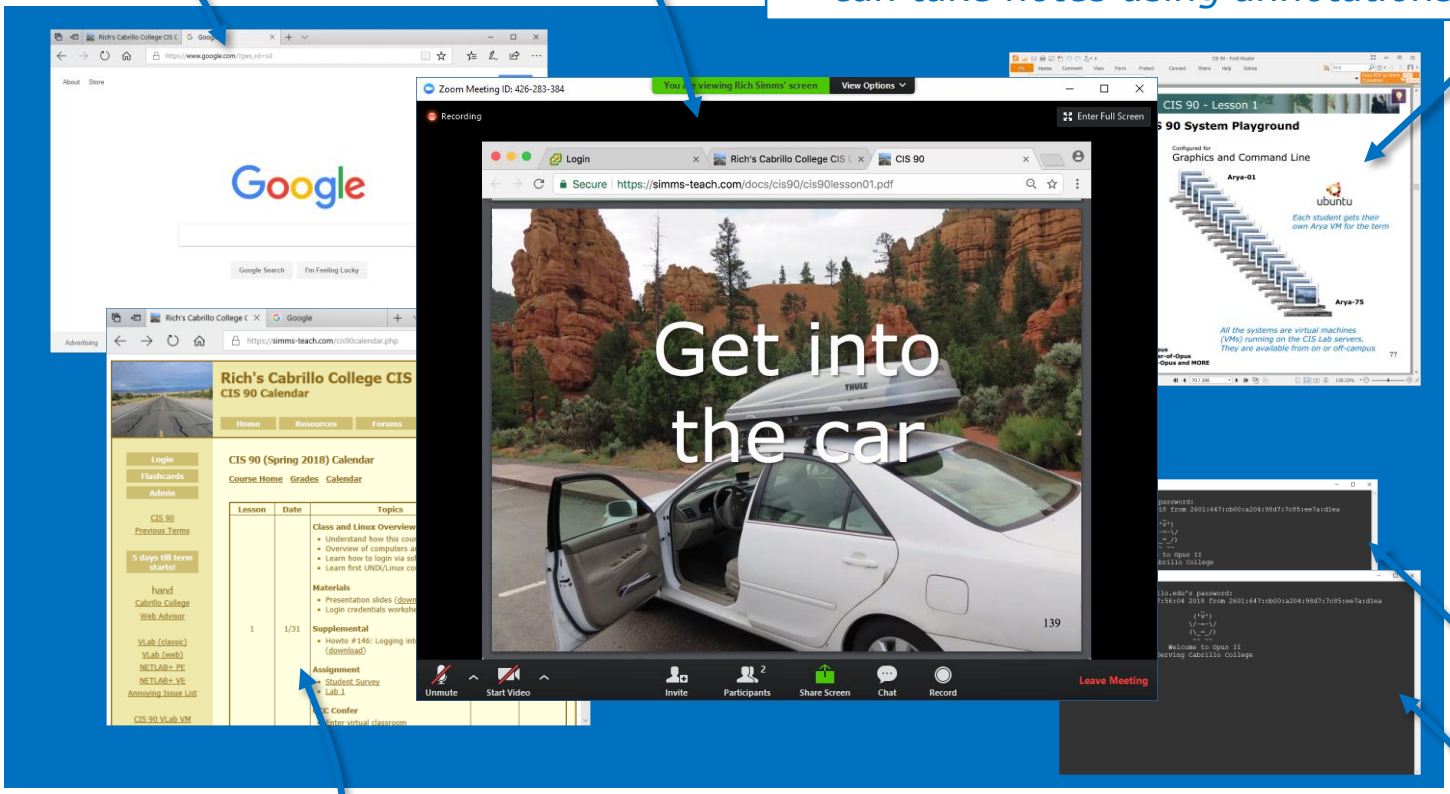
Student checklist - Before class starts



1. Browse to: **http://simms-teach.com**
2. Click the **CIS 90** link.
3. Click the **Calendar** link.
4. Locate today's lesson.
5. Find the **Presentation slides** for the lesson and **download** for easier viewing.
6. Click the **Enter virtual classroom** link to join ConferZoom.
7. Log into Opus-II with Putty or ssh command.

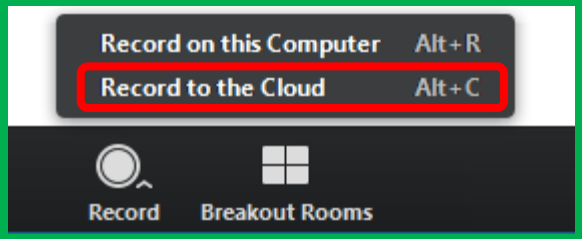
Student checklist - Before class starts

- ❑ *Google*
- ❑ *ConferZoom*
- ❑ *Downloaded PDF of Lesson Slides. I like Foxit Reader so I can take notes using annotations.*



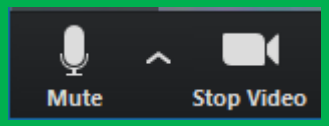
- ❑ *CIS 90 website Calendar page*
- ❑ *One or more login sessions to Opus-II*

Start



Start Recording

Audio Check



Start Recording

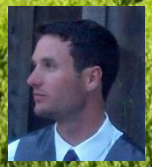
Audio & video Check

CIS 90 - Lesson 13

Instructor: **Rich Simms**
Dial-in: **408-638-0968 (toll)**
Meeting ID: **426 283 384**


Mikey


Jona


Joseph


Tara Marie


Fredi


Carina

Isaac


Matthew

Erik


Tony

Branden

Dominic

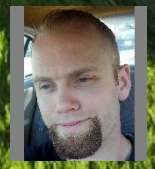
Ryan L.


Alejandra

Blair


Zari

Victor


Danny

Gabriel

Janelly


Austin


Aaron


Ryan M.

Email me (risimms@cabrillo.edu) a relatively current photo of your face for 3 points extra credit

First Minute Quiz #10

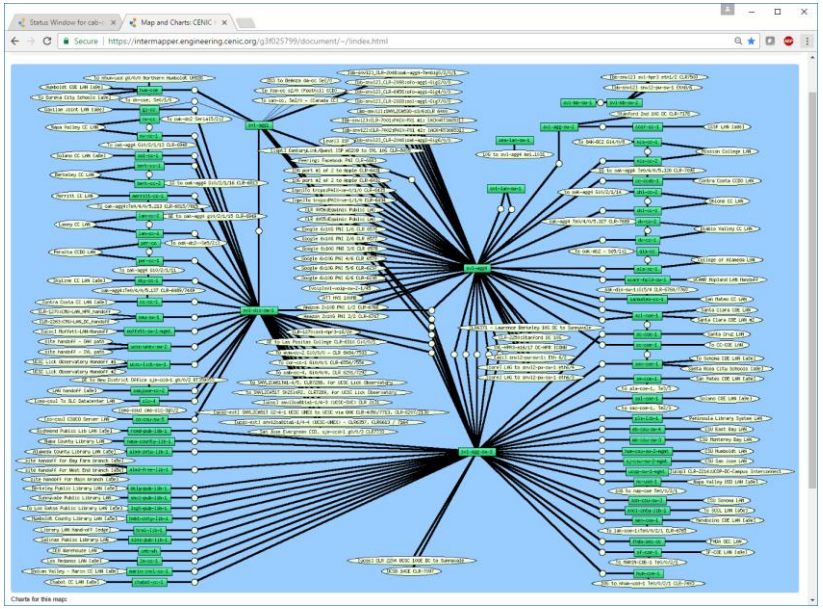
Please answer these questions **in the order** shown:

Use ConferZoom White Board

email answers to: risimms@cabrillo.edu

(answers must be emailed within the first few minutes of class for credit)

Network Check

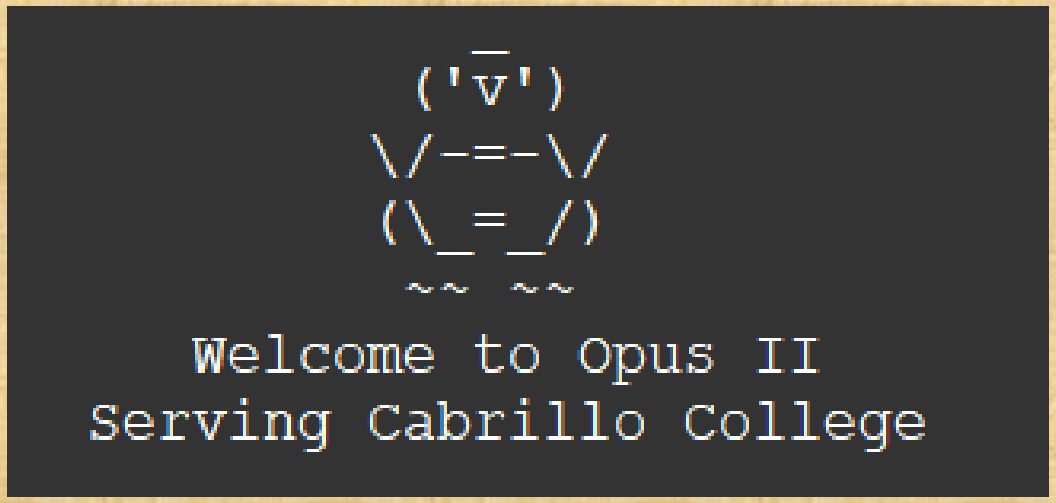


https://intermapper.engineering.cenic.org/g3f025799/document/~/!index.html

Shell Scripting and Printing

Objectives	Agenda
• Understand how to write a script and how they run. • Learn how to print and manage print jobs waiting to print.	• Quiz • Questions • Breaking things in Lab 10 • Extra Credit Answer • Lesson 12 review • Grok that? • Housekeeping • Printers • Printing in Linux • Managing print jobs • $(cmd), date part II, exit status, color, sleep • Shell scripting 101 • Final project myscript • Final project grading rubric • Final project permissions • Umask again! • Final project getting started • Final project forum tips • Scripting tips - echo • Don't name your scripts "script" • Review how scripts are run • Assignment • Wrap up

Class Activity



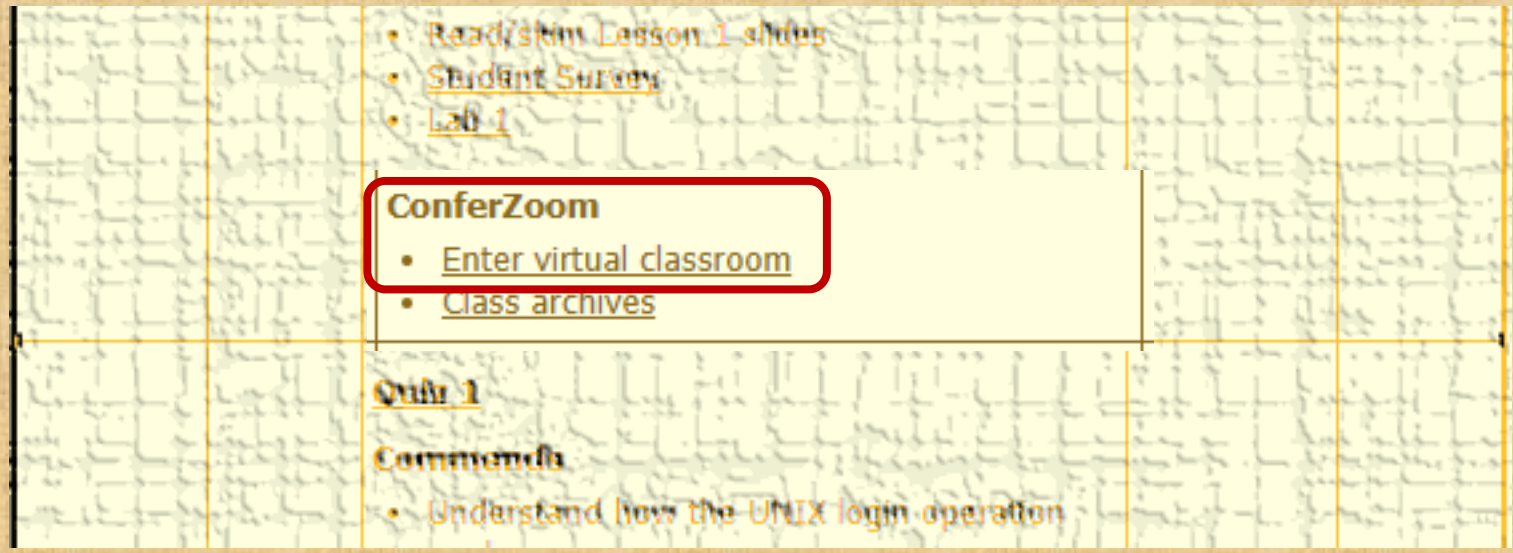
If you haven't already, log into Opus-II

Class Activity

https://simms-teach.com/cis90calendar.php

If you haven't already, download the lesson slides

Class Activity



https://simms-teach.com/cis90calendar.php

If you haven't already, join ConferZoom classroom

Questions

Questions?

Lesson material?

Labs? Tests?

How this course works?

- Graded work & tests in home directories
- Answers in /home/cis90/answers

Who questions much, shall learn much, and retain much.
- Francis Bacon

If you don't ask, you don't get.
- Mahatma Gandhi

Chinese Proverb	他問一個問題，五分鐘是個傻子，他不問一個問題仍然是一個傻瓜永遠。 *He who asks a question is a fool for five minutes; he who does not ask a question remains a fool forever.*

```
alias bill="cd /home/cis90/${LOGNAME%90}/poems/Shakespeare"
```

What the heck was this all about?

```
/home/cis90/milhom $ echo $LOGNAME
milhom90
/home/cis90/milhom $ echo ${#LOGNAME}
8
/home/cis90/milhom $ echo ${LOGNAME%90}
milhom
/home/cis90/milhom $ echo ${LOGNAME:3:3}
hom
/home/cis90/milhom $ echo ${LOGNAME#mil}
hom90
```

Length of the string

Extracts "90" from end of string

Substring extraction from position 3 length 3

Extracts "mil" from front of string

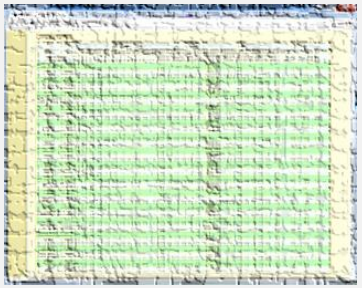
For MANY MORE ways to manipulate strings Google "bash string manipulation" or browse to:

http://tldp.org/LDP/abs/html/string-manipulation.html

Review your progress in the course

Check the website Grades page

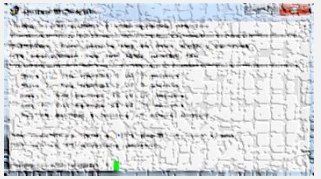
http://simms-teach.com/cis90grades.php



Or check on Opus-II

checkgrades *codename*

(where codename is your LOR codename)



Written by Jesse Warren a past CIS 90 Alumnus

- **Send me your survey to get your LOR codename.**
- **Graded labs and tests are in your home directories.**

Percentage	Total Points	Letter Grade	Pass/No Pass
90% or higher	504 or higher	A	Pass
80% to 89.9%	448 to 503	B	Pass
70% to 79.9%	392 to 447	C	Pass
60% to 69.9%	336 to 391	D	No pass
0% to 59.9%	0 to 335	F	No pass

At the end of the term I'll add up all your points and assign you a grade using this table

Points that could have been earned:	
9 quizzes:	27 points
9 labs:	270 points
2 tests:	60 points
3 forum quarters:	60 points
Total:	**417 points**

Extra Credit

On the forum

Be sure to monitor the forum as I may post extra credit opportunities without any other notice!

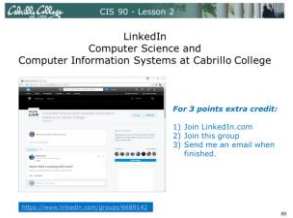
On some labs

Extra credit (2 points)

For a small taste of what you would learn in CIS 191 let's add a new user to your Arya VM. Once added we will see how the new account is represented in */etc/passwd* and */etc/shadow*.

1. Log into your Arya VM as the cis90 user. Make sure it's your VM and not someone else's.
2. Install the latest updates:
   ```
   sudo apt-get update
   sudo apt-get upgrade
   ```
3. Add a new user account for yourself. You may make whatever username you wish. The example below shows how Benji would make the same username he uses on Opus:
   ```
   sudo useradd -G sudo -c "Benji Simms" -m -s /bin/bash simben90
   ```

In lesson slides (search for extra credit)

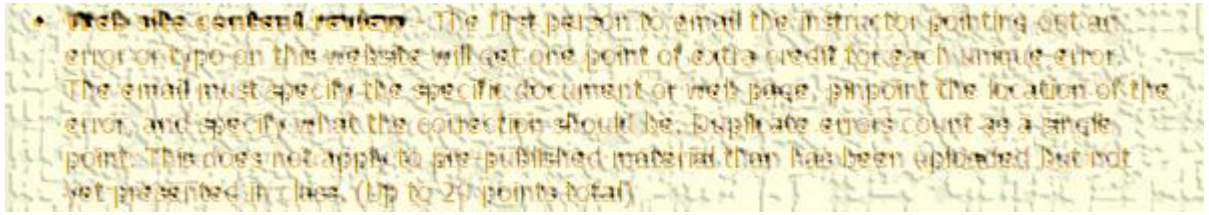


On the website

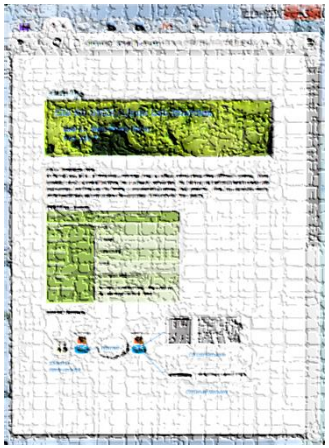
http://simms-teach.com/cis90grades.php

For some flexibility, personal preferences or family emergencies there is an additional 90 points available of **extra credit** activities.

http://simms-teach.com/cis90extracredit.php



Lab Assignments -- Pearls of Wisdom



- Don't wait till the last minute to start.
- Plan for things to go wrong and give yourself time to ask questions and get answers.
- The *slower* you go the *sooner* you will be finished.
- A few minutes reading the forum can save you hour(s).
- Line up materials, references, equipment and software ahead of time.
- It's best if you fully understand each step as you do it. Use Google or refer back to lesson slides to understand the commands you are using.
- Keep a growing cheat sheet of commands and examples.
- Study groups are very productive and beneficial.
- Use the forum to collaborate, ask questions, get clarifications and share tips you learned while doing a lab.
- **Late work is not accepted** so submit what you have for partial credit.

Getting Help When Stuck on an Assignment

- Google the topic/error message.
- Search the Lesson Slides (they are PDFs) for a relevant example on how to do something.
- Check the forum. Someone else may have run into the same issue and found a way past it. If not start a new topic, explain what you are trying to do and what you have tried so far.
- Talk to a STEM center tutor/assistant.
- Come see me during my office or lab hours:
  https://www.cabrillo.edu/salsa/listing.php?staffId=1426
  I'm in the CTC (room 1403) every Tuesday from 3:30-5:00 pm.
- Make use of the Open Questions time at the start of every class.
- Make a cheat sheet of commands and examples so you never again get stuck on the same thing!

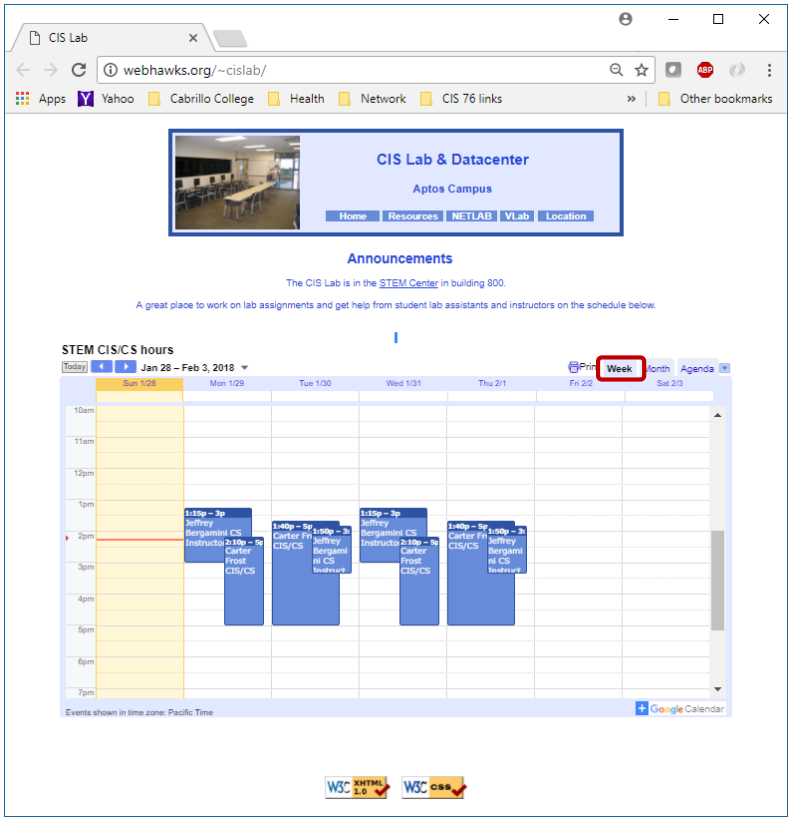
CIS Labs always involve some troubleshooting!

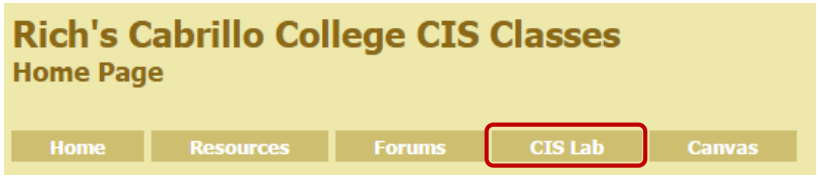
CTC - Building 1400
On lower campus

I will be in the CTC (room 1403) every Tuesday afternoon from 3:30-5:00

Help Available in the CIS Lab

Instructors, lab assistants and equipment are available for CIS students to work on assignments.





To see schedule, click the CIS Lab link on the website and use the "Week" calendar view

The slippery slope

1) If you didn't submit the last lab ...

2) If you were in class and didn't submit the last quiz ...

3) If you didn't send me the student survey assigned in Lesson 1 ...

4) If you haven't made a forum post in the last quarter of the course ...

5) If you had trouble doing the last test ...

Please contact me by email, see me during my office hours or when I'm in the CTC

Email: risimms@cabrillo.edu

Breaking things in Lab 10

Did you break your path in Lab 10?

```
/home/cis90/simben $ type echo tty scavenge allscripts tryme dogbone
echo is a shell builtin
tty is /usr/bin/tty
scavenge is /usr/local/bin/scavenge
allscripts is /home/cis90/simben/../bin/allscripts
tryme is /home/cis90/simben/bin/tryme
dogbone is ./dogbone
```

Are you getting unexpected "Command not found" errors today? If the highlighted directories above are not on your path then you will get them!

Note the echo command is built into the shell. We can always run it even if our shell path is broken.

Review of the path (PATH) variable

- Lab 10 often results in clobbered paths and students may think some or all of the commands have disappeared!
- The path is a list of directories each containing commands, programs and scripts.
- The path is used by the shell, during the search step, to locate commands to run.
- The PATH variable defines the directories (separated by ":"s) and the search order.
- NOTE: If your path gets clobbered it is still possible to run commands. However to do that you must specify the full absolute or relative pathname. For example, without a path you can still run the **tty** and **tryme** commands as follows:

```
/home/cis90/simben $ /usr/bin/tty      Using an absolute pathname
/dev/pts/0
/home/cis90/simben $ bin/tryme         Using a relative pathname
My name is "tryme"
I am pleased to make your acquaintance, Benji Simms
/tmp
```

The path (PATH) variable … a Review

Examine your path:

After you finish Lab 10 this one will be simplified

```
/home/cis90/simben $ echo $PATH
/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:
/home/cis90/simben/../bin:/home/cis90/simben/bin:.
```

1. **Determine the 2nd directory on the path above.**

2. **What is the name of the first command, in alphabetic order, found in this directory?**

Put your answer in the chat window

The first command in /usr/bin is [

The path (PATH) variable … a Review

Examine your path:

```
/home/cis90/simben $ echo $PATH
/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:
/home/cis90/simben/../bin:/home/cis90/simben/bin:.
```

On Opus-II why is /bin and /sbin not needed on your path?

Put your answer in the chat window

Because on RHEL 7 the /bin and /usr/bin directories have been merged with /bin symbolically linked to /usr/bin. Same for /sbin and /usr/sbin.

RHEL 7 merged /bin and /usr/bin, /sbin and /usr/sbin

```
[simben90@opus-ii ~]$ ls -ld /bin /usr/bin /sbin /usr/sbin
lrwxrwxrwx. 1 root root     7 Aug  4  2017 /bin -> usr/bin
lrwxrwxrwx. 1 root root     8 Aug  4  2017 /sbin -> usr/sbin
dr-xr-xr-x. 2 root root 32768 Mar 14 18:26 /usr/bin
dr-xr-xr-x. 2 root root 16384 Jan  7 14:48 /usr/sbin
[simben90@opus-ii ~]$
```

Note that /bin is symbolically linked to /usr/bin. Likewise with /sbin and /usr/sbin

Life without a path

Clobber your path on purpose

```
/home/cis90/simben $ oldpath=$PATH
/home/cis90/simben $ unset PATH
```

Backing up then breaking the current path

```
/home/cis90/simben $ tty
-bash: tty: No such file or directory
```

The tty command can no longer be run by typing just it's name.

```
/home/cis90/simben $ /usr/bin/tty
/dev/pts/0
```

On Opus-II the tty command is in the /usr/bin directory.

If we know that, a temporary workaround is to specify the command with an absolute pathname.

Life without a path

Some commands still work without a path … why?

```
/home/cis90/simben $ echo "I want my path back"
I want my path back

/home/cis90/simben $ type echo
echo is a shell builtin

/home/cis90/simben $ type type
type is a shell builtin
```

The shell has some commands built into it. The shell does not have to search the path to find these commands so they are always available.

Class Activity

Backup and clobber your path variable:

```
/home/cis90/simben $ oldpath=$PATH

/home/cis90/simben $ unset PATH
/home/cis90/simben $ echo $PATH

/home/cis90/simben $ tty
/home/cis90/simben $ /usr/bin/tty

/home/cis90/simben $ echo "I want my path back"
/home/cis90/simben $ type echo
/home/cis90/simben $ type type
```

Why does the echo command work with no path?

Put your answer in the chat window

The echo is built into the shell

Life without a path

✘
```
/home/cis90/simben $ ls letter
-bash: ls: No such file or directory
```

✓
```
/home/cis90/simben $ /usr/bin/ls letter
letter
```

*On Opus-II the **ls** command is in the /usr/bin directory.*

If we know that, a temporary workaround is to specify the full absolute pathname of the command.

Making a path from scratch

Fixing the path, one directory at a time …

✘
```
/home/cis90/simben $ ls letter
-bash: ls: command not found
```

✔
```
/home/cis90/simben $ PATH=/usr/bin
/home/cis90/simben $ ls letter
letter
```

*The **ls** command is in /usr/bin so lets put that directory on the path.*

```
/home/cis90/simben $ echo $PATH
/usr/bin
```

You try it

Enter these commands:

```
ls letter
PATH=/usr/bin
echo $PATH
ls letter
```

What is your shell path now?

Put your answer in the chat window

Making a path from scratch

✘
```
/home/cis90/simben $ allscripts
-bash: allscripts: command not found
```

✔
```
/home/cis90/simben $ PATH=$PATH:/home/cis90/bin
/home/cis90/simben $ allscripts
```

*The **allscripts** shell script is in /home/cis90/bin so let's append that directory to the path as well.*

```
**************************************************************
*                Fall 2012 CIS 90 Online Projects            *
**************************************************************
1) Andrew
2) Ben
3) Benji
4) Bryn
5) Carlile
6) Carlos
  <snipped>
21) Ray
22) Rita
23) Sean C.
24) Sean F.
25) Shahram

99) Exit

Enter Your Choice:
```

You try it

Enter these commands:

```
allscripts
PATH=$PATH:/home/cis90/bin
echo $PATH
allscripts
```

What is your shell path now?
Put your answer in the chat window

Making a path from scratch

```
/home/cis90/simben $ scavenge
-bash:  scavenge: command not found
```

```
/home/cis90/simben $ PATH=$PATH:/usr/local/bin
/home/cis90/simben $ scavenge
#####################################
#   S C A V E N G E R   H U N T   #
#####################################

Welcome Benji,
<snipped>
```

*The **scavenge** shell script is in /usr/local/bin so let's add that directory to the path as well*

You try it

Enter these commands:

```
scavenge
PATH=$PATH:/usr/local/bin
echo $PATH
scavenge
```

What is your shell path now?
Put your answer in the chat window

Making a path from scratch

```
/home/cis90/simben $ tryme
-bash: tryme: command not found
```

```
/home/cis90/simben $ PATH=$PATH:/home/cis90/simben/bin
/home/cis90/simben $ tryme
My name is "tryme"
I am pleased to make your acquaintance, Benji Simms
/tmp
/home/cis90/simben $
```

*The **tryme** shell script is in your own bin directory so lets add that to the path as well*

You try it

Enter these commands:

```
tryme
PATH=$PATH:/home/cis90/simben/bin
echo $PATH
tryme
```

Change this to your own home directory

or

```
tryme
PATH=$PATH:$HOME/bin
echo $PATH
tryme
```

What is your shell path now?
Put your answer in the chat window

Making a path from scratch

```
/home/cis90/simben $ cp ../depot/scripts/dogbone .
/home/cis90/simben $ chmod +x dogbone
/home/cis90/simben $ dogbone
-bash: dogbone: command not found
```

```
/home/cis90/simben $ ./dogbone
What is your name? Benji
What is your favorite bone? Chicken
Hi Benji, your favorite bone is Chicken
```

A temporary workaround is to put a ./ in front of the command

How can I run a script in the current directory without having to put a ./ in front of it?

Making a path from scratch

```
/home/cis90/simben $ dogbone
-bash: dogbone: command not found
```

```
/home/cis90/simben $ PATH=$PATH:.
/home/cis90/simben $ dogbone
What is your name? Benji
What is your favorite bone? Chicken
Hi Benji, your favorite bone is Chicken
```

Easy, just add the "." directory to the path

You try it

```
cd
cp ../depot/scripts/dogbone .
```

Did you do this the hard way or use tab completes?

```
chmod +x dogbone

dogbone
./dogbone

PATH=$PATH:.
dogbone
```

What is your shell path now?

Put your answer in the chat window

Making a path from scratch

Rebuilding the path by appending directories one at a time

```
/home/cis90/simben $ unset PATH
/home/cis90/simben $ echo $PATH

/home/cis90/simben $ PATH=/usr/bin
/home/cis90/simben $ echo $PATH
/usr/bin
```

Start with /usr/bin which has all essential and auxiliary UNIX/Linux commands

```
/home/cis90/simben $ PATH=$PATH:/home/cis90/bin
/home/cis90/simben $ echo $PATH
/usr/bin:/home/cis90/bin
```

Append the CIS 90 class bin directory

```
/home/cis90/simben $ PATH=$PATH:/usr/local/bin
/home/cis90/simben $ echo $PATH
/usr/bin:/home/cis90/bin:/usr/local/bin
```

Append the /usr/local/bin directory

```
/home/cis90/simben $ PATH=$PATH:/home/cis90/simben/bin
/home/cis90/simben $ echo $PATH
/usr/bin:/home/cis90/bin:/usr/local/bin:/home/cis90/simben/bin
```

Append your own student bin directory

```
/home/cis90/simben $ PATH=$PATH:.
/home/cis90/simben $ echo $PATH
/usr/bin:/home/cis90/bin:/usr/local/bin:/home/cis90/simben/bin:.
```

Append the current directory

CIS 90 class bin directory | */usr/local/bin directory* | *Student bin directory* | *Current directory*

.bash_profile

Making the path permanent using .bash_profile

```
/home/cis90/simben $ cat .bash_profile
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
        . ~/.bashrc
fi

# User specific environment and startup programs

PATH=$PATH:/home/cis90/bin:$HOME/bin:.
BASH_ENV=$HOME/.bashrc
USERNAME=""
PS1='$PWD $ '
export USERNAME BASH_ENV PATH
umask 002
set -o ignoreeof
stty susp
eval `tset -s -m vt100:vt100 -m :\?${TERM:-ansi} -r -Q `

/home/cis90/simben $
```

This customizes the normal path by appending the class bin directory, the student's bin directory and the "current" directory

Extra Credit Special Answer

Extra Credit Special (from Lesson 12)

1) Why did the prompt change?

```
/home/cis90/simben $ bash
[simben@opus ~]$ exit
exit
/home/cis90/simben $
```

2) What command could be issued prior to the bash command above that would prevent the prompt from changing?

For 2 points extra credit, email risimms@cabrillo.edu answers to **both** questions before the Lesson 13 class starts

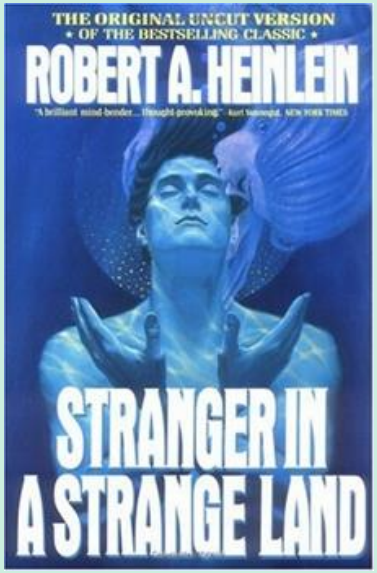
Lesson 12 Review

The rules of the road for variables

Process Rule #1: When a shell forks a child, only copies of exported variables are made available to the child.

Process Rule #2: A child can modify the variables it receives but those modifications will not change the parent's variables.

grok that?


THE ORIGINAL UNCUT VERSION
OF THE BESTSELLING CLASSIC
ROBERT A. HEINLEIN
STRANGER IN A STRANGE LAND

The flowers script
/home/cis90/bin/flowers

```
#!/bin/bash
#
# Useful alias:
#   alias go='echo roses are \"$roses\" and violets are \"$violets\"'
#
echo
echo "==> Entering child process <=="
ps -f
echo "==> showing variables in child <=="
echo "  " roses are '"'$roses'"'
echo "  " violets are '"'$violets'"'
echo "==> setting variables in child <=="
roses=black
violets=orange
echo "  " roses are '"'$roses'"'
echo "  " violets are '"'$violets'"'
echo "==> Leaving child process <=="
echo
```

Show the parent, child and the ps processes

Show the values of the roses and violets variables

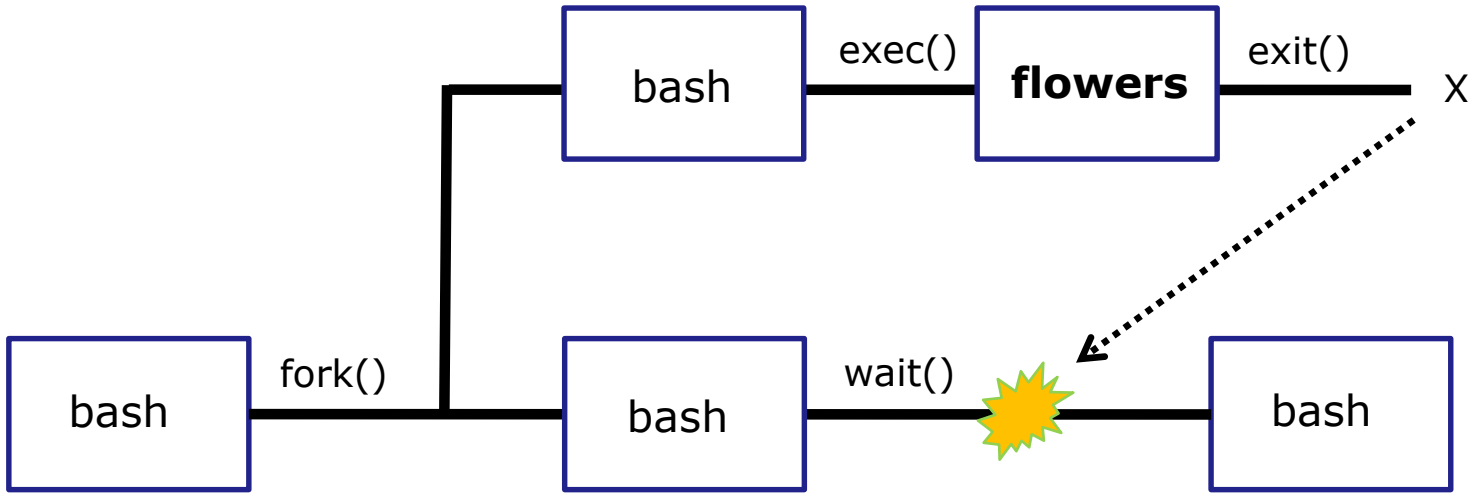
Set the values of the roses and violets variables to new values

The flowers script
/home/cis90/bin/flowers

```
/home/cis90/simben $ flowers

==> Entering child process <==
UID        PID  PPID  C STIME TTY          TIME CMD
simben90 17518 17512  0 08:32 pts/0    00:00:00 -bash
simben90 17568 17518  0 08:33 pts/0    00:00:00 /bin/bash /home/cis90/bin/flowers
simben90 17575 17568  8 08:33 pts/0    00:00:00 ps -f
==> showing variables in child <==
   roses are ""
   violets are ""
==> setting variables in child <==
   roses are "black"
   violets are "orange"
==> Leaving child process <==

/home/cis90/simben $
```

```
#!/bin/bash
#
# Useful alias:
#   alias go='echo roses are \"$roses\" and violets are \"$violets\"'
#
echo
echo "==> Entering child process <=="
ps -f
echo "==> showing variables in child <=="
echo "   " roses are '"'$roses'"'
echo "   " violets are '"'$violets'"'
echo "==> setting variables in child <=="
roses=black
violets=orange
echo "   " roses are '"'$roses'"'
echo "   " violets are '"'$violets'"'
echo "==> Leaving child process <=="
echo
```

The flowers script
/home/cis90/bin/flowers



*Use the **flowers** script to test your understanding of how variables are handled with child processes*

Create an alias to show variable values

Note, the double quotes are escaped. We don't want bash to treat them as special metacharacters. We just want the double quotes preserved so they can be seen in the output of the echo command.

```
/home/cis90/simben $ alias go='echo roses are \"$roses\" and violets
are \"$violets\"'

/home/cis90/simben $ alias go
alias go='echo roses are \"$roses\" and violets are \"$violets\"'

/home/cis90/simben $ go
roses are "" and violets are ""
```

Since there are no shell variables named roses or violets the echo command prints nothing for them.

Activity

Setup this alias so you can use it in activities that follow:

```
alias go='echo roses are \"$roses\" and violets are \"$violets\"'
```

What happens now when you type the go command?

Type your answer in the chat window

Use the alias to show the values of the two variables

```
/home/cis90/simben $ go
roses are "" and violets are ""
```

```
/home/cis90/simben $ roses=red
/home/cis90/simben $ go
roses are "red" and violets are ""
```

Now the roses variable has been created and initialized

```
/home/cis90/simben $ violets=blue
/home/cis90/simben $ go
roses are "red" and violets are "blue"
```

Now the violets variable has been created and initialized

Use the alias to show the values of the two variables

```
/home/cis90/simben $ unset roses
/home/cis90/simben $ go
roses are "" and violets are "blue"
```

Now the roses variable no longer exists

```
/home/cis90/simben $ unset violets
/home/cis90/simben $ go
roses are "" and violets are ""
```

Now the violets variable no longer exists

Activity

```
/home/cis90/simben $ roses=red; violets=blue
/home/cis90/simben $ go
roses are "red" and violets are "blue"
/home/cis90/simben $ env | grep roses
/home/cis90/simben $ env | grep violets
/home/cis90/simben $ flowers
```

When the flowers script runs will it see the values of the roses and violets variables?

Write your answer in the chat window

***NO**, the roses and violets variables were not exported*

```
/home/cis90/simben $ flowers

==> Entering child process <==
UID        PID  PPID  C STIME TTY          TIME CMD
simben90 25106 25059  0 17:16 pts/8    00:00:00 -bash
simben90 27052 25106  0 17:19 pts/8    00:00:00 /bin/bash /home/cis90/bin/flowers
simben90 27059 27052  0 17:19 pts/8    00:00:00 ps -f
==> showing variables in child <==
   roses are ""
   violets are ""
==> setting variables in child <==
   roses are "black"
   violets are "orange"
==> Leaving child process <==

/home/cis90/simben $
```

The child cannot view the values of the parent's non-exported variables (Rule #1)

Activity

```
/home/cis90/simben $ roses=red; violets=blue
/home/cis90/simben $ export roses
/home/cis90/simben $ env | grep roses
roses=red
/home/cis90/simben $ env | grep violets
/home/cis90/simben $ go
roses are "red" and violets are "blue"
/home/cis90/simben $ flowers
```

When the flowers script runs will it see the value of the roses variable or the violets variable?

Write your answer in the chat window

***Yes**, the flowers script can see the roses variable now which was exported*

```
/home/cis90/simben $ flowers

==> Entering child process <==
UID        PID  PPID  C STIME TTY          TIME CMD
simben90 25106 25059  0 17:16 pts/8    00:00:00 -bash
simben90 32147 25106  0 17:27 pts/8    00:00:00 /bin/bash /home/cis90/bin/flowers
simben90 32154 32147  0 17:27 pts/8    00:00:00 ps -f
==> showing variables in child <==
   roses are "red"
   violets are ""
==> setting variables in child <==
   roses are "black"
   violets are "orange"
==> Leaving child process <==

/home/cis90/simben $
```

The child now sees the value of roses but not violets (Rule #1)

Activity

```
/home/cis90/simben $ roses=red; violets=blue
/home/cis90/simben $ export roses violets
/home/cis90/simben $ env | grep roses
roses=red
/home/cis90/simben $ env | grep violets
violets=blue
/home/cis90/simben $ go
roses are "red" and violets are "blue"
/home/cis90/simben $ flowers
```

Will the flowers process change the values of the roses and violets variables?

Write your answer in the chat window

***No**, the flowers script which runs as a child process cannot change the parent's variables*

```
/home/cis90/simben $ flowers

==> Entering child process <==
UID        PID  PPID  C STIME TTY          TIME CMD
simben90 28732 28724  0 17:51 pts/0    00:00:00 -bash
simben90 29383 28732  0 18:11 pts/0    00:00:00 /bin/bash /home/cis90/bin/flowers
simben90 29390 29383  0 18:11 pts/0    00:00:00 ps -f
==> showing variables in child <==
   roses are "red"
   violets are "blue"
==> setting variables in child <==
   roses are "black"
   violets are "orange"
==> Leaving child process <==

/home/cis90/simben $ go
roses are "red" and violets are "blue"
/home/cis90/simben $
```

The child can only change copies of the parents variables

The child cannot change the parent's variables (Rule #2)

Activity

```
/home/cis90/simben $ roses=red; violets=blue
/home/cis90/simben $ export roses violets
/home/cis90/simben $ env | grep roses
roses=red
/home/cis90/simben $ env | grep violets
violets=blue
/home/cis90/simben $ go
roses are "red" and violets are "blue"
/home/cis90/simben $ . flowers
```

Now will the flowers process change the values of the roses and violets variables?

Write your answer in the chat window

***Yes**, if sourced, flowers will NOT run as a child process and so the variables are changed*

```
/home/cis90/simben $ . flowers

==> Entering child process <==
UID        PID  PPID  C STIME TTY          TIME CMD
simben90 28732 28724  0 17:51 pts/0    00:00:00 -bash
simben90 29480 28732  0 18:15 pts/0    00:00:00 ps -f
==> showing variables in child <==
   roses are "red"
   violets are "blue"
==> setting variables in child <==
   roses are "black"
   violets are "orange"
==> Leaving child process <==

/home/cis90/simben $ go
roses are "black" and violets are "orange"
/home/cis90/simben $
```

```
/home/cis90/rodduk $ cat .bash_profile
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
        . ~/.bashrc
fi

# User specific environment and startup programs

PATH=$PATH:$HOME/../bin:$HOME/bin:.
BASH_ENV=$HOME/.bashrc
USERNAME=""
PS1='$PWD $ '
export USERNAME BASH_ENV PATH
umask 002
set -o ignoreeof
stty susp
eval `tset -s -m vt100:vt100 -m :

/home/cis90/rodduk $
```

And now you know why the bash login scripts are sourced rather than run as child processes.

*Note: the **.** (dot) and **source** commands are equivalent*

```
/home/cis90/rodduk $ cat .bashrc
# .bashrc

# User specific aliases and functions

# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi
alias print="echo -e"
```

Activity

```
/home/cis90/simben $ roses=red; violets=blue
/home/cis90/simben $ export roses violets
/home/cis90/simben $ env | grep roses
roses=red
/home/cis90/simben $ env | grep violets
violets=blue
/home/cis90/simben $ go
roses are "red" and violets are "blue"
/home/cis90/simben $ exec flowers
```

What will happen if flowers is exec'ed?

Write your answer in the chat window

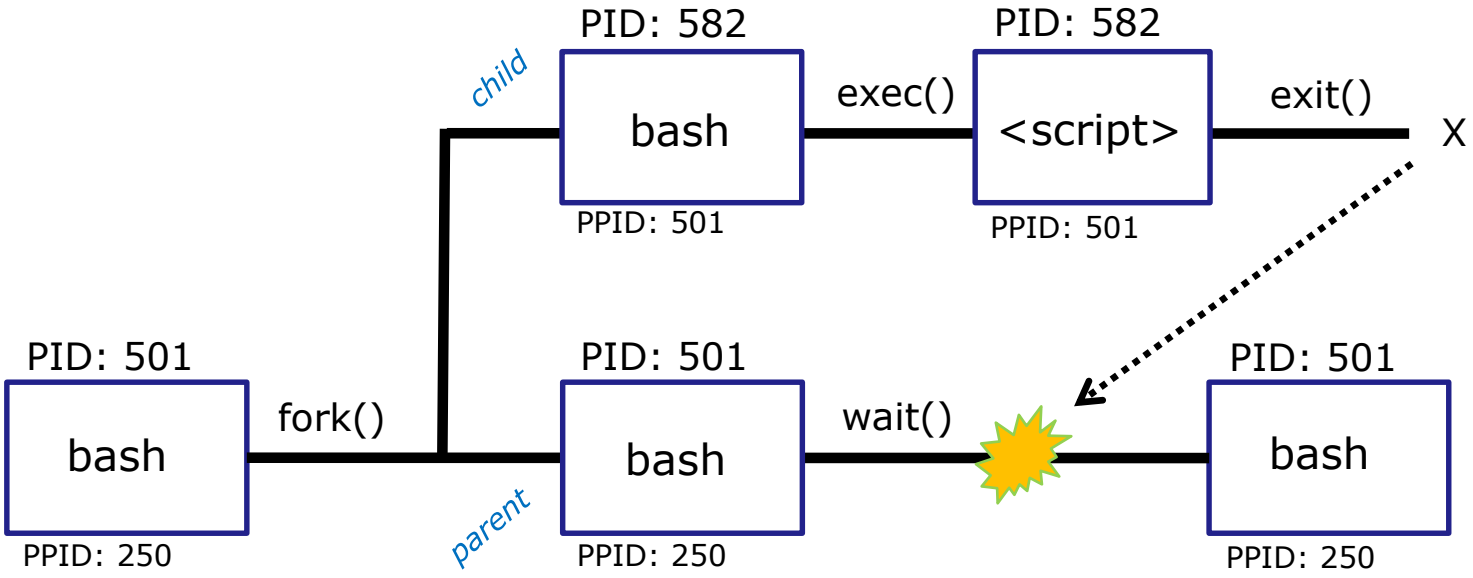
The flowers script overlays and replaces the bash code in your current process. It runs to completion and your session ends!

The rules of the road for variables

Process Rule #1: When a shell forks a child, only copies of exported variables are made available to the child.

Process Rule #2: A child can modify the variables it receives but those modifications will not change the parent's variables.

Running a script



Scripts run as a child process and the rules apply:

- When a shell forks a child process, only copies of exported variables are made available to the child.
- A child process can modify the variables it receives but those modifications will not change the parent's variables.

But what if we want a script to change the parent's variables?

. and source

Sometimes it is desirable to run a shell script (like .bash_profile or .bashrc) that will initialize or change shell variables in the parent environment.

