

## Lesson Module Status

- Slides - draft
- Properties - done
- Flash cards - NA
- First minute quiz - done
- Web calendar summary - done
- Web book pages - gillay done
- Commands - done
- Lab - done
- Opus accounts created for students submitting Lab 1 -
- CCC Confer room whiteboard - done
- Backup headset charged - done
- Backup slides, VC, handouts on flash drive -



- [ ] Has the phone bridge been added?
- [ ] Is recording on?
- [ ] Does the phone bridge have the mike?
- [ ] Share slides, putty, VB, eko and Chrome
- [ ] Disable spelling on PowerPoint



Dennis



Sean B.



Christopher



Salena



Abd



Sarah



Astitow



Mike D.



Alex



Christine



Steven



Nathan



Tony



James G.



Anthony



Fernando



Miguel



Lars



Jennifer



Rudy



Laura P.



Nick R.



Juan



Jacob



Nic S.



Jose



Luke



Saulius



Eric



Brian B.



Rich

Instructor: **Rich Simms**

Dial-in: **888-450-4821**

Passcode: **761867**

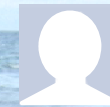
### Online Class Students



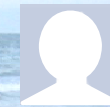
James B.



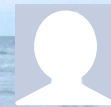
Liz



Casady



Jason



Dale



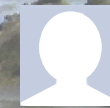
Aaron



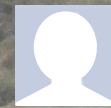
Stephen



Matthew



Songul



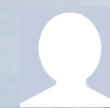
Tanya



Stephanie



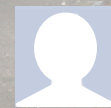
Adriana



Laura S.



Janelle



Victor

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

## First Minute Quiz



Please close your books, notes, lesson materials, forum and answer these questions:

1. What command shows the other users logged in to the computer?
2. What is the lowest level, inner-most component of a UNIX/Linux Operating System called?
3. What part of UNIX/Linux is both a user interface and a programming language?

*Online students, email me  
your answers at  
risimms@cabrillo.edu*

*Classroom students, email me  
your answers at  
risimms@cabrillo.edu  
(or you can write your answers on a  
scrap piece of paper and hand it in)*

## Commands

Objectives	Agenda
<ul style="list-style-type: none"><li>• Understand how the UNIX login operation works.</li><li>• Meet John the Ripper and learn how vulnerable a poor password is.</li><li>• Understand basic command syntax and operation.</li><li>• Understand program files and what happens when they are run.</li><li>• Understand how the shell works and environment variables.</li><li>• Understand how to get documentation when online.</li></ul>	<ul style="list-style-type: none"><li>• Quiz</li><li>• Review</li><li>• SSH hopping</li><li>• Deep dive on logging in</li><li>• Personal Opus accounts</li><li>• Passwords and cracking them</li><li>• Making strong passwords</li><li>• Programs files</li><li>• Running programs/processes</li><li>• Command line syntax</li><li>• Environment variables</li><li>• Life of the shell</li><li>• Metacharacters</li><li>• Docs</li><li>• Wrap up</li></ul>

## Last weeks Assignment

1. Student surveys due today
2. Lab 1 due (by midnight)
3. Question on previous material?

# Review and clarifications

# What is a computer?

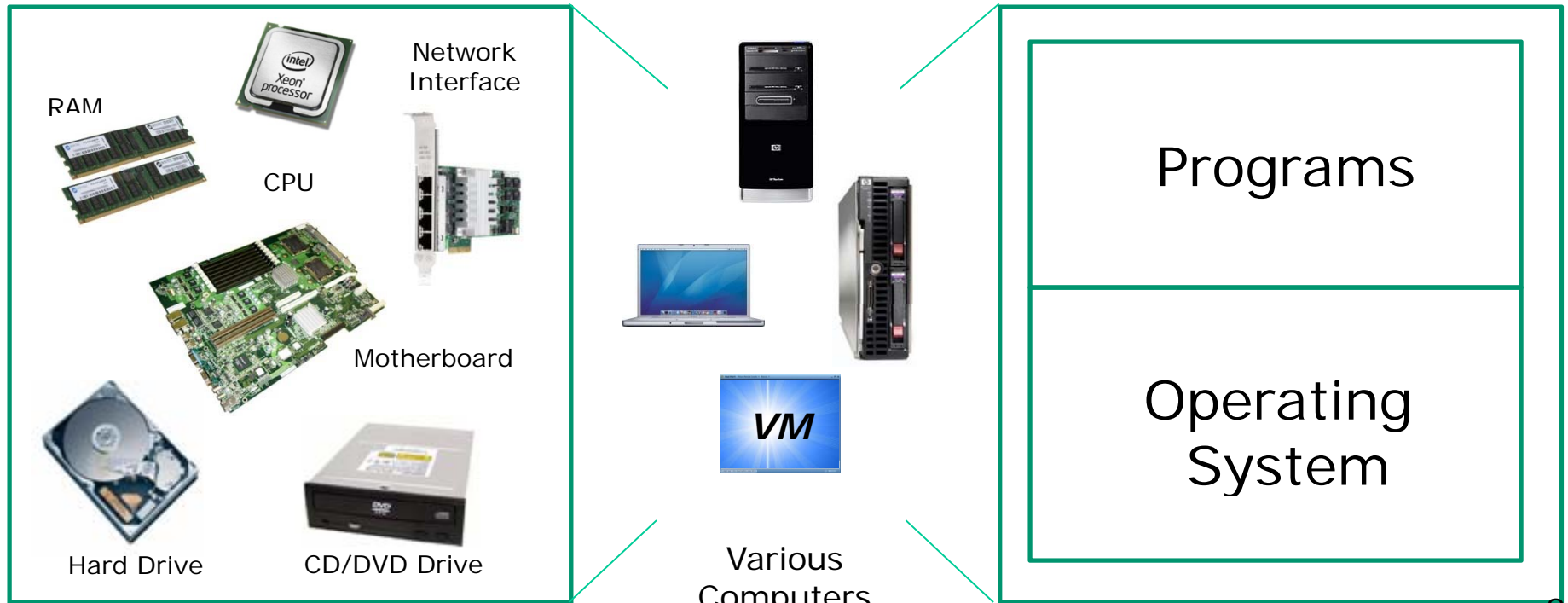
Desktops, Mobiles, Servers and Virtual Machines

*one or more users*



Hardware

Software





## Software - Programs and Operating Systems

Users



Software

### Programs (e.g. Word, Apache, PhotoShop, vi)

- Some programs come as part of the OS
- Some programs are add-ons purchases or downloads
- Programs include commands and utilities
- Depend on an OS for all access to the hardware

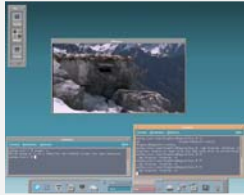
### Operating System (e.g. Linux, Windows, UNIX)

- Interface to the hardware
- Shares hardware resources
- Schedules/executes programs
- Process management
- Input/output services
- System monitoring
- Network stack

Hardware



# UNIX and Unix-like Operating Systems



HP-UX



Sun Solaris



AT&T UNIX  
(1969)



SCO



IBM AIX



BSD UNIX



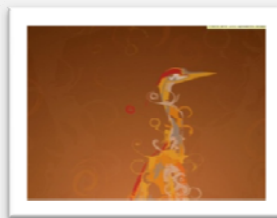
Mac OS X and iOS

*Apple operating systems  
use the