. *<script>*
source *<script>*

} *equivalent*

To do this, the shell (bash) provides a **.** (dot) or **source** command, which instructs the shell to execute the shell script itself, without spawning a child process to run the script, and then continue on where it left off.

In the generic example above, the commands in the file *<script-name>* are run by the parent process, and therefore, any changes made to the environment will last for the duration of the login session.

Compare running vs sourcing a script

```
echo "smartphone=android" > google
echo 'echo smartphone is $smartphone' >> google
cat google
chmod +x google
```

Check that your google file contains:

```
smartphone=android
smartphone is $smartphone
```

```
smartphone=iPhone
echo $smartphone
```

Should be iPhone

Method 1

```
google
echo $smartphone
```

Run google script as a child process

Method 2

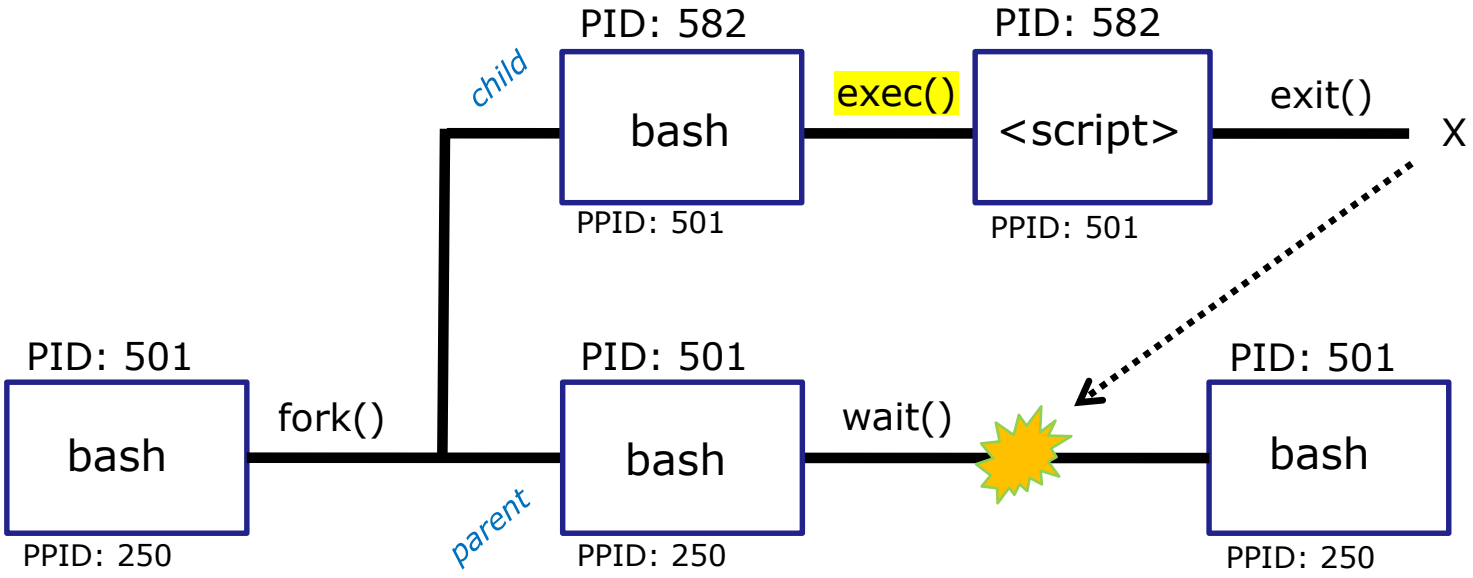
```
. google
echo $smartphone
```

Source google script so it runs as part of the parent process

Which method of running a script above changed the parent's smartphone variable?

Put your answer in the chat window

The exec system call



The exec() system call overlays the the child process with new code for the command being run

exec command

exec <*command*>

If a UNIX command is run using the **exec** <*command*>, the bash code in the process is overlaid by the <*command*> code, when finished the process will terminate.

Using exec command

```
/home/cis90/simben $ bash
[simben90@opus-ii ~]$ ps -l
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1201 23951 23950  0  80   0 - 28881 do_wai pts/2    00:00:00 bash
0 S  1201 24018 23951  0  80   0 - 28880 do_wai pts/2    00:00:00 bash
0 R  1201 24062 24018  0  80   0 - 37235 -      pts/2    00:00:00 ps
[simben90@opus-ii ~]$ exec sh
sh-4.2$ ps -l
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1201 23951 23950  0  80   0 - 28881 do_wai pts/2    00:00:00 bash
0 S  1201 24018 23951  0  80   0 - 28848 do_wai pts/2    00:00:00 sh
0 R  1201 24111 24018  0  80   0 - 37235 -      pts/2    00:00:00 ps
sh-4.2$ exec ksh
$ ps -l
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1201 23951 23950  0  80   0 - 28881 do_wai pts/2    00:00:00 bash
0 S  1201 24018 23951  0  80   0 - 29280 do_wai pts/2    00:00:00 ksh
0 R  1201 24188 24018  0  80   0 - 37235 -      pts/2    00:00:00 ps
$ exec bash
[simben90@opus-ii ~]$ ps -l
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1201 23951 23950  0  80   0 - 28881 do_wai pts/2    00:00:00 bash
0 S  1201 24018 23951  0  80   0 - 28881 do_wai pts/2    00:00:00 bash
0 R  1201 24252 24018  0  80   0 - 37235 -      pts/2    00:00:00 ps
[simben90@opus-ii ~]$ exit
exit
/home/cis90/simben $
```

Run second bash as child process

Replaces second bash process code with sh code

Replaces sh code with ksh code

Replaces ksh code with bash code

Exit back to parent bash process

You try it

```
bash
ps -l
exec sh
ps -l
exec ksh
ps -l
exec bash
ps -l
exit
```

Did the shell PID change each time you exec'ed a different shell?

Put your answer in the chat window

Housekeeping

1. Lab 10 due by 11:59PM tonight
2. Use the **check10** script to check your work
3. Don't forget to **submit your work**!
4. Check you Opus-II mail to verify your submission was successful and complete.
5. After you submit your lab10 file you may comment out your riddle command in *.bash_profile*
6. The Extra Credit Labs X1 and X2 (30 points each) are available. They will be graded the day after the final. Use **checkx2** to the second lab.
7. The **Final Project is available and due in two weeks**.

Heads up on Final Exam

Test #3 (final exam) is **MONDAY December 10th 1-3:50PM**

Mon	12/10	**Test #3 (the final exam)** **Time** • MONDAY 1:00PM - 3:50PM in Room 828 or online **Materials** • Presentation slides (download) • Test (canvas) **ConferZoom** • Enter virtual classroom • Class archives		5 posts Lab X1 Lab X2

Extra credit Labs X1/X2 and final posts ***due by 11:59PM***

Final grades *available by the end of the next day*

- All students will take the test at the same time. The test must be completed by **3:50PM**.
- Working and long distance students can take the test online via ConferZoom and Canvas.
- Working students will need to plan ahead to arrange time off from work for the test.
- Test #3 is **mandatory** (even if you have all the points you want)

FALL 2018 FINAL EXAMINATIONS SCHEDULE
DECEMBER 10 TO DECEMBER 15

DAYTIME FINAL SCHEDULE

Daytime Classes: All times in bold refer to the beginning times of classes. **MW/Daily** means Monday alone, Wednesday alone, Monday and Wednesday **or any 3** or more days in any combination. **TTH** means Tuesday alone, Thursday alone, or Tuesday and Thursday. **Classes meeting other combinations of days and/or hours not listed must have a final schedule approved by the Division Dean.**

STARTING CLASS TIME / DAY(S)	EXAM HOUR	EXAM DATE
Classes starting between:		
6:30 am and 8:55 am, MW/Daily	7:00 am-9:50 am	Monday, December 10
9:00 am and 10:15 am, MW/Daily	7:00 am-9:50 am	Wednesday, December 12
10:20 am and 11:35 am, MW/Daily	10:00 am-12:50 pm	Monday, December 10
11:40 am and 12:55 pm, MW/Daily	10:00 am-12:50 pm	Wednesday, December 12
1:00 pm and 2:15 pm, MW/Daily	1:00 pm-3:50 pm	Monday, December 10
2:20 pm and 3:35 pm, MW/Daily	1:00 pm-3:50 pm	Wednesday, December 12
3:40 pm and 5:30 pm, MW/Daily	4:00 pm-6:50 pm	Wednesday, December 12

CIS 90 **Introduction to UNIX/Linux**

Provides a technical overview of the UNIX/Linux operating system, including hands-on experience with commands, files, and tools. Recommended Preparation: CIS 1L or CIS 72.

Transfer Credit: Transfers to CSU;UC

Section	Days	Times	Units	Instructor	Room
1	W	1:00PM-4:05PM	3.00	R.Simms	OL
&	Arr.	Arr.		R.Simms	OL

Section 1 is an ONLINE course. Meets weekly throughout the semester online during the scheduled times by remote technology with an additional 50 min online lab per week. For details, see instructor's web page at go.cabrillo.edu/online. This course has zero cost for textbooks.

2	W	1:00PM-4:05PM	3.00	R.Simms	828
&	Arr.	Arr.		R.Simms	OL

Section 2 is a Hybrid ONLINE course. Meets weekly throughout the semester at the scheduled times with an additional 50 min online lab per week. For details, see instructor's web page at go.cabrillo.edu/online. This course has zero cost for textbooks.

Printers

Two predominate types of printers

- Thermal inkjet technology
- Laser, drum, toner technology

Printing in Linux

Printing Commands

The ATT System V way
- lp (to print)
- lpstat (queue management)
- cancel (to remove jobs)

The BSD (Berkeley Software Distribution) way
- lpr (to print)
- lpq (queue management)
- lprm (to remove jobs)

BSD is a branch of UNIX that was developed at the University of California, Berkeley

And now CUPS …
- Provides both System V and Berkeley based command-line interfaces
- Supports new Internet Printing Protocol
- Works with Samba

CUPS
lpstat command

Syntax: **lpstat** [*options*]

Example (Not on Opus-II):

```
rsimms@hugo:~$ lpstat -p
printer HP_LaserJet_1320_series is idle.  enabled since Tue 08 May
2012 08:46:45 PM PDT
```

The –p option will show the available printers

```
rsimms@hugo:~$ lpstat -p -d
printer HP_LaserJet_1320_series is idle.  enabled since Tue 08 May
2012 08:46:45 PM PDT
system default destination: HP_LaserJet_1320_series
```

The –d option will identify the default printer

CUPS

lpstat command

On Opus-II

What printers are available?

Which is the default printer?

Write your answers in the chat window

CUPS

lp and lpr commands

*Use **lp** (or **lpr**) to print files*

```
/home/cis90/simben $ lp lab10
request id is hplaser-5 (1 file(s))

/home/cis90/simben $ lp -d hplaser lab10
request id is hplaser-6 (1 file(s))
```

*With **lp**, use the –d option to manually select the printer*

```
/home/cis90/simben $ lpr lab10

/home/cis90/simben $ lpr -P hplaser lab10
```

*With **lpr**, use the –P option to manually select a printer*

CUPS

lp and lpr commands

```
/home/cis90/simben $ echo "Print Me Quietly" | lpr -P hplaser
/home/cis90/simben $
```

Note that both lp and lpr will read from stdin.

This allows output from another command to be piped in

CUPS

Practice Printing

On Opus, print your lab10 and letter files

```
lp lab10
lpstat

lpr letter
lpstat

echo "Print Me Quietly" | lpr -P hplaser
lpstat
```

When finished type "done" in the chat window

Managing Print Jobs

CUPS

Showing jobs waiting to print

```
[root@benji ~]# lpq
hp7550 is not ready
Rank    Owner   Job     File(s)
Total Size
1st     root    22      myfile
1024 bytes
2nd     root    23      myfile
1024 bytes
3rd     root    24      myfile
1024 bytes
4th     root    25      myfile
1024 bytes
```

*Use **lpq** or **lpstat** with no options to show spooled print jobs*

```
[root@benji ~]# lpstat
hp7550-22               root              1024   Sat
15 Nov 2008 12:20:23 PM PST
hp7550-23               root              1024   Sat
15 Nov 2008 12:20:28 PM PST
hp7550-24               root              1024   Sat
15 Nov 2008 12:20:31 PM PST
hp7550-25               root              1024   Sat
15 Nov 2008 12:20:34 PM PST
```

CUPS

Removing/canceling pending print jobs

```
[root@benji ~]# lpq
hp7550 is not ready
Rank    Owner   Job     File(s)
Total Size
1st     root    22      myfile
1024 bytes
2nd     root    23      myfile
1024 bytes
3rd     root    24      myfile
1024 bytes
4th     root    25      myfile
1024 bytes

[root@benji ~]# cancel 22
[root@benji ~]# cancel 23
[root@benji ~]# lprm 24
[root@benji ~]# lprm 25

[root@benji ~]# lpq
hp7550 is not ready
no entries

[root@benji ~]# lpstat
[root@benji ~]#
```

*Use **cancel** or **lprm** to remove print jobs*

CUPS

Practice Printing

Cancel your print jobs on Opus-II

```
lpq
lpstat

cancel <print job number>
lpq

lprm <print job number>
lpq
```

When finished type "gone" in the chat window

$(*some-command*)

Utilizing $(*some-command*)

The **$** metacharacter provides the "value" of:

- variables, e.g. $PS1
- commands, e.g. $(*some-command*)

```
/home/cis90/simben $ echo $PS1
$PWD $

/home/cis90/simben $ echo $(grep love poems/Shakespeare/* | wc -l)
11

/home/cis90/simben $ myname=$(grep $LOGNAME /etc/passwd | cut -f5 -d":")
/home/cis90/simben $ echo My name is $myname
My name is Benji Simms
```

This is useful when you want to insert the output of a command into a sentence being echoed

Activity

```
dir=$(echo $PATH | cut -f6 -d":")
echo The 6th directory on my PATH is $dir
```

Paste the output of your echo command into the chat window

date command part II

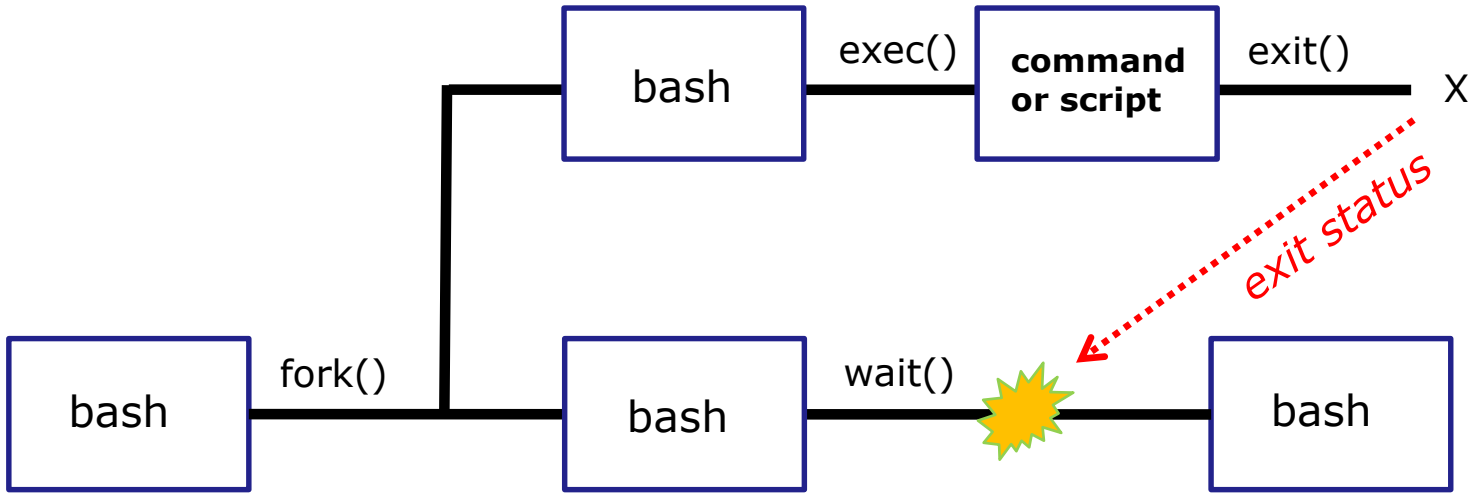
Utilizing the date command

```
/home/cis90/milhom/bin $ date
Tue Nov 24 14:33:41 PST 2015

/home/cis90/milhom/bin $ date +'%r'
02:33:53 PM

/home/cis90/milhom/bin $ date +'%A'
Tuesday

/home/cis90/milhom/bin $ date +'%m/%d/%Y'
11/24/2015
```

See the man page on date for lots of other % codes

Communicating status back to parent

The child can communicate status back to the parent



The child process makes an exit() system call to release all resources. The child remains a zombie until the exit status is communicated to the parent.

Utilizing the status

Yes, there is a variable named ?

This variable will be set to the exit status of the command or script that just ran.

```
/home/cis90/milhom/bin $ grep bogus /etc/passwd > /dev/null
/home/cis90/milhom/bin $ echo $?
1
```
status=1 (grep found no matches)

```
/home/cis90/milhom/bin $ grep $LOGNAME /etc/passwd > /dev/null
/home/cis90/milhom/bin $ echo $?
0
```
status=0 (grep found one or more matches)

A status=0 typically indicates success and non-zero values are error codes

Utilizing the status

```
/home/cis90/milhom/bin $ ping -c1 sun-hwa-iv.cis.cabrillo.edu
PING sun-hwa-iv.cis.cabrillo.edu (172.20.90.61) 56(84) bytes of data.

--- sun-hwa-iv.cis.cabrillo.edu ping statistics ---
1 packets transmitted, 0 received, 100% packet loss, time 0ms
/home/cis90/milhom/bin $ echo $?
1
```

status=1 (sun-hwa-iv is down right now)

```
/home/cis90/milhom/bin $ ping -c1 simms-teach.com
PING simms-teach.com (208.113.154.64) 56(84) bytes of data.
64 bytes from apache2-dap.giles.dreamhost.com (208.113.154.64): icmp_seq=1 ttl=43 time=78.9 ms

--- simms-teach.com ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 164ms
rtt min/avg/max/mdev = 78.957/78.957/78.957/0.000 ms
/home/cis90/milhom/bin $ echo $?
0
```

status=0 (simms-teach.com website is up right now)

Utilizing the status

```
/home/cis90/milhom/bin $ ping -c1 cousin-of-opus.simms-teach.com
ping: cousin-of-opus.simms-teach.com: Name or service not known
/home/cis90/milhom/bin $ echo $?
2
```

status=2 (there is no cousin-of-Opus system in that domain)

Color

Using Color

Black 0;30	Green 0;32	Red 0;31	Brown 0;33
Dark Gray 1;30	Light Green 1;32	Light Red 1;31	Yellow 1;33
Blue 0;34	Cyan 0;36	Purple 0;35	Light Gray 0;37
Light Blue 1;34	Light Cyan 1;36	Light Purple 1;35	White 1;37

```
/home/cis90/simben/bin $ echo -e "\e[00;31mMy favorite color is RED\e[00m"
My favorite color is RED
/home/cis90/simben/bin $ echo -e "\e[00;34mMy favorite color is BLUE\e[00m"
My favorite color is BLUE
/home/cis90/simben/bin $ echo -e "\e[00;32mMy favorite color is GREEN\e[00m"
My favorite color is GREEN
/home/cis90/simben/bin $
```

*Use **echo -e "\e[0**n;nn**m"** to turn on color and **\e[00m** to turn it off.*

(the -e option enables interpretation of backslash escapes)

More: https://misc.flogisoft.com/bash/tip_colors_and_formatting

Using Color

```
/home/cis90/simben/bin $ echo -e "\e[00;32m"

/home/cis90/simben/bin $ head -4 ~/letter
Hello Mother!  Hello Father!

Here I am at Camp Granada.  Things are very entertaining,
and they say we'll have some fun when it stops raining.
/home/cis90/simben/bin $ echo -e '\e[00m'

/home/cis90/simben/bin $
```

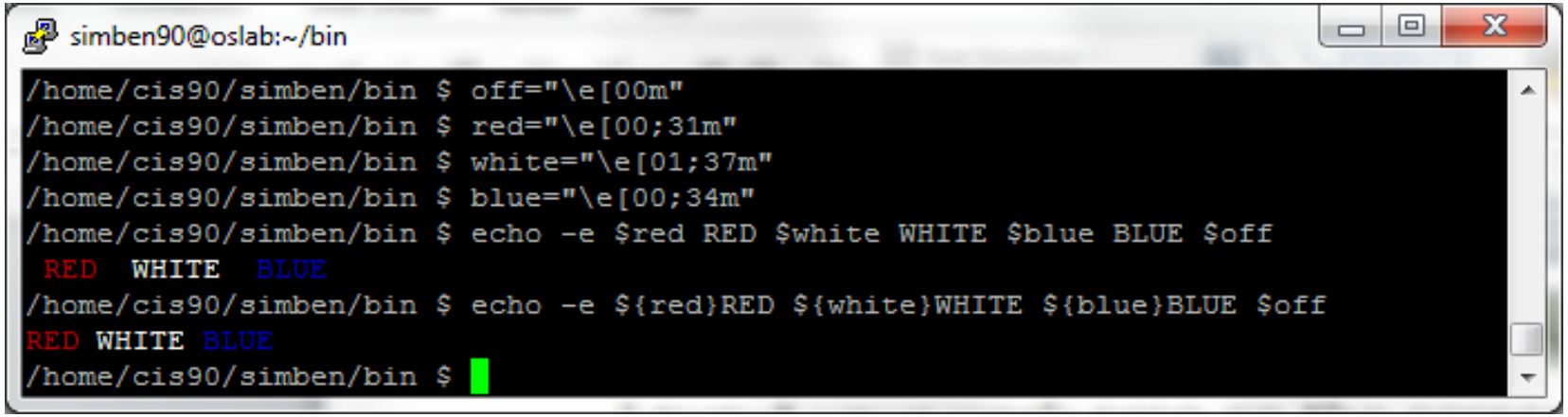
Change to color green

Revert color back to normal

More: https://misc.flogisoft.com/bash/tip_colors_and_formatting

Using Color

```
off="\e[00m"
red="\e[00;31m"
white="\e[01;37m"
blue="\e[00;34m"
echo -e $red RED $white WHITE $blue BLUE $off
echo -e ${red}RED ${white}WHITE ${blue}BLUE $off
```



Demonstrating the use of variables and curly braces to make color easier to use.

Curly braces are used to clearly separate the variable name from adjacent text strings:

```
$redRED is null
${red}RED is "\e[00;31mRED"
```

Final Project myscript

milhom90@oslab:~/bin

```
#!/bin/bash
#
# menu: A simple menu template
#
while true
do
        clear
        echo -n "
                CIS 90 Final Project
        1) Task 1
        2) Task 2
        3) Task 3
        4) Task 4
        5) Task 5
        6) Exit

        Enter Your Choice: "
        read RESPONSE
        case $RESPONSE in
          1)    # Commands for Task 1
                ;;
          2)    # Commands for Task 2
                ;;
          3)    # Commands for Task 3
                ;;
          4)    # Commands for Task 4
                ;;
          5)    # Commands for Task 5
                ;;
          6)    exit 0
                ;;
          *)    echo "Please enter a number between 1 and 6"
                ;;
        esac
        echo -n "Hit the Enter key to return to menu "
        read dummy
done
~
                                                        1,1           All
```

You will modify and extend this script for your final project

Final Project

If you did not do this last week, please do so now

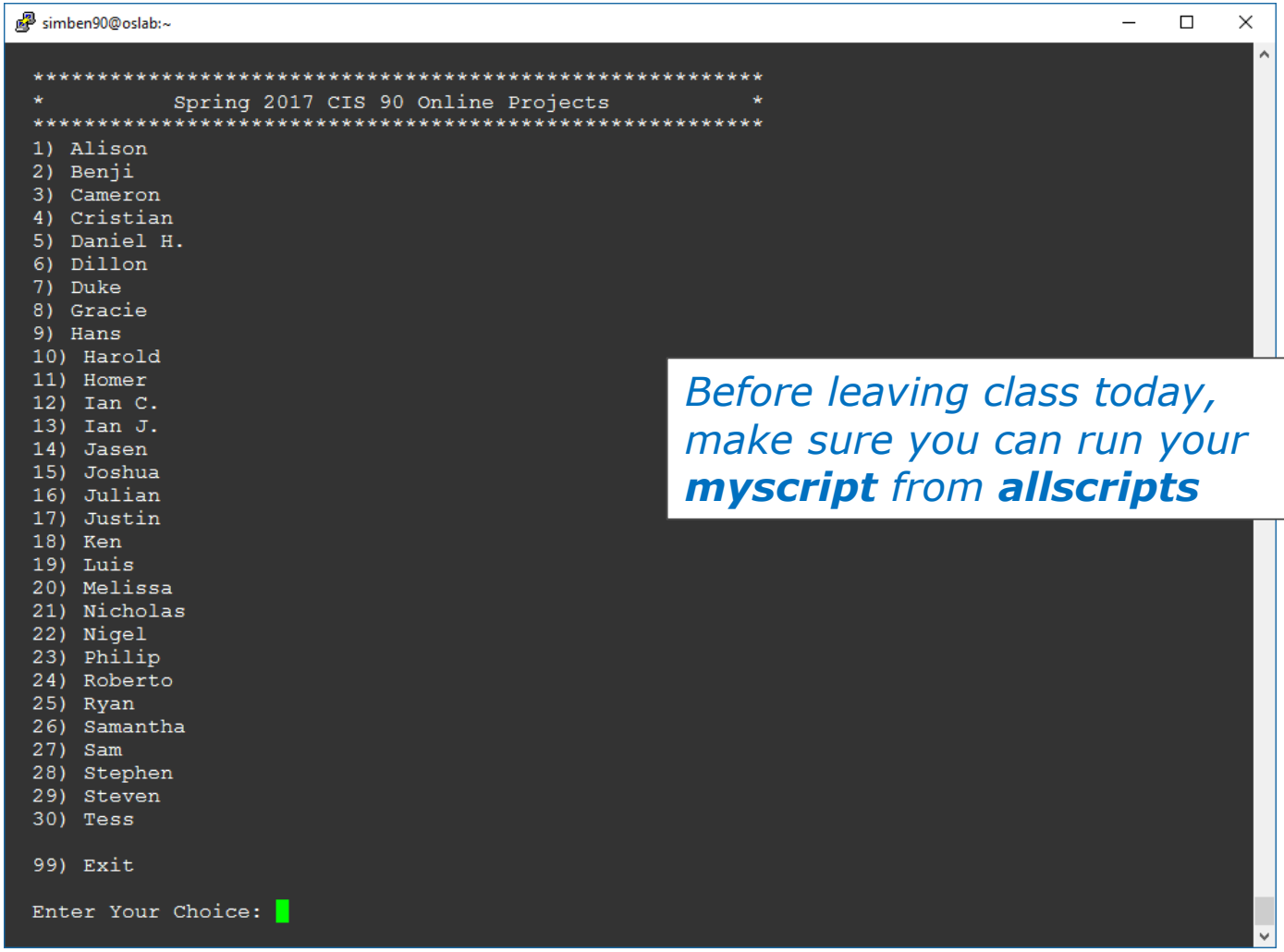
Getting Started

1) On Opus-II, cd to your home directory and enter:
   cd
   cp ../depot/myscript bin/

2) Give your script execute permissions with:
   chmod +x bin/myscript

3) Run the script:
   myscript

Final Project



*Before leaving class today, make sure you can run your **myscript** from **allscripts***

Final Project Grading Rubric

Possible Points	Requirements
30	Implementing all five tasks (6 points each): • Requirements for each task: – Minimum of 12 "original" lines of bash script – Has one or more non-generic comments to explain what it is doing – Has user interaction
24	At least six bash constructs from this list: • Redirecting stdin (4 points) • Redirecting stdout (4 points) • Redirecting stderr (4 points) • Use of permissions (4 points) • Use of filename expansion characters (4 points) • Use of absolute path (4 points) • Use of relative path (4 points) • Use of a PID (4 points) • Use of inodes (4 points) • Use of links (4 points) • Use of color (4 points) • Use of scheduling (4 points) • Use of a GID or group (4 points) • Use of a UID or user (4 points) • Use of a /dev/tty device (4 points) • Use of a signal (4 points) • Use of piping (4 points) • Use of an environment variable (4 points) • Use of /bin/mail (4 points) • Use of a conditional (4 points) • Use of $(*command*) The maximum for this section is 24 points.
6	Present your script to the class
Points lost	
-15	Fails to run from **allscripts**
-15	Other students in the class are unable to read and execute your script.
-15	Error messages are displayed when running one or more tasks
-up to 90	No credit for any task which contains unoriginal script code that: • Doesn't give full credit to the original author. • Doesn't indicate where the code was obtained from. • Doesn't include licensing terms. • Violates copyright or licensing terms.
-up to 90	For any "malware" scripts that steal credentials, exfiltrate confidential information, remove or encrypt a user's files or creates a denial of service condition on Opus-II.
Extra credit	
30	Up to three additional tasks (10 points each)

Grading Rubric for Final Project

Plagiarizing another author's code is a NO-NO! All points lost!

Scripts that result in unauthorized hacking" is a NO-NO! All points lost!

Final Project permissions

Permissions

A past forum post …

Ha Ha Class

by [redacted] on Tue May 12, 2009 12:22 pm

I'm sure this is some kind of payback for last week "Hacking" attempt

I will find out who did this

~~~~~~

ps. Im going to pass

Uh, oh … someone got hacked!
~~~~~~

Group Write Permissions

ls -l /home/cis90/*/bin/myscript

```
rsimms@opus-ii:~
[rsimms@opus-ii ~]$ date
Sun Apr 29 18:39:03 PDT 2018
[rsimms@opus-ii ~]$ ls -l /home/cis90/*/bin/myscript
-rwxrwxr-x. 1 arrdav90 cis90 739 Apr 25 10:07 /home/cis90/arrdav/bin/myscript
-rwxrwxr-x. 1 ausedg90 cis90 700 Apr 25 10:01 /home/cis90/ausedg/bin/myscript
-rwxrwxr-x. 1 banric90 cis90 781 Apr 25 10:01 /home/cis90/banric/bin/myscript
-rwxrwxr-x. 1 broada90 cis90 782 Apr 25 10:21 /home/cis90/broada/bin/myscript
-rwxrwxr-x. 1 chudar90 cis90 549 Apr 26 13:59 /home/cis90/chudar/bin/myscript
-rwxrwxr-x. 1 farcia90 cis90 549 Apr 25 10:51 /home/cis90/farcia/bin/myscript
-rwxrwxr-x. 1 milhom90 cis90 781 Apr 24 12:56 /home/cis90/milhom/bin/myscript
-rwxrwxr-x. 1 monele90 cis90 698 Apr 25 10:09 /home/cis90/monele/bin/myscript
-rwxrwxr-x. 1 ohapau90 cis90 717 Apr 25 10:04 /home/cis90/ohapau/bin/myscript
-rwxrwxr-x. 1 ottlai90 cis90 689 Apr 25 10:03 /home/cis90/ottlai/bin/myscript
-rwxrwxr-x. 1 pernat90 cis90 702 Apr 25 10:12 /home/cis90/pernat/bin/myscript
-rwxrwxr-x. 1 plabra90 cis90 700 Apr 25 10:17 /home/cis90/plabra/bin/myscript
-rwxrwxr-x. 1 ranlui90 cis90 751 Apr 25 10:12 /home/cis90/ranlui/bin/myscript
-rwxrw-r--. 1 rocces90 cis90 725 Apr 25 10:00 /home/cis90/rocces/bin/myscript
-rwxr-x---. 1 simben90 cis90 549 Apr 23 16:11 /home/cis90/simben/bin/myscript
-rwxrwxr-x. 1 telnat90 cis90 711 Apr 25 10:17 /home/cis90/telnat/bin/myscript
-rwxrwxr-x. 1 vanjoa90 cis90 795 Apr 29 18:27 /home/cis90/vanjoa/bin/myscript
-rwxrwxr-x. 1 wilnov90 cis90 704 Apr 25 09:59 /home/cis90/wilnov/bin/myscript
-rwxrwxr-x. 1 winsha90 cis90 701 Apr 25 10:02 /home/cis90/winsha/bin/myscript
[rsimms@opus-ii ~]$
```

*Which **myscript** files can only be edited by their owner?*
Which ones could be edited by anyone in the CIS 90 class?
Which ones could be edited by anyone on Opus-II?

Group Read and Execute Permissions

rsimms@opus-ii:~

```
[rsimms@opus-ii ~]$ date
Sun Apr 29 18:39:40 PDT 2018
[rsimms@opus-ii ~]$ /home/cis90/bin/checkmyscripts
-rwxr-x---. 1 simben90 cis90 549 Apr 23 16:11 /home/cis90/simben/bin/myscript
-rwxrwxr-x. 1 milhom90 cis90 781 Apr 24 12:56 /home/cis90/milhom/bin/myscript
ls: cannot access /home/cis90/rodduk/bin/myscript: No such file or directory
ls: cannot access /home/cis90/angjak/bin/myscript: No such file or directory
-rwxrwxr-x. 1 ausedg90 cis90 700 Apr 25 10:01 /home/cis90/ausedg/bin/myscript
-rwxrwxr-x. 1 banric90 cis90 781 Apr 25 10:01 /home/cis90/banric/bin/myscript
ls: cannot access /home/cis90/bilfri/bin/myscript: No such file or directory
-rwxrwxr-x. 1 chudar90 cis90 549 Apr 26 13:59 /home/cis90/chudar/bin/myscript
ls: cannot access /home/cis90/klenat/bin/myscript: No such file or directory
-rwxrwxr-x. 1 ohapau90 cis90 717 Apr 25 10:04 /home/cis90/ohapau/bin/myscript
ls: cannot access /home/cis90/olscla/bin/myscript: No such file or directory
-rwxrwxr-x. 1 ottlai90 cis90 689 Apr 25 10:03 /home/cis90/ottlai/bin/myscript
-rwxrwxr-x. 1 pernat90 cis90 702 Apr 25 10:12 /home/cis90/pernat/bin/myscript
ls: cannot access /home/cis90/ragjet/bin/myscript: No such file or directory
-rwxrw-r--. 1 rocces90 cis90 725 Apr 25 10:00 /home/cis90/rocces/bin/myscript
-rwxrwxr-x. 1 vanjoa90 cis90 795 Apr 29 18:27 /home/cis90/vanjoa/bin/myscript
-rwxrwxr-x. 1 wilnov90 cis90 704 Apr 25 09:59 /home/cis90/wilnov/bin/myscript
-rwxrwxr-x. 1 arrdav90 cis90 739 Apr 25 10:07 /home/cis90/arrdav/bin/myscript
-rwxrwxr-x. 1 broada90 cis90 782 Apr 25 10:21 /home/cis90/broada/bin/myscript
-rwxrwxr-x. 1 farcia90 cis90 549 Apr 25 10:51 /home/cis90/farcia/bin/myscript
ls: cannot access /home/cis90/fuldan/bin/myscript: No such file or directory
ls: cannot access /home/cis90/kankim/bin/myscript: No such file or directory
-rwxrwxr-x. 1 monele90 cis90 698 Apr 25 10:09 /home/cis90/monele/bin/myscript
ls: cannot access /home/cis90/padhen/bin/myscript: No such file or directory
-rwxrwxr-x. 1 plabra90 cis90 700 Apr 25 10:17 /home/cis90/plabra/bin/myscript
-rwxrwxr-x. 1 ranlui90 cis90 751 Apr 25 10:12 /home/cis90/ranlui/bin/myscript
-rwxrwxr-x. 1 telnat90 cis90 711 Apr 25 10:17 /home/cis90/telnat/bin/myscript
-rwxrwxr-x. 1 winsha90 cis90 701 Apr 25 10:02 /home/cis90/winsha/bin/myscript
[rsimms@opus-ii ~]$
```

Which myscript files cannot be run by classmates?

Class Activity

Note: One of the requirements for the final project is setting permissions on your script so that all cis90 members can read and run it.

To meet this requirement use:

```
cd
chmod 750 bin bin/myscript
ls -ld bin bin/myscript
```

When finished check that your script can be run by other CIS 90 students:

```
su - tbd1090
   (use the "funny Cabrillo" password)
allscripts
exit
```

Run you script and write "success" or "not working" into the chat window

umask again!

Permissions

Why can other classmates modify my scripts?

Before Lab 10

```
/home/cis90/simben/bin $ umask
0002
/home/cis90/simben $ rm newscript; touch newscript
/home/cis90/simben $ ls -l newscript
-rw-rw-r-- 1 simben cis90 0 Nov 23 16:17 newscript
/home/cis90/simben $ chmod +x newscript
/home/cis90/simben $ ls -l newscript
-rwxrwxr-x 1 simben cis90 0 Nov 23 16:17 newscript
```

After Lab 10

```
/home/cis90/simben $ umask
0006
/home/cis90/simben $ rm newscript; touch newscript
/home/cis90/simben $ ls -l newscript
-rw-rw---- 1 simben cis90 0 May 12 08:44 newscript
/home/cis90/simben $ chmod +x newscript
/home/cis90/simben $ ls -l newscript
-rwxrwx--x 1 simben cis90 0 May 12 08:44 newscript
```

Because your umask setting allows group members to have write permission on any new files you create!

Permissions

```
[rodduk90@opus-ii bin]$ cat /home/cis90/rodduk/.bash_profile
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
        . ~/.bashrc
fi

# User specific environment and startup programs

PATH=$PATH:$HOME/../bin:$HOME/bin:.
BASH_ENV=$HOME/.bashrc
USERNAME=""
PS1='$PWD $ '
export USERNAME BASH_ENV PATH
umask 002
set -o ignoreeof
stty susp
eval `tset -s -m vt100:vt100 -m :\?${TERM:-ansi} -r -Q `
```

Note your umask is defined in .bash_profile which runs every time you login. In lab 10 you change this setting to 006.

Class Activity

- Change your umask to 026
- Can group or other users modify future new files now?
- Try it, **touch** a new file and check the permissions with **ls -l**

How would you make this a permanent umask setting?

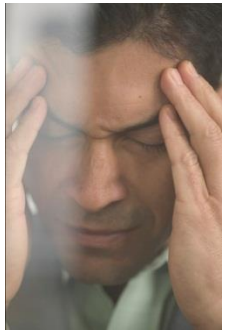
Write your answer in the chat window

Final Project Getting Started

What takes longer?

Writing the script?

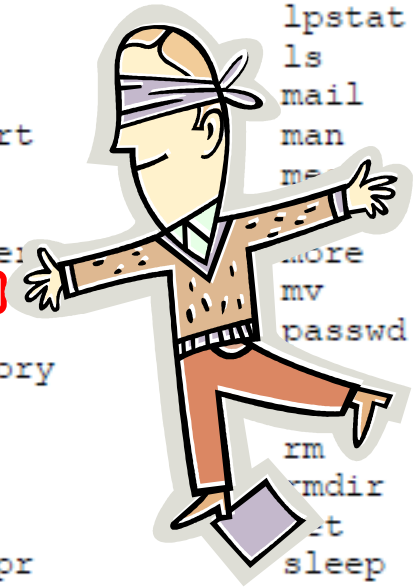
Or deciding what to script?



One way to get started … select a random command to build a script around

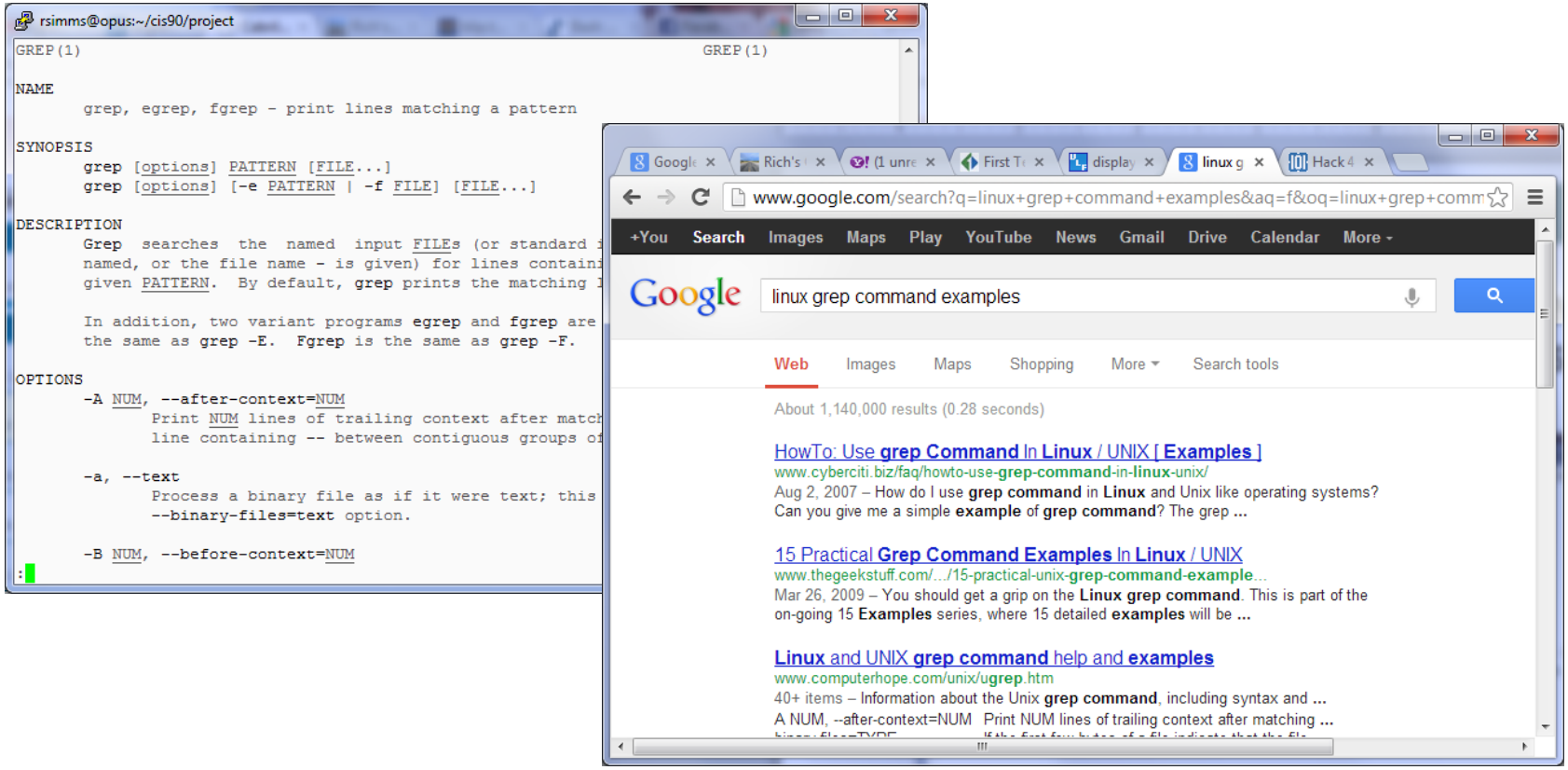
Commands

.	echo	lpstat	sort
at	env	ls	spell
banner	exit	mail	su
bash	export	man	tail
bc	file	me	tee
cal	find		touch
cancel	finger	more	type
cat	grep	mv	umask
cd	head	passwd	uname
chgrp	history		unset
chmod	id		vi
chown	jobs	rm	wc
clear	kill	rmdir	who
cp	ln		write
date	lp/lpr	sleep	xxd



For this example we will pick the grep command

Research your command by reading the man page and googling examples



Review the various options and arguments for the command

Next, decide what you want to do with the command you selected. For this example we will:

1. Start a new task in **myscript**
2. Customize the menu for the new task
3. Start with a simple **grep** command
4. Add some simple interaction
5. Add successive grep commands that experiment with different options
6. Iterate till happy with it.

Start hacking the menu!

Customize the menu options for Task 1

rodduk90@oslab:~/bin

```
#!/bin/bash
#
# menu: A simple menu template
#
while true
do
        clear
        echo -n "
                CIS 90 Final Project
        1) Task 1
        2) Task 2
        3) Task 3
        4) Task 4
        5) Task 5
        6) Exit

        Enter Your Choice: "
        read RESPONSE
        case $RESPONSE in
          1)    # Commands for Task
                ;;
          2)    # Commands for Task
                ;;
"myscript" 37L, 546C
```

Before

After

rodduk90@oslab:~/bin

```
#!/bin/bash
#
# menu: A simple menu template
#
while true
do
        clear
        echo -n "
                CIS 90 Final Project
        1) Hacking with the grep command
        2) Task 2
        3) Task 3
        4) Task 4
        5) Task 5
        6) Exit

        Enter Your Choice: "
        read RESPONSE
        case $RESPONSE in
          1)    # Commands for Task 1
                ;;
          2)    # Commands for Task 2
                ;;
-- INSERT --                                    10,5-12       Top
```

www.catb.org/jargon/html/H/hacker.html

hacker: n.

[originally, someone who makes furniture with an axe]

1. A person who enjoys exploring the details of programmable systems and how to stretch their capabilities, as opposed to most users, who prefer to learn only the minimum necessary. RFC1392, the *Internet Users' Glossary*, usefully amplifies this as: A person who delights in having an intimate understanding of the internal workings of a system, computers and computer networks in particular.

2. One who programs enthusiastically (even obsessively) or who enjoys programming rather than just theorizing about programming.

3. A person capable of appreciating *hack value*.

4. A person who is good at programming quickly.

5. An expert at a particular program, or one who frequently does work using it or on it; as in 'a Unix hacker'. (Definitions 1 through 5 are correlated, and people who fit them congregate.)

6. An expert or enthusiast of any kind. One might be an astronomy hacker, for example.

7. One who enjoys the intellectual challenge of creatively overcoming or circumventing limitations.

8. [deprecated] A malicious meddler who tries to discover sensitive information by poking around. Hence password hacker, network hacker. The correct term for this sense is *cracker*.

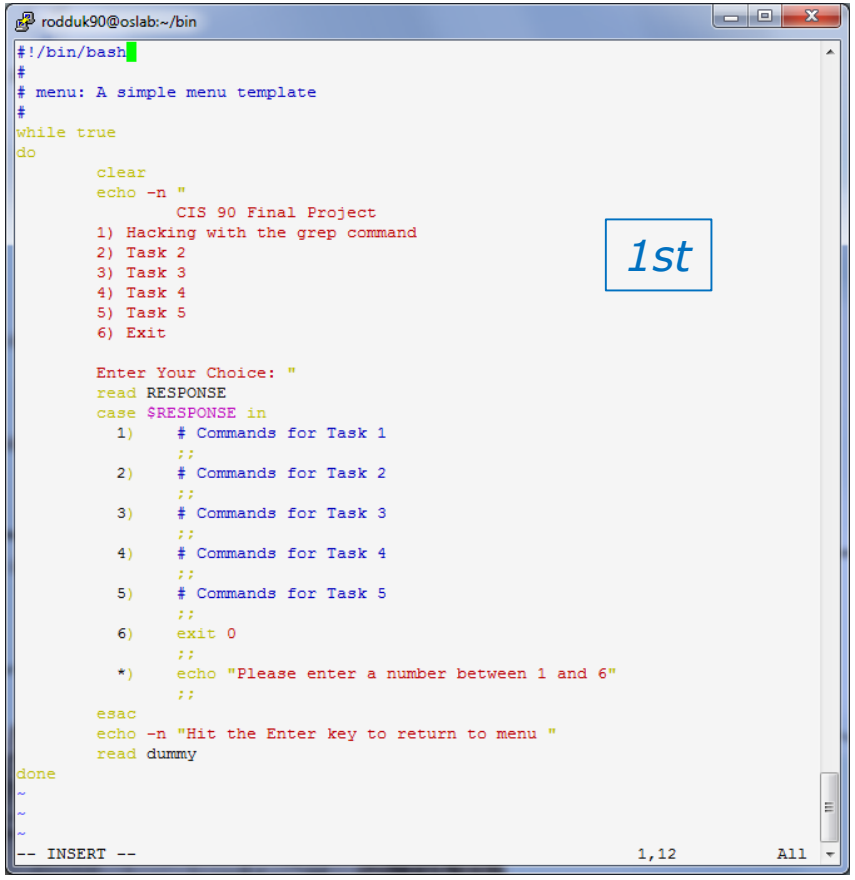
The term 'hacker' also tends to connote membership in the global community defined by the net (see *the network*. For discussion of some of the basics of this culture, see the How To Become A Hacker FAQ. It also implies that the person described is seen to subscribe to some version of the hacker ethic (see *hacker ethic*).

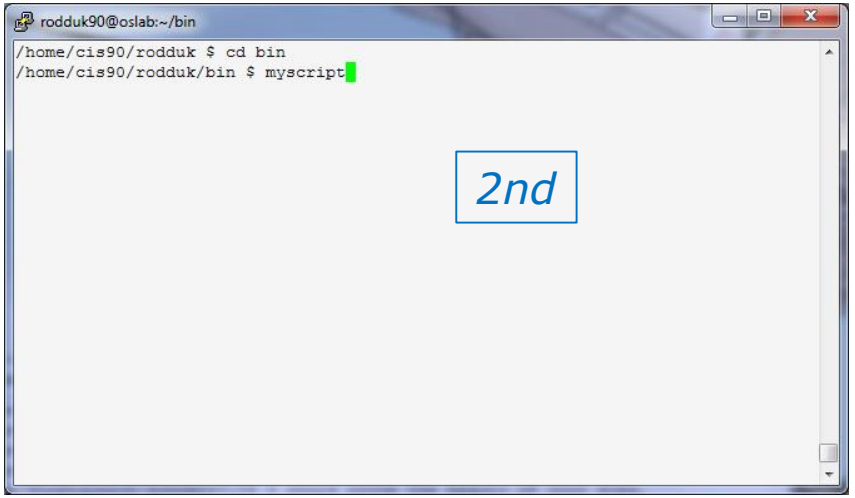
It is better to be described as a hacker by others than to describe oneself that way. Hackers consider themselves something of an elite (a meritocracy based on ability), though one to which new members are gladly welcome. There is thus a certain ego satisfaction to be had in identifying yourself as a hacker (but if you claim to be one and are not, you'll quickly be labeled *bogus*). See also *geek*, *wannabee*.

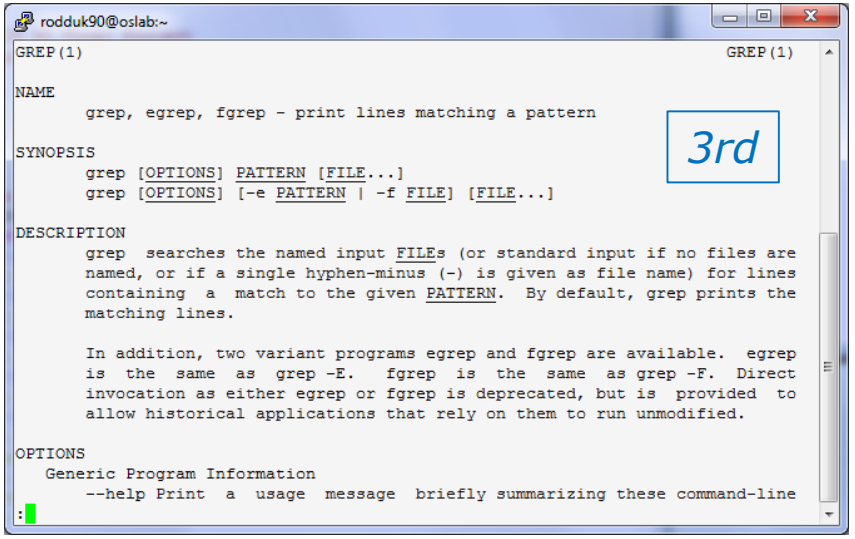
This term seems to have been first adopted as a badge in the 1960s by the hacker culture surrounding TMRC and the MIT AI Lab. We have a report that it was used in a sense close to this entry's by teenage radio hams and electronics tinkerers in the mid-1950s.

Hacking (building, exploring) is not cracking (malicious)

Layout your work area on the screen







Utilize screen real estate with multiple windows:

- *the 1st for vi,*
- *the 2nd for testing **myscript**,*
- *and a 3rd for experimenting or showing man pages*

Test your menu change

rodduk90@oslab:~/bin

```
#!/bin/bash
#
# menu: A simple menu template
#
while true
do
        clear
        echo -n "
                CIS 90 Final Project
        1) Hacking with the grep command
        2) Task 2
        3) Task 3
        4) Task 4
        5) Task 5
        6) Exit

        Enter Your Choice: "
        read RESPONSE
        case $RESPONSE in
          1)    # Commands for Task 1
                ;;
          2)    # Commands for Task 2
                ;;
          3)    # Commands for Task 3
                ;;
          4)    # Commands for Task 4
                ;;
          5)    # Commands for Task 5
                ;;
          6)    exit 0
                ;;
          *)    echo "Please enter a number between 1 and 6"
                ;;
        esac
        echo -n "Hit the Enter key to return to menu "
        read dummy
done
~
~
~
"myscript" 37L, 569C written                                1,11          All
```

rodduk90@oslab:~/bin

```
                CIS 90 Final Project
        1) Hacking with the grep command
        2) Task 2
        3) Task 3
        4) Task 4
        5) Task 5
        6) Exit

        Enter Your Choice:
```

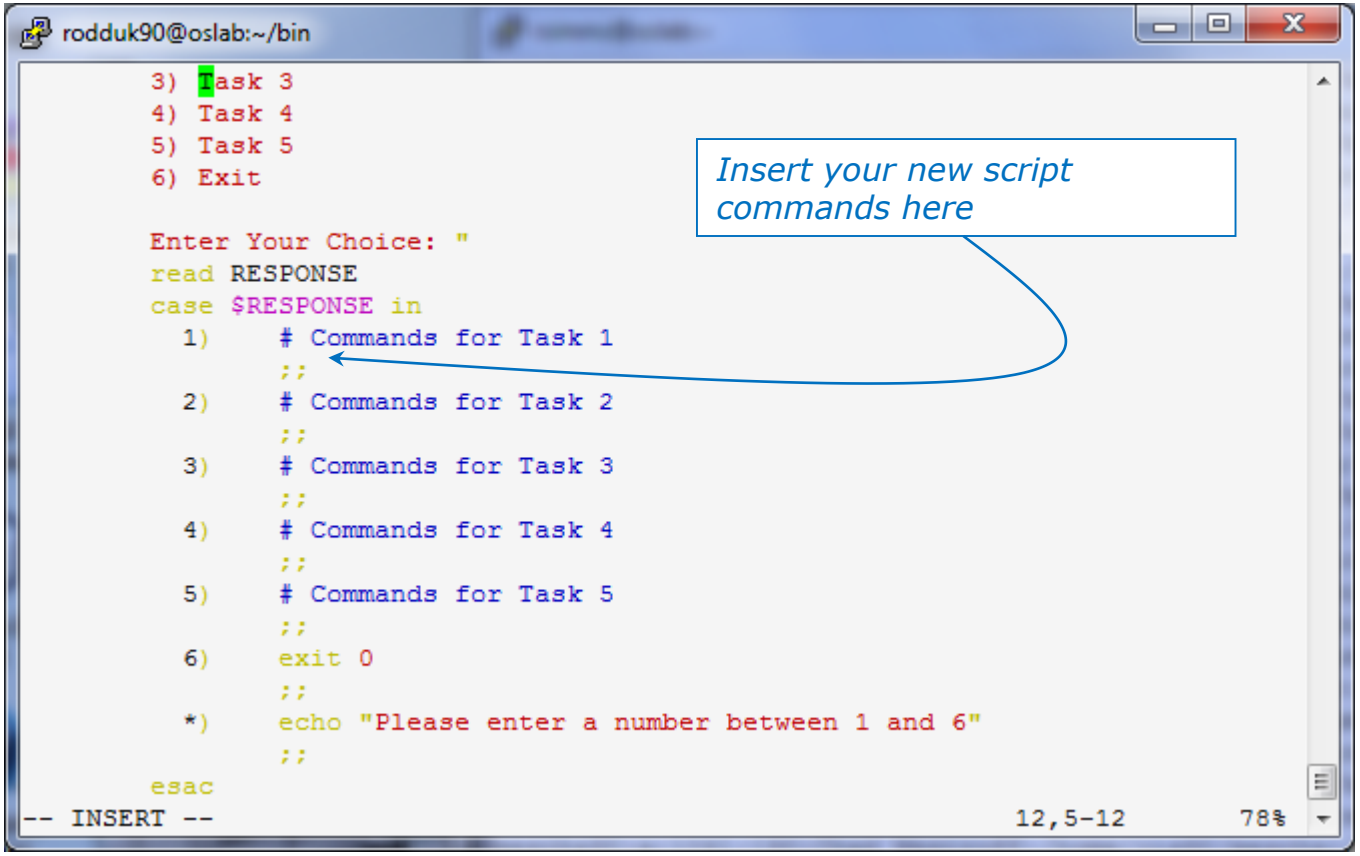
Changes work!

rodduk90@oslab:~

```
GREP(1)                                                          GREP(1)

NAME
       grep, egrep, fgrep - print lines matching a pattern

SYNOPSIS
       grep [OPTIONS] PATTERN [FILE...]
       grep [OPTIONS] [-e PATTERN | -f FILE] [FILE...]

DESCRIPTION
       grep  searches the named input FILEs (or standard input if no files are
       named, or if a single hyphen-minus (-) is given as file name) for lines
       containing  a  match to the given PATTERN.  By default, grep prints the
       matching lines.

       In addition, two variant programs egrep and fgrep are available.  egrep
       is  the  same  as  grep -E.  fgrep  is  the  same  as grep -F.  Direct
       invocation as either egrep or fgrep is deprecated, but is  provided  to
       allow historical applications that rely on them to run unmodified.

OPTIONS
   Generic Program Information
       --help Print  a  usage  message  briefly summarizing these command-line
:
```

*Run **myscript** in the 2nd window and verify your changes work*

Find the location to insert your new task commands



Now its time to add some commands to the task.

*Be sure to insert commands **after** the generic comment and **before** the ;;*

Add a simple command first and test it

rodduk90@oslab:~/bin

```
#!/bin/bash
#
# menu: A simple menu template
#
while true
do
        clear
        echo -n "
                CIS 90 Final Project
        1) Hacking with the grep command
        2) Task 2
        3) Task 3
        4) Task 4
        5) Task 5
        6) Exit

        Enter Your Choice: "
        read RESPONSE
        case $RESPONSE in
          1)    # Commands for Task 1
                grep beauty poems/*/*
                ;;
          2)    # Commands for Task 2
                ;;
          3)    # Commands for Task 3
                ;;
          4)    # Commands for Task 4
                ;;
          5)    # Commands for Task 5
                ;;
          6)    exit 0
                ;;
          *)    echo "Please enter a number between 1 and 6"
                ;;
        esac
        echo -n "Hit the Enter key to return to menu "
        read dummy
done
~
~
"myscript" 38L, 593C written                                21,15-29      All
```

*Experiment with a **grep** command in 3rd window*

*In the 1st window add the new grep command then save with **<esc>:w** (don't quit vi)*

*Run **myscript** in the 2nd second window to test change.*

rodduk90@oslab:~/bin

```
                CIS 90 Final Project
        1) Hacking with the grep command
        2) Task 2
        3) Task 3
        4) Task 4
        5) Task 5
        6) Exit

        Enter Your Choice: 1
grep: poems/*/*: No such file or directory
Hit the Enter key to return to menu
```

Oops, the change broke the script! Why? Because the relative path (beauty poems//*) does not work from the bin directory*

rodduk90@oslab:~

```
/home/cis90/rodduk $ grep beauty poems/*/*
poems/Shakespeare/sonnet1:That thereby beauty's rose might never die,
poems/Shakespeare/sonnet10:     That beauty still may live in thine or thee.
poems/Shakespeare/sonnet11:Herein lives wisdom, beauty, and increase;
poems/Shakespeare/sonnet17:If I could write the beauty of your eyes,
poems/Shakespeare/sonnet2:And dig deep trenches in thy beauty's field,
poems/Shakespeare/sonnet2:Then being ask'd, where all thy beauty lies,
poems/Shakespeare/sonnet2:How much more praise deserv'd thy beauty's use,
poems/Shakespeare/sonnet2:Proving his beauty by succession thine.
poems/Shakespeare/sonnet4:Upon thyself thy beauty's legacy?
poems/Shakespeare/sonnet4:      Thy unus'd beauty must be tomb'd with thee,
poems/Shakespeare/sonnet5:Beauty's effect with beauty were bereft,
poems/Shakespeare/sonnet7:Yet mortal looks adore his beauty still,
poems/Shakespeare/sonnet9:But beauty's waste hath in the world an end,
poems/Yeats/old:And loved your beauty with love false or true,
/home/cis90/rodduk $
```

Fix it and test again

rodduk90@oslab:~/bin

```
#!/bin/bash
#
# menu: A simple menu template
#
while true
do
        clear
        echo -n "
                CIS 90 Final Project
        1) Hacking with the grep command
        2) Task 2
        3) Task 3
        4) Task 4
        5) Task 5
        6) Exit

        Enter Your Choice: "
        read RESPONSE
        case $RESPONSE in
          1)    # Commands for Task 1
                grep beauty /home/cis90/rodduk/poems/*/*
                ;;
          2)    # Commands for Task 2
                ;;
          3)    # Commands for Task 3
                ;;
          4)    # Commands for Task 4
                ;;
          5)    # Commands for Task 5
                ;;
          6)    exit 0
                ;;
          *)    echo "Please enter a number between 1 and 6"
                ;;
        esac
        echo -n "Hit the Enter key to return to menu "
        read dummy
done
~
~
"myscript" 38L, 612C written                                21,33-47      All
```

Fix task in 1st window by using an absolute pathname then save with ***<esc>:w***

Re-run ***myscript*** *in the 2nd second window and test your change. To do this quickly hit* ***Ctrl-C*** *then <****up arrow****> key.*

rodduk90@oslab:~/bin

```
                CIS 90 Final Project
        1) Hacking with the grep command
        2) Task 2
        3) Task 3
        4) Task 4
        5) Task 5
        6) Exit

        Enter Your Choice: 1
/home/cis90/rodduk/poems/Shakespeare/sonnet1:That thereby beauty's rose might ne
ver die,
/home/cis90/rodduk/poems/Shakespeare/sonnet10:  That beauty still may live in th
ine or thee.
/home/cis90/rodduk/poems/Shakespeare/sonnet11:Herein lives wisdom, beauty, and i
ncrease;
/home/cis90/rodduk/poems/Shakespeare/sonnet17:If I could write the beauty of you
r eyes,
/home/cis90/rodduk/poems/Shakespeare/sonnet2:And dig deep trenches in thy beauty
's field,
/home/cis90/rodduk/poems/Shakespeare/sonnet2:Then being ask'd, where all thy bea
uty lies,
/home/cis90/rodduk/poems/Shakespeare/sonnet2:How much more praise deserv'd thy b
eauty's use,
/home/cis90/rodduk/poems/Shakespeare/sonnet2:Proving his beauty by succession th
ine.
/home/cis90/rodduk/poems/Shakespeare/sonnet4:Upon thyself thy beauty's legacy?
/home/cis90/rodduk/poems/Shakespeare/sonnet4:   Thy unus'd beauty must be tomb'd
 with thee,
/home/cis90/rodduk/poems/Shakespeare/sonnet5:Beauty's effect with beauty were be
reft,
/home/cis90/rodduk/poems/Shakespeare/sonnet7:Yet mortal looks adore his beauty s
till,
/home/cis90/rodduk/poems/Shakespeare/sonnet9:But beauty's waste hath in the worl
d an end,
/home/cis90/rodduk/poems/Yeats/old:And loved your beauty with love false or true
,
Hit the Enter key to return to menu
```

Fix worked!

```
poems/Shakespeare/sonnet5:Beauty's effect with beauty were bereft,
poems/Shakespeare/sonnet7:Yet mortal looks adore his beauty still,
poems/Shakespeare/sonnet9:But beauty's waste hath in the world an end,
poems/Yeats/old:And loved your beauty with love false or true,
/home/cis90/rodduk $
```

Add some interaction

Let's add some interaction

```
1)  # Commands for Task 1
    echo "Are you ready to search for beauty in the poems?"
    read response
    grep beauty /home/cis90/rodduk/poems/*/*
    ;;
```

rodduk90@oslab:~/bin

```
#!/bin/bash
#
# menu: A simple menu template
#
while true
do
        clear
        echo -n "
                CIS 90 Fin
        1) Hacking with th
        2) Task 2
        3) Task 3
        4) Task 4
        5) Task 5
        6) Exit

        Enter Your Choice:
        read RESPONSE
        case $RESPONSE in
          1)    # Commands for Task 1
                echo "Are you ready to search for beauty i
                read dummy
                grep beauty /home/cis90/rodduk/poems/*/*
                ;;
          2)    # Commands for Task 2
                ;;
          3)    # Commands for Task 3
                ;;
          4)    # Commands for Task 4
                ;;
          5)    # Commands for Task 5
                ;;
          6)    exit 0
                ;;
          *)    echo "Please enter a number between 1 and
                ;;
        esac
        echo -n "Hit the Enter key to return to menu "
        read dummy
done
"myscript" 40L, 711C written
```

rodduk90@oslab:~/bin

```
                CIS 90 Final Project
        1) Hacking with the grep command
        2) Task 2
        3) Task 3
        4) Task 5
        5) Task 5
        6) Exit

        Enter Your Choice: 1
Are you ready to search for beauty in the poems?

/home/cis90/rodduk/poems/Shakespeare/sonnet1:That thereby beauty's rose might never die,
/home/cis90/rodduk/poems/Shakespeare/sonnet10:  That beauty still may live in thine or thee.
/home/cis90/rodduk/poems/Shakespeare/sonnet11:Herein lives wisdom, beauty, and increase;
/home/cis90/rodduk/poems/Shakespeare/sonnet17:If I could write the beauty of your eyes,
/home/cis90/rodduk/poems/Shakespeare/sonnet2:And dig deep trenches in thy beauty's field,
/home/cis90/rodduk/poems/Shakespeare/sonnet2:Then being ask'd, where all thy beauty lies,
/home/cis90/rodduk/poems/Shakespeare/sonnet2:How much more praise deserv'd thy beauty's use,
/home/cis90/rodduk/poems/Shakespeare/sonnet2:Proving his beauty by succession thine.
/home/cis90/rodduk/poems/Shakespeare/sonnet4:Upon thyself thy beauty's legacy?
/home/cis90/rodduk/poems/Shakespeare/sonnet4:   Thy unus'd beauty must be tomb'd with thee,
/home/cis90/rodduk/poems/Shakespeare/sonnet5:Beauty's effect with beauty were bereft,
/home/cis90/rodduk/poems/Shakespeare/sonnet7:Yet mortal looks adore his beauty still,
/home/cis90/rodduk/poems/Shakespeare/sonnet9:But beauty's waste hath in the world an end,
/home/cis90/rodduk/poems/Yeats/old:And loved your beauty with love false or true,
Hit the Enter key to return to menu
```

And it works!

Try a new option on the command

rodduk90@oslab:~/bin

```
#!/bin/bash
#
# menu: A simple menu template
#
while true
do
        clear
        echo -n "
                CIS 9
        1) Hacking wit
        2) Task 2
        3) Task 3
        4) Task 4
        5) Task 5
        6) Exit

        Enter Your Cho
        read RESPONSE
        case $RESPONSE in
          1)    # Commands for Task 1
                echo "Are you ready to search
                read dummy
                grep -h beauty /home/cis90/ro
                ;;
          2)    # Commands for Task 2
                ;;
          3)    # Commands for Task 3
                ;;
          4)    # Commands for Task 4
                ;;
          5)    # Commands for Task 5
                ;;
          6)    exit 0
                ;;
          *)    echo "Please enter a number b
                ;;
        esac
        echo -n "Hit the Enter key to return
        read dummy
done
"myscript" 40L, 714C written
```

Let's try the -h option and not print the leading file names

```
1)  # Commands for Task 1
    echo "Are you ready to search for beauty in the poems?"
    read response
    grep -h beauty /home/cis90/rodduk/poems/*/*
    ;;
```

rodduk90@oslab:~/bin

```
                CIS 90 Final Project
        1) Hacking with the grep command
        2) Task 2
        3) Task 3
        4) Task 4
        5) Task 5
        6) Exit

        Enter Your Choice: 1
Are you ready to search for beauty in the poems?
1
That thereby beauty's rose might never die,
        That beauty still may live in thine or thee.
Herein lives wisdom, beauty, and increase;
If I could write the beauty of your eyes,
And dig deep trenches in thy beauty's field,
Then being ask'd, where all thy beauty lies,
How much more praise deserv'd thy beauty's use,
Proving his beauty by succession thine.
Upon thyself thy beauty's legacy?
        Thy unus'd beauty must be tomb'd with thee,
Beauty's effect with beauty were bereft,
Yet mortal looks adore his beauty still,
But beauty's waste hath in the world an end,
And loved your beauty with love false or true,
Hit the Enter key to return to menu
```

And it works!

Add a new feature

Let's count the strings found now

```
1)  # Commands for Task 1
    echo "Are you ready to search for beauty in the poems?"
    read response
    grep -h beauty /home/cis90/rodduk/poems/*/*
    echo "Ready to count them?"
    read response
    grep -h beauty /home/cis90/rodduk/poems/*/* | wc -l
    ;;
```

```
#!/bin/bash
#
# menu: A s
#
while true
do
        cle
        ech

        1)
        2)
        3)
        4)
        5)
        6)

        Ent
        rea
        case $RESPONSE in
          1)    # Commands for Task 1
                echo "Are you ready to search for beauty in the poems?"
                read dummy
                grep -h beauty /home/cis90/rodduk/poems/*/*
                echo "Ready to count them?"
                read dummy
                grep -h beauty /home/cis90/rodduk/poems/*/* | wc -l
                ;;
          2)    # Commands for Task 2
                ;;
          3)    # Commands for Task 3
                ;;
          4)    # Commands for Task 4
                ;;
          5)    # Commands for Task 5
                ;;
          6)    exit 0
                ;;
          *)    echo "Please enter a number between 1 and 6"
                ;;
        esac
"myscript" 43L, 839C written                                  26,53-67
```

```
rodduk90@oslab:~/bin

                CIS 90 Final Project
        1) Hacking with the grep command
        2) Task 2
        3) Task 3
        4) Task 4
        5) Task 5
        6) Exit

        Enter Your Choice: 1
Are you ready to search for beauty in the poems?

That thereby beauty's rose might never die,
        That beauty still may live in thine or thee.
Herein lives wisdom, beauty, and increase;
If I could write the beauty of your eyes,
And dig deep trenches in thy beauty's field,
Then being ask'd, where all thy beauty lies,
How much more praise deserv'd thy beauty's use,
Proving his beauty by succession thine.
Upon thyself thy beauty's legacy?
        Thy unus'd beauty must be tomb'd with thee,
Beauty's effect with beauty were bereft,
Yet mortal looks adore his beauty still,
But beauty's waste hath in the world an end,
And loved your beauty with love false or true
Ready to count them?

14
Hit the Enter key to return to menu
```

Test it and it works!

How many points so far?

```
1)  # Commands for Task 1
    echo "Are you ready to search for beauty in the poems?"
    read response
    grep -h beauty /home/cis90/rodduk/poems/*/*
    echo "Ready to count them?"
    read response
    grep -h beauty /home/cis90/rodduk/poems/*/* | wc -l
    ;;
```

We haven't met the requirements yet for a task but we do have 12 points from using 3 of the constructs on the list.

Implementing all five tasks (6 points each):
- Requirements for each task:

NO – Minimum of 12 "original" lines of bash script
NO – Has one or more non-generic comments to explain what it is doing
✓ – Has user interaction

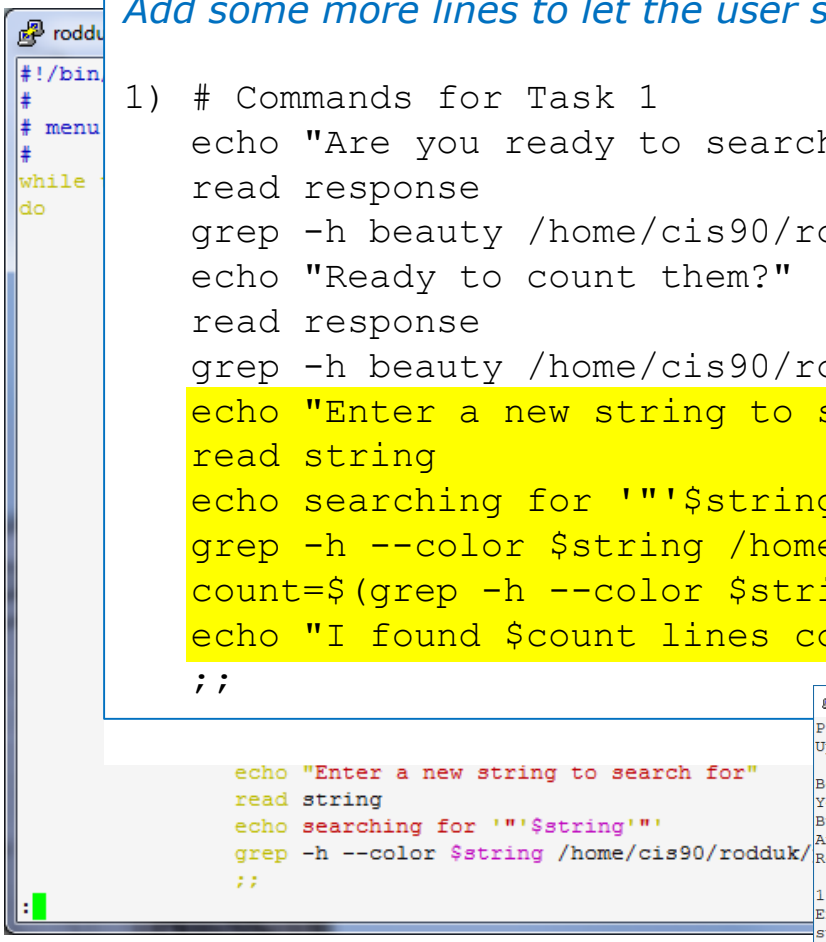
At least six bash constructs from this list:
- Redirecting stdin (4 points)
- Redirecting stdout (4 points)
- Redirecting stderr (4 points)
- Use of permissions (4 points)
- ✓ Use of filename expansion characters (4 points)
- ✓ Use of absolute path (4 points)
- Use of relative path (4 points)
- Use of a PID (4 points)
- Use of inodes (4 points)
- Use of links (4 points)
- Use of color (4 points)
- Use of scheduling (4 points)
- Use of a GID or group (4 points)
- Use of a UID or user (4 points)
- Use of a /dev/tty device (4 points)
- Use of a signal (4 points)
- ✓ Use of piping (4 points)
- Use of an environment variable (4 points)
- Use of /bin/mail (4 points)
- Use of a conditional (4 points)
- Use of $(*command*)

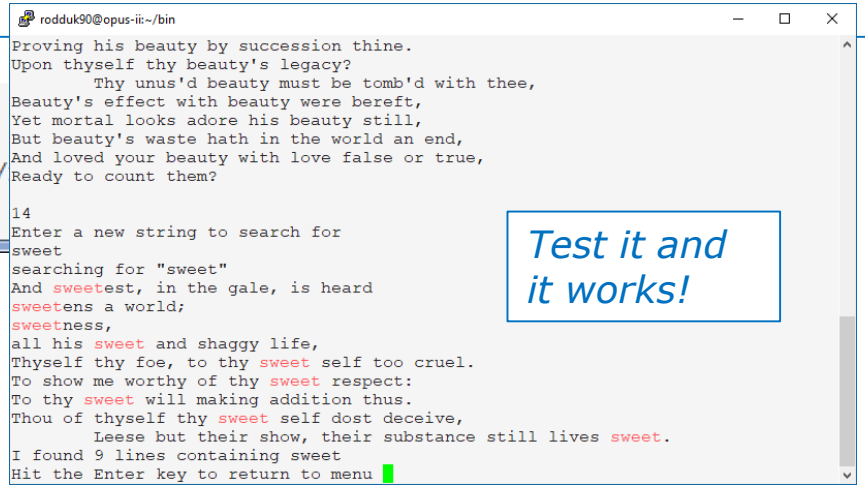
The maximum for this section is 24 points.

Let's add some more code

Add some more lines to let the user specify the string to search for

```
1)  # Commands for Task 1
    echo "Are you ready to search for beauty in the poems?"
    read response
    grep -h beauty /home/cis90/rodduk/poems/*/*
    echo "Ready to count them?"
    read response
    grep -h beauty /home/cis90/rodduk/poems/*/* | wc -l
    echo "Enter a new string to search for"
    read string
    echo searching for '"'$string'"'
    grep -h --color $string /home/cis90/rodduk/poems/*/*
    count=$(grep -h --color $string /home/cis90/rodduk/poems/*/* | wc -l)
    echo "I found $count lines containing $string"   ;;
    ;;
```





Test it and it works!

Check the score again

```
1)  # Commands for Task 1
    echo "Are you ready to search for beauty in the poems?"
    read response
    grep -h beauty /home/cis90/rodduk/poems/*/*
    echo "Ready to count them?"
    read response
    grep -h beauty /home/cis90/rodduk/poems/*/* | wc -l
    echo "Enter a new string to search for"
    read string
    echo searching for '"'$string'"'
    grep -h --color $string /home/cis90
    count=$(grep -h --color $string /ho                                    c -l)
    echo "I found $count lines containi
    ;;
```

Implementing all five tasks (6 points each):
- Requirements for each task:
  - ✓ – Minimum of 12 "original" lines of bash script
  - NO – Has one or more non-generic comments to explain what it is doing
  - ✓ – Has user interaction

At least six bash constructs from this list:
- Redirecting stdin (4 points)
- Redirecting stdout (4 points)
- Redirecting stderr (4 points)
- Use of permissions (4 points)
- ✓ Use of filename expansion characters (4 points)
- ✓ Use of absolute path (4 points)
- Use of relative path (4 points)
- Use of a PID (4 points)
- Use of inodes (4 points)
- Use of links (4 points)
- Use of color (4 points)
- Use of scheduling (4 points)
- Use of a GID or group (4 points)
- Use of a UID or user (4 points)
- Use of a /dev/tty device (4 points)
- Use of a signal (4 points)
- ✓ Use of piping (4 points)
- Use of an environment variable (4 points)
- Use of /bin/mail (4 points)
- Use of a conditional (4 points)
- ✓ Use of $(*command*)

The maximum for this section is 24 points.

We have at least 12 lines of code now and we have 16 points from using 4 constructs on the list.

Lets add some non-generic comments

Use non-generic comments to help others understand what you are doing

```
1)      # Task 1 - grep command explored
        # Simple grep for "beauty" with -h option to suppress filenames
        echo "Are you ready to search for beauty in the poems?"
        read response
        grep -h beauty /home/cis90/rodduk/poems/*/*
        # This time count the matches
        echo "Ready to count them?"
        read response
        grep -h beauty /home/cis90/rodduk/poems/
        # Let user select search string
        echo "Enter a new string to search for"
        read string
        echo searching for '"'$string'"'
        grep -h --color $string /home/cis90/rodd
        count=$(grep -h --color $string /home/ci
        echo "I found $count lines containing $s
        ;;
```

Implementing all five tasks (6 points each):
- Requirements for each task:
  - ✓ Minimum of 12 "original" lines of bash script
  - ✓ Has one or more non-generic comments to explain what it is doing
  - ✓ Has user interaction

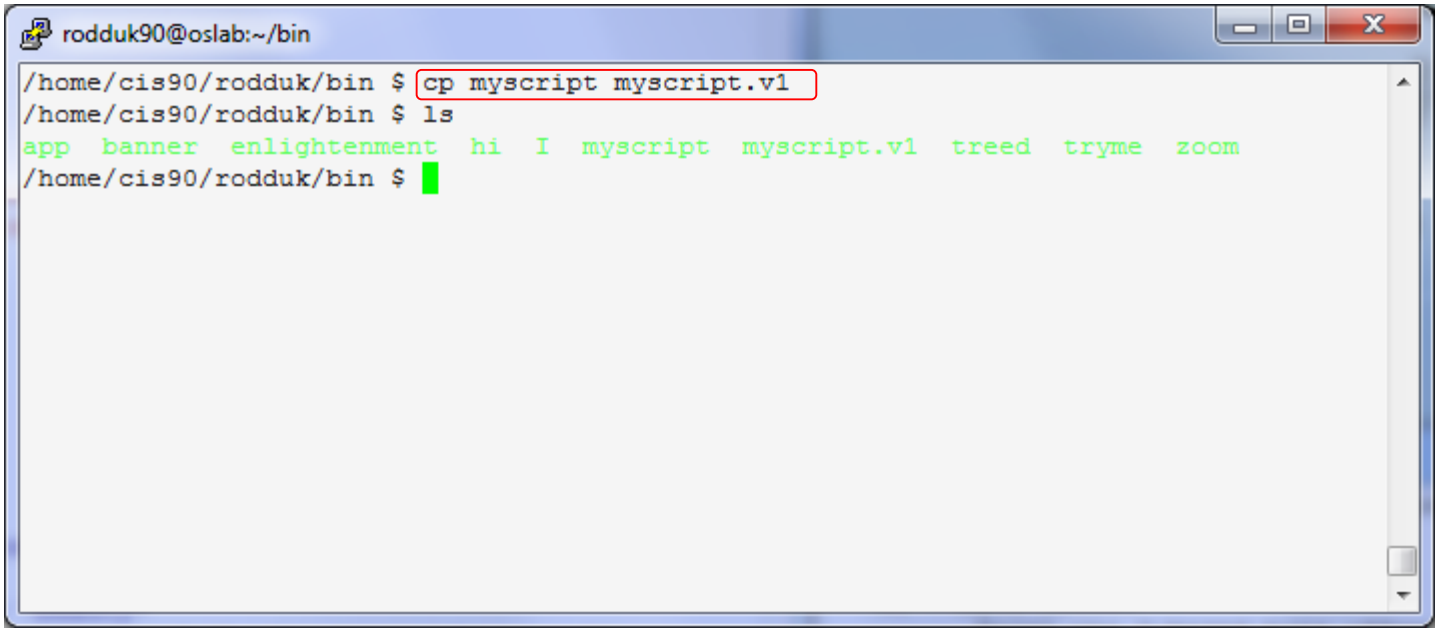
At least six bash constructs from this list:
- Redirecting stdin (4 points)
- Redirecting stdout (4 points)
- Redirecting stderr (4 points)
- Use of permissions (4 points)
- ✓ Use of filename expansion characters (4 points)
- ✓ Use of absolute path (4 points)
- Use of relative path (4 points)
- Use of a PID (4 points)
- Use of inodes (4 points)
- Use of links (4 points)
- Use of color (4 points)
- Use of scheduling (4 points)
- Use of a GID or group (4 points)
- Use of a UID or user (4 points)
- Use of a /dev/tty device (4 points)
- Use of a signal (4 points)
- ✓ Use of piping (4 points)
- Use of an environment variable (4 points)
- Use of /bin/mail (4 points)
- Use of a conditional (4 points)
- ✓ Use of $(*command*)

The maximum for this section is 24 points.

Yay ... we have one task finished and four constructs implemented for 22 points!

Backup your work!

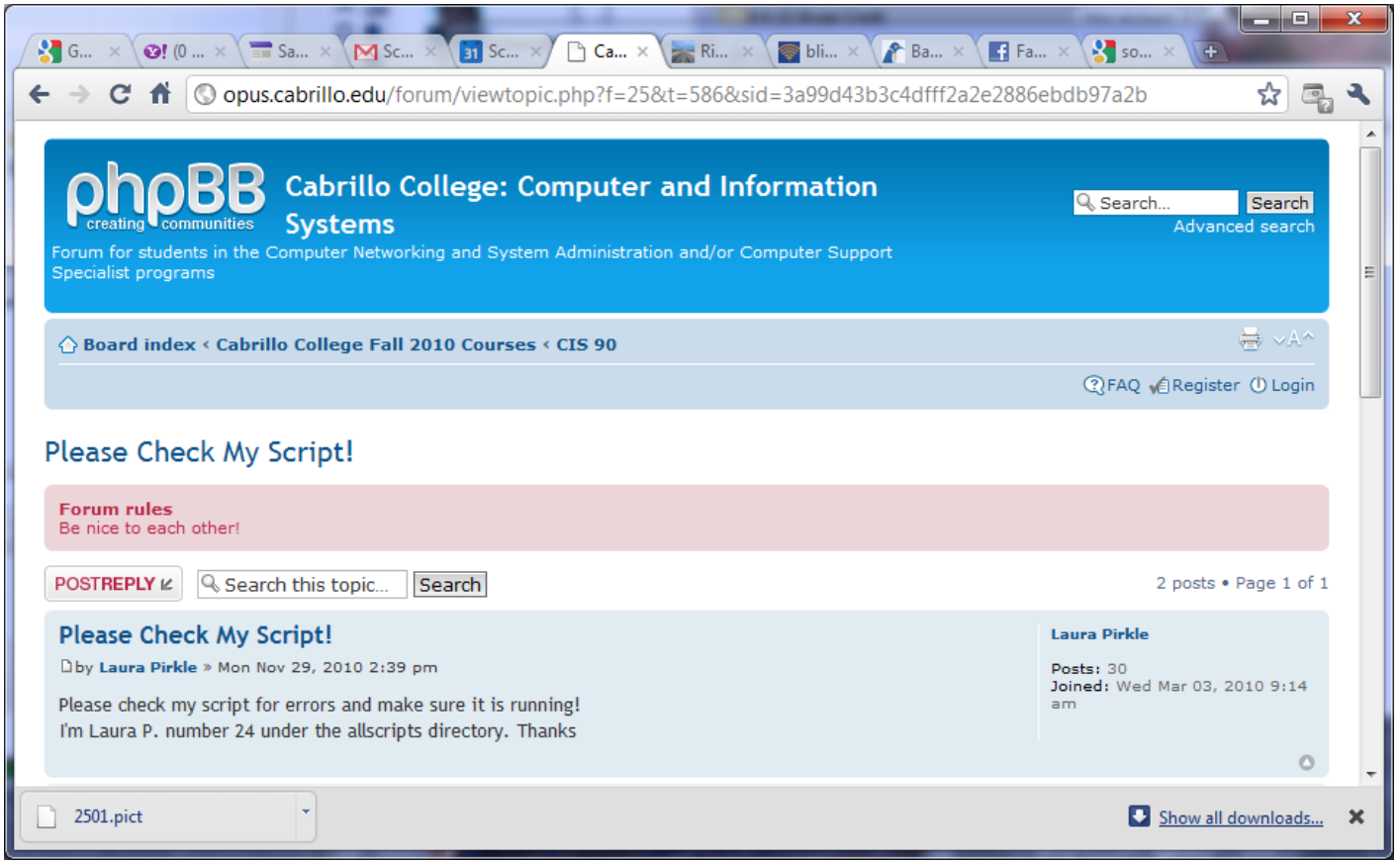
cp myscript myscript.v1 *after first day of work*



cp myscript myscript.v2 *after second day of work*
cp myscript myscript.v3 *and so on …*
cp myscript myscript.v4

Always be able to revert back to an earlier version in case you clobber the current one!

Have others test your script



The ask others on the forum to check your script and give you feedback

Plan extra time for:

- Figuring our how to do what you really want to do!
- Removing syntax errors
- Removing logic errors
- Posting script code on the forum and asking others to view it and suggest how to fix it
- Sleeping on it

Don't wait till the last minute to start your project!

Final Project forum tips

Use the forum effectively to get help

Not so good …

Preview:
Help!

My script is getting weird error

- Homer

Neither code nor output is shown which makes it very hard for someone else to understand what you are trying to do.

Use the forum effectively to get help

Better … but requires viewer to log into Opus-II and you may have modified the script since posting

Preview:

Help!

My script is getting weird error

My script is here:
/home/cis90/milhom/bin/myscript

And this is the error:

CODE: SELECT ALL

```
/home/cis90/simben/bin $ ./script99
simben90
-rwxr-x--- 1 simben90 cis90 10489 Apr 30 07:33 /home/cis90/simben/bin/myscript
./script99: line 8: unexpected EOF while looking for matching `"'
./script99: line 16: syntax error: unexpected end of file
/home/cis90/simben/bin $
```

- Homer

This post provides the location of the script and the error message which enables others to help you find and fix the problem

Use the forum effectively to get scripting help

B | i | u | Quote | Code | List | List= | [*] | Img | URL | Flash | Normal | Font colour

Preview:

Help!

My script is getting weird error

This is the script:

CODE: SELECT ALL

```
#!/bin/bash
# Test script
#
echo $LOGNAME
dir=/home/cis90/simben
ls -l $dir/bin/myscript
if [ -f "$dir/bin/myscript ]; then
  echo you have a myscript file in the bin directory
else
  echo there is no myscript file in your bin directory!]
fi
exit
```

And this is the error:

CODE: SELECT ALL

```
/home/cis90/simben/bin $ ./script99
simben90
-rwxr-x--- 1 simben90 cis90 10489 Apr 30 07:33 /home/cis90/simben/bin/myscript
./script99: line 8: unexpected EOF while looking for matching `"'
./script99: line 16: syntax error: unexpected end of file
/home/cis90/simben/bin $
```

- Homer

Best ...

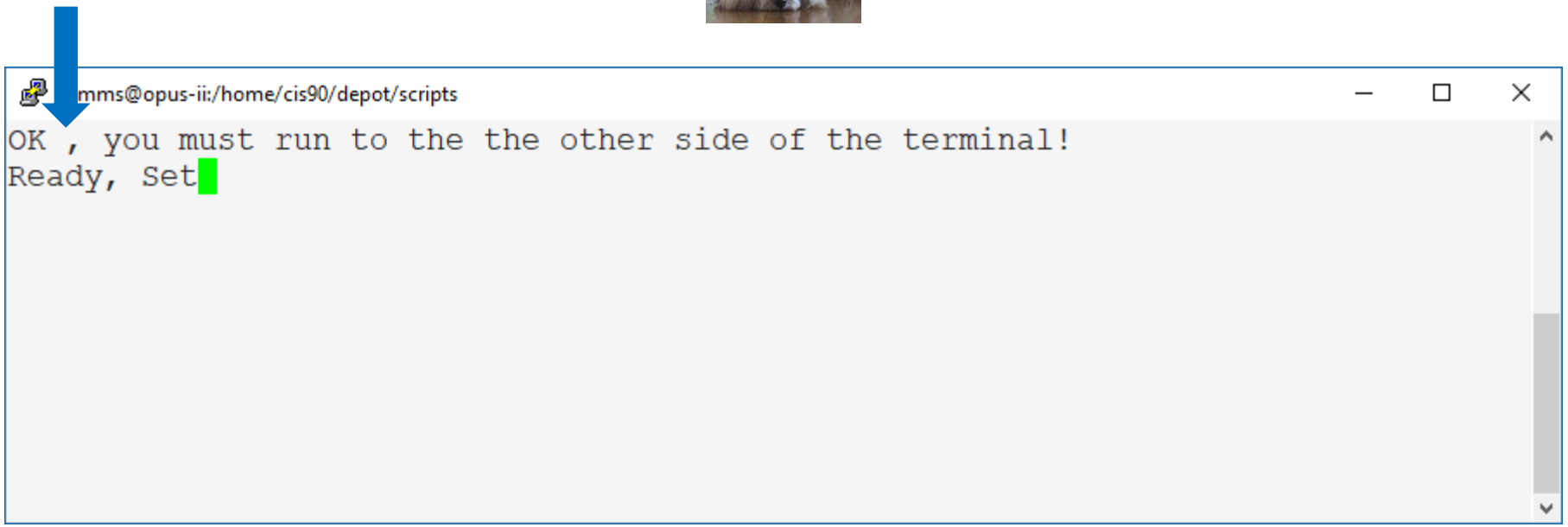
This post shows both the code and the output using code tags which makes it a lot easier for others to understand what you are doing and offer help.

Debugging Tips

Tracing your code with echos and pauses

Crud! The first name is not showing up.

Let's look at your code.



See: /home/cis90/depot/scripts/broken

Tracing your code with echos and pauses

```
#!/bin/bash
count=60
gid=$(grep "^cis90:" /etc/group | cut -f3 -d":")
numStudents=$(grep ":$gid:" /etc/passwd | grep -v tbd | grep -v "^cis90:" |
wc -l)
studentNum=$((RANDOM%$numStudents))
userRecord=$(grep ":$gid:" /etc/passwd | grep -v tbd | grep -v "^cis90:" |
head -n $studentNum | tail -n1)
first=$(echo $userRecord | cut -f5 -d ":" | cut -f1 -d " ")
clear
echo "OK $first, you must run to the the other side of the terminal!"
echo -n "Ready"; sleep 2; echo -n ", Set"; sleep 2; banner GO; sleep 1
for (( i=1; i<=$count; i++ )); do
  clear
  row=2; col=$i; foregroundColor=1
  tput setf $foregroundColor
  tput cup $row $col
  echo -n "$first"
  sleep .05
done
tput reset
banner $first made it!
exit
```

It's broken, first should get set to a random first name.

Try some tracing.

See: /home/cis90/depot/scripts/broken

Tracing your code with echos and pauses

What's tracing?

Add a bunch of echos and pauses to see what is going on. Like this.

```
gid=$(grep "^cis90:" /etc/group | cut -f3 -d":")
echo TRACE gid=$gid

numStudents=$(grep ":$gid:" /etc/passwd | grep -v tbd | grep -v "^cis90:" | wc -l)
echo TRACE numStudents=$numStudents

studentNum=$((RANDOM%$numstudents))
echo TRACE studentNum=$studentNum

userRecord=$(grep ":$gid:" /etc/passwd | grep -v tbd | grep -v "^cis90:" | head -n
$studentNum | tail -n1)
echo TRACE userRecord=$userRecord
read -p "Press Enter to continue ..." reply
first=$(echo $userRecord | cut -f5 -d ":" | cut -f1 -d " ")
```

Tracing your code with echos and pauses

What the heck! An error on line 7 and studentNum is not getting set!

Let's look at the code and see why.

```
milhom90@opus-ii:~/bin
/home/cis90/milhom/bin $ ./broken
TRACE gid=1090
TRACE numStudents=28
./broken: line 7: RANDOM%: syntax error: operand expected (error token is "%")
TRACE studentNum=
head: option requires an argument -- 'n'
Try 'head --help' for more information.
TRACE userRecord=
Press Enter to continue ...
```

Tracing your code with echos and pauses

Oops! The "S" in the numStudents variable was not capitalized.

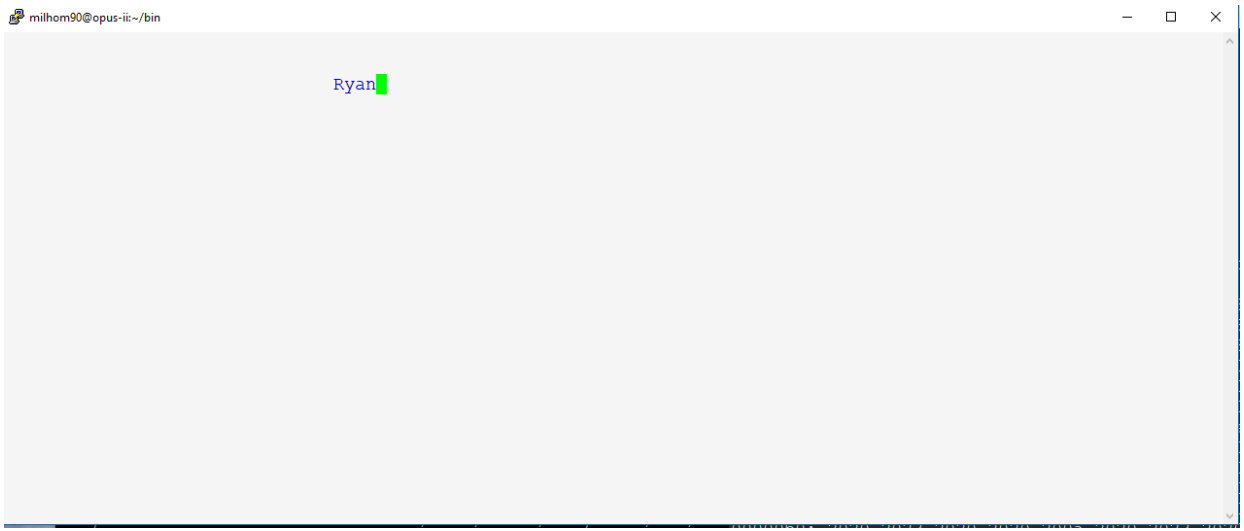
Fix it and let's try again.

```
gid=$(grep "^cis90:" /etc/group | cut -f3 -d":")
echo TRACE gid=$gid
numStudents=$(grep ":$gid:" /etc/passwd | grep -v tbd | grep -v "^cis90:" | wc -l)
echo TRACE numStudents=$numStudents
studentNum=$((RANDOM%$numstudents))
echo TRACE studentNum=$studentNum
userRecord=$(grep ":$gid:" /etc/passwd | grep -v tbd | grep -v "^cis90:" | head -n
$studentNum)
echo TRACE userRecord=$userRecord
read -p "Press Enter to continue ..." reply
first=$(echo $userRecord | cut -f5 -d ":" | cut -f1 -d " ")
```

Tracing your code with echos and pauses

That's better. I see userRecord was correctly assigned a random line from /etc/passwd now

Cool! I see Ryan's name moving across the screen now!



Tracing your code with echos and pauses

It works! I'm going to take out those tracing statements now

Just search for TRACE in vi to light them up.

Shell Scripting 101

Shell Scripts

- In its simplest form a shell script can just be a list of commands in a file .
- Read "r" and execute "x" permissions must be enabled on the script file.
- The script must be on your path or you must use an absolute or relative pathname to run it.
- Putting #!/bin/bash on line 1 specifies which program should be used to execute the script. The default, if not specified, is /bin/bash. Note this enables vi to use color syntax.
- Putting the exit command at the end triggers a system call to the kernel to terminate the process and release all resources. Note a numerical status can be specified as an argument (e.g. exit 20) which will be communicated back to the parent process.

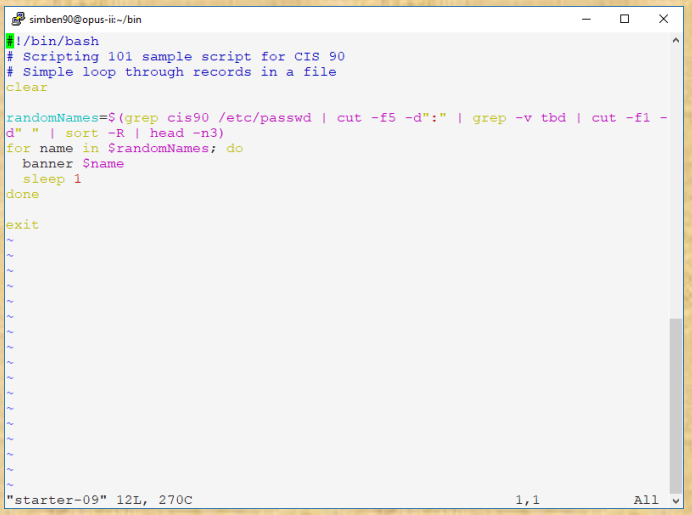
Example Scripts

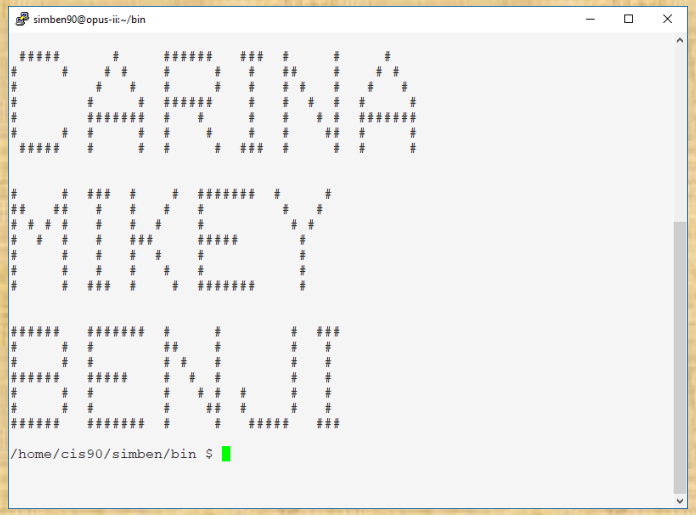
starter-00:# Description: Hello World
starter-01:# Description: Just a bunch of commands
starter-02:# Description: Sh-bang, comments and exit
starter-03:# Description: Using variables and $(command) construct
starter-04:# Description: Clearing and pausing
starter-05:# Description: Reading user input
starter-06:# Description: Arguments and exit codes
starter-07:# Description: Using color
starter-08:# Description: Simple loop through list
starter-09:# Description: Simple loop through records in a file
starter-10:# Description: Simple loop through range of integers
starter-11:# Description: Simple loop for counting and parsing words in random poem lines
starter-12:# Description: Demonstrate simple if statement
starter-13:# Description: Scrape a web page for data
starter-14:# Description: Remotely control via ssh a Hue smart light
starter-15:# Description: Remotely turn off via ssh a Hue smart light
starter-16:# Description: Random numbers, terminal text placement and color
starter-17:# Description: Display a message on the STEM center LEDs

Scripting 101

It is helpful to use two terminal sessions for this module.

That way you can view the script using vi in one terminal and run the script in the other.





Hello World

```
/home/cis90/simben $ cd bin
/home/cis90/simben/bin $ cp ~/../depot/scripts/starter-00 .
/home/cis90/simben/bin $ vi starter-00
```

Use Esc **:wq** *to save file and quit vi*

```
/home/cis90/simben/bin $ chmod +x starter-00
/home/cis90/simben/bin $ starter-00
```

What was the output from your script?

Put your answer in the chat window

```
/home/cis90/simben/bin $ ./starter-00
Hello World
/home/cis90/simben/bin $
```

Just a bunch of commands

```
/home/cis90/simben $ cd bin
/home/cis90/simben/bin $ cp ~/../depot/scripts/starter-01 .
/home/cis90/simben/bin $ vi starter-01
```

simben90@opus-ii:~/bin

```
clear
echo "Sample script: Starter-01"
echo "My name is: Xxxxx"
echo "My distro is:  Cxxxxx Lxxxx N"
echo
echo "My favorite dog poem is:"
head -n10 ~/poems/Anon/nursery | tail -n4
echo
echo Xxxxx
echo CAN DO
banner It
~
"starter-01" 11L, 218C                                1,1           All
```

Modify:
"Xxxxx" to your first name (in two places)
and
"Cxxxxx Lxxxx N" to "Centos Linux 7"

Use Esc **:wq** *to save file and quit vi*

```
/home/cis90/simben/bin $ chmod +x starter-01
/home/cis90/simben/bin $ starter-01
```

What was the first line of output from the script's head command?

Put your answer in the chat window

```
/home/cis90/simben/bin $ ./starter-01
Sample script: Starter-01
My name is: Xxxxx
My distro is:  Cxxxxx Lxxxx N

My favorite dog poem is:
Hark! Hark!  The dogs do bark!
The beggars are coming to town.
Some in rags, some in tags,
and some in velvet gowns.

Xxxxx
CAN DO

###  #######
 #       #
 #       #
 #       #
 #       #
 #       #
###      #

/home/cis90/simben/bin $
```

Sh-bang, comments and exit

```
/home/cis90/simben $ cd bin
/home/cis90/simben/bin $ cp ~/../depot/scripts/starter-02 .
/home/cis90/simben/bin $ vi starter-02
```

rsimms@opus-ii:/home/cis90/depot/scripts

```
#!/bin/bash
# Scripting 101 sample script for CIS 90
# Description: Sh-bang, comments and exit
clear
echo "Sample script: Starter-02"
echo "My name is: Xxxxx"
echo "My distro is:  Cxxxxx Lxxxx N"
echo
echo "My favorite dog poem is:"
head -n10 ~/poems/Anon/nursery | tail -n4
echo
echo Xxxxx
echo CAN DO
banner It
exit
~
"starter-02" 15L, 318C                                5,1           All
```

Modify:
"Xxxxx" to your first name (in two places)
and
"Cxxxxx Lxxxx N" to "Centos Linux 7"

Use Esc **:wq** *to save file and quit vi*

```
/home/cis90/simben/bin $ chmod +x starter-02
/home/cis90/simben/bin $ starter-02
```

What does vi do when it sees #!/bin/bash as the first line of the file?

Put your answer in the chat window

```
/home/cis90/simben/bin $ ./starter-02
Sample script: Starter-02
My name is: Xxxxx
My distro is:  Cxxxxx Lxxxx N

My favorite dog poem is:
Hark! Hark!  The dogs do bark!
The beggars are coming to town.
Some in rags, some in tags,
and some in velvet gowns.

Xxxxx
CAN DO

###  #######
 #       #
 #       #
 #       #
 #       #
 #       #
###      #

/home/cis90/simben/bin $
```

Throwing in variables

```
/home/cis90/simben $ cd bin
/home/cis90/simben/bin $ cp ~/../depot/scripts/starter-03 .
/home/cis90/simben/bin $ vi starter-03
```

simben90@opus-ii:~/bin

```
#!/bin/bash
# Scripting 101 sample script for CIS 90
# Using variables and $(command) construct
clear
first=$(grep $LOGNAME /etc/passwd | cut -f5 -d":" | cut -f1 -d" ")
scriptName=$(basename $0)
distro=$(cat /etc/*-release | grep PRETTY | cut -f2 -d"=")
echo "Sample script: $scriptName"
echo "My name is: $first"
echo "My distro is: $distro"
echo
echo "My favorite dog poem is:"
head -n10 ~/poems/Anon/nursery | tail -n4
echo
echo $first
echo CAN DO
banner It
exit
~
"starter-03" 18L, 466C                                1,1           All
```

Notice the use of $(some command) construct to initialize variables to the output of the command.

The basename command extracts the last file on a pathname.

Use Esc **:wq** *to save file and quit vi*

```
/home/cis90/simben/bin $ chmod +x starter-03
/home/cis90/simben/bin $ starter-03
```

If you run Benji's script (~simben90/bin/starter-03) is your first name still correct?

Put your answer in the chat window

```
/home/cis90/simben/bin $ ./starter-03
Sample script: starter-03
My name is: Benji
My distro is: "CentOS Linux 7 (Core)"

My favorite dog poem is:
Hark! Hark!  The dogs do bark!
The beggars are coming to town.
Some in rags, some in tags,
and some in velvet gowns.

Benji
CAN DO

###  #######
 #       #
 #       #
 #       #
 #       #
 #       #
###      #

/home/cis90/simben/bin $
```

Clearing and Pausing

```
/home/cis90/simben/bin $ cd ~/bin
/home/cis90/simben/bin $ cp ~/../depot/scripts/starter-04 .
/home/cis90/simben/bin $ vi starter-04
```

simben90@opus-ii:~/bin

```
#!/bin/bash
# Scripting 101 sample script for CIS 90
# Clearing and pausing
clear
echo -n "And the top three finalists are (drum roll please) "
sleep .5; echo -n .; sleep .5; echo -n .; sleep .5; echo .; sleep 1
clear
banner $(grep cis90 /etc/passwd | cut -f5 -d":" | grep -v tbd | cut -f1 -d" " | sort -R | head -n1)
sleep 2
clear
banner $(grep cis90 /etc/passwd | cut -f5 -d":" | grep -v tbd | cut -f1 -d" " | sort -R | head -n1)
sleep 2
clear
banner $(grep cis90 /etc/passwd | cut -f5 -d":" | grep -v tbd | cut -f1 -d" " | sort -R | head -n1)
sleep 2
clear
exit
"starter-04" 17L, 565C                                        1,1           All
```

The -R option on sort does a random order sort.

Use Esc **:wq** *to save file and quit vi*

```
/home/cis90/simben/bin $ chmod +x starter-04
/home/cis90/simben/bin $ starter-04
```

What does "sort -R | head -n1" in the pipeline do?

Put your answer in the chat window

```
/cis90/simben/bin $ ./starter-04
And the top three finalists are (drum roll please) ...

######   #######  #     #  ###  #     #  ###   #####
#     #  #        ##   ##   #   ##    #   #   #     #
#     #  #        # # # #   #   # #   #   #   #
#     #  #        #  #  #   #   #  #  #   #   #
#     #  #        #     #   #   #   # #   #   #
#     #  #        #     #   #   #    ##   #   #     #
######   #######  #     #  ###  #     #  ###   #####

      #  #######  #     #     #
      #  #        ##    #    # #
      #  #        # #   #   #   #
      #  #        #  #  #  #     #
#     #  #        #   # #  #######
#     #  #        #    ##  #     #
 #####   #######  #     #  #     #

#######     #     ######   ###
     #     # #    #     #   #
    #     #   #   #     #   #
   #      #     #  ######    #
  #       #######  #   #     #
#         #     #  #    #    #
#######   #     #  #     #   ###
```

The names change randomly each time this script is run

Reading input from the user

```
/home/cis90/simben/bin $ cd ~/bin
/home/cis90/simben/bin $ cp ~/../depot/scripts/starter-05 .
/home/cis90/simben/bin $ vi starter-05
```

simben90@opus-ii:~/bin

```
#!/bin/bash
# Scripting 101 sample script for CIS 90
# Reading user input
clear
first=$(grep $LOGNAME /etc/passwd | cut -f5 -d":" | cut -f1 -d" ")
read -p "What is your favorite planet? " planet
echo "$first, your favorite planet is $planet"
exit
~
~
"starter-05" 8L, 248C                                 1,1           All
```

The -p option on read produces a prompt.

Use Esc **:wq** *to save file and quit vi*

```
/home/cis90/simben/bin $ chmod +x starter-05
/home/cis90/simben/bin $ starter-05
```

What does the -p option on the read command do?

Put your answer in the chat window

```
/home/cis90/simben/bin $ ./starter-05
What is your favorite planet? Mars
Benji, your favorite planet is Mars
/home/cis90/simben/bin $
```

Arguments and Exit Status

```
/home/cis90/simben/bin $ cd ~/bin
/home/cis90/simben/bin $ cp ~/../depot/scripts/starter-06 .
/home/cis90/simben/bin $ vi starter-06
```

simben90@opus-ii:~/bin

```
#!/bin/bash
# Scripting 101 sample script for CIS 90
# Arguments and exit codes
clear
echo The first argument passed by the shell to this script is: $1
echo The second argument passed by the shell to this script is: $2
echo
read -p "Enter an integer between 0 and 255 to return to the parent process: " code
exit $code
~
"starter-06" 9L, 319C                                         1,1           All
```

Use Esc **:wq** *to save file and quit vi*

```
/home/cis90/simben/bin $ chmod +x starter-06
/home/cis90/simben/bin $ ls [be]*
/home/cis90/simben/bin $ starter-06 [be]*
/home/cis90/simben/bin $ echo $?
```

Be sure to add some arguments when you run starter-06

Did the script ever see the "[be]*" typed by the user?

Put your answer in the chat window

```
/home/cis90/simben/bin $ ./starter-06 "Santa Cruz" Capitola
The first argument passed by the shell to this script is: Santa Cruz
The second argument passed by the shell to this script is: Capitola

Enter an integer between 0 and 255 to return to the parent process: 101
/home/cis90/simben/bin $ echo $?
101
```

Using color

```
/home/cis90/simben/bin $ cd ~/bin
/home/cis90/simben/bin $ cp ~/../depot/scripts/ starter-07 .
/home/cis90/simben/bin $ vi starter-07
```

simben90@opus-ii:~/bin

```
#!/bin/bash
# Scripting 101 sample script for CIS 90
# Using color
clear
off="\e[00m"
red="\e[00;31m"
white="\e[01;37m"
blue="\e[00;34m"
green="\e[00;32m"
echo -e Hi $LOGNAME, you look a little ${green}GREEN${off} today!
sleep 2
echo -e $blue
banner Stay COOL
echo -e $off
exit
"starter-07" 15L, 278C                                1,1           All
```

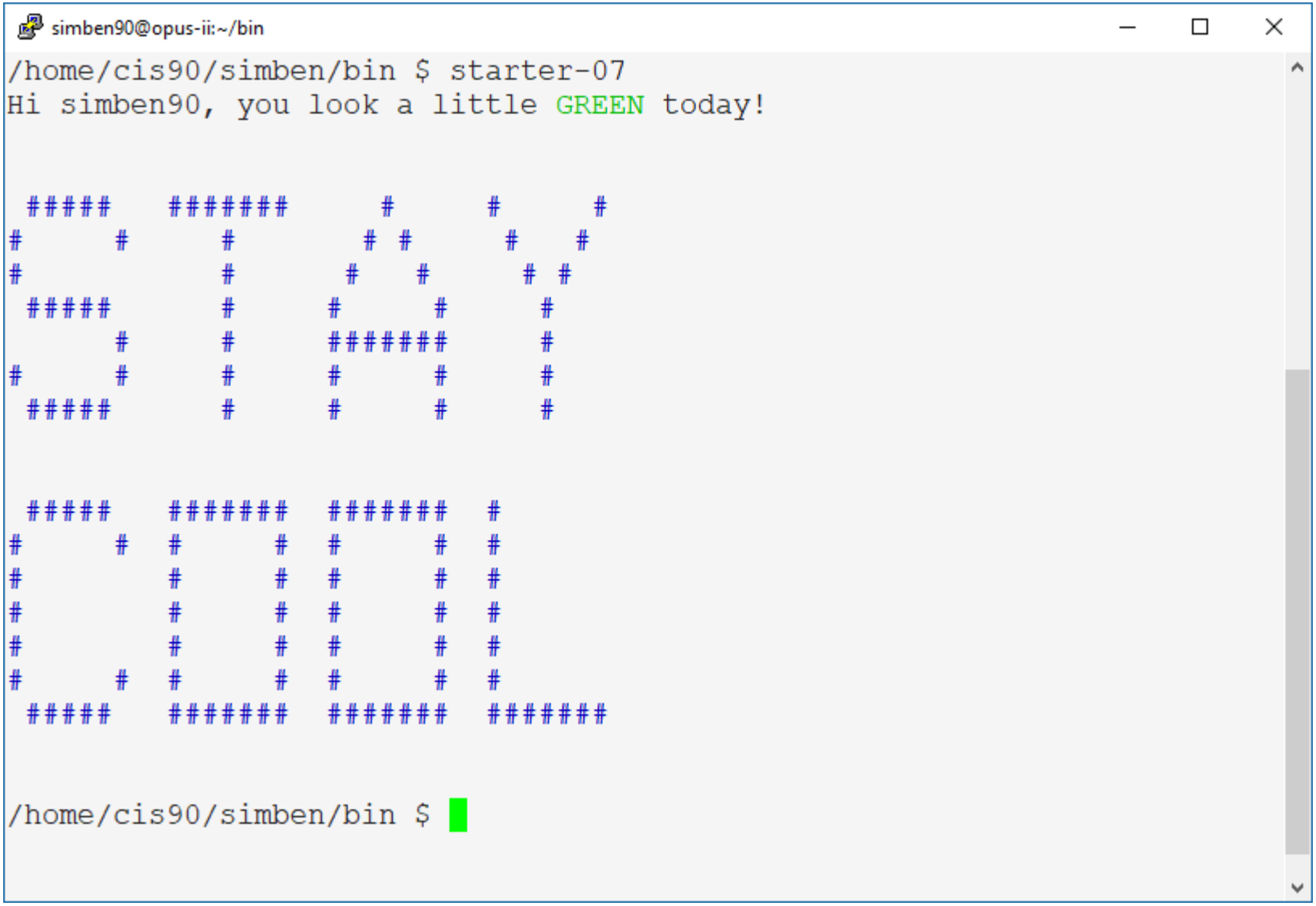
Don't forget the -e option on echo so the escape sequences will work.

Use Esc **:wq** *to save file and quit vi*

```
/home/cis90/simben/bin $ chmod +x starter-07
/home/cis90/simben/bin $ starter-07
```

What would happen if you deleted the "echo -e $off" line?

Put your answer in the chat window


simben90@opus-ii:~/bin
/home/cis90/simben/bin $ starter-07
Hi simben90, you look a little GREEN today!
/home/cis90/simben/bin $

Simple loop through list

```
/home/cis90/simben/bin $ cd ~/bin
/home/cis90/simben/bin $ cp ~/../depot/scripts/ starter-08 .
/home/cis90/simben/bin $ vi starter-08
```

simben90@opus-ii:~/bin

```
#!/bin/bash
# Scripting 101 sample script for CIS 90
# Simple loop through list
clear

for place in Rome Paris Milpitas; do
  echo "I want to go to $place"
  sleep 1
done

exit
~
~
"starter-08" 11L, 177C                    1,1           All
```

Indenting the commands that are looped will make your code more readable by others.

Use Esc **:wq** *to save file and quit vi*

```
/home/cis90/simben/bin $ chmod +x starter-08
/home/cis90/simben/bin $ starter-08
```

Try and add a new place after Milpitas. Did it work?

Put your answer in the chat window

```
/home/cis90/simben/bin $ starter-08
I want to go to Rome
I want to go to Paris
I want to go to Milpitas
```

Simple loop through records in a file

```
/home/cis90/simben/bin $ cd ~/bin
/home/cis90/simben/bin $ cp ~/../depot/scripts/ starter-09 .
/home/cis90/simben/bin $ vi starter-09
```

simben90@opus-ii:~/bin

```
#!/bin/bash
# Scripting 101 sample script for CIS 90
# Simple loop through records in a file
clear

randomNames=$(grep cis90 /etc/passwd | cut -f5 -d":" | grep -v tbd | cut -f1 -d" " | sort -R | head -n3)
for name in $randomNames; do
  banner $name
  sleep 1
done

exit
~
"starter-09" 12L, 270C                                          1,1           All
```

Use Esc **:wq** *to save file and quit vi*

```
/home/cis90/simben/bin $ chmod +x starter-09
/home/cis90/simben/bin $ starter-09
```

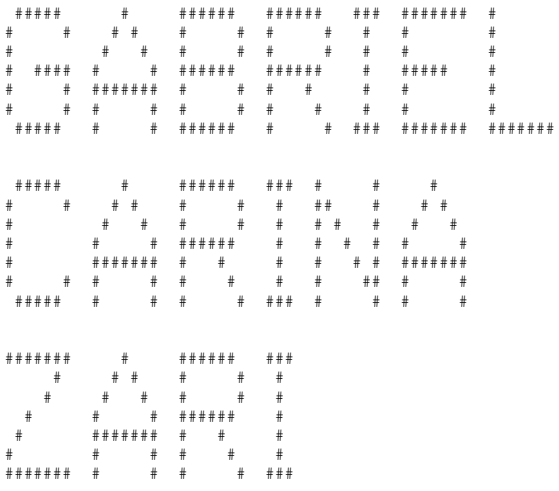
What was the final name your script output?

Put your answer in the chat window

```
/home/cis90/simben/bin $ starter-09
```

The names change randomly each time this script is run

```
/home/cis90/simben/bin $ starter-09
```



Simple loop through range of integers

```
/home/cis90/simben/bin $ cd ~/bin
/home/cis90/simben/bin $ cp ~/../depot/scripts/ starter-10 .
/home/cis90/simben/bin $ vi starter-10
```

simben90@opus-ii:~/bin

```
#!/bin/bash
# Scripting 101 sample script for CIS 90
# Simple loop through range of integers
clear

echo "The hosts in the 192.168.1.0/24 network:"
sleep 1
for ((i=1; i<=254; i++)); do
  echo IP address = 192.168.1.$i
done

exit
~
~
"starter-10" 12L, 229C                                    1,1           All
```

Use Esc **:wq** *to save file and quit vi*

```
/home/cis90/simben/bin $ chmod +x starter-10
/home/cis90/simben/bin $ starter-10
```

How would you pipe the output of this script to the more command?

Put your answer in the chat window

```
/home/cis90/simben/bin $ starter-10
The hosts in the 192.168.1.0/24 network:
IP address = 192.168.1.1
IP address = 192.168.1.2
IP address = 192.168.1.3
IP address = 192.168.1.4
IP address = 192.168.1.5
IP address = 192.168.1.6
IP address = 192.168.1.7
IP address = 192.168.1.8
IP address = 192.168.1.9
IP address = 192.168.1.10
IP address = 192.168.1.11
IP address = 192.168.1.12
IP address = 192.168.1.13
<snipped>
IP address = 192.168.1.249
IP address = 192.168.1.250
IP address = 192.168.1.251
IP address = 192.168.1.252
IP address = 192.168.1.253
IP address = 192.168.1.254
/home/cis90/simben/bin $
```

Simple loop for parsing a line and counting arguments

```
/home/cis90/simben/bin $ cd ~/bin
/home/cis90/simben/bin $ cp ~/../depot/scripts/ starter-11 .
/home/cis90/simben/bin $ vi starter-11
```

simben90@opus-ii:~/bin

```
#!/bin/bash
# Scripting 101 sample script for CIS 90
# Simple loop for counting and parsing words in random poem lines
clear

randomPoem=$(find ~/poems -type f | sort -R | head -n1)
echo Random poem = $randomPoem
randomLine=$(cat $randomPoem | sort -R | head -n1)
echo Random line in poem = $randomLine
echo Parsing the random line using set and shift:
count=1
set $randomLine
while [ "$1" != "" ]; do
  echo "  argument $count is $1"
  shift
  let count=$count+1
done

exit
~
"starter-11" 19L, 475C                                    1,1           All
```

For more loop examples google: bash loop examples

Use Esc **:wq** *to save file and quit vi*

```
/home/cis90/simben/bin $ chmod +x starter-11
/home/cis90/simben/bin $ starter-11
```

What does the let command do inside the loop?

Put your answer in the chat window

```
/home/cis90/simben/bin $ starter-11
Random poem = /home/cis90/simben/poems/Anon/ant
Random line in poem = 'till one who seemed the least
Parsing the random line using set and shift:
  argument 1 is 'till
  argument 2 is one
  argument 3 is who
  argument 4 is seemed
  argument 5 is the
  argument 6 is least
```

The poem and line in the poem changes randomly each time this script is run.

```
/home/cis90/simben/bin $ starter-11
Random poem = /home/cis90/simben/poems/Angelou/bird
Random line in poem = his bars of rage
Parsing the random line using set and shift:
  argument 1 is his
  argument 2 is bars
  argument 3 is of
  argument 4 is rage
/home/cis90/simben/bin $
```

Simple if-then-else conditional

```
/home/cis90/simben/bin $ cd ~/bin
/home/cis90/simben/bin $ cp ~/../depot/scripts/ starter-12 .
/home/cis90/simben/bin $ vi starter-12
```

simben90@opus-ii:~/bin

```
#!/bin/bash
# Scripting 101 sample script for CIS 90
# Demonstrate simple if statement
clear

off="\e[00m"
blue="\e[00;34m"
read -p "Guess what my favorite color is: " color
if [ "$color" = "blue" ]; then
   echo -e $blue
   echo "That's correct!"
   echo "You must have read my mind!"
   echo -e $off
else
   echo "Sorry!"
   echo "Please try again."
fi

exit

"starter-12" 20L, 362C                                1,1           All
```

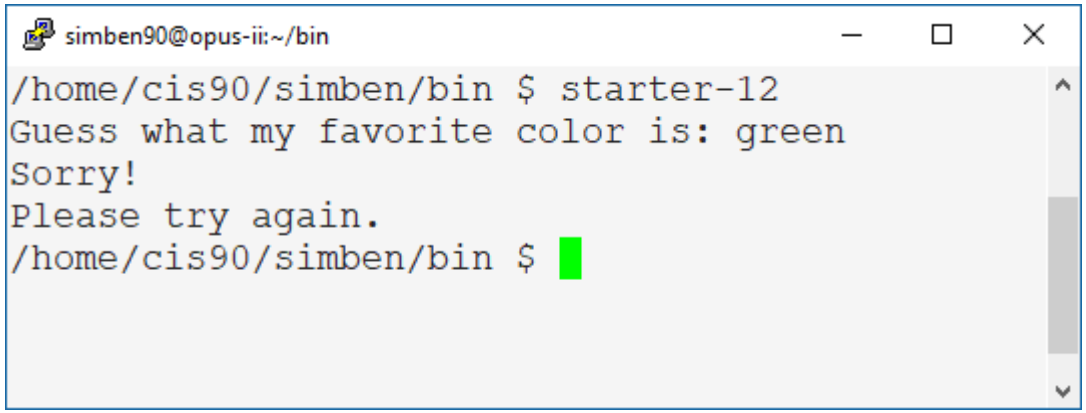
For more conditional examples google: bash if statement

Use Esc **:wq** *to save file and quit vi*

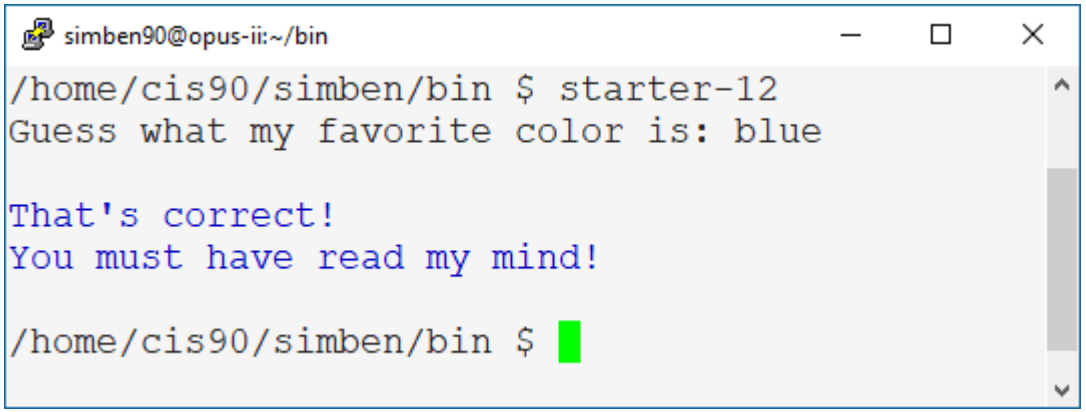
```
/home/cis90/simben/bin $ chmod +x starter-12
/home/cis90/simben/bin $ starter-12
```

What command closes the if statement?

Put your answer in the chat window



Blue is the correct answer!



Scraping data from a web page

```
/home/cis90/simben/bin $ cd ~/bin
/home/cis90/simben/bin $ cp ~/../depot/scripts/ starter-13 .
/home/cis90/simben/bin $ vi starter-13
```

simben90@opus-ii:~/bin

```
#!/bin/bash
# Scripting 101 sample script for CIS 90
# Scrape a web page for data
clear

url="http://aqicn.org/city/california/santa-cruz/santa-cruz-soquel-avenue/"
aqi=$(curl $url 2> /dev/null | sed 's/></>\n</g' | grep aqiwgtvalue | grep -o ">.*<" | tr -d "><")
echo "Current AQI (Air Quality Index)"
echo "================================"
banner " $aqi"
echo "
Good (0-50)
Moderate (51-100)
Unhealthy for Sensitive Groups (101-150)
Unhealthy (151-200)
Very Unhealthy (201-300)
Hazardous (301-500)
"
exit

"starter-13" 20L, 508C                                    1,1           All
```

curl *downloads the web page specified by the URL argument*

*Using **tr** to delete any ">" or "<" characters*

*Using **sed** to insert a newline character between every "><" found on the web page*

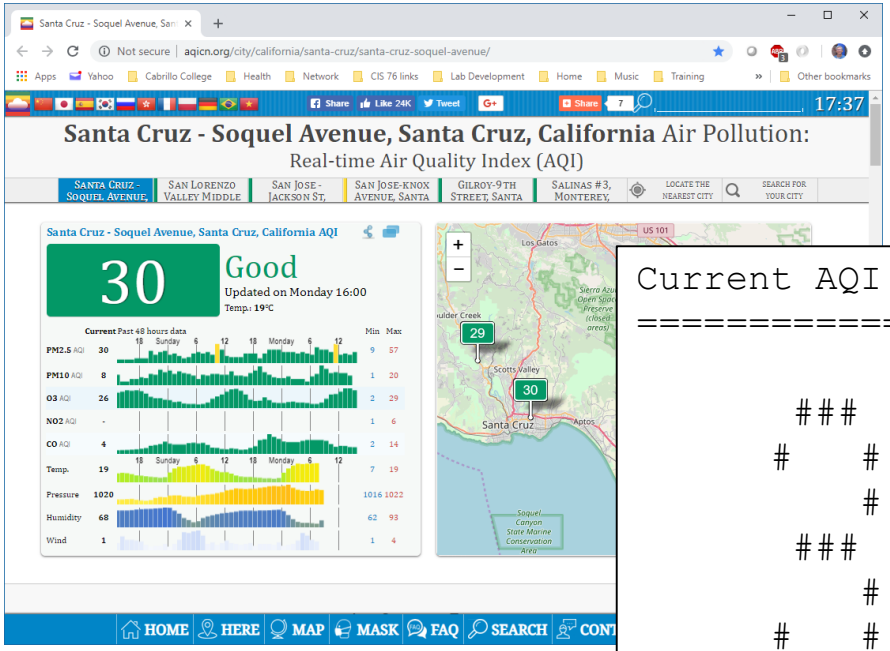
*The **-o** option on **grep** only outputs the matched text*

Use Esc **:wq** *to save file and quit vi*

```
/home/cis90/simben/bin $ chmod +x starter-13
/home/cis90/simben/bin $ starter-13
```

Open the web page in your browser to check the AQI value. Is your script correct?

Put your answer in the chat window




```
Current AQI (Air Quality Index)
================================

        ###     ####
       #   #   #    ##
           #   #   # #
        ###    # #   #
           #   ##    #
       #   #   #     #
        ###     ####


Good (0-50)
Moderate (51-100)
Unhealthy for Sensitive Groups (101-150)
Unhealthy (151-200)
Very Unhealthy (201-300)
Hazardous (301-500)

/home/cis90/simben/bin $
```

Remotely controlling a Hue smart light's brightness

/home/cis90/simben/bin $ **cd ~/bin**
/home/cis90/simben/bin $ **cp ~/../depot/scripts/ starter-14 .**
/home/cis90/simben/bin $ **vi starter-14**

rsimms@opus-ii:/home/cis90/depot/scripts

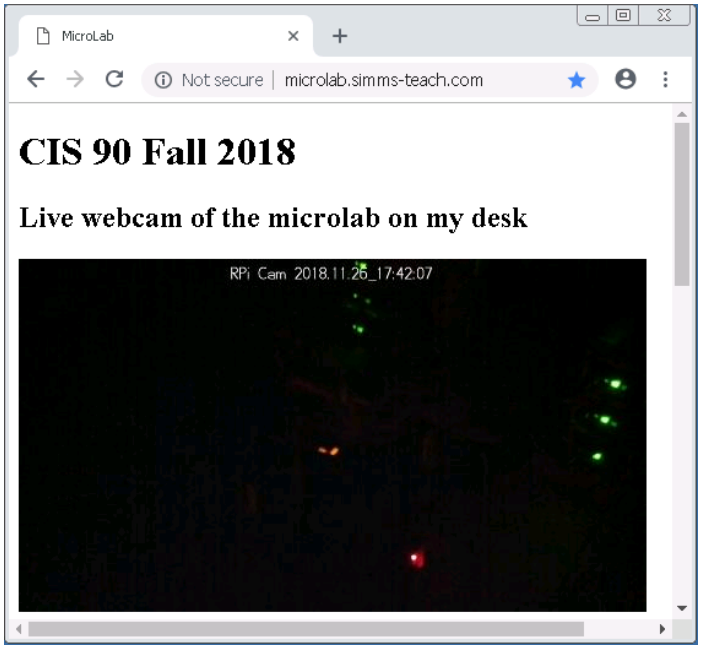
```
#!/bin/bash
# Scripting 101 sample script for CIS 90
# Description: Remotely control via ssh a Hue smart light
clear
echo Browse to: http://microlab.simms-teach.com

hostname=brienne.simms-teach.com
port=2225
hueBridge=[illegible]
hueUser=[illegible]

read -p "Enter brightness value (0-255) [50]: " custom
if [ "$custom" = "" ]; then custom=50; fi

settings={\"on\":true,\"bri\":$custom}
url="http://$hueBridge/api/$hueUser/lights/1/state"
encoded=$(echo "curl -H Accept:application/json -X PUT --data '$settings' $url > hue-status" | base64 -w 0)
ssh -p $port $LOGNAME@$hostname "echo $encoded | base64 -d > hue-script; chmod +x hue-script; ./hue-script; echo; cat hue-status"
exit
~
"starter-14" 19L, 715C                                                    3,1           All
```

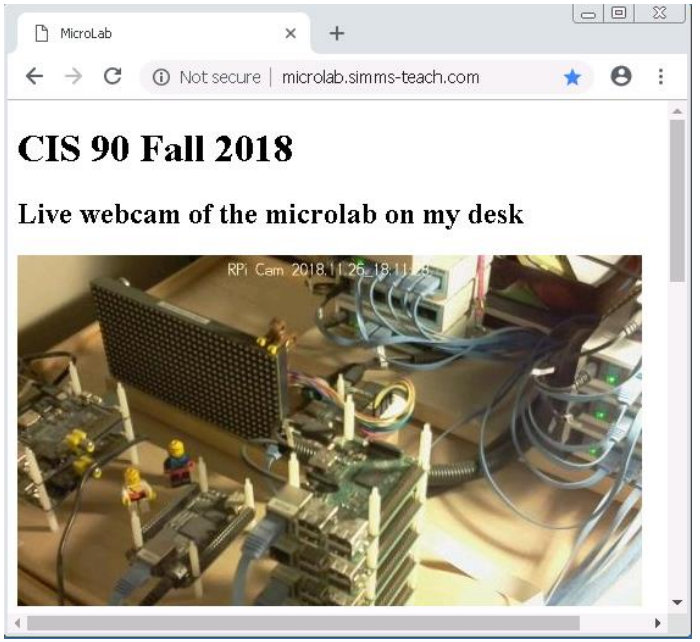
Do this first!

Use Esc **:wq** *to save file and quit vi*

/home/cis90/simben/bin $ **chmod +x starter-14**
/home/cis90/simben/bin $ **starter-14**

Can you turn my light on to full brightness level (255)?
Put your answer in the chat window






```
[simben90@son-of-opus bin]$ starter-14
Browse to: http://microlab.simms-teach.com
Enter brightness value (0-255) [50]: 210
simben90@brienne.simms-teach.com's password:
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
[{"success":{"/lights/1/state/on":true}},{"success":{"/lights/1/state/bri":210}}]100   102    0
81  100    21   5178   1342 --:--:-- --:--:-- --:--:--  5400
[simben90@son-of-opus bin]$
```

Remotely turning off a Hue smart light

```
/home/cis90/simben/bin $ cd ~/bin
/home/cis90/simben/bin $ cp ~/../depot/scripts/ starter-15 .
/home/cis90/simben/bin $ vi starter-15
```

rsimms@opus-ii:/home/cis90/depot/scripts

```
#!/bin/bash
# Scripting 101 sample script for CIS 90
# Description: Remotely turn off via ssh a Hue smart light
clear
echo Browse to: http://microlab.simms-teach.com

hostname=brienne.simms-teach.com
port=2225
scriptFile=hue-script
hueBridge=[illegible]
hueUser=[illegible]

settings={\"on\":false}
url="http://$hueBridge/api/$hueUser/lights/1/state"
encoded=$(echo "curl -H Accept:application/json -X PUT --data '$settings' $url > hue-status" | base64 -w 0)
ssh -p $port $LOGNAME@$hostname "echo $encoded | base64 -d > hue-script; chmod +x hue-script; ./hue-script; echo; cat hue-status"
exit
~
~
~
"starter-15" 17L, 625C                                                    1,1           All
```

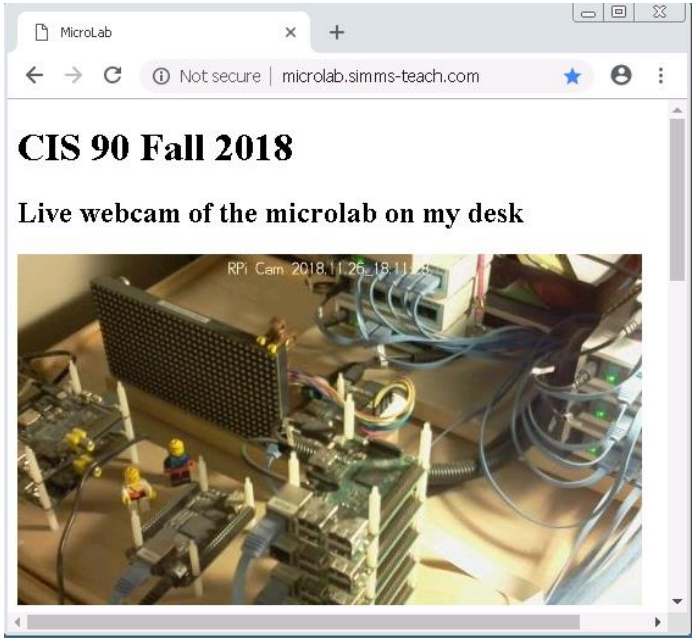
Do this first!

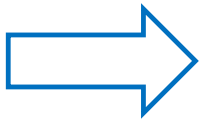
Use Esc **:wq** *to save file and quit vi*

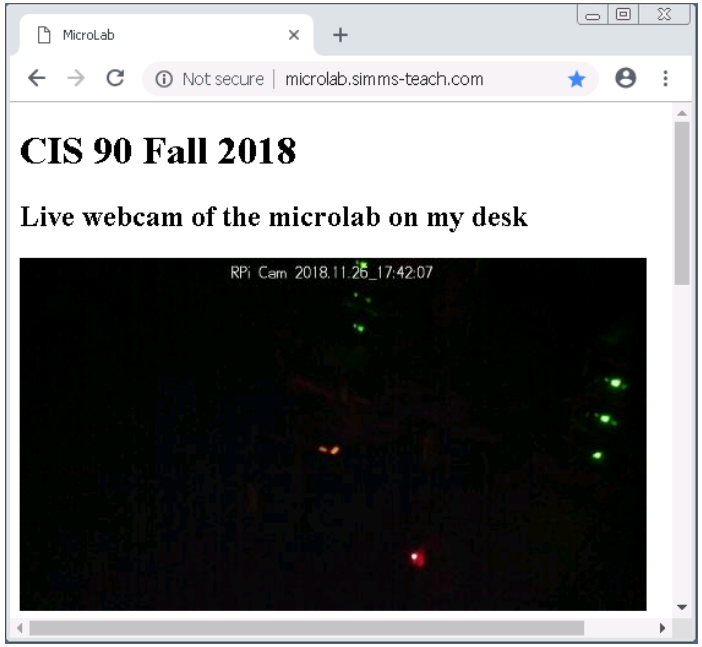
```
/home/cis90/simben/bin $ chmod +x starter-15
/home/cis90/simben/bin $ starter-15
```

Can you turn my light off?

Put your answer in the chat window








```
[simben90@son-of-opus bin]$ starter-15
Browse to: http://microlab.simms-teach.com
simben90@brienne.simms-teach.com's password:
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    54    0    42    0     0      0      0 --:--:-- --:--:-- --:--:--     0
 100    12  2764    789 --:--:-- --:--:-- --:--:--  2800
[{"success":{"/lights/1/state/on":false}}][simben90@son-of-opus bin]$
```

Random numbers, terminal text placement and color

```
/home/cis90/simben/bin $ cd ~/bin
/home/cis90/simben/bin $ cp ~/../depot/scripts/ starter-16 .
/home/cis90/simben/bin $ vi starter-16
```

rsimms@opus-ii:/home/cis90/depot/scripts

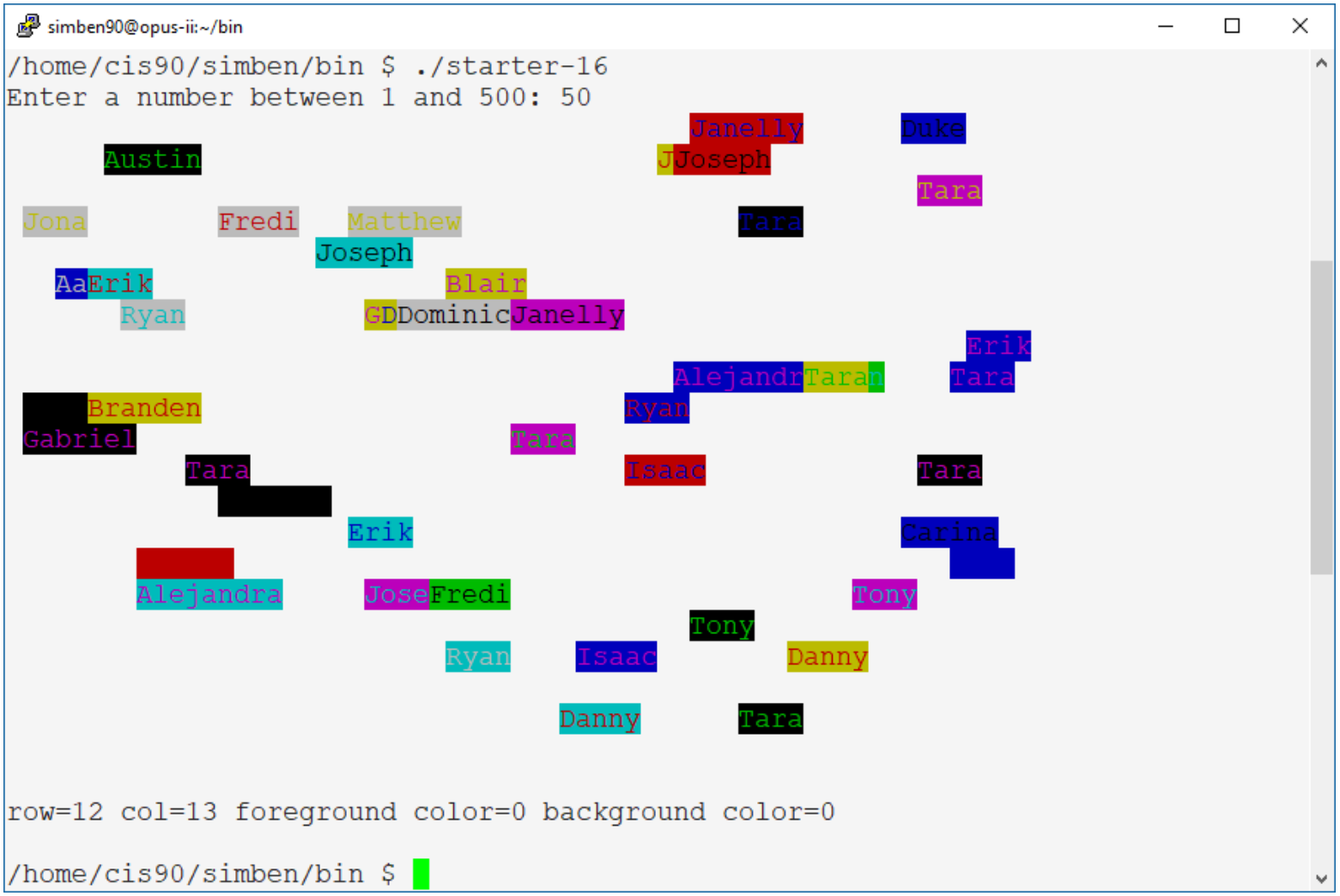
```
#!/bin/bash
# Scripting 101 sample script for CIS 90
# Description: Random numbers, terminal text color and placement
read -p "Enter a number between 1 and 500: " size
gid=$(grep "^cis90:" /etc/group | cut -f3 -d":")
numStudents=$(grep ":$gid:" /etc/passwd | grep -v tbd | grep -v "^cis90:" | wc -l)
clear
for (( i=1; i<=$size; i++ )); do
  studentNum=$((RANDOM%$numStudents))
  first=$(grep ":$gid:" /etc/passwd | grep -v tbd | grep -v "^cis90:" | head -n $studentNum | tail -n1 | cut -f5 -d ":" | cut -f1 -d " ")
  foregroundColor=$((RANDOM%8))
  backgroundColor=$((RANDOM%8))
  row=$((RANDOM%20))
  col=$((RANDOM%60))
  tput setf $foregroundColor
  tput setb $backgroundColor
  tput cup $row $col
  echo -n "$first"
  echo -en "\e[0m"
  tput cup 22 0
  echo -n "row=$row col=$col foreground color=$foregroundColor background color=$backgroundColor    "
  sleep .5
done
tput reset
echo $PS1
exit

"starter-16" 27L, 899C                                                    1,1           All
```

Use Esc **:wq** *to save file and quit vi*

```
/home/cis90/simben/bin $ chmod +x starter-16
/home/cis90/simben/bin $ starter-16
```

How could you spread the text across more columns?

Put your answer in the chat window


simben90@opus-ii:~/bin
/home/cis90/simben/bin $ ./starter-16
Enter a number between 1 and 500: 50
Janelly
Duke
Austin
JJoseph
Tara
Jona
Fredi
Matthew
Tara
Joseph
AaErik
Blair
Ryan
GDDominicJanelly
Erik
AlejandrTara
Tara
Branden
Ryan
Gabriel
Tara
Tara
Isaac
Tara
Erik
Carina
Alejandra
JoseFredi
Tony
Tony
Ryan
Isaac
Danny
Danny
Tara
row=12 col=13 foreground color=0 background color=0
/home/cis90/simben/bin $

Display a message on the STEM center LEDs

```
/home/cis90/simben/bin $ cd ~/bin
/home/cis90/simben/bin $ cp ~/../depot/scripts/ starter-17 .
/home/cis90/simben/bin $ vi starter-17
```

simben90@opus-ii:~/bin

```
#!/bin/bash
# Scripting 101 sample script for CIS 90
# Description: Display text on the LED panel in the CIS Lab

clear
echo From the CIS network browse to: http://ulab-webcam/
read -p "Which message buffer file? 1=/tmp/uLab 2=/tmp/microlab 3=/tmp/LEDs [1] : " choice
if [ "$choice" = "" ]; then choice=1; fi
case $choice in
  1) file=/tmp/uLab ;;
  2) file=/tmp/microlab ;;
  3) file=/tmp/LEDs ;;
  *) file=/tmp/uLab
esac

echo Use a . to between shorts words to display at teh same time
read -p "Enter a one-line message: " message
if [ "$choice" = "1" ]; then
  color=5; time=1
  read -p "Select color 1=magenta, 2=red, 3=blue, 4=green, 5=turquoise, 6=orange [4]: " color
  if [ "$color" = "" ]; then color=4; fi
  read -p "Select time to display each word in seconds (1-5) [1]: " time
  if [ "$time" = "" ]; then time=1; fi
  if [ "$time" -gt "5" ]; then time=5; fi
  if [ "$time" -lt "1" ]; then time=1; fi
  echo $color:$time > ${file}-config
fi
echo "$message" > $file
exit
~
"starter-17" 29L, 983C                                        1,1           All
```

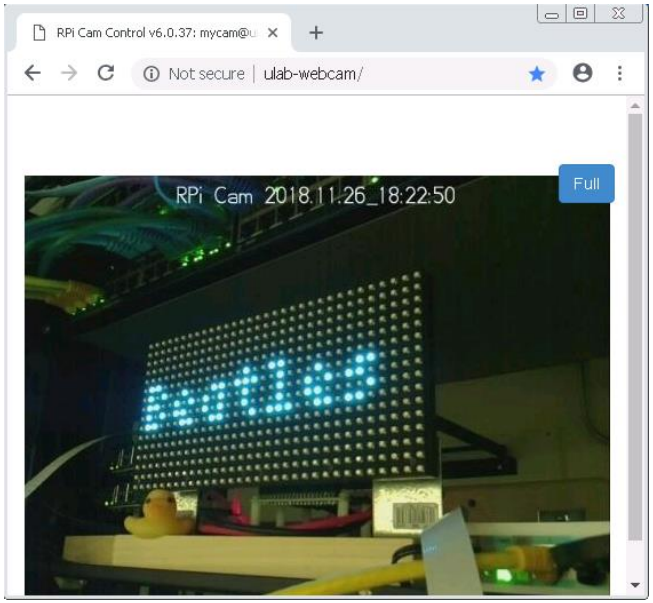
Use Esc **:wq** *to save file and quit vi*

```
/home/cis90/simben/bin $ chmod +x starter-17
/home/cis90/simben/bin $ starter-17
```

If you are off campus view the webcam from your Arya VM (via VLab).
Did it work?

Put your answer in the chat window

```
/home/cis90/simben/bin $ ./starter-17
From the CIS network browse to: http://ulab-webcam/
Which message buffer file? 1=/tmp/uLab 2=/tmp/microlab 3=/tmp/LEDs [1] : 1
Use a . to between shorts words to display at teh same time
Enter a one-line message: Beatles
Select color 1=magenta, 2=red, 3=blue, 4=green, 5=turquoise, 6=orange [4]: 5
Select time to display each word in seconds (1-5) [1]: 10
/home/cis90/simben/bin $
```



Don't name your scripts "script"!

Don't name your scripts "script"

```
[simben90@opus bin]$ ls -l script
-rwxr-x--- 1 simben90 cis90 47 Nov 23 16:44 script

[simben90@opus bin]$ cat script
echo "Hello from the script file named script"
```

What would happen if your ran the script above?

Don't name your scripts "script"

```
[simben90@opus bin]$ cat script
echo "Hello from the script file named script"

[simben90@opus bin]$ script
Script started, file is typescript
```

Why the heck doesn't my script do what it's supposed to do?

Don't name your scripts "script"

```
[simben90@opus bin]$ cat script
echo "Hello from the script file named script"

[simben90@opus bin]$ script
Script started, file is typescript
```

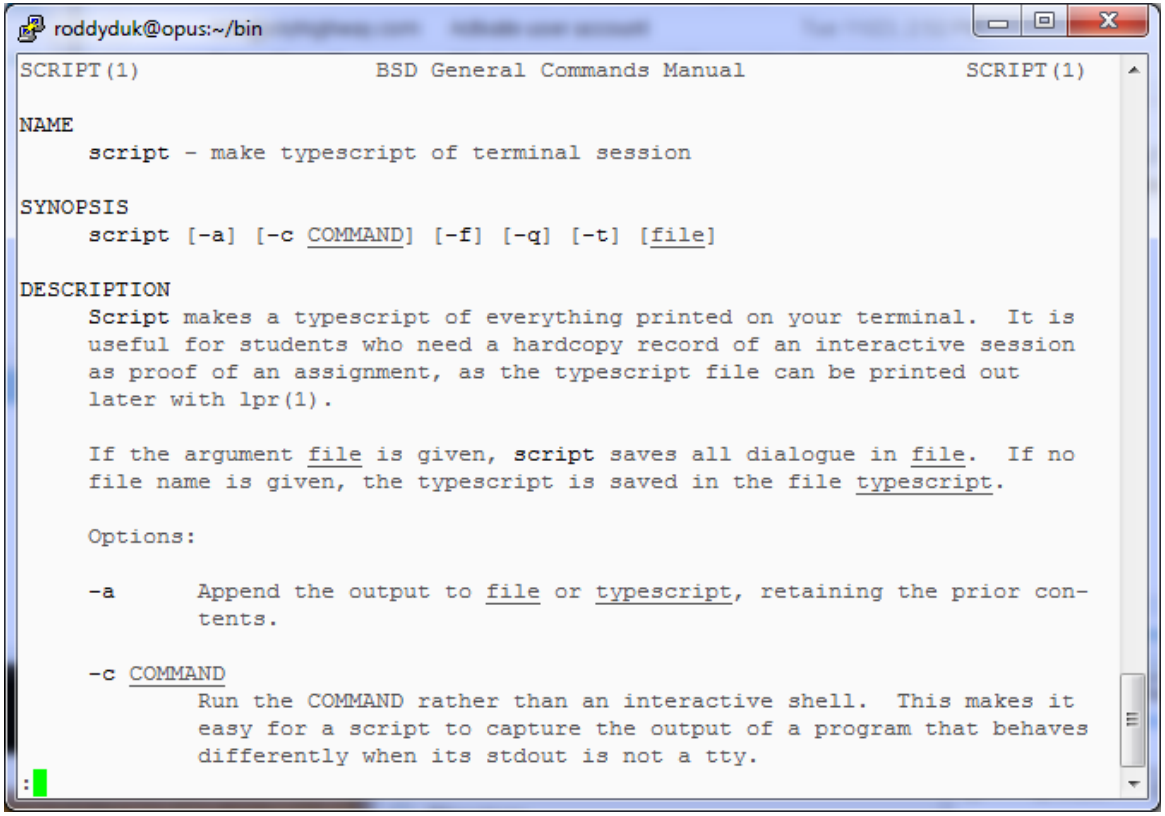
Why the heck doesn't my script do what it's supposed to do?

```
[simben90@opus bin]$ Where is my script?
bash: Where: command not found
[simben90@opus bin]$ exit
Script done, file is typescript
[simben90@opus bin]$ cat typescript
Script started on Wed 13 May 2009 08:00:02 AM PDT
[simben90@opus bin]$ Where is my script?
bash: Where: command not found
[simben90@opus bin]$ exit

Script done on Wed 13 May 2009 08:00:47 AM PDT
[simben90@opus bin]$
```

Don't name your scripts "script"

Why doesn't script do what it is supposed to do? … because script is the name of an existing UNIX command!

```
[simben90@opus bin]$ man script
[simben90@opus bin]$
```



Don't name your scripts "script"

There are (at least) two files named script on Opus

```
[simben90@opus bin]$ type script
script is hashed (/usr/bin/script)
[simben90@opus bin]$ file /usr/bin/script
/usr/bin/script: ELF 32-bit LSB executable, Intel 80386, version 1
(SYSV), for GNU/Linux 2.6.9, dynamically linked (uses shared libs),
for GNU/Linux 2.6.9, stripped
```

```
[simben90@opus bin]$ type /home/cis90/simben/bin/script
/home/cis90/simben/bin/script is /home/cis90/simben/bin/script
[simben90@opus bin]$ file /home/cis90/simben/bin/script
/home/cis90/simben/bin/script: ASCII text
[simben90@opus bin]$
```

***Question**: Why did bash run the script in /usr/bin instead of the script in /home/cis90/simben/bin?*

Don't name your scripts "script"

***Question**: Why did bash run the script in /usr/bin instead of the script in /home/cis90/simben/bin?*

*The Linux **script** command is in this directory*

```
[simben90@opus bin]$ echo $PATH
/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/home/cis90/bin:
/home/cis90/simben/bin:.
```

*Our script, named **script**, is in this directory*

***Answer**: bash searches the path in the order the directories are listed. It finds the script command in /user/bin first.*

Don't name your scripts "script"

To override the PATH you can always specify an absolute pathname to the file you want to run:

```
[simben90@opus bin]$ /home/cis90/simben/bin/script
Hello from the script file named script

[simben90@opus bin]$ ./script
Hello from the script file named script
```

Note the shell treats the . above as "here" which in this case is /home/cis90/simben/bin

Try the script command

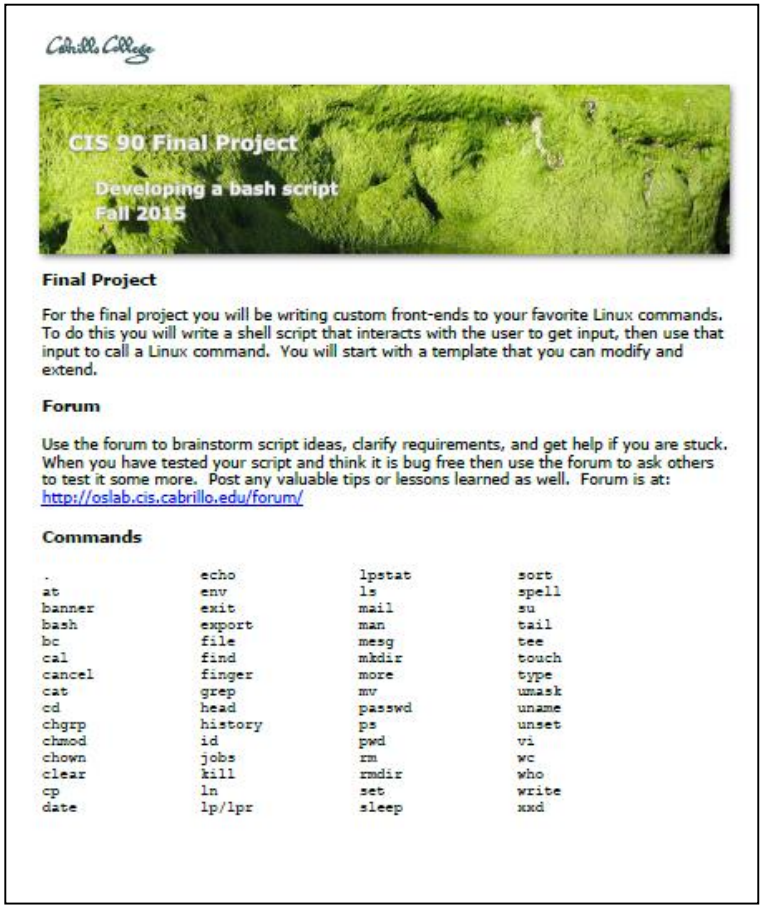
- Use the **script** command to start recording
- Type various commands of your choice
- Type **exit** or hit **Ctrl-D** to end recording
- Use **cat typescript** to see what you recorded

This would be a good way to record a session such as working one of the lab assignments for future reference.

When finished type "done" in the chat window


Assignment

Start your project!



Cabrillo College

CIS 90 Final Project

Developing a bash script

Fall 2015

Final Project

For the final project you will be writing custom front-ends to your favorite Linux commands. To do this you will write a shell script that interacts with the user to get input, then use that input to call a Linux command. You will start with a template that you can modify and extend.

Forum

Use the forum to brainstorm script ideas, clarify requirements, and get help if you are stuck. When you have tested your script and think it is bug free then use the forum to ask others to test it some more. Post any valuable tips or lessons learned as well. Forum is at: http://oslab.cis.cabrillo.edu/forum/

Commands

.	echo	lpstat	sort
at	env	ls	spell
banner	exit	mail	su
bash	export	man	tail
bc	file	mesg	tee
cal	find	mkdir	touch
cancel	finger	more	type
cat	grep	mv	umask
cd	head	passwd	uname
chgrp	history	ps	unset
chmod	id	pwd	vi
chown	jobs	rm	wc
clear	kill	rmdir	who
cp	ln	set	write
date	lp/lpr	sleep	xxd

Start early and finish on time!


Wrap up

Commands:

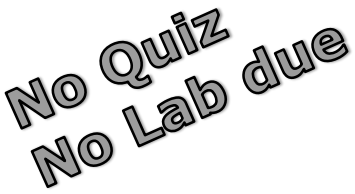
lp, lpr	- Linux print command
cancel, lprm	- cancel print job
lpq, lpstat	- Show print queue

Web:

http://hostname:631	- CUPS web based management utility
http://hostname:9100	- HP JetDirect printer

Next Class

Assignment: Check Calendar Page on web site to see what is due next week.



Work on final project - due in two weeks!

Optional extra credit labs

Project Workshop

- See if you can get one "starter" task scripted and working before leaving class today.
- Grade your starter script using the Final Project rubric.

Implementing all five tasks (6 points each): • Requirements for each task: – Minimum of 12 "original" lines of bash script – Has one or more non-generic comments to explain what it is doing – Has user interaction
At least six bash constructs from this list: • Redirecting stdin (4 points) • Redirecting stdout (4 points) • Redirecting stderr (4 points) • Use of permissions (4 points) • Use of filename expansion characters (4 points) • Use of absolute path (4 points) • Use of relative path (4 points) • Use of a PID (4 points) • Use of inodes (4 points) • Use of links (4 points) • Use of color (4 points) • Use of scheduling (4 points) • Use of a GID or group (4 points) • Use of a UID or user (4 points) • Use of a /dev/tty device (4 points) • Use of a signal (4 points) • Use of piping (4 points) • Use of an environment variable (4 points) • Use of /bin/mail (4 points) • Use of a conditional (4 points) • Use of $(*command*) The maximum for this section is 24 points.

End Meeting

Backup

Review

```
function runningScript ()
{
```

The rules of the road for variables

- Rule 1: A child process can only see variables the parent has exported.
- Rule 2: A child process cannot change the parent's variables.

Running a Script

```
/home/cis90/simben $ cat mydate
#!/bin/bash
echo "Hola $LOGNAME"
date +'%m/%d/%Y'
echo $myvar1 $myvar2 $myvar3
```

Add this line to the last script we made

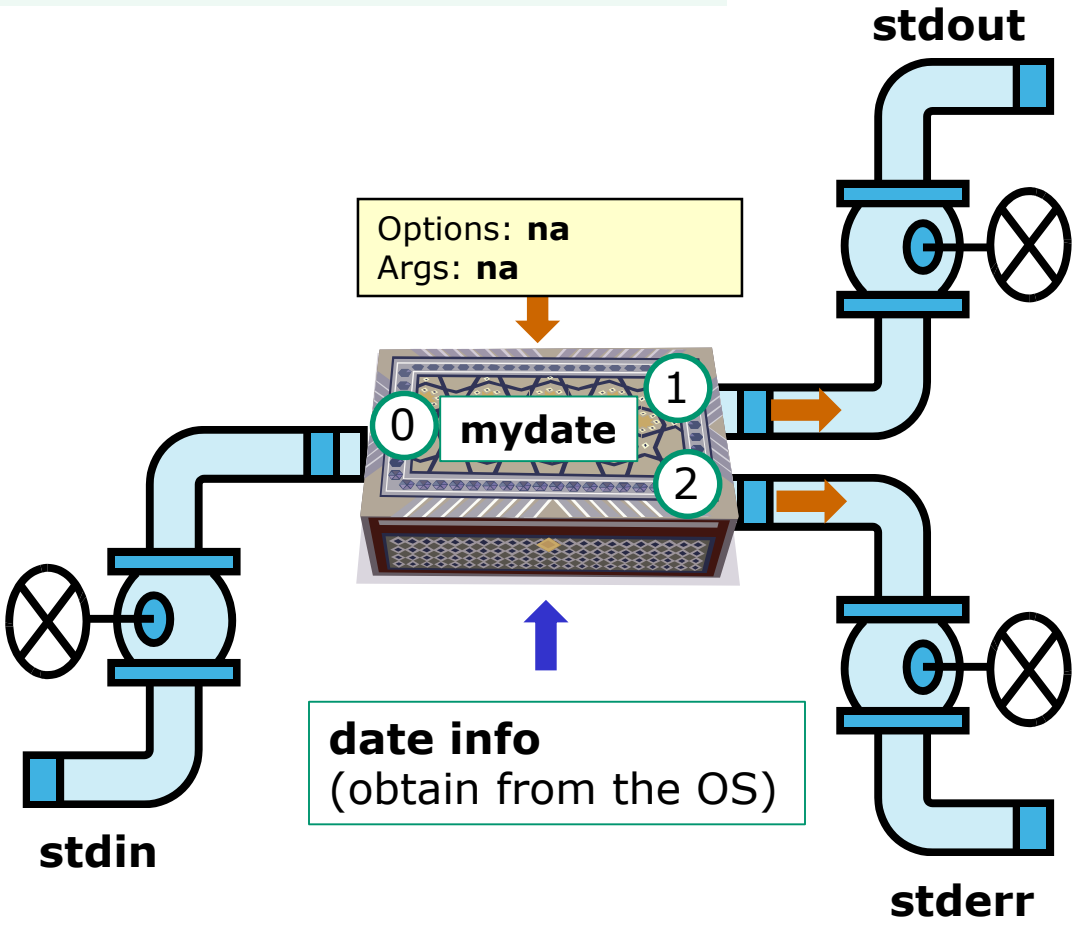
Don't initialize them yet

```
/home/cis90/simben $ mydate
Hola simben90
05/16/2013

/home/cis90/simben $
```

Because the variables don't exist yet the last echo statement prints a blank line

Running a Script

$ mydate



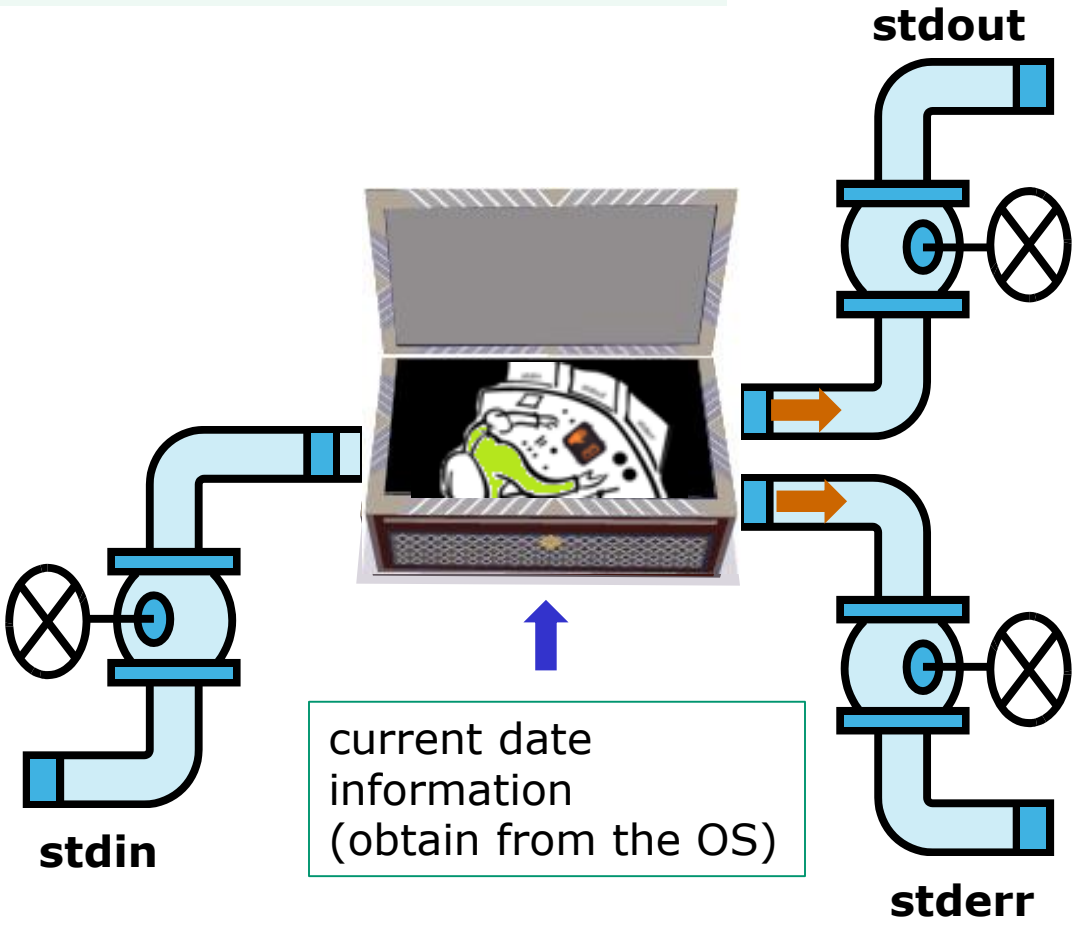

```
Hola simben90
05/09/2013
```

*In this example, ouptput from **myscript** goes to stdout.*

stdout has not been redirected so it goes to the default terminal device (your screen).

Running a Script

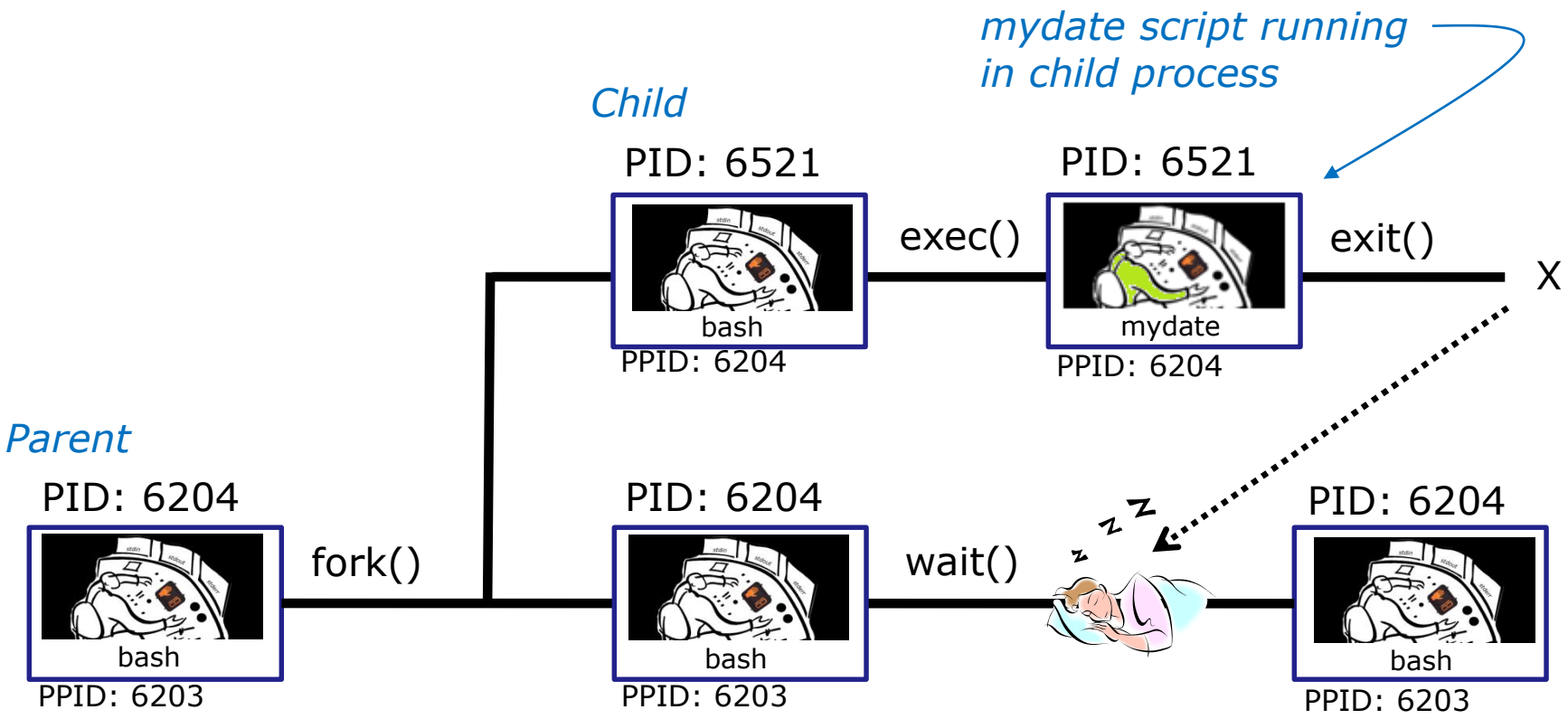
$ mydate




```
Hola simben90
05/16/2012
```

A sneak peek into memory to see what our process looks like!

Running a Script



*Whenever you run any command, program, or script it runs as a **child process***

Running a Script

```
/home/cis90/simben $ cat mydate
#!/bin/bash
echo "Hola $LOGNAME"
date +'%m/%d/%Y'
echo $myvar1 $myvar2 $myvar3
```

In the parent process, initialize the three variables

```
/home/cis90/simben $ myvar1=Tic; myvar2=Tac; myvar3=Toe
/home/cis90/simben $ echo $myvar1 $myvar2 $myvar3
Tic Tac Toe
```

*What happens if we run **mydate** now?*

Running a Script

```
/home/cis90/simben $ cat mydate
#!/bin/bash
echo "Hola $LOGNAME"
date +'%m/%d/%Y'
echo $myvar1 $myvar2 $myvar3

/home/cis90/simben $ myvar1=Tic; myvar2=Tac; myvar3=Toe
/home/cis90/simben $ echo $myvar1 $myvar2 $myvar3
Tic Tac Toe

/home/cis90/simben $ mydate
Hola simben90
05/09/2012

/home/cis90/simben $
```

*Running **mydate***
(as a child process)

Why no Tic Tac Toe output?

Running a Script

```
/home/cis90/simben $ export myvar1
/home/cis90/simben $ mydate
Hola simben90
05/09/2012
Tic

/home/cis90/simben $ export myvar2
/home/cis90/simben $ mydate
Hola simben90
05/09/2012
Tic Tac

/home/cis90/simben $ export myvar3
/home/cis90/simben $ mydate
Hola simben90
05/09/2012
Tic Tac Toe
```

Rule 1: A child process can only see variables the parent has exported

Running a Script

```
/home/cis90/simben $ echo $myvar1 $myvar2 $myvar3
Tic Tac Toe

/home/cis90/simben $ cat mydate
#!/bin/bash
echo "Hola $LOGNAME"
date +'%m/%d/%Y'
echo $myvar1 $myvar2 $myvar3
myvar1=red myvar2=white myvar3=blue
echo $myvar1 $myvar2 $myvar3

/home/cis90/simben $ mydate
Hola simben90
05/09/2012
Tic Tac Toe
red white blue

/home/cis90/simben $ echo $myvar1 $myvar2 $myvar3
Tic Tac Toe
```

Add these new lines

Rule 2: A child process cannot change the parent's variables.

Running a Script

Unless we want them to

```
/home/cis90/simben $ echo $myvar1 $myvar2 $myvar3
Tic Tac Toe

/home/cis90/simben $ source mydate
Hola simben90
05/09/2012
Tic Tac Toe
red white blue

/home/cis90/simben $ echo $myvar1 $myvar2 $myvar3
red white blue
```

Sourcing a script causes the instructions to be run in the parent process. A child process is not created

```
}
while не розумію
do
  runningScript
done
```

RPi Zero Envy 4500 Configuration via CUPS

CUPS Demo on RPi and HP Envy 4500

Raspberry Pi configuration (Jessie)

1. Bootup with monitor
2. Connect to wireless uLab network (might need HDMI monitor and keyboard)
3. As root:
   usermod -a -G lpadmin *username*
   apt-get update
   apt-get install cups cups-bsd
   apt-get install hplip
   apt-get install sysvbanner
   apt-get install tightvncserver
   Enable remote administration on CUPS
4. As *username*:
   vncserver

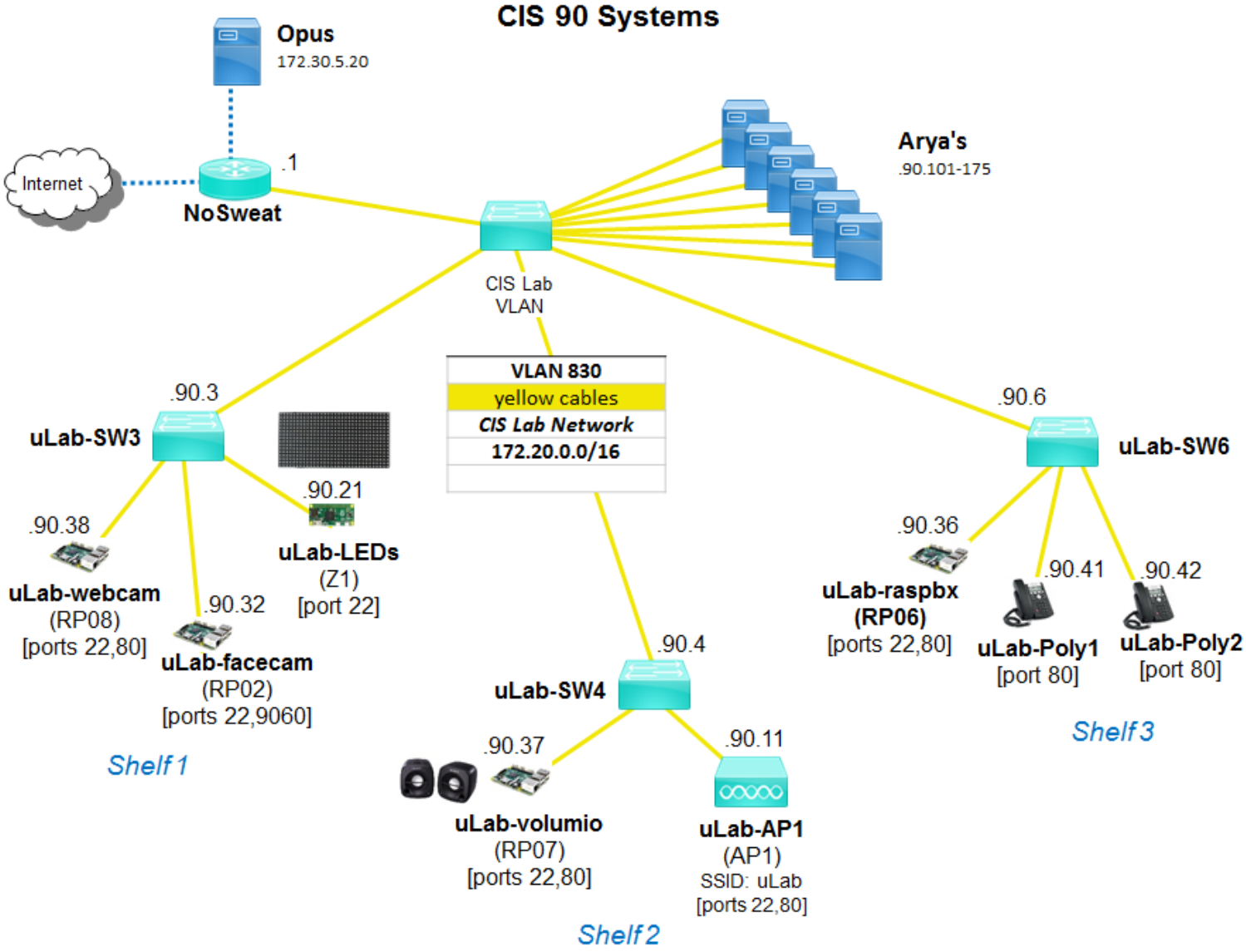
Classroom Instructor PC

- Browse to http://<printer-IP>:631
- Run Elmo Image Mate in expert mode and rotate image

Troubleshoot if needed:

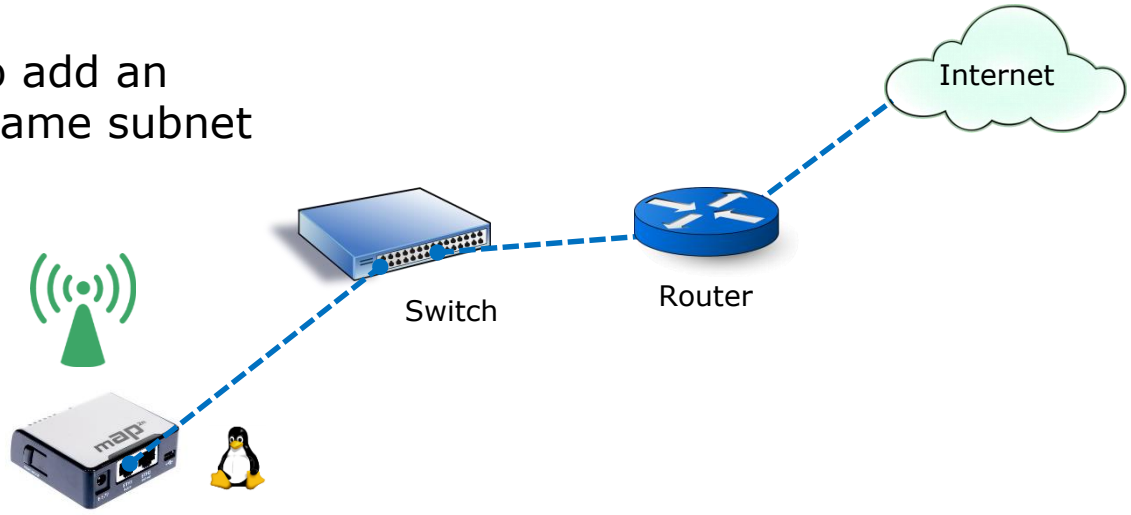
- Instructor PC: install tightvnc from http://www.tightvnc.com/
- Run TightVNC Viewer and connect to: <Raspberry Pi IP>:5901

Hostname	MAC	IP	Ports
home-PanTilt	b8:27:eb:66:ce:79	172.20.90.230	http://<ip-address>:80 and 9595
home-ZeroW	b8:27:eb:00:58:f0	172.20.90.231	http://<ip-address>:631
home-HPEnvy4500	58:20:B1:F1:E2:66	172.20.90.232	http://<ip-address>:80

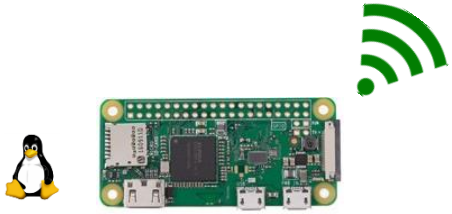

CIS 90 Systems
Opus
172.30.5.20
Internet
.1
NoSweat
Arya's
.90.101-175
CIS Lab
VLAN
VLAN 830
yellow cables
CIS Lab Network
172.20.0.0/16
.90.3
uLab-SW3
.90.21
uLab-LEDs
(Z1)
[port 22]
.90.38
uLab-webcam
(RP08)
[ports 22,80]
.90.32
uLab-facecam
(RP02)
[ports 22,9060]
Shelf 1
.90.4
uLab-SW4
.90.37
uLab-volumio
(RP07)
[ports 22,80]
.90.11
uLab-AP1
(AP1)
SSID: uLab
[ports 22,80]
Shelf 2
.90.6
uLab-SW6
.90.36
uLab-raspbx
(RP06)
[ports 22,80]
.90.41
uLab-Poly1
[port 80]
.90.42
uLab-Poly2
[port 80]
Shelf 3

CUPS Demo on RPi Zero and HP Envy 4500

This example will show how to add an HP Envy 4500 printer on the same subnet as the Linux print server.



uLab-AP1
MikroTik mAP 2n
Access Point
SSID: uLab

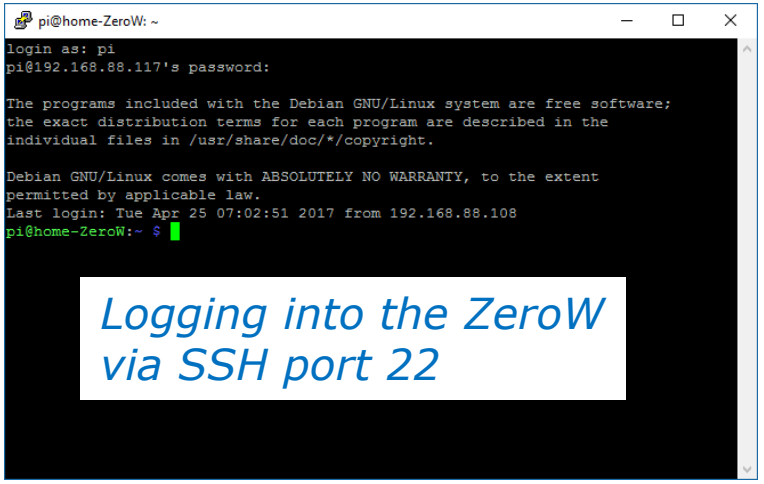


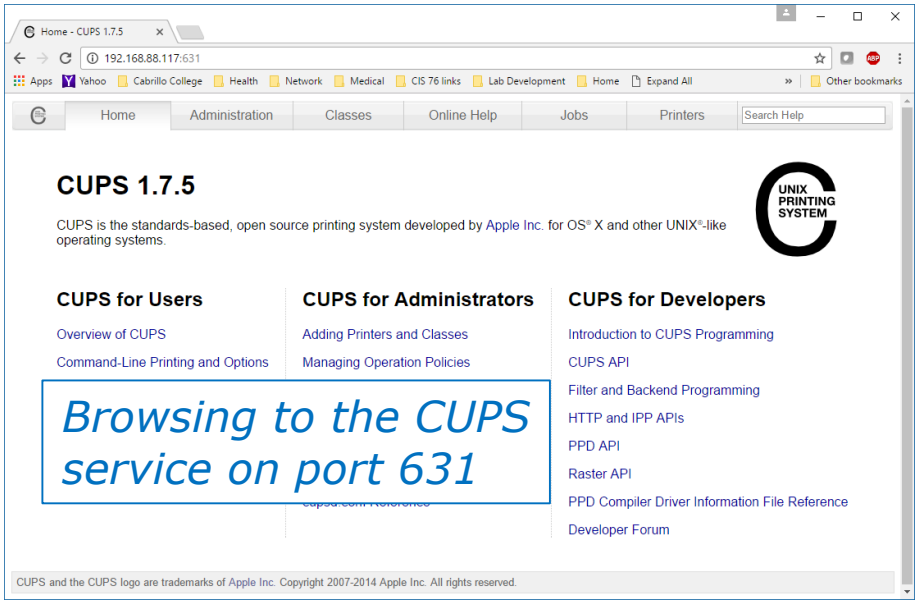
home-ZeroW
Raspberry Pi ZeroW
Raspian GNU/Linux 8 (Jessie)
Mac: b8:27:eb:00:58:f0

home-HPEnvy4500
HP Envy 4500 Printer
(Wireless Interface)
Mac: 58:20:B1:F1:E2:66

Raspberry Pi ZeroW

IP Address for this printer is:
192.168.88.117 (home)
172.30.90.231 (room 828)

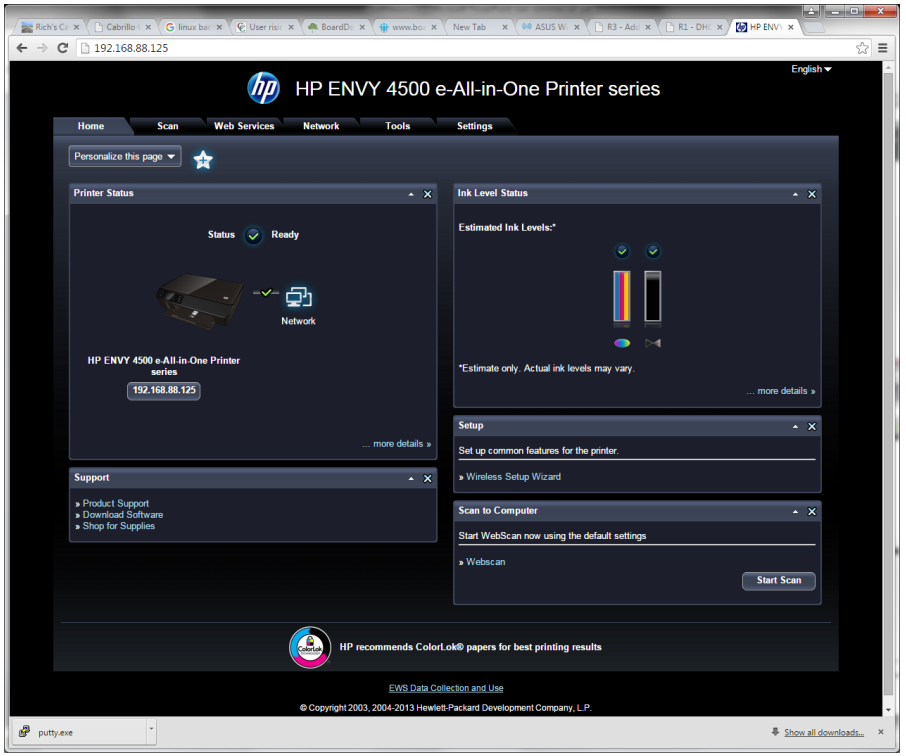




HP Envy 4500 Printer

IP Address for this printer is:
192.168.88.115 (home)
172.30.90.232 (room 828)

Networked HP printers have a built in web-server



Browsing to the IP address of the printer

RPi Envy 4500 Configuration via CUPS

CUPS Demo on RPi and HP Envy 4500

Raspberry Pi configuration (Jessie)

1. Bootup with monitor
2. Connect to wireless uLab network (might need HDMI monitor and keyboard)
3. As root:
   usermod -a -G lpadmin *username*
   apt-get update
   apt-get install cups cups-bsd
   apt-get install hplip
   apt-get install sysvbanner
   apt-get install tightvncserver
   Enable remote administration on CUPS
4. As *username*:
   vncserver

Classroom Instructor PC

- Instructor PC: install tightvnc from http://www.tightvnc.com/
- Browse to http://<printer-IP>:631
- Run Elmo Image Mate in expert mode and rotate image

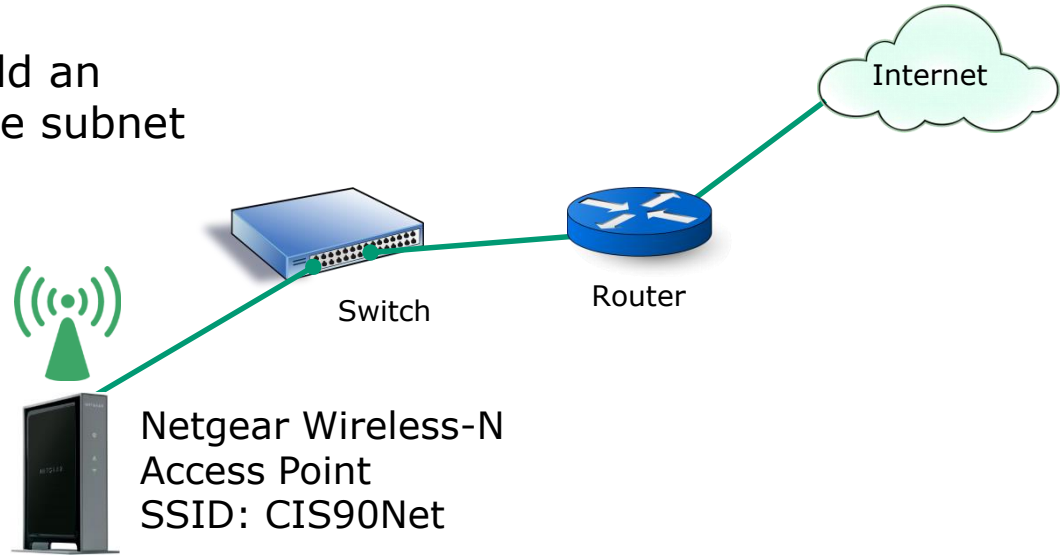
Troubleshoot if needed:

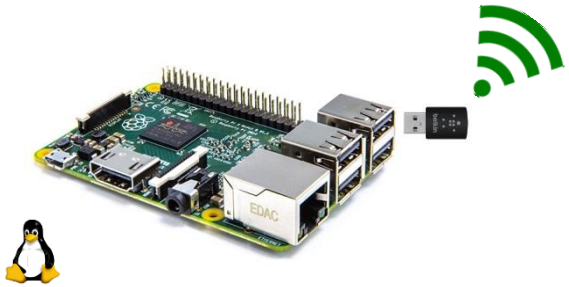
- Run TightVNC Viewer and connect to: <Raspberry Pi IP>:5901

Hostname	MAC	IP	Ports
home-PanTilt	b8:27:eb:66:ce:79	172.20.90.230	http://<ip-address>:80 and 9595
home-ZeroW	b8:27:eb:00:58:f0	172.20.90.231	http://<ip-address>:631
home-HPEnvy4500	58:20:B1:F1:E2:66	172.20.90.232	http://<ip-address>:80

CUPS Demo on RPi and HP Envy 4500

This example will show how to add an HP Envy 4500 printer on the same subnet as the Linux server.



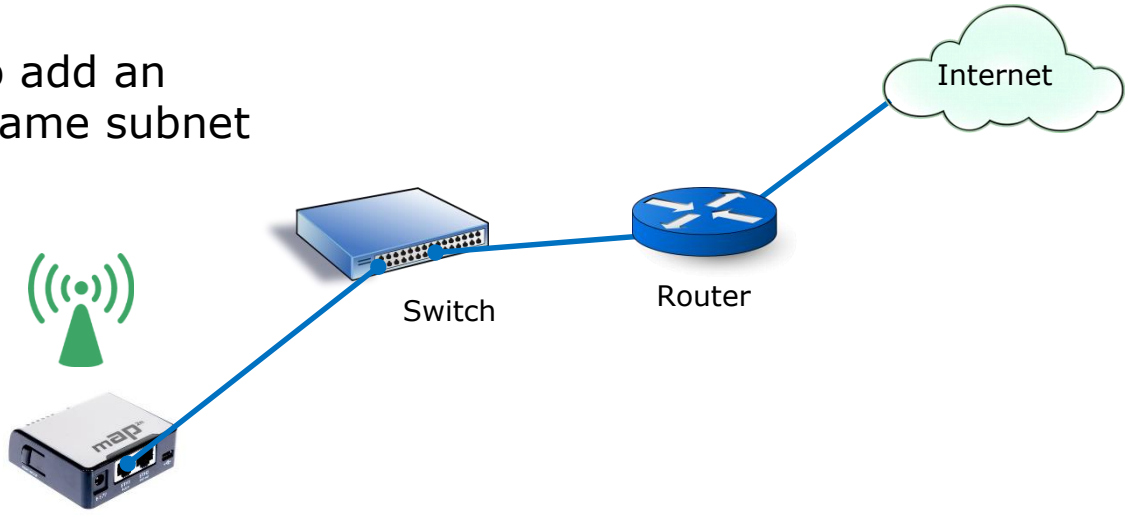


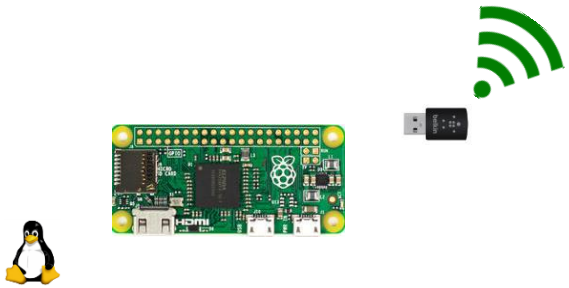
Raspberry Pi 2
Raspian GNU/Linux 8 (Jessie)
+ Belkin N300 Wireless USB Adapter
Mac: EC:1A:59:D6:2C:C0

HP Envy 4500 Printer
(Wireless Interface)
Mac: 58:20:B1:F1:E2:66

CUPS Demo on RPi Zero and HP Envy 4500

This example will show how to add an HP Envy 4500 printer on the same subnet as the Linux server.





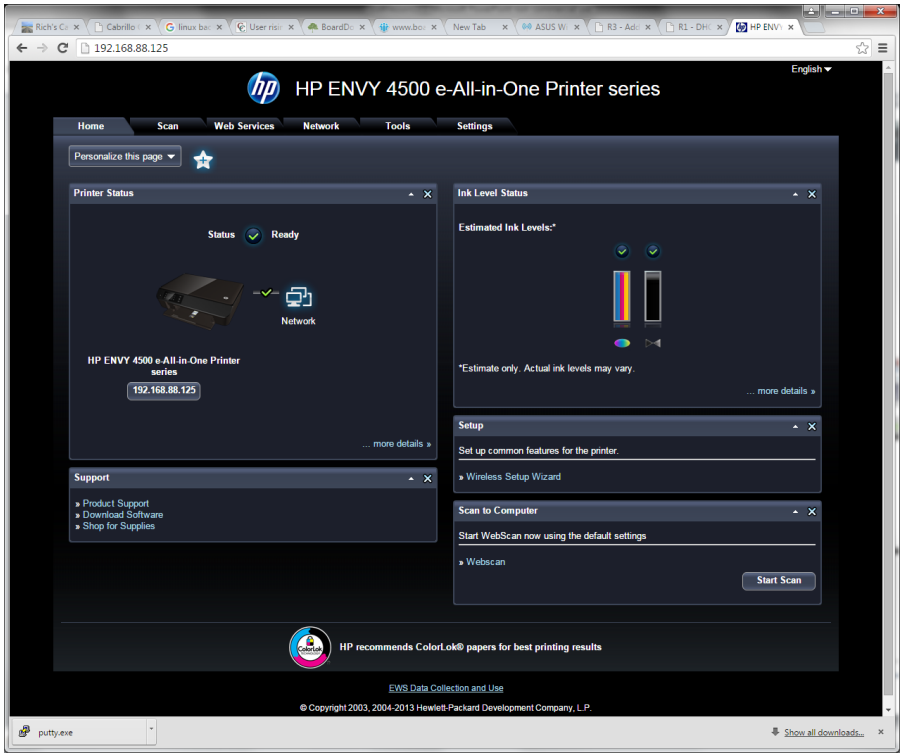
Raspberry Pi 2
Raspian GNU/Linux 8 (Jessie)
+ Belkin N300 Wireless USB Adapter
Mac: EC:1A:59:D6:2C:C0

HP Envy 4500 Printer
(Wireless Interface)
Mac: 58:20:B1:F1:E2:66

CUPS Demo on RPi and HP Envy 4500

IP Address for this printer is:
192.168.88.125 (home)
172.30.1.35 (room 828)

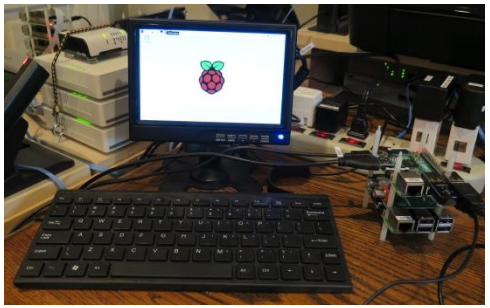
Networked HP printers have a built in web-server



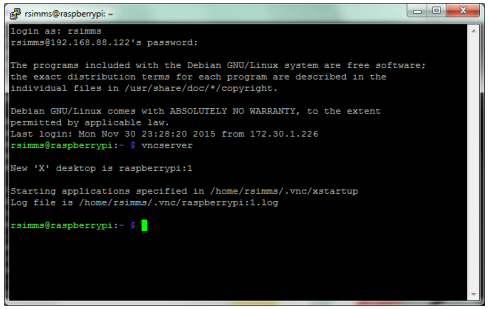
Browsing to the IP address of the printer

CUPS Demo on RPi and HP Envy 4500

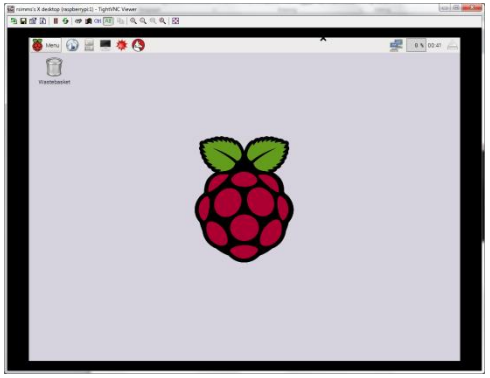
IP Address for this RPi is:
- 192.168.88.122 (home)
- 172.30.1.34 (room 828)



Local access with monitor, keyboard and mouse

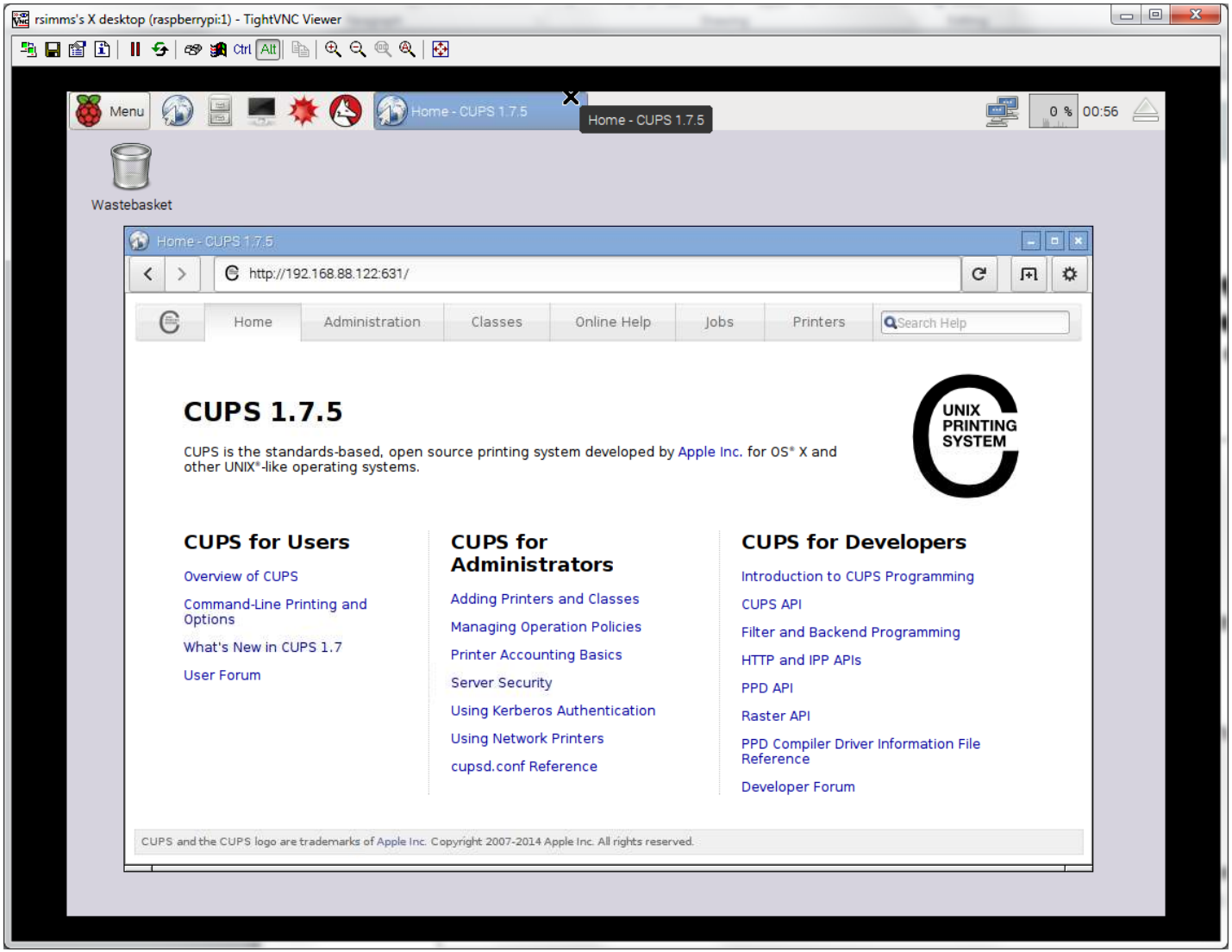


SSH access over network

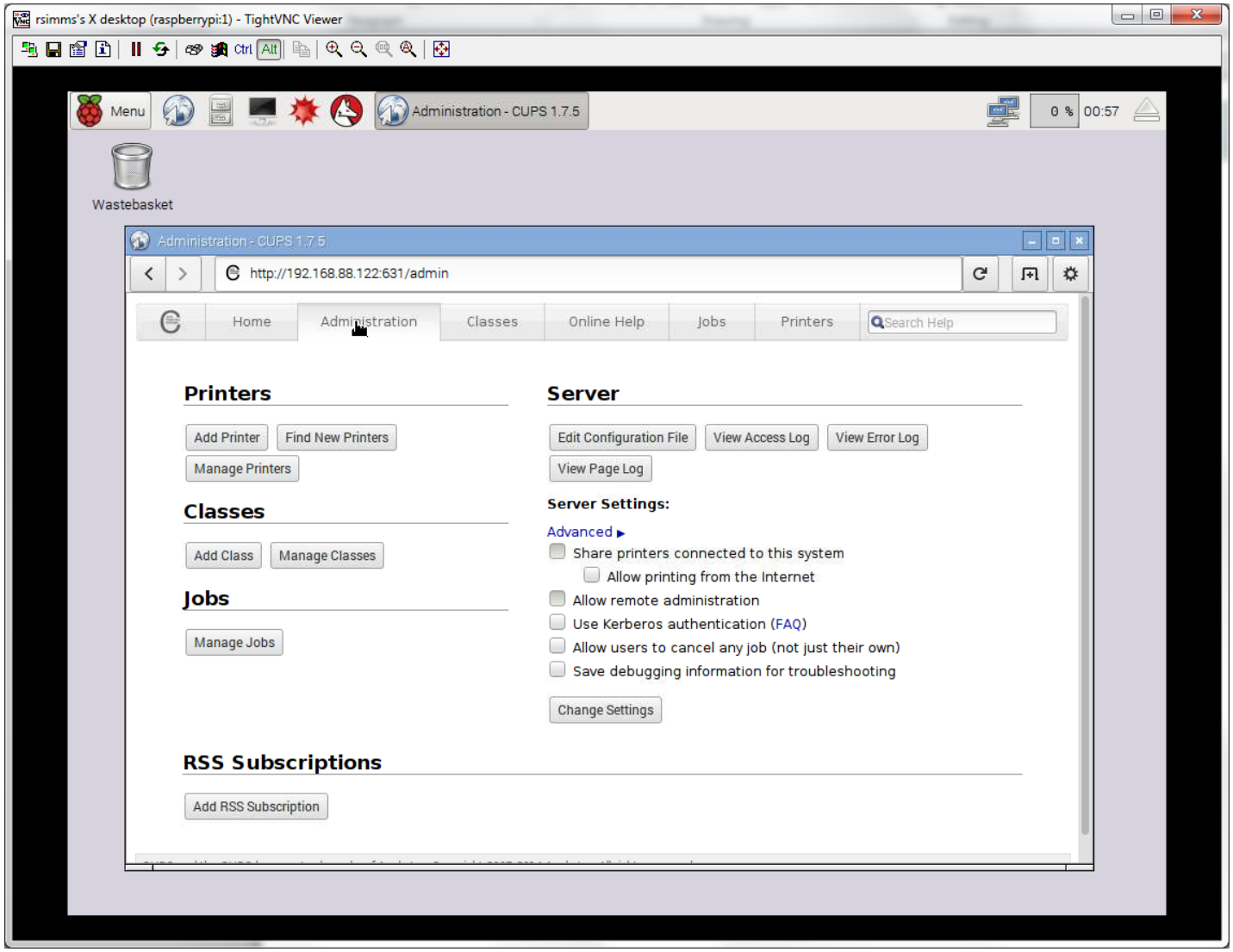


VNC access over network

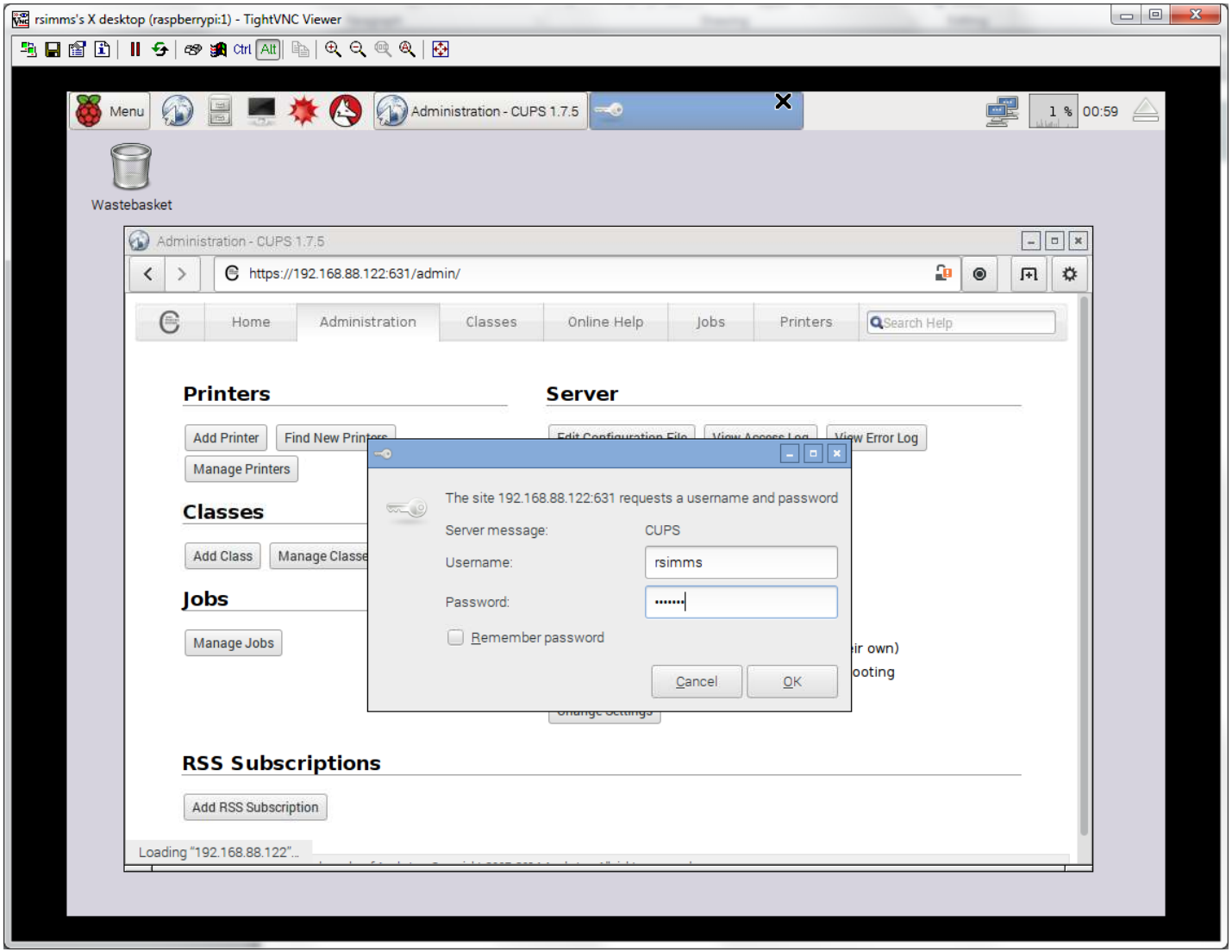
Browse to CUPS service at <server-ip-address>:631



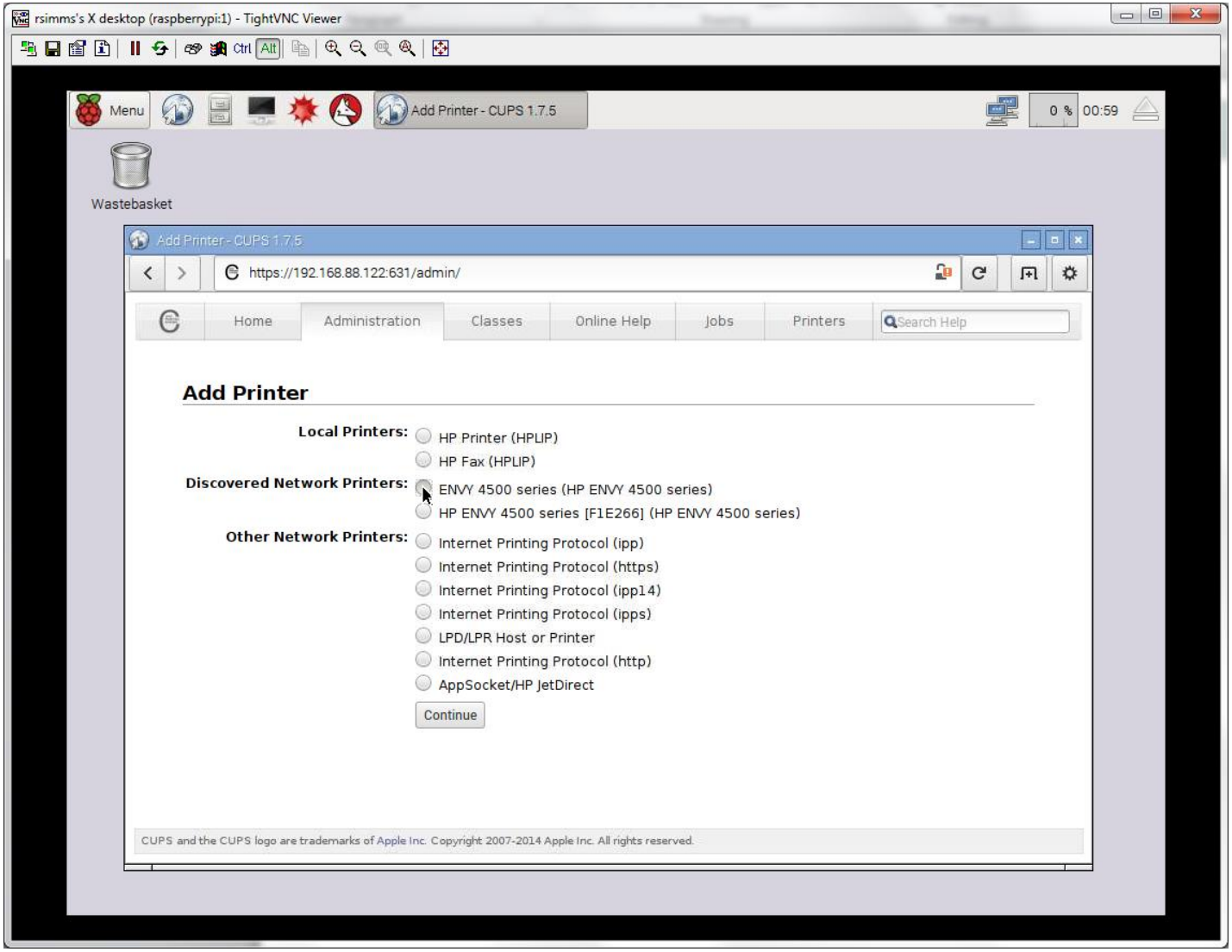
Select Administration tab



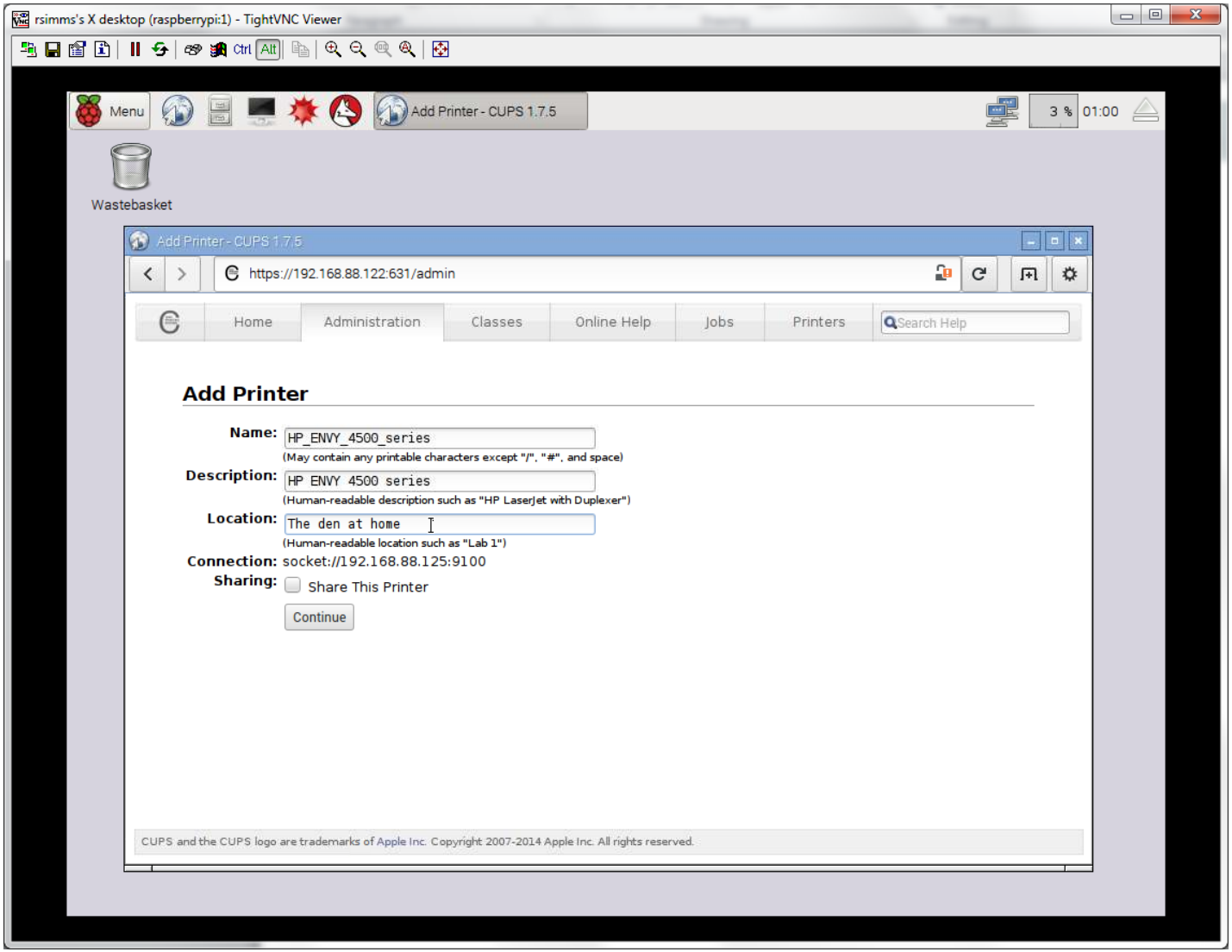
Click Add Printer button and authenticate with user belonging to lpadmin group



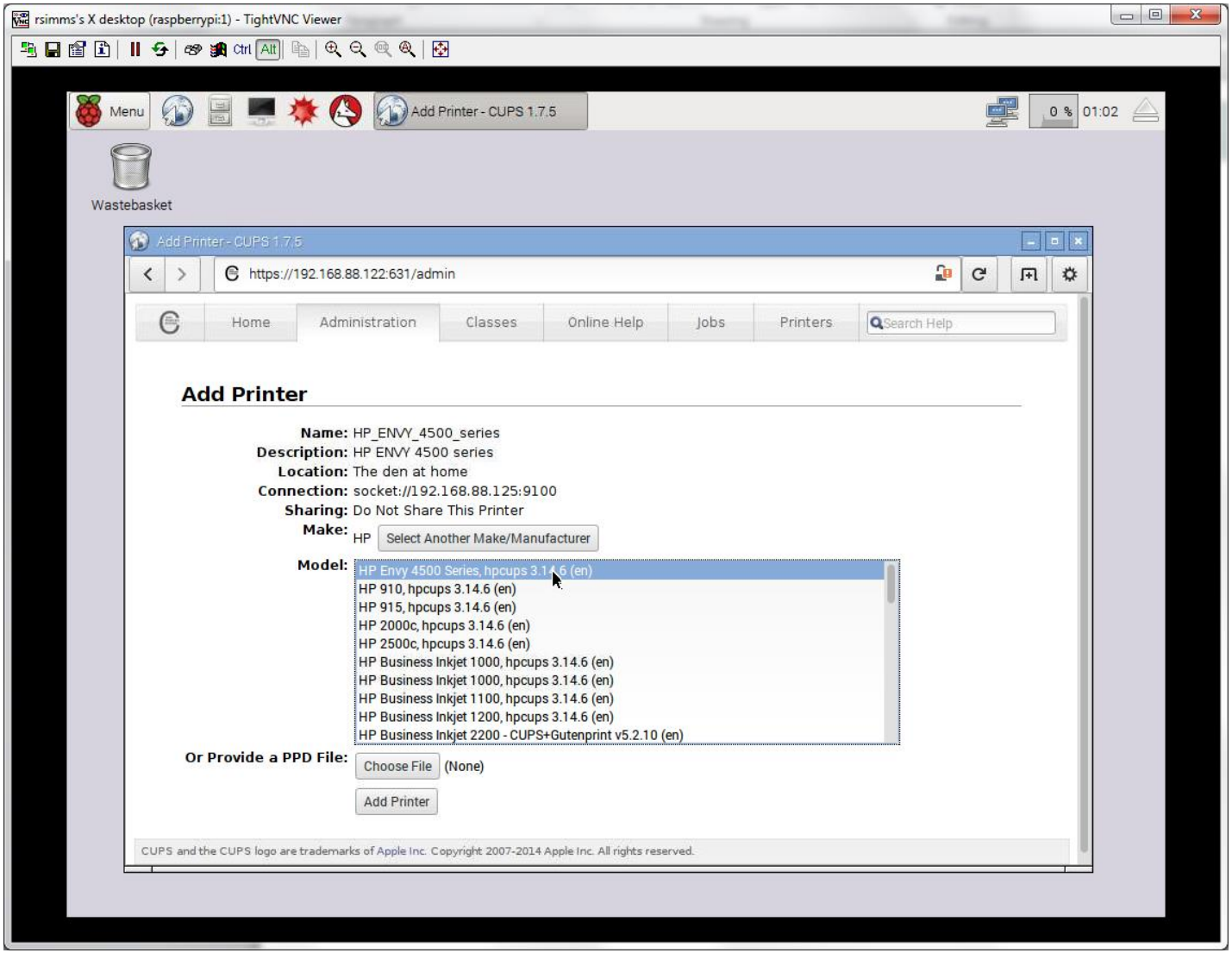
CUPS discovers and displays printers found on network. Select the printer to install.



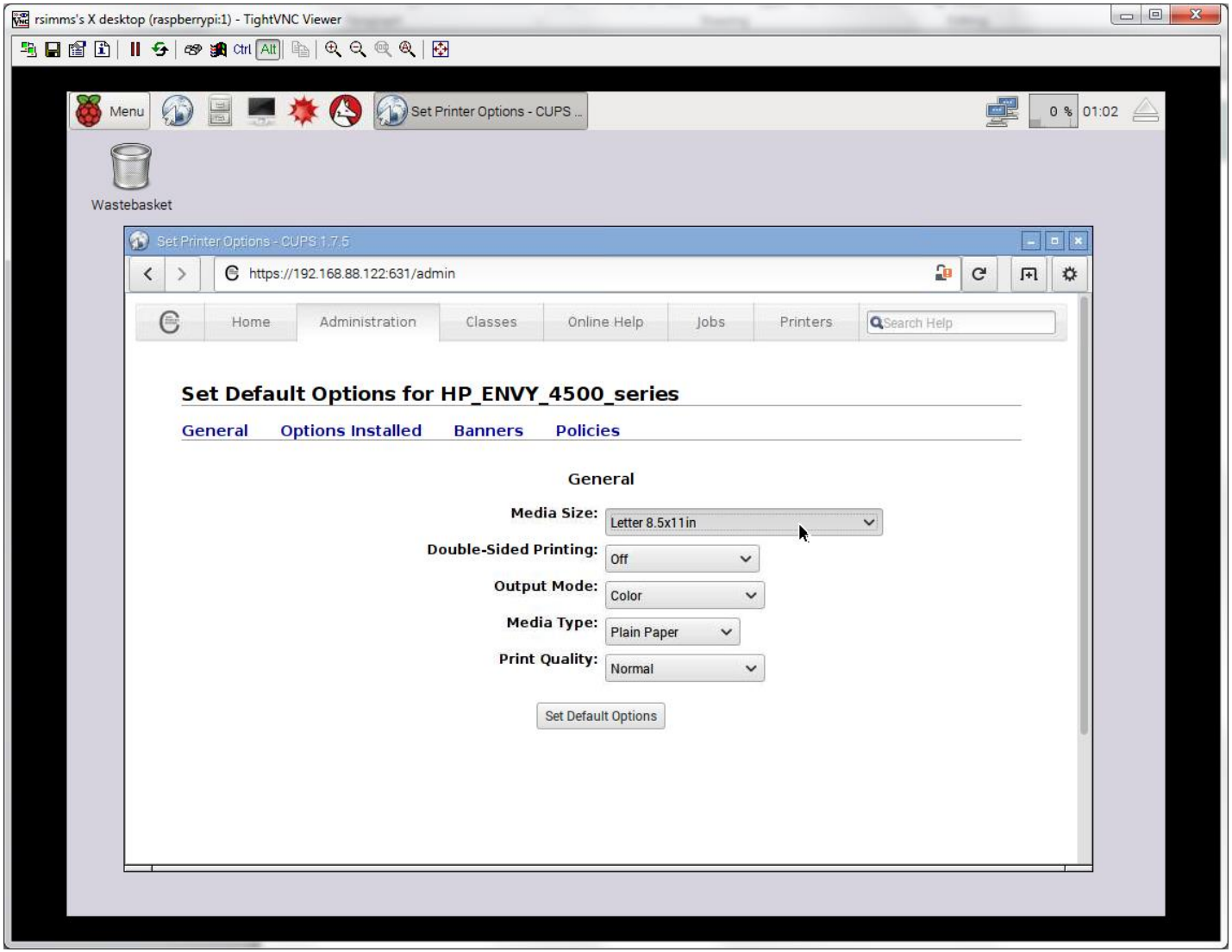
Add some information about the printer



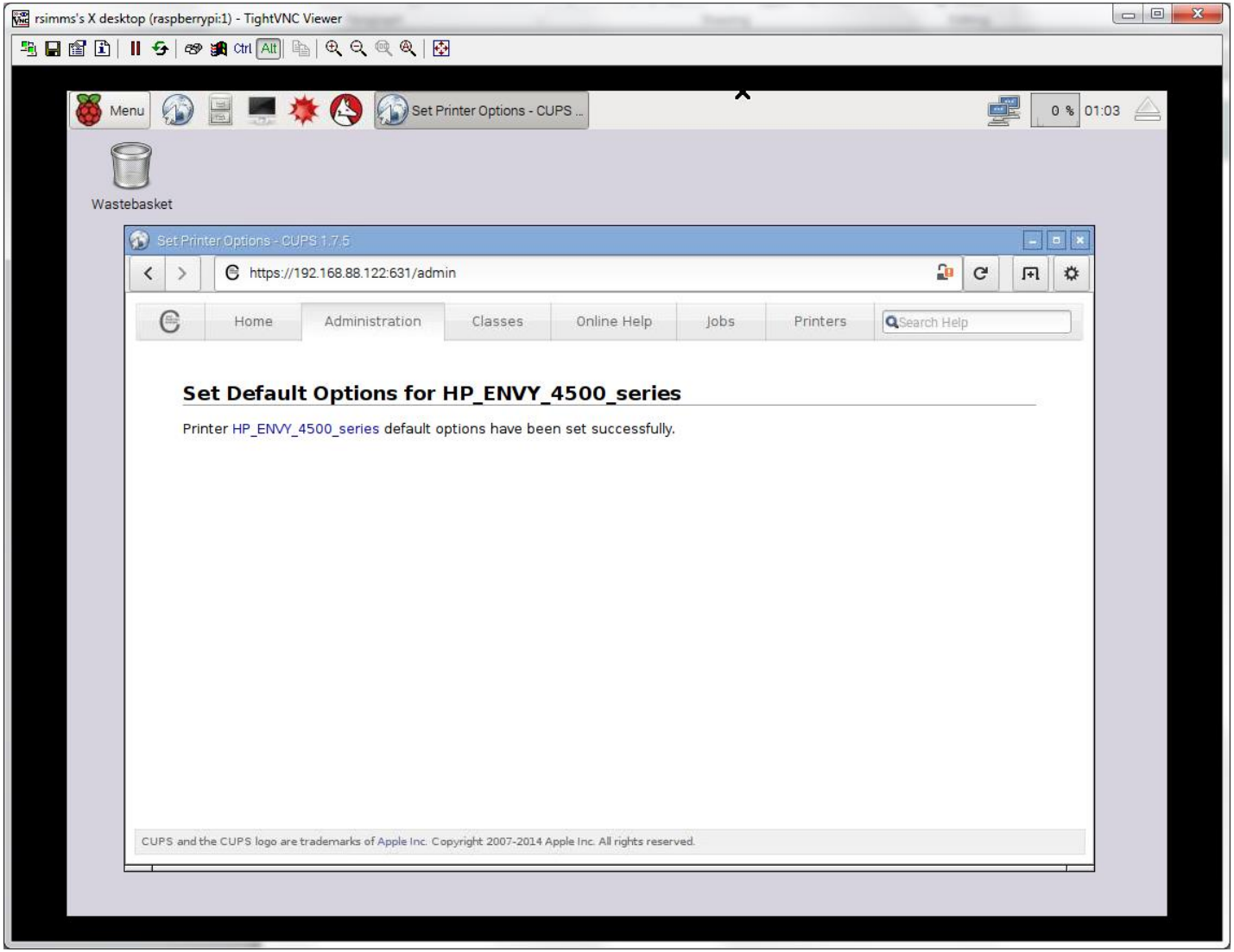
Add the printer



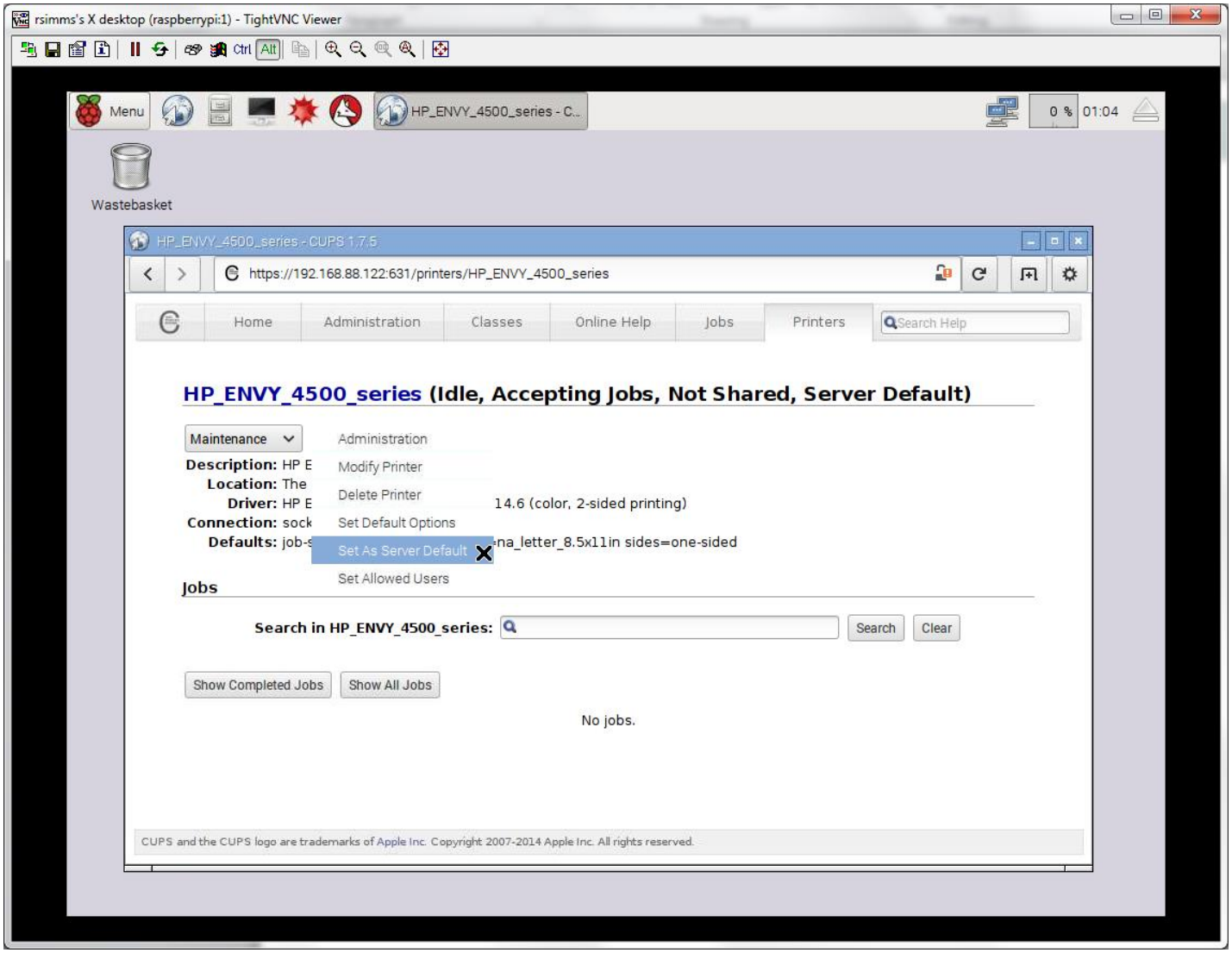
Set printing defaults



Printer added and ready!

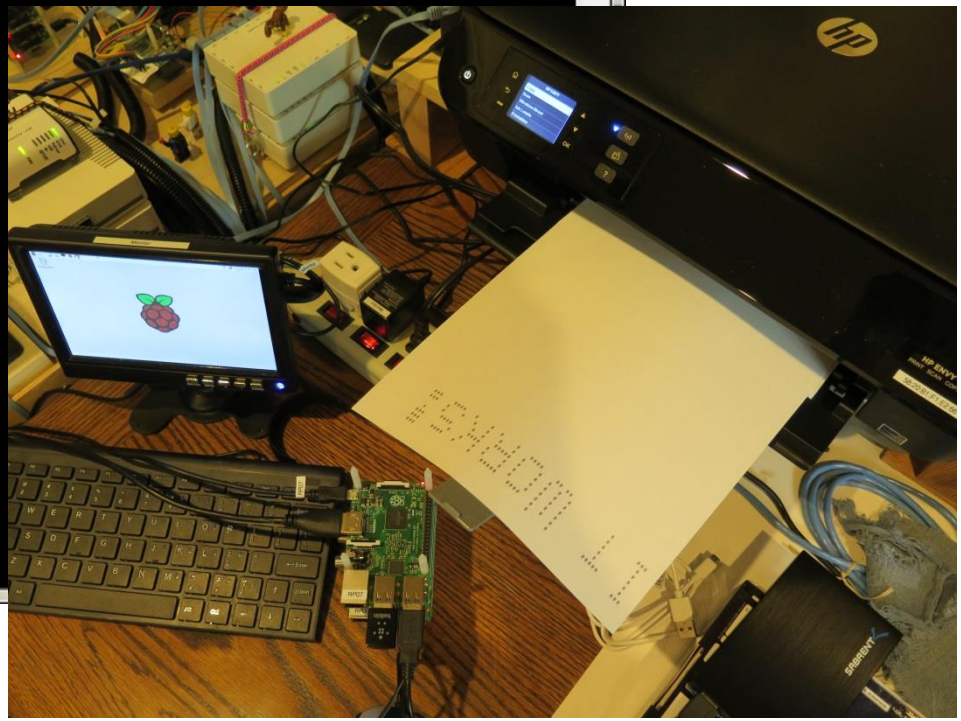


Make it the default printer



Test from the command line to verify it works

```
rsimms@raspberrypi: ~
rsimms@raspberrypi:~ $ banner "It Works!" | lp
request id is HP_ENVY_4500_series-8 (0 file(s))
rsimms@raspberrypi:~ $
```



Photosmart c309n Configuration via CUPS

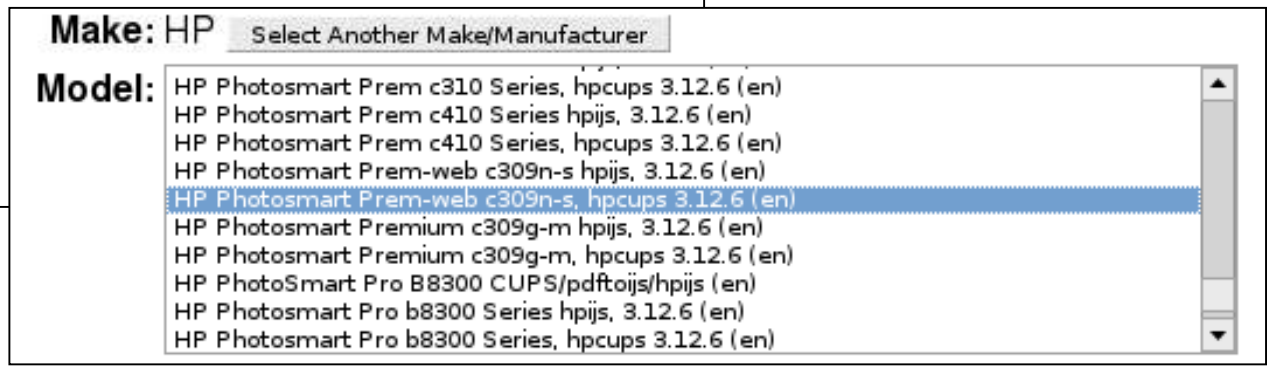
Instructor Configuration Notes

(portwenn) NoPar#**show ip dhcp binding**
HP Photosmart Premium 18:A9:05:01:2D:30 => 172.30.1.xxx
Banana Pi (BP01) 02:d5:09:c0:f0:0f => 172.30.1.xxx

apt-get update
apt-get install tightvncserver
vncserver
apt-get install cups
apt-get install iceweasel

ssh <bp01-ip> 'vncserver'
TightVNC Viewer (www.tightvnc.com, typical install)
(opus) Remote Host: <bp01-ip>:5901

service cups start
Internet > iceweasel
http://localhost:631
AppSocket/HP JetDirect
socket://<printer-ip>:9100



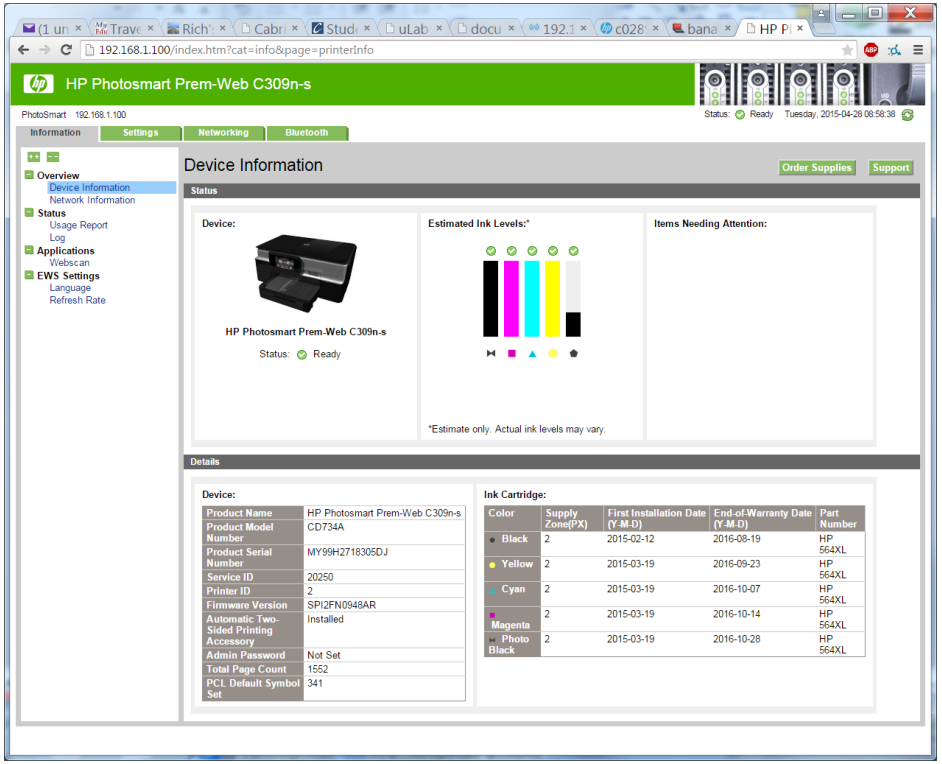
CUPS

Example printer configuration

Printer: HP PhotoSmart Premium C309n-s
Connection: LAN

CUPS

Networked HP printers have a built in web-server



IP Address for this printer is 192.168.1.100

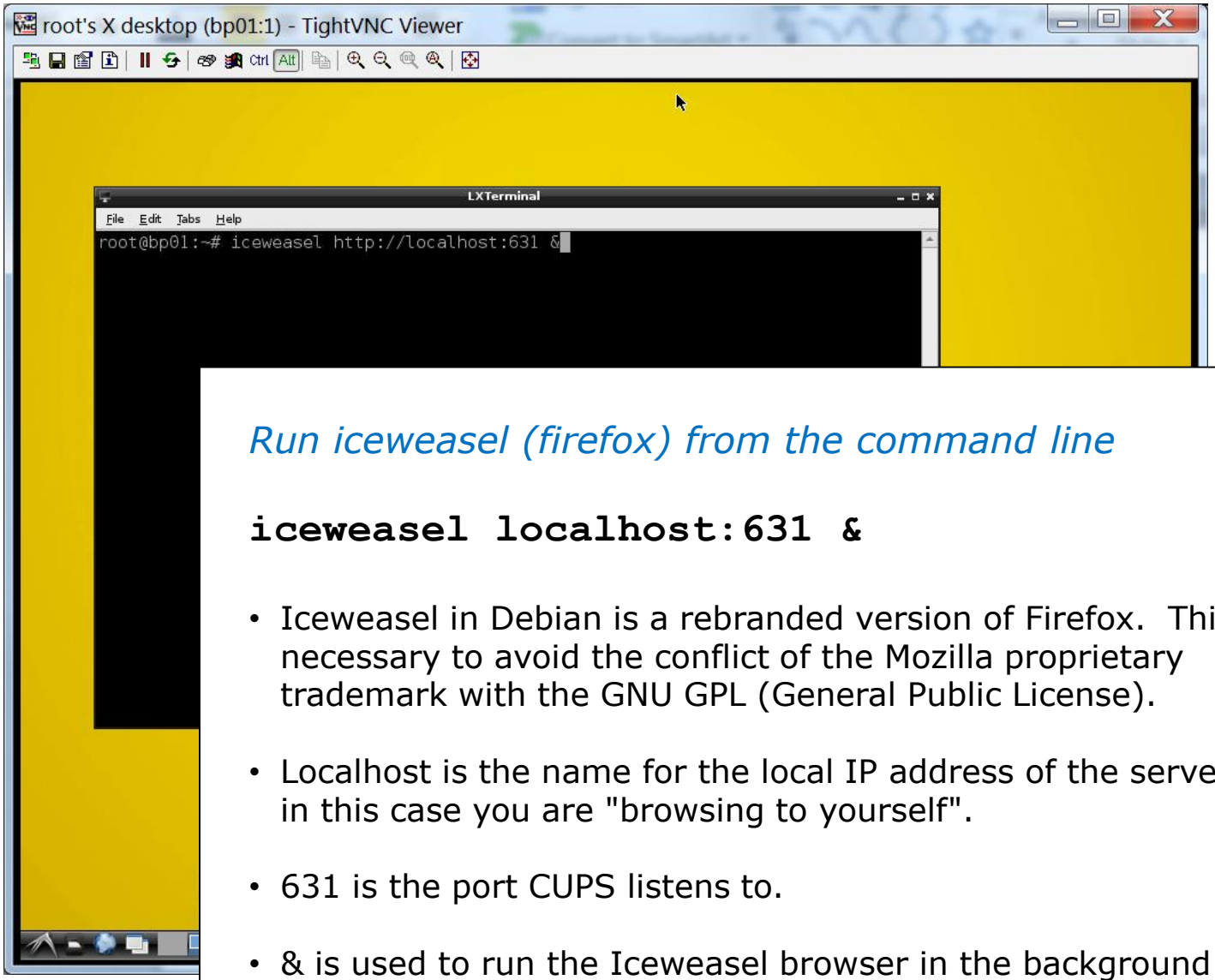
CUPS

FYI
CIS 192

This example will show how to add an HP PhotoSmart Premium C309n-s printer located on a different subnet than the Linux server.

Banana Pi running Raspian GNU/Linux 7 (bp01)
IP: 192.168.88.148

HP PhotoSmart Premium C309n-s (inky)
IP: 192.168.1.100



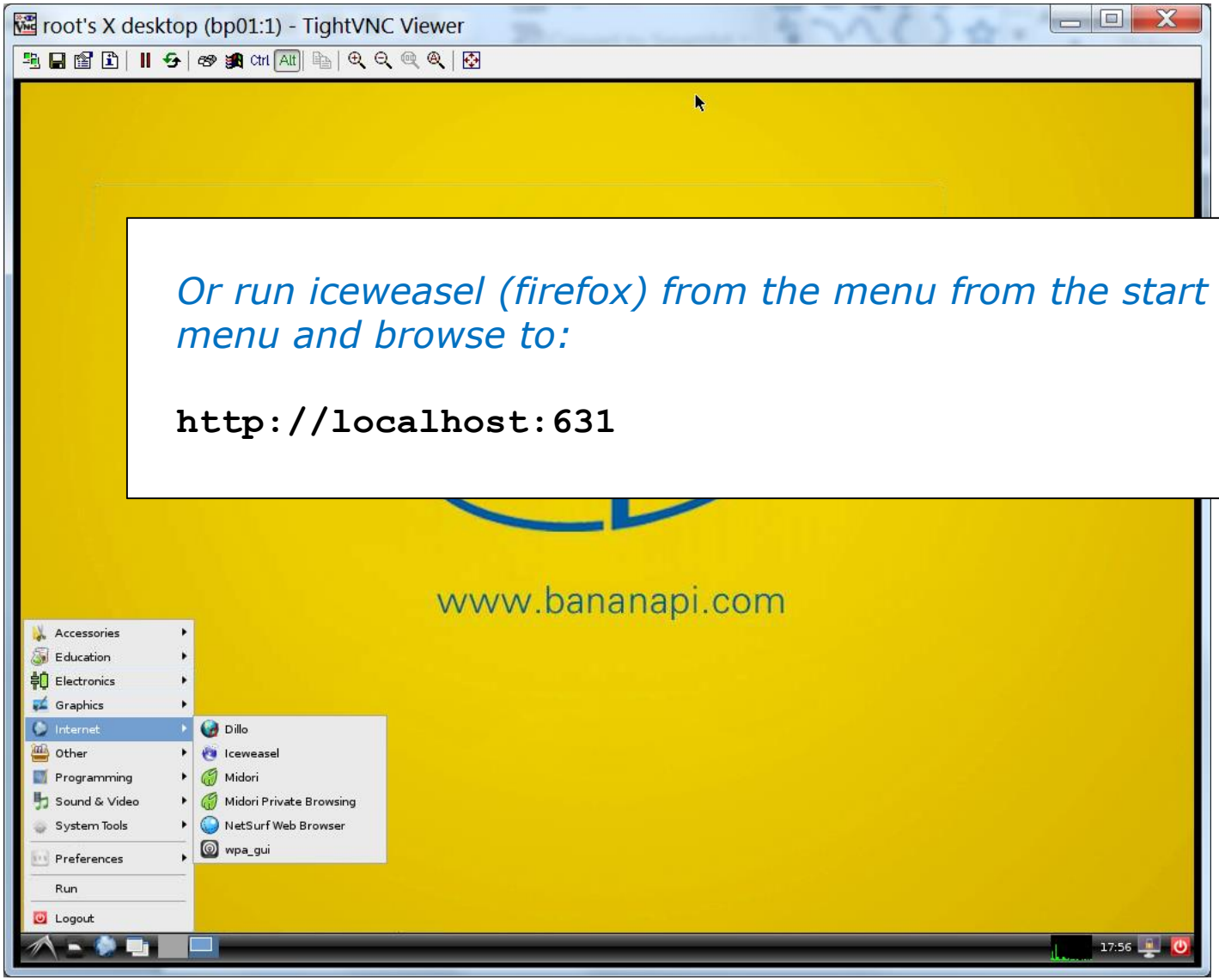
FYI
CIS 192

Run iceweasel (firefox) from the command line

`iceweasel localhost:631 &`

- Iceweasel in Debian is a rebranded version of Firefox. This was necessary to avoid the conflict of the Mozilla proprietary trademark with the GNU GPL (General Public License).
- Localhost is the name for the local IP address of the server since in this case you are "browsing to yourself".
- 631 is the port CUPS listens to.
- & is used to run the Iceweasel browser in the background so we can continue to enter more commands in the terminal session if desired.

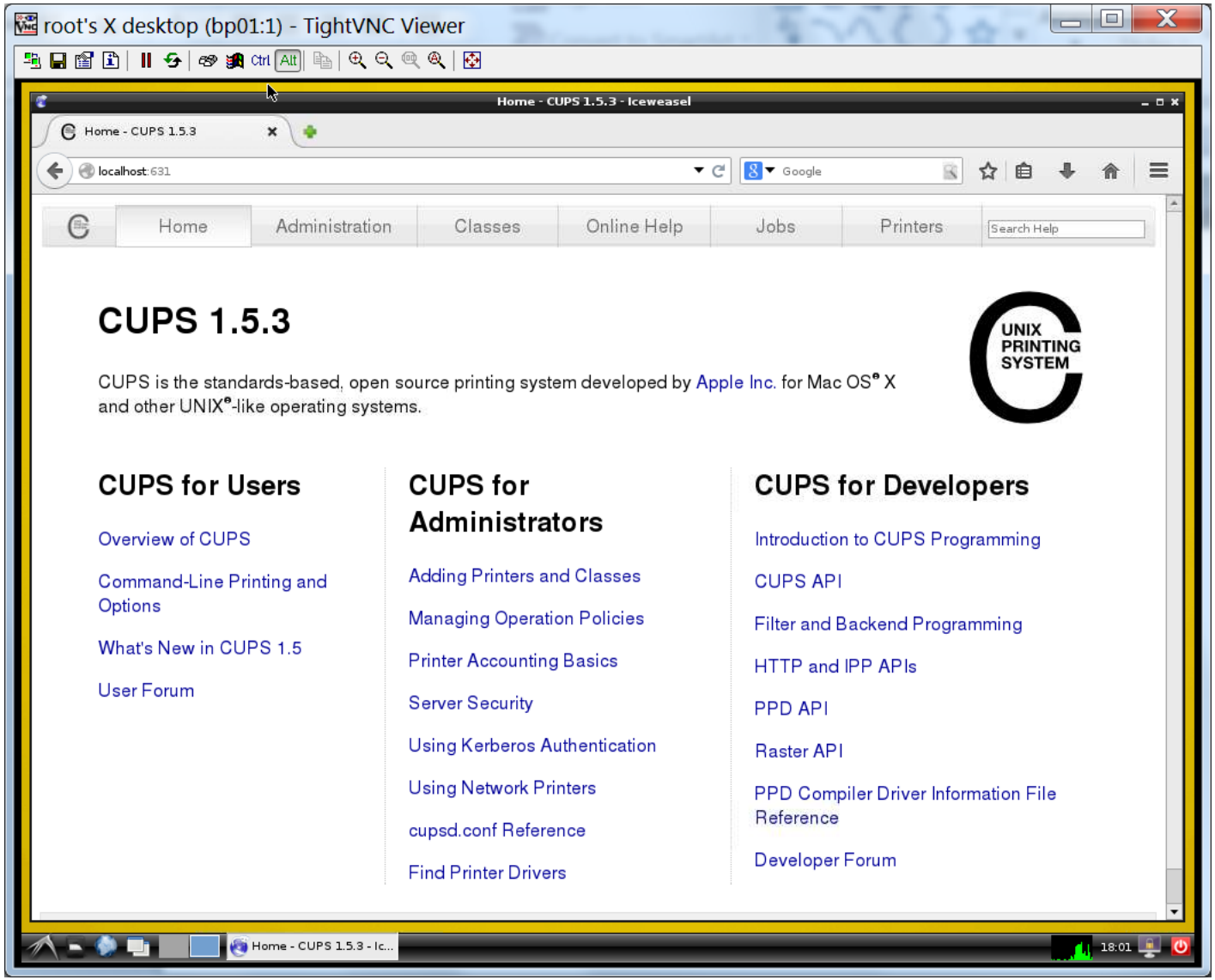
CUPS



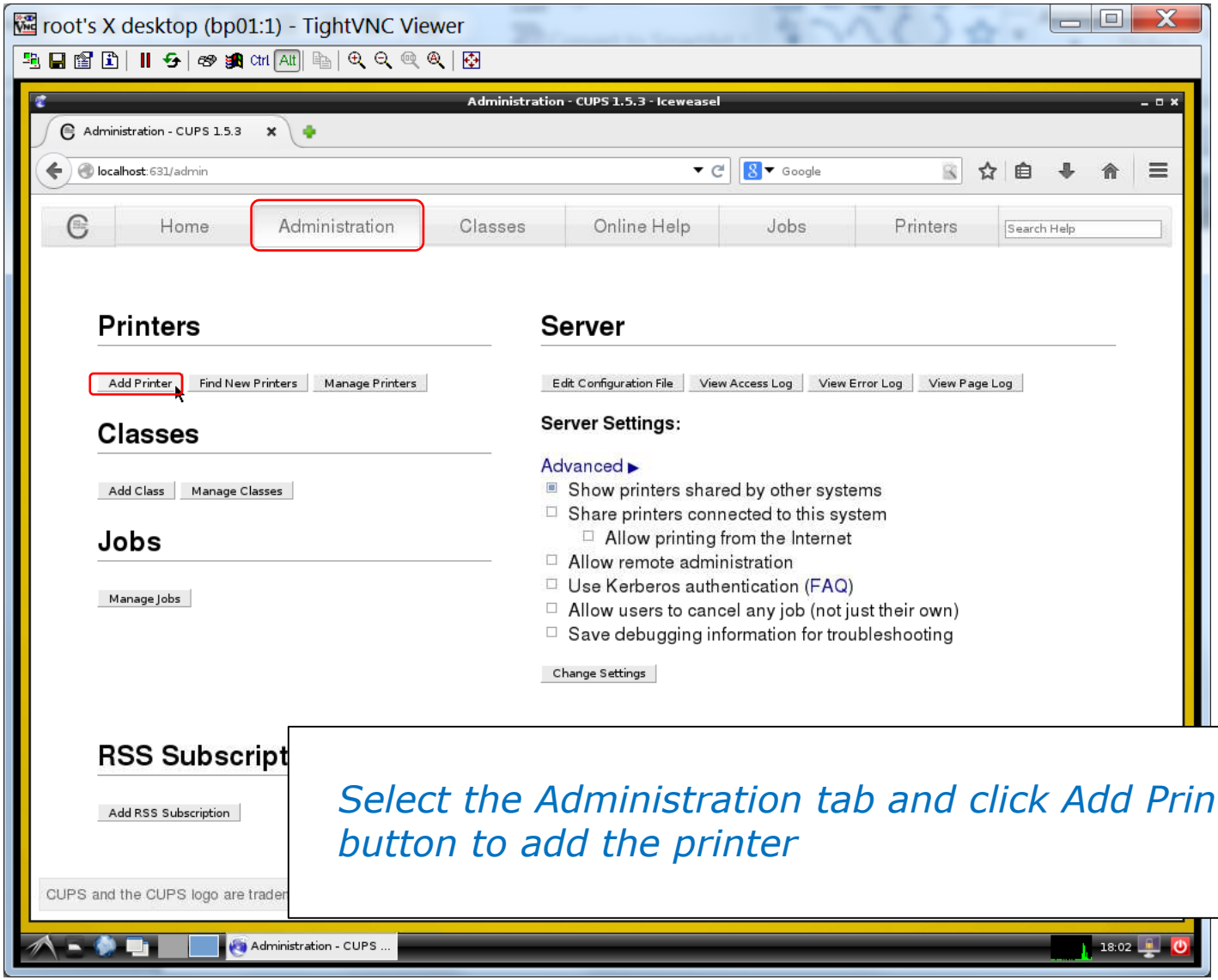
FYI
CIS 192

Or run iceweasel (firefox) from the menu from the start menu and browse to:

`http://localhost:631`

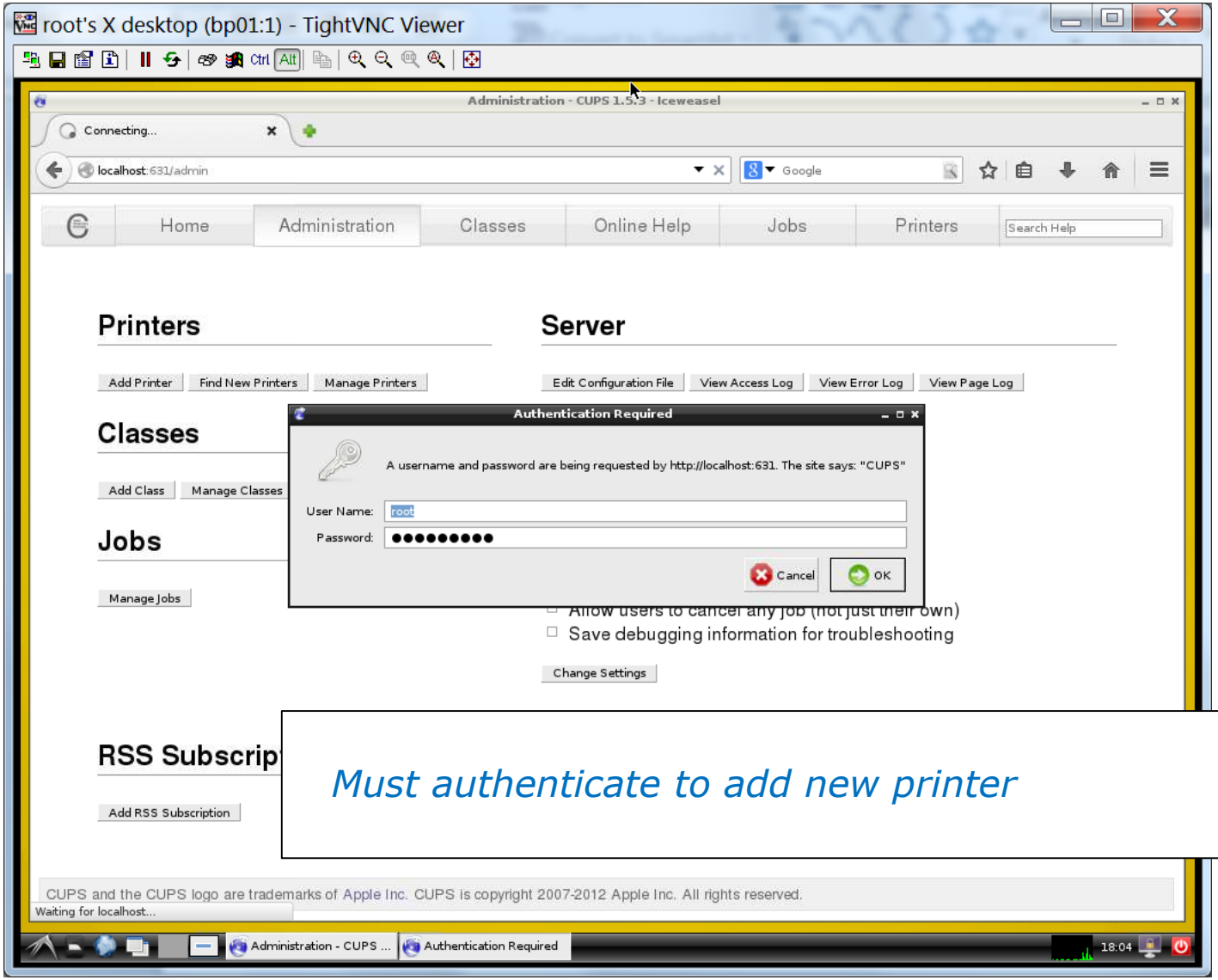

root's X desktop (bp01:1) - TightVNC Viewer
Home - CUPS 1.5.3 - Iceweasel
Home - CUPS 1.5.3
localhost:631
Google
Home
Administration
Classes
Online Help
Jobs
Printers
Search Help
CUPS 1.5.3
CUPS is the standards-based, open source printing system developed by Apple Inc. for Mac OS® X and other UNIX®-like operating systems.
UNIX PRINTING SYSTEM
CUPS for Users
Overview of CUPS
Command-Line Printing and Options
What's New in CUPS 1.5
User Forum
CUPS for Administrators
Adding Printers and Classes
Managing Operation Policies
Printer Accounting Basics
Server Security
Using Kerberos Authentication
Using Network Printers
cupsd.conf Reference
Find Printer Drivers
CUPS for Developers
Introduction to CUPS Programming
CUPS API
Filter and Backend Programming
HTTP and IPP APIs
PPD API
Raster API
PPD Compiler Driver Information File Reference
Developer Forum
Home - CUPS 1.5.3 - Ic...
18:01

FYI CIS 192

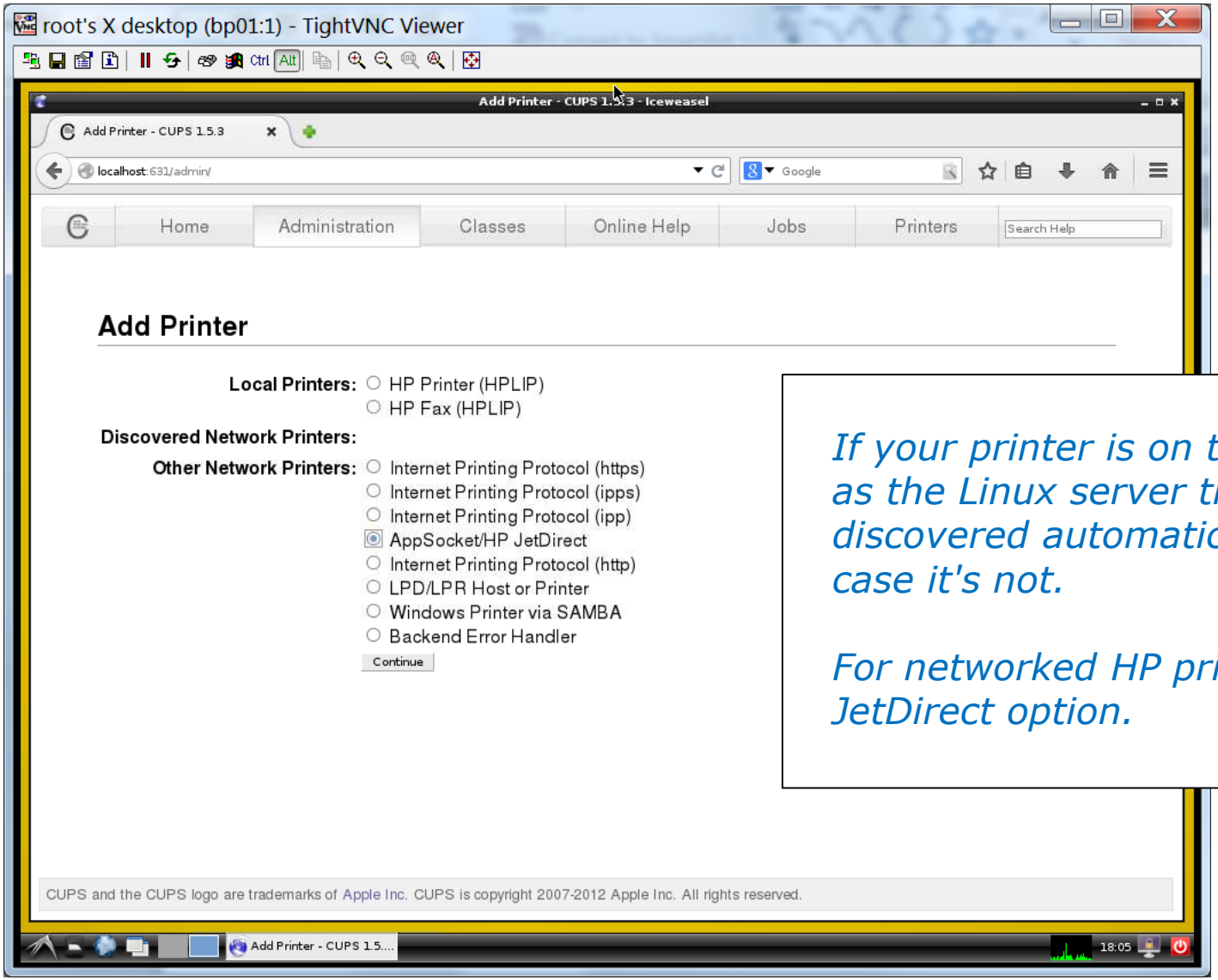


Select the Administration tab and click Add Printer button to add the printer

FYI CIS 192

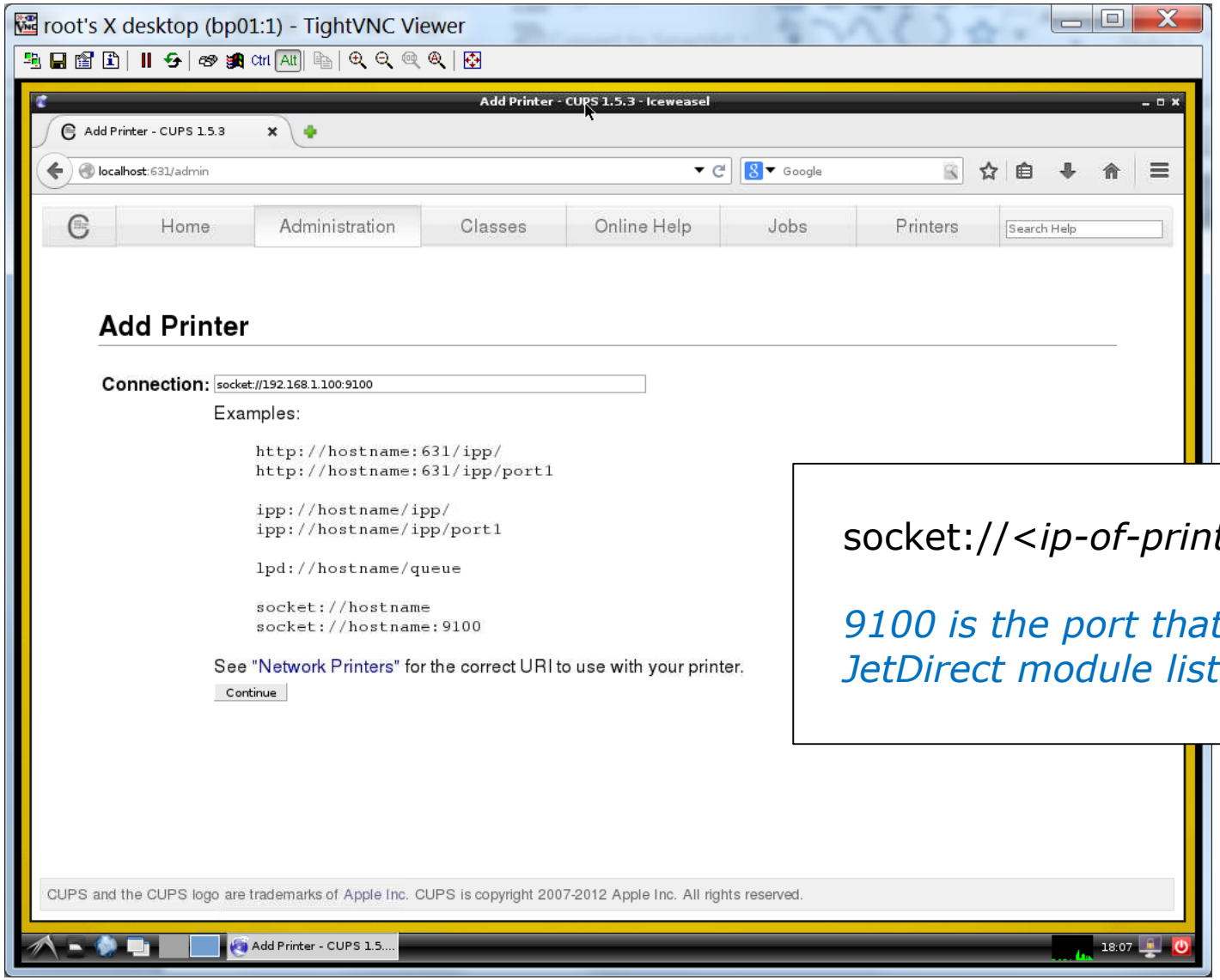


Must authenticate to add new printer



If your printer is on the same subnet as the Linux server then it will be discovered automatically. In this case it's not.

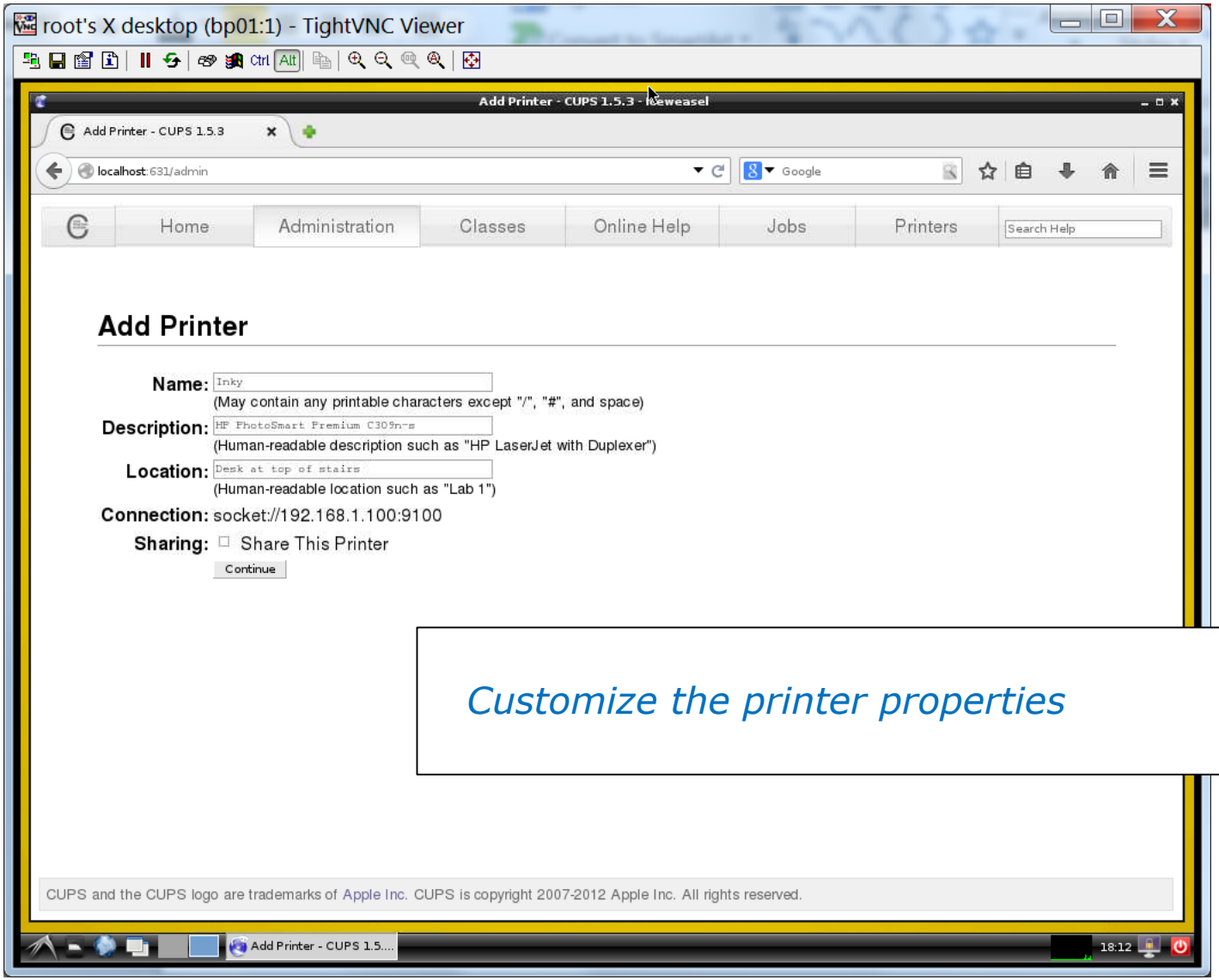
For networked HP printers select the JetDirect option.



FYI
CIS 192

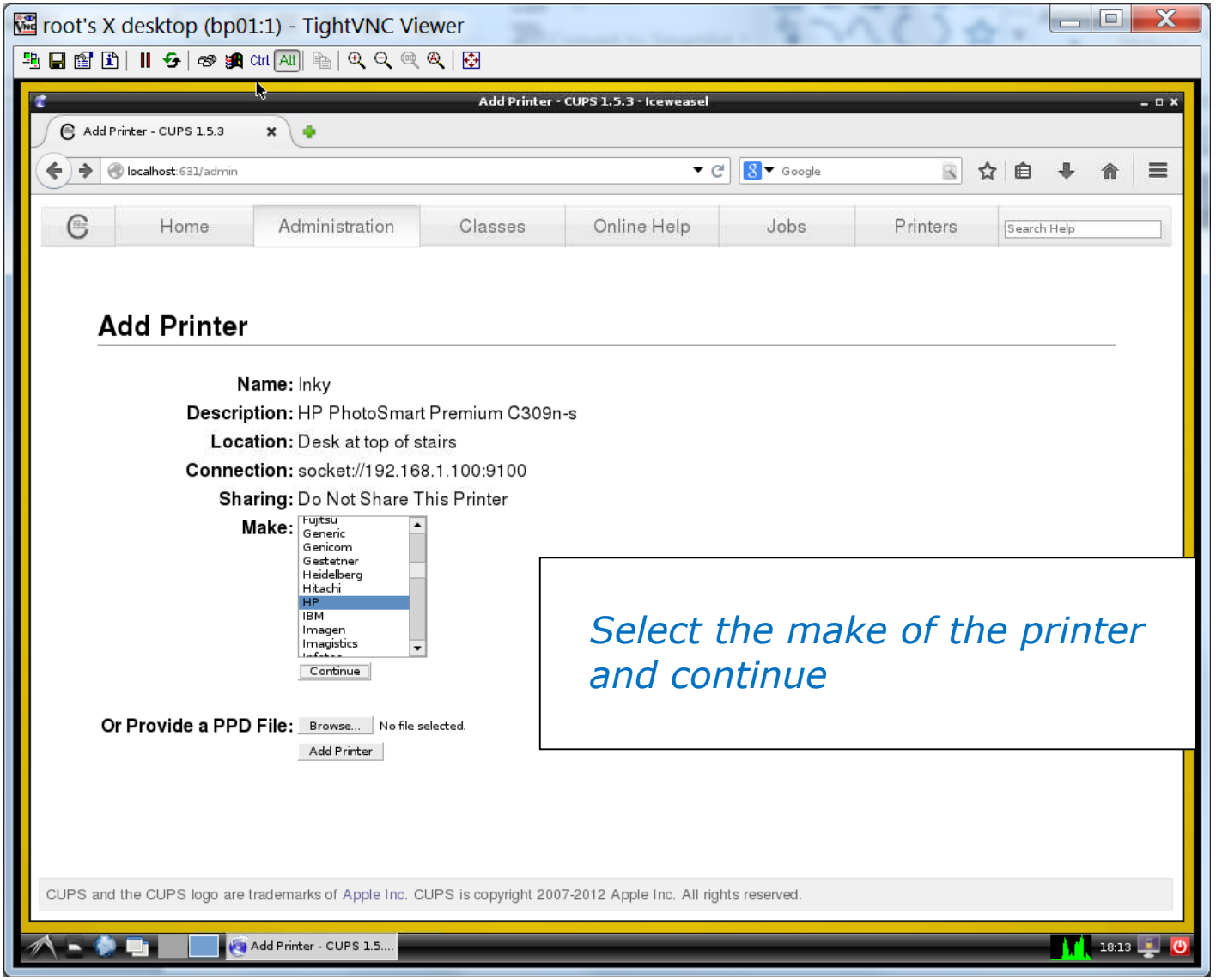
socket://<*ip-of-printer*>:9100

9100 is the port that the HP JetDirect module listens to



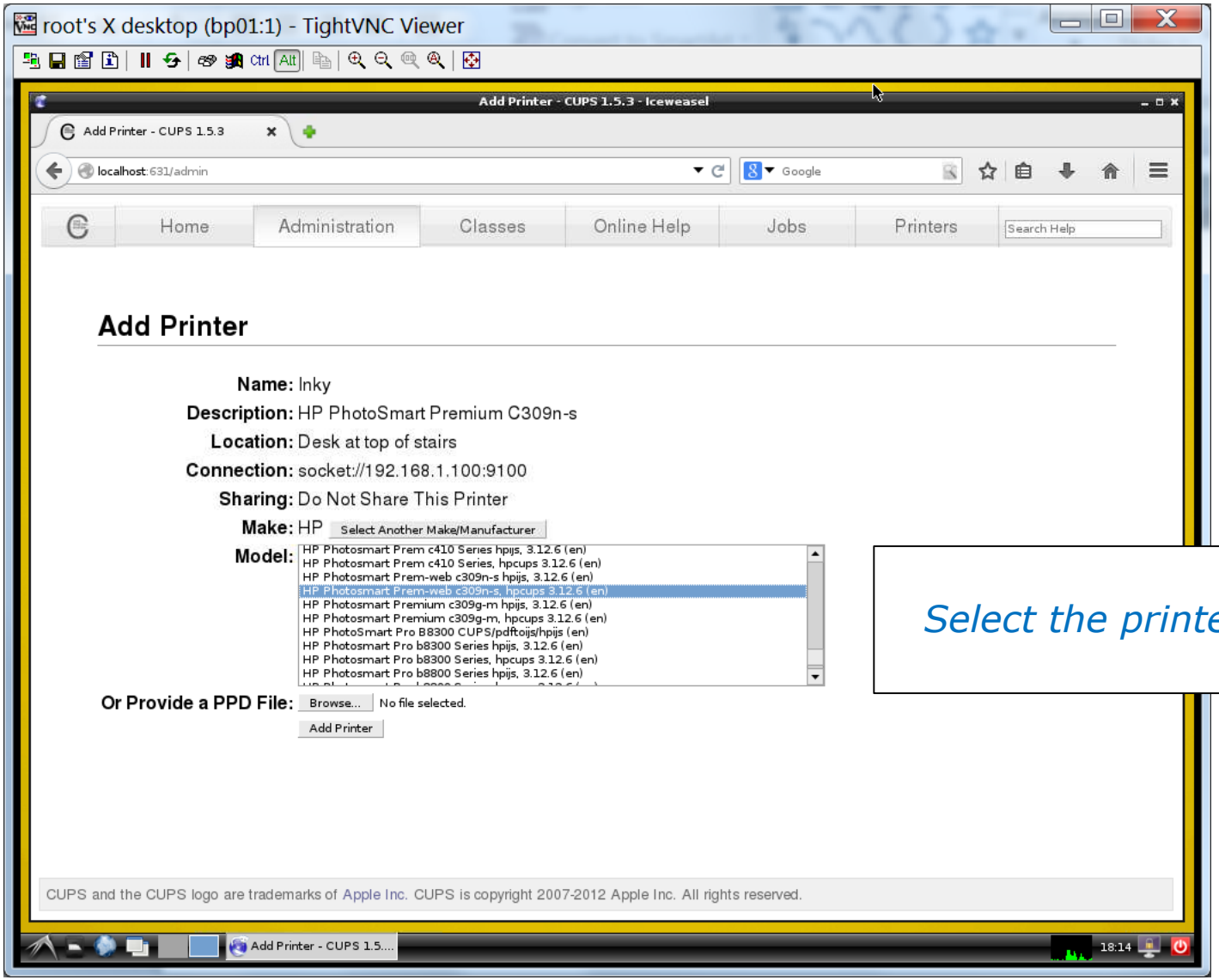
FYI
CIS 192

Customize the printer properties



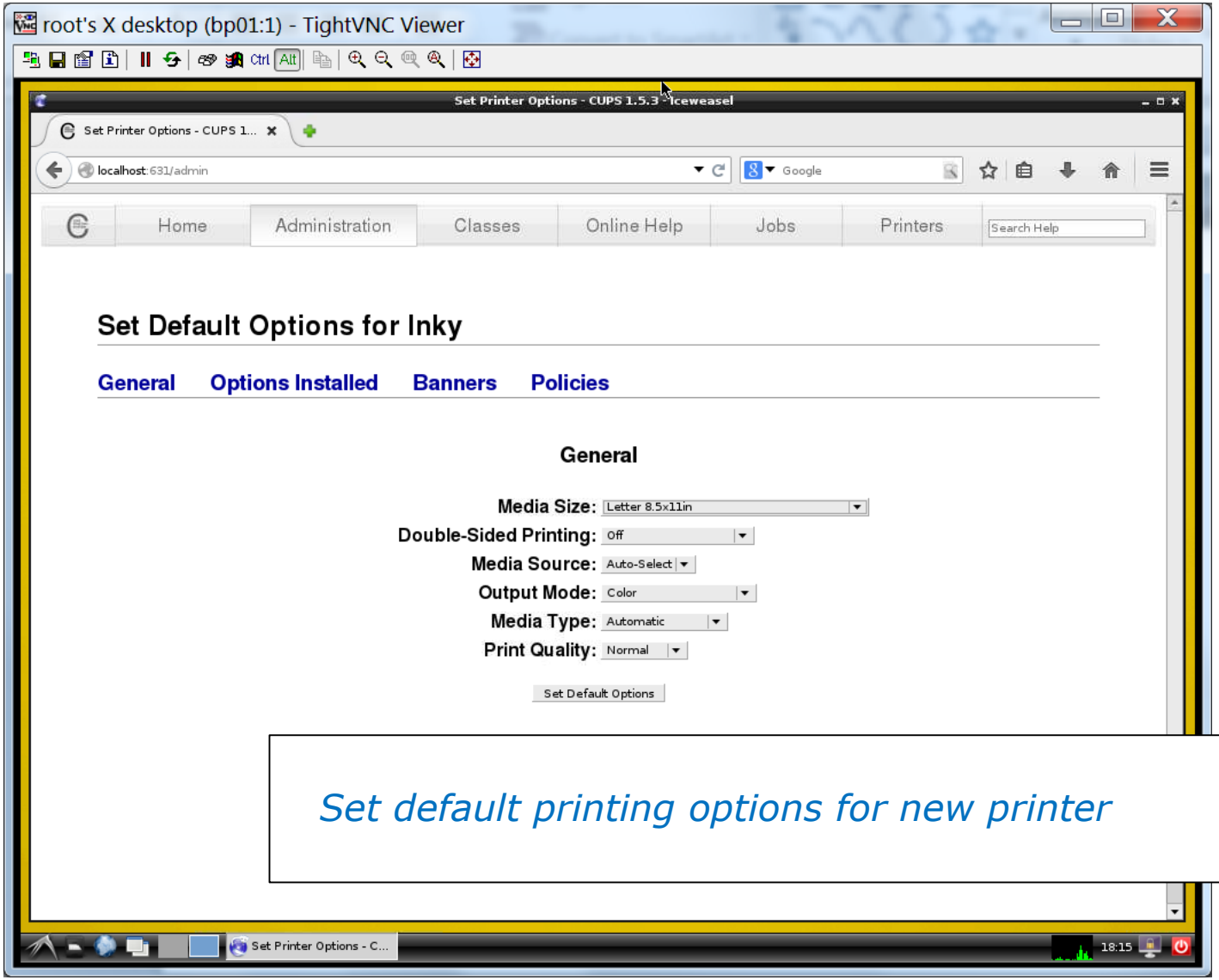
Select the make of the printer and continue

FYI
CIS 192



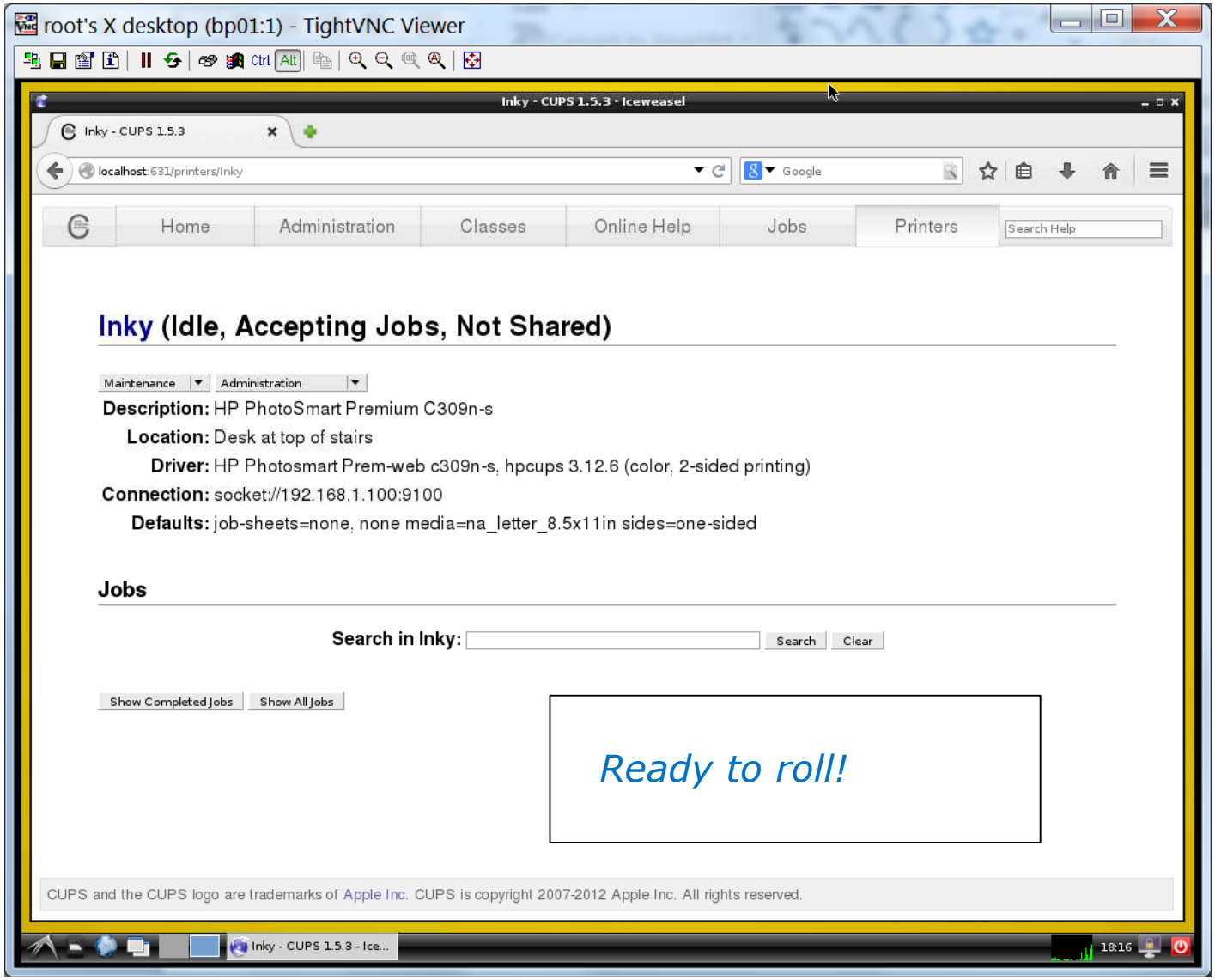
FYI
CIS 192

Select the printer driver



FYI
CIS 192

Set default printing options for new printer


root's X desktop (bp01:1) - TightVNC Viewer
Inky - CUPS 1.5.3 - Iceweasel
Inky - CUPS 1.5.3
localhost:631/printers/Inky
Google
Home
Administration
Classes
Online Help
Jobs
Printers
Search Help
Inky (Idle, Accepting Jobs, Not Shared)
Maintenance
Administration
Description: HP PhotoSmart Premium C309n-s
Location: Desk at top of stairs
Driver: HP Photosmart Prem-web c309n-s, hpcups 3.12.6 (color, 2-sided printing)
Connection: socket://192.168.1.100:9100
Defaults: job-sheets=none, none media=na_letter_8.5x11in sides=one-sided
Jobs
Search in Inky:
Search
Clear
Show Completed Jobs
Show All Jobs
Ready to roll!
CUPS and the CUPS logo are trademarks of Apple Inc. CUPS is copyright 2007-2012 Apple Inc. All rights reserved.
Inky - CUPS 1.5.3 - Ice...
18:16


FYI
CIS 192

LaserJet 1320n Configuration via CUPS

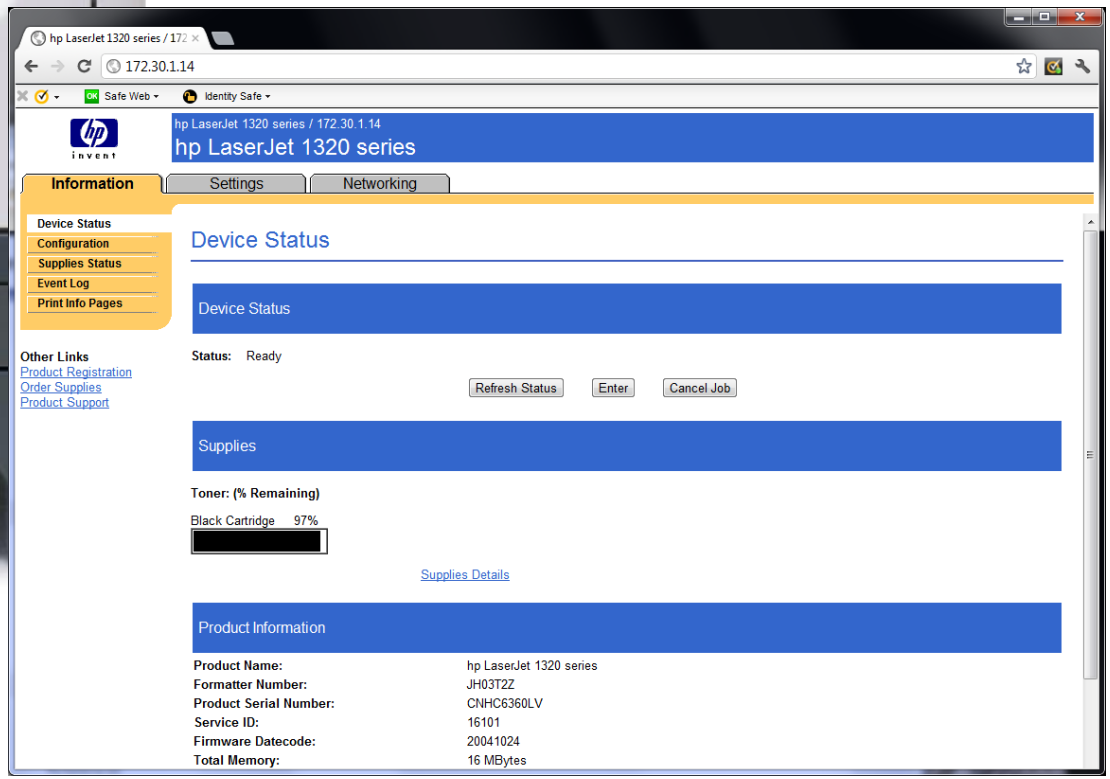
CUPS

Example printer configuration

Printer: HP LaserJet 1320n
Connection: LAN

CUPS

The LaserJets have a web-based management utility

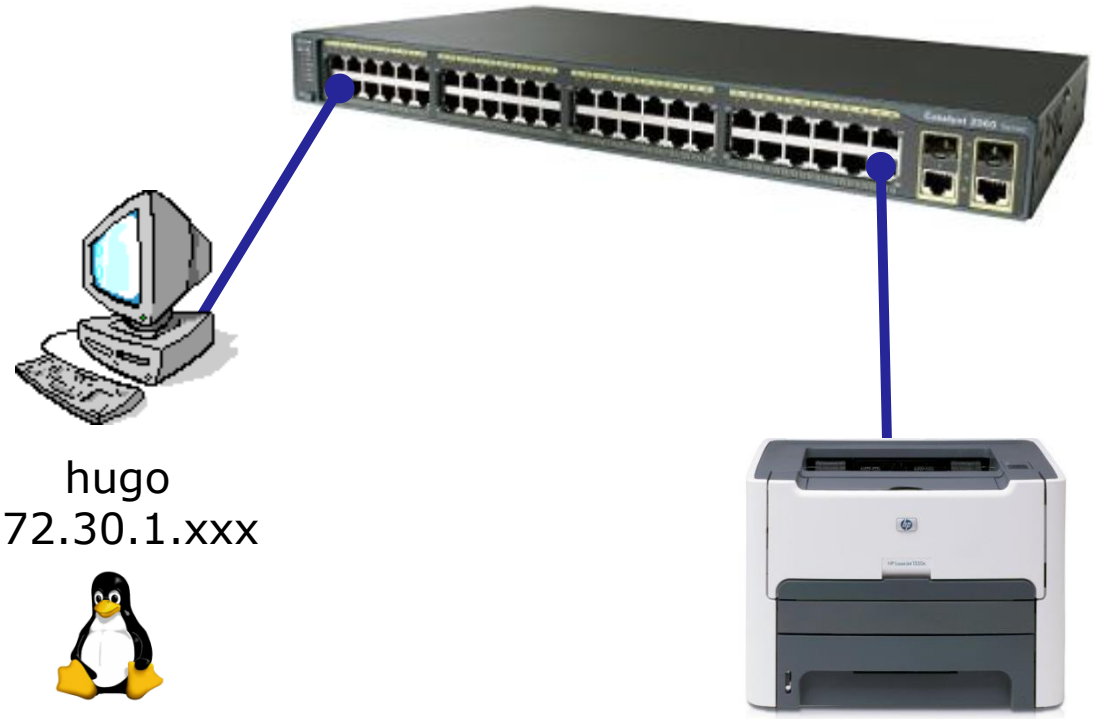


IP Address for this 1320n is 172.30.1.14

CUPS

FYI
CIS 192

This example will show how to add the HP 1320n as a networked printer.

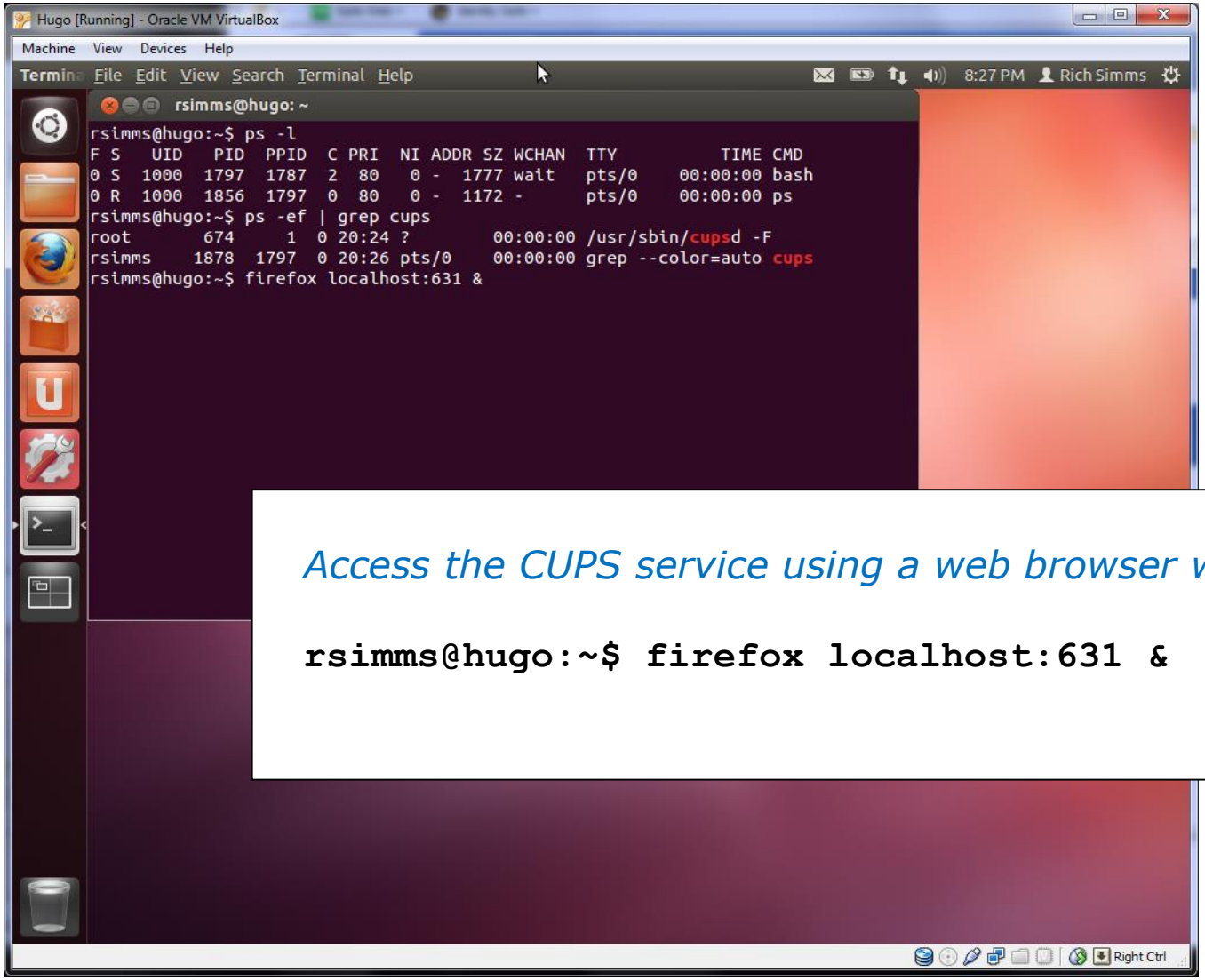


hugo
172.30.1.xxx

hp1320n
172.30.1.14

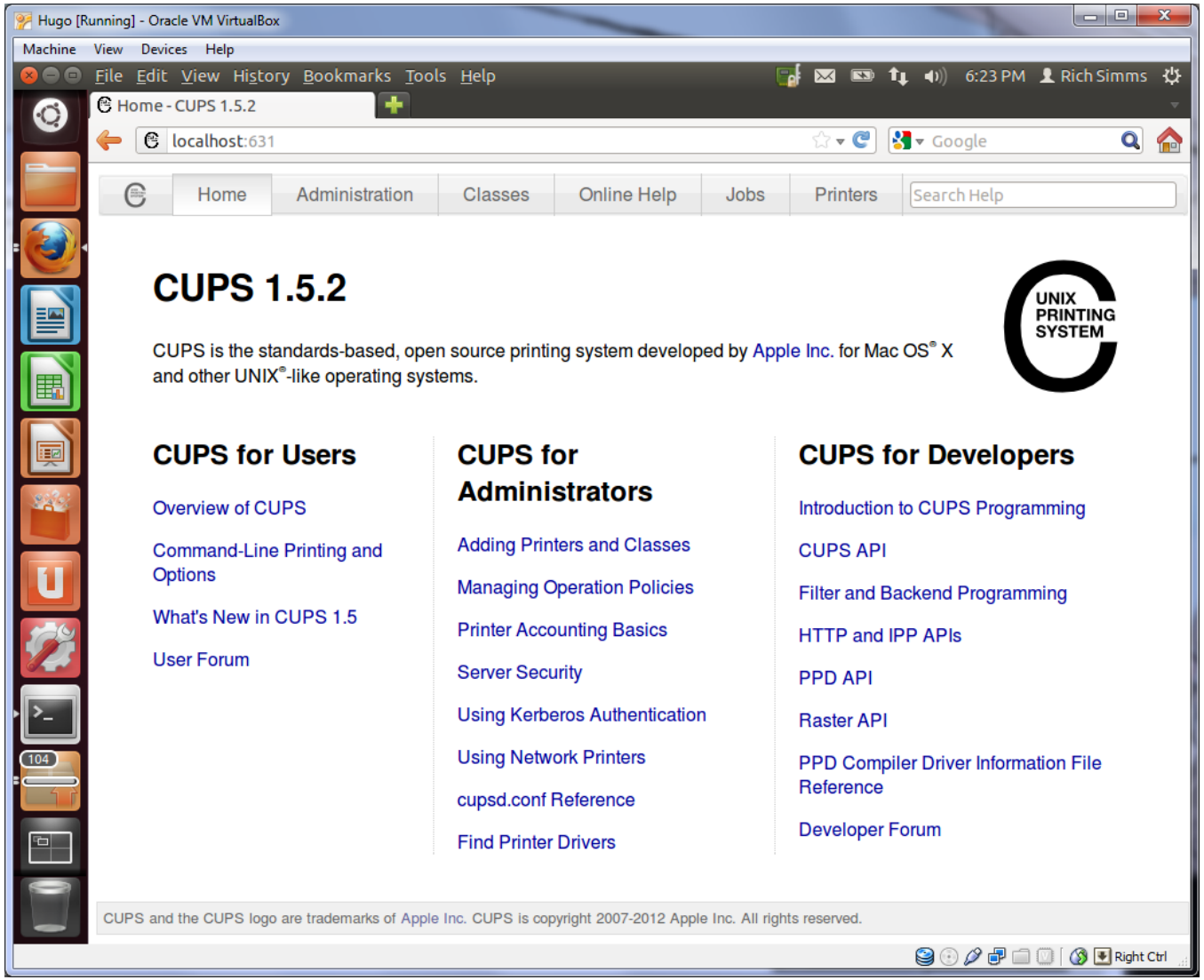
CUPS

FYI
CIS 192



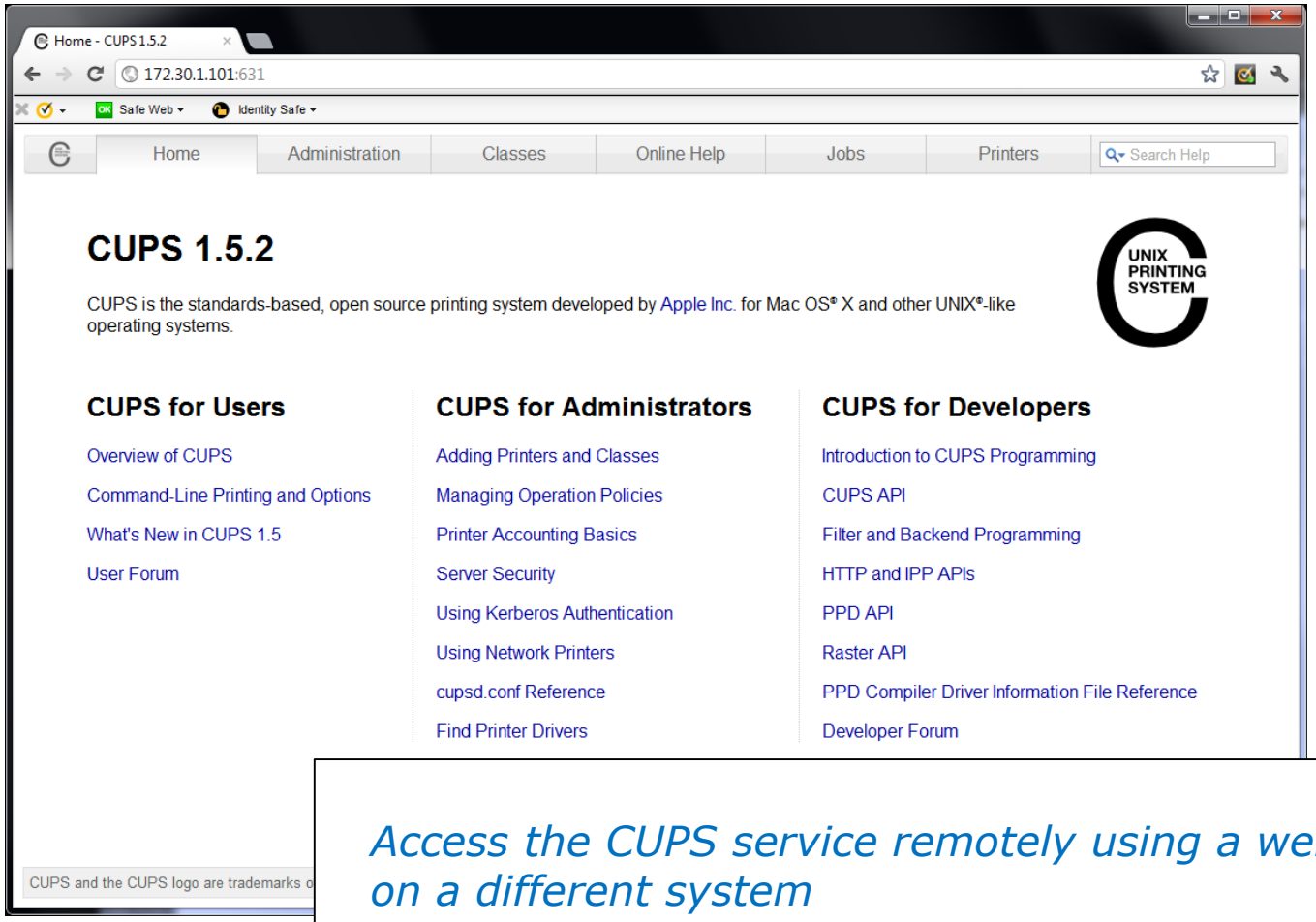
Access the CUPS service using a web browser with

```
rsimms@hugo:~$ firefox localhost:631 &
```


Hugo [Running] - Oracle VM VirtualBox
Machine View Devices Help
File Edit View History Bookmarks Tools Help
6:23 PM Rich Simms
Home - CUPS 1.5.2
localhost:631
Google
Home Administration Classes Online Help Jobs Printers
Search Help
CUPS 1.5.2
CUPS is the standards-based, open source printing system developed by Apple Inc. for Mac OS® X and other UNIX®-like operating systems.
UNIX PRINTING SYSTEM
CUPS for Users
Overview of CUPS
Command-Line Printing and Options
What's New in CUPS 1.5
User Forum
CUPS for Administrators
Adding Printers and Classes
Managing Operation Policies
Printer Accounting Basics
Server Security
Using Kerberos Authentication
Using Network Printers
cupsd.conf Reference
Find Printer Drivers
CUPS for Developers
Introduction to CUPS Programming
CUPS API
Filter and Backend Programming
HTTP and IPP APIs
PPD API
Raster API
PPD Compiler Driver Information File Reference
Developer Forum
CUPS and the CUPS logo are trademarks of Apple Inc. CUPS is copyright 2007-2012 Apple Inc. All rights reserved.
Right Ctrl

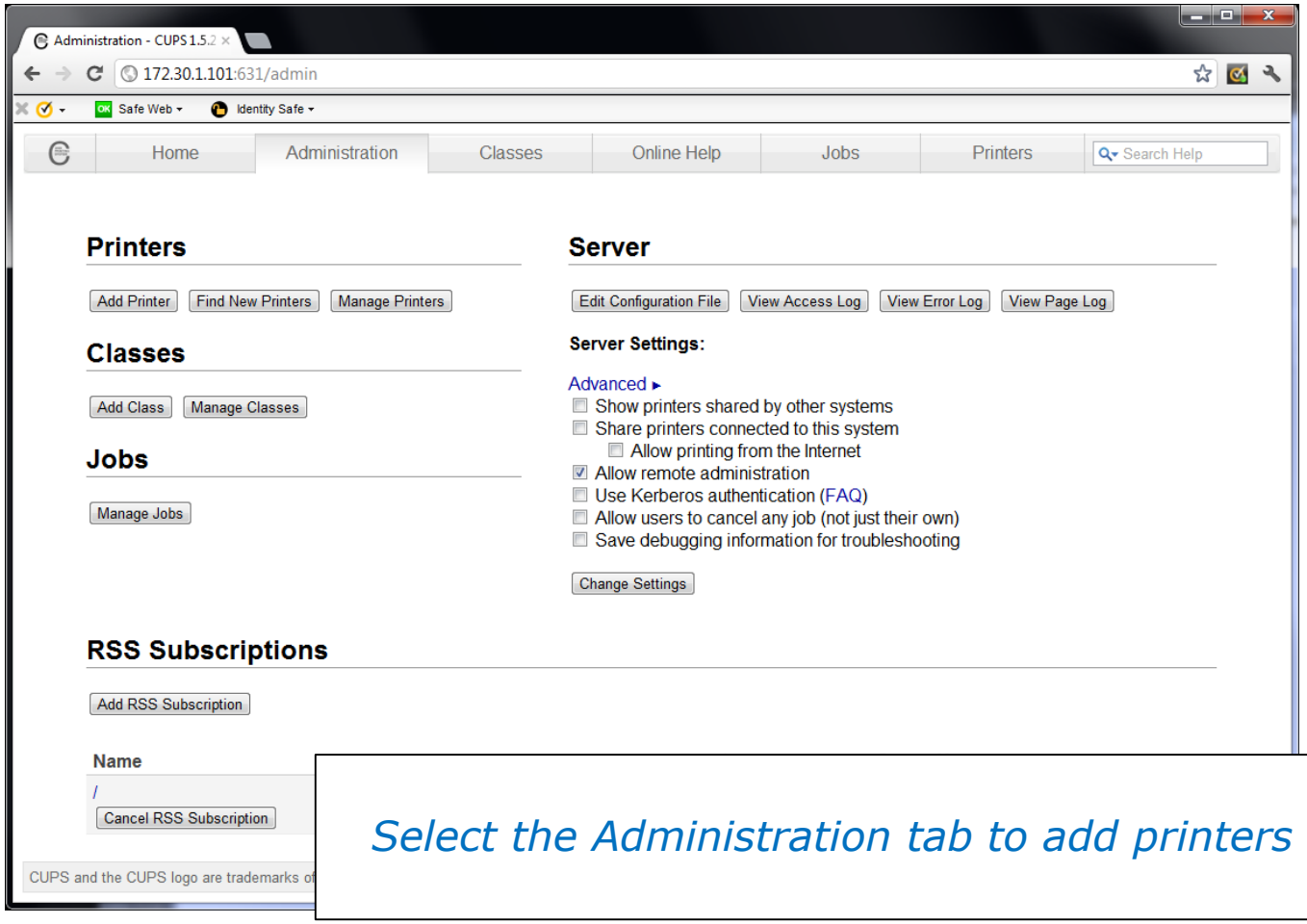
FYI CIS 192

FYI CIS 192

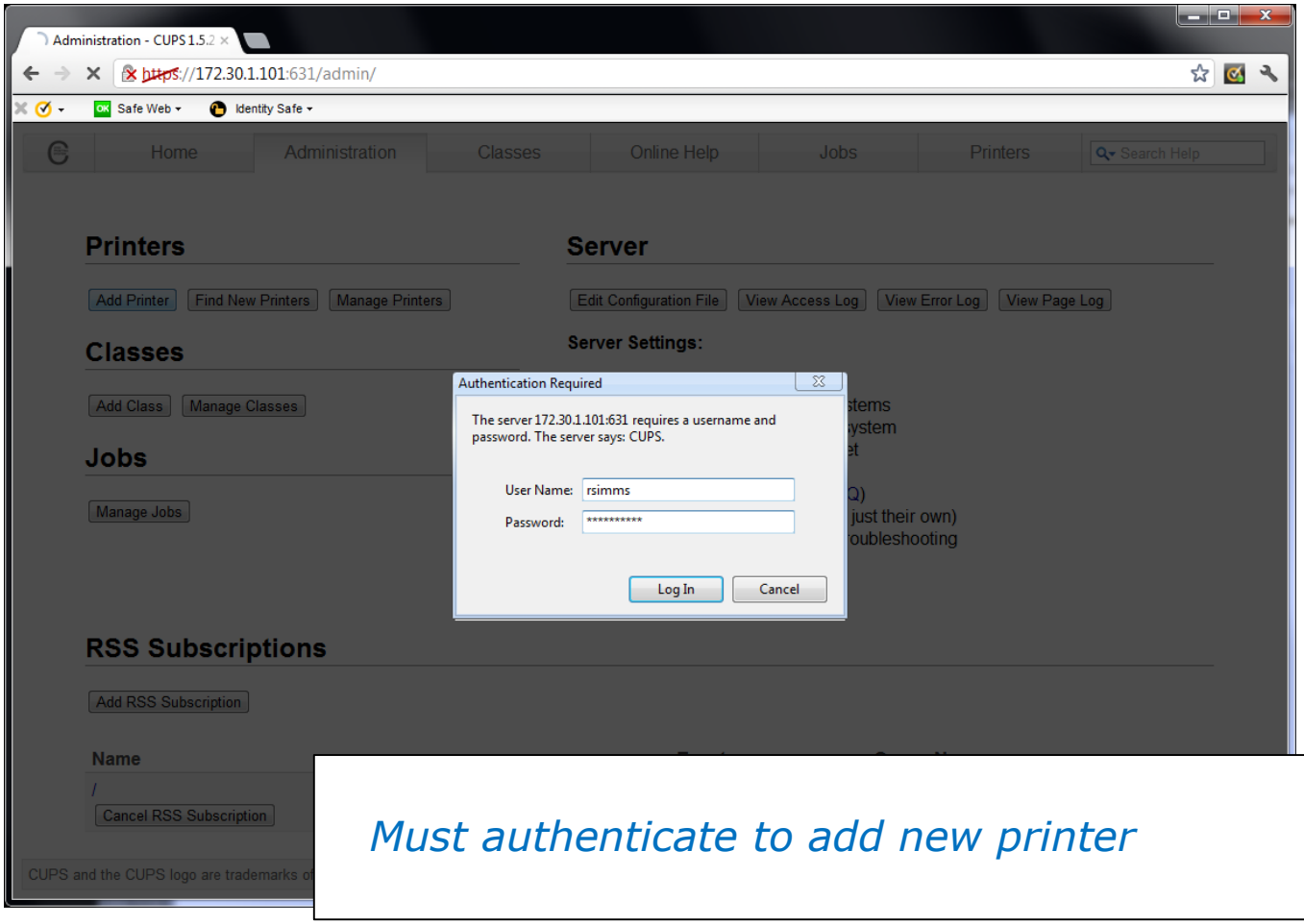


Access the CUPS service remotely using a web browser on a different system

FYI CIS 192



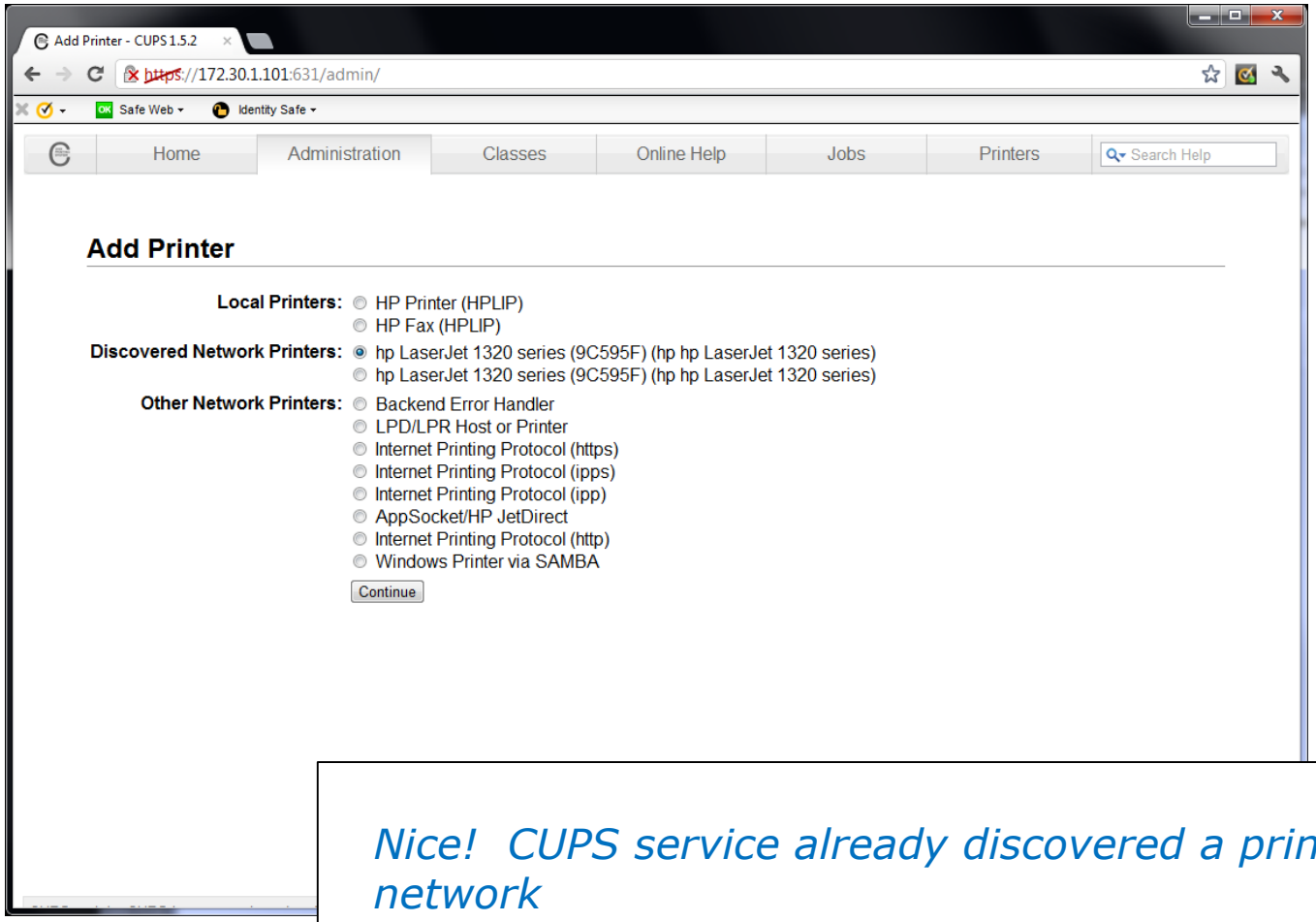
Select the Administration tab to add printers



FYI
CIS 192

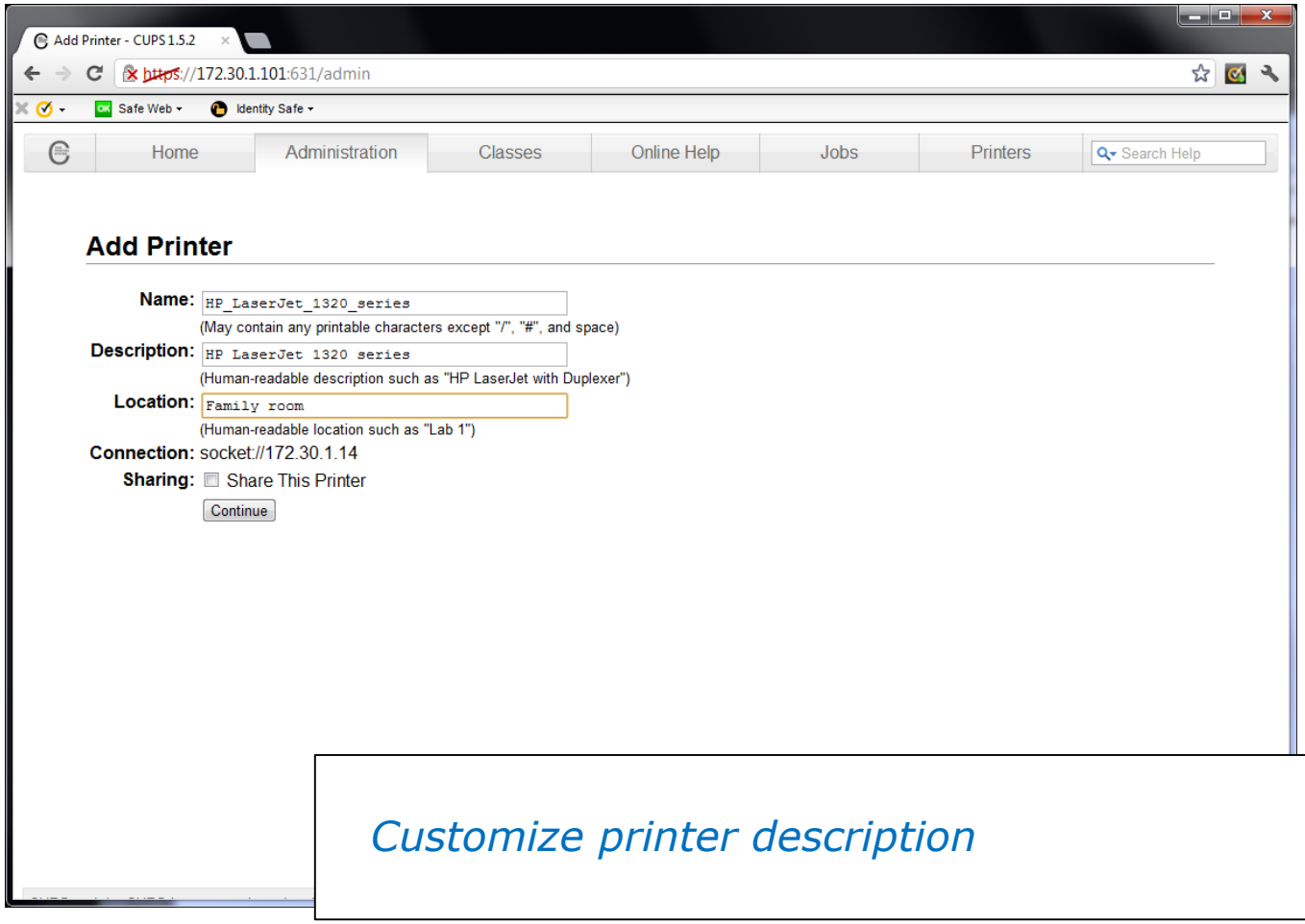
Must authenticate to add new printer

FYI CIS 192



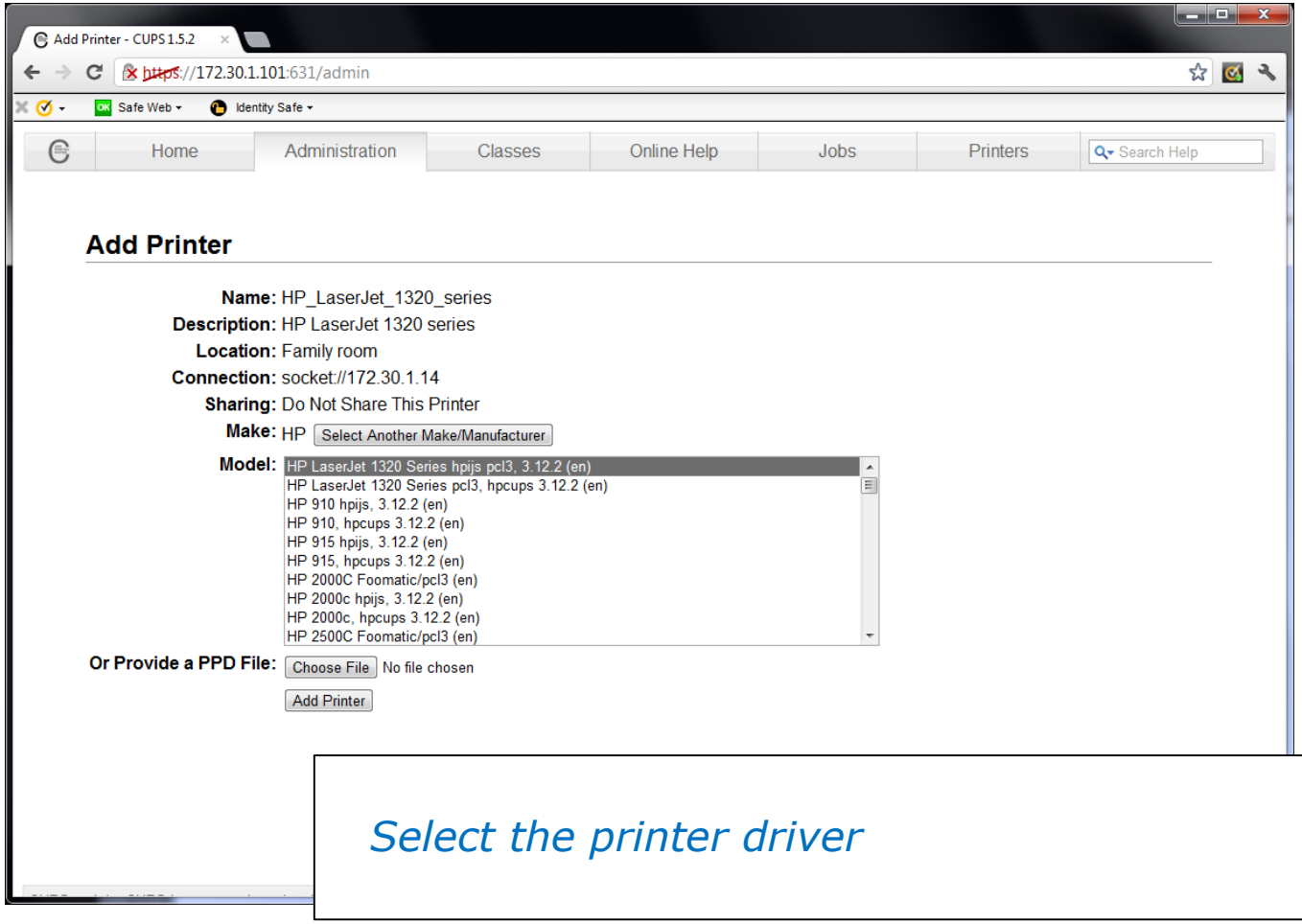
Nice! CUPS service already discovered a printer on the network

FYI CIS 192



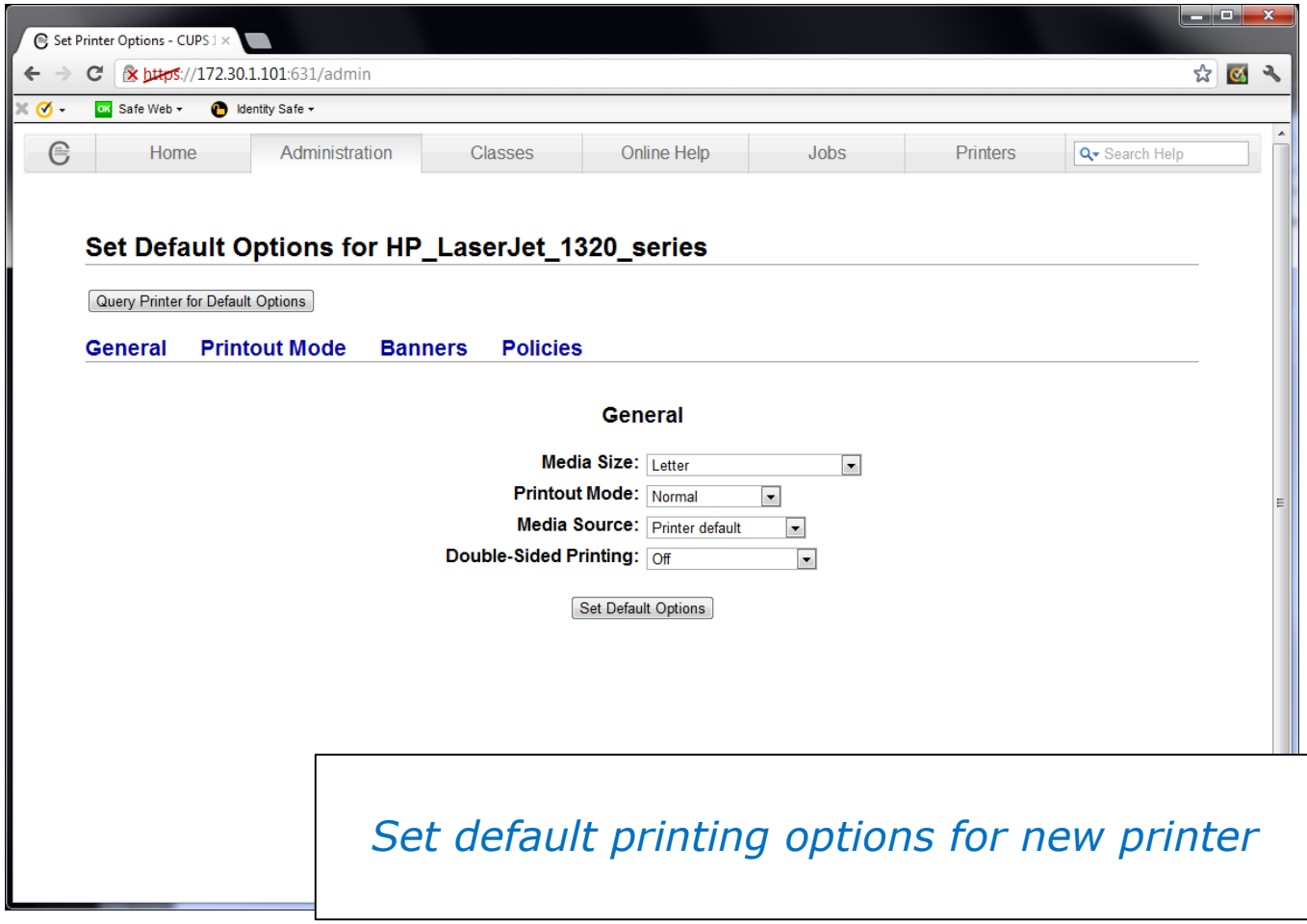
Customize printer description

FYI CIS 192

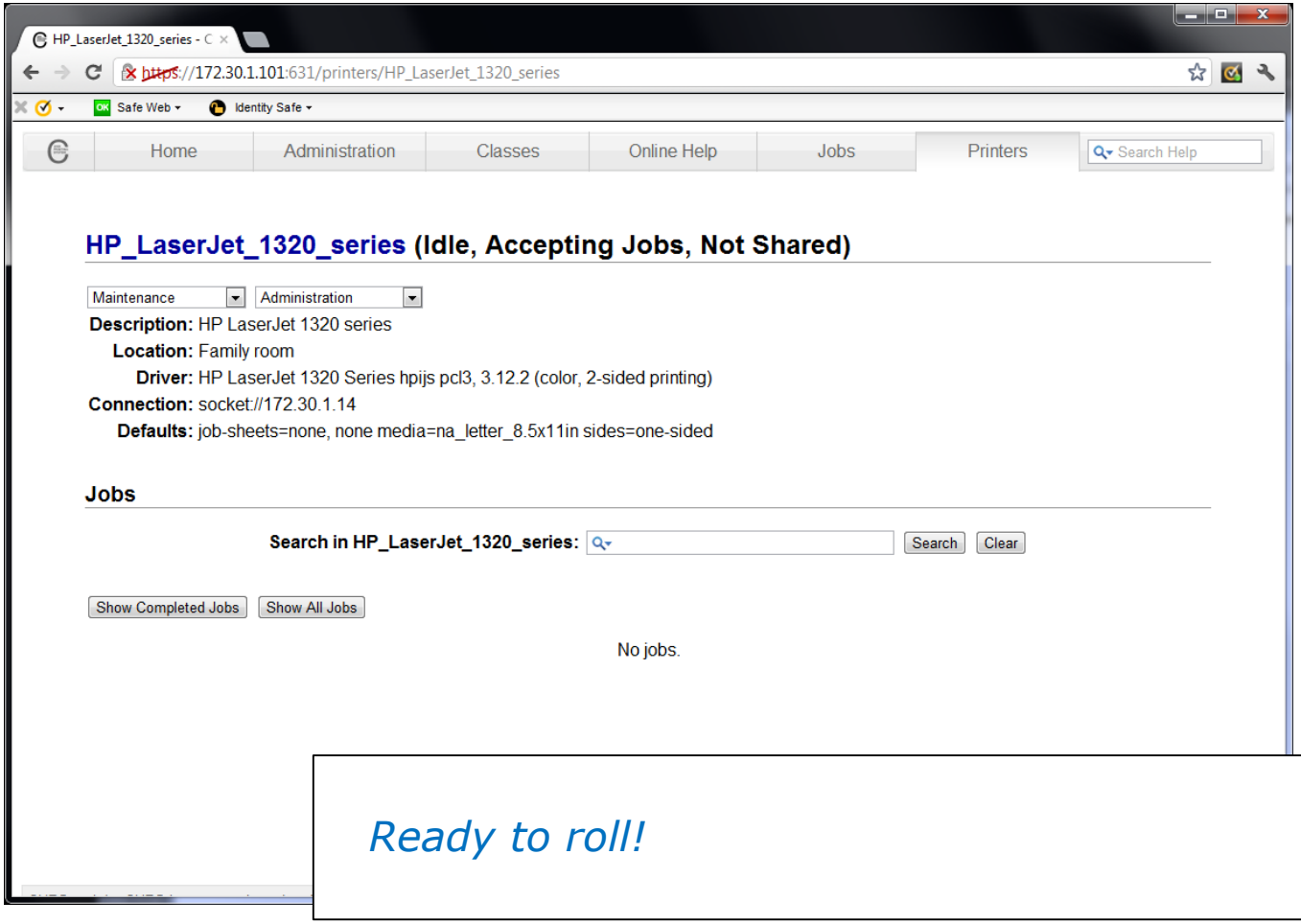


Select the printer driver

FYI CIS 192



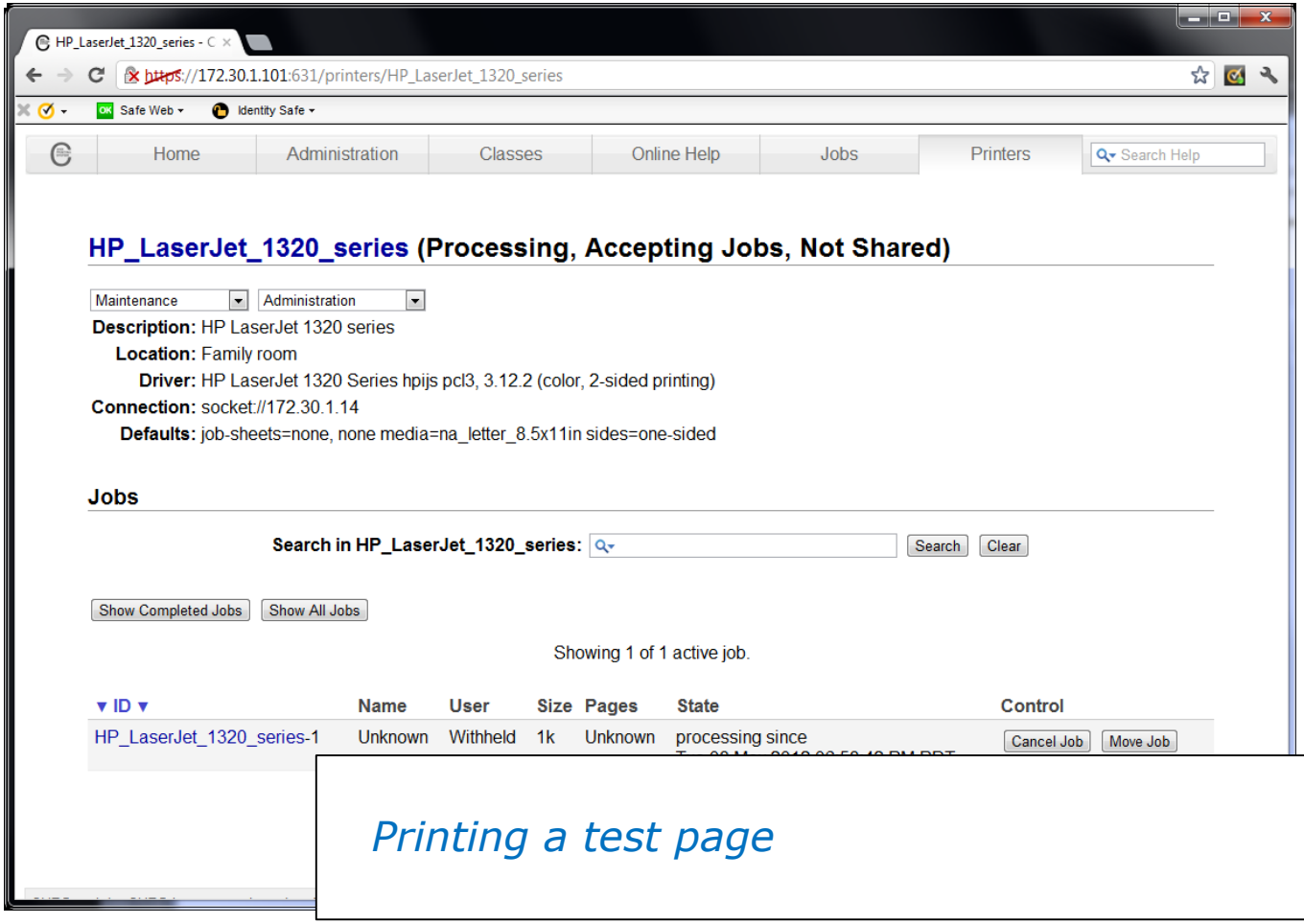
Set default printing options for new printer



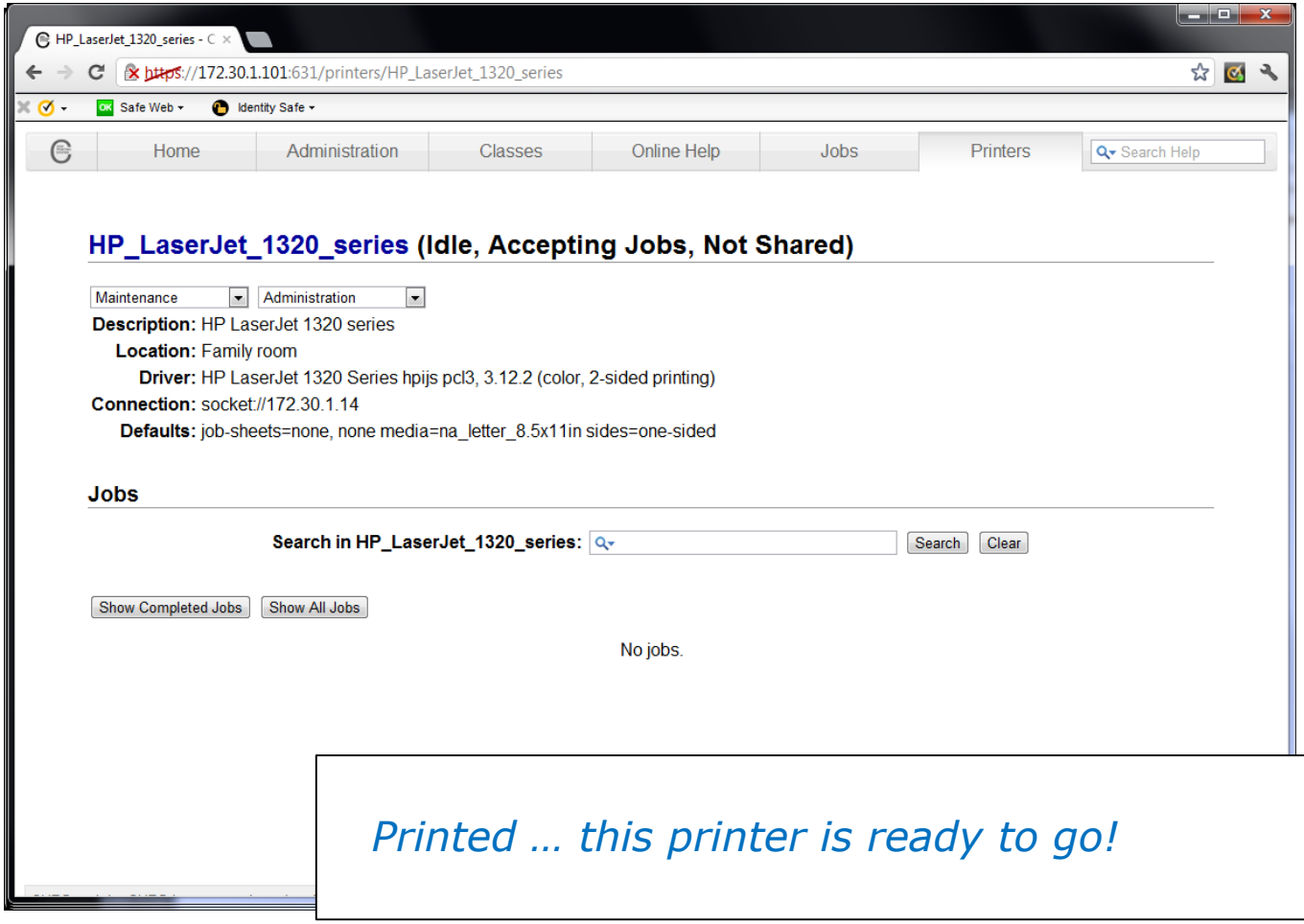
FYI
CIS 192

Ready to roll!

FYI CIS 192



Printing a test page



FYI
CIS 192

Printed … this printer is ready to go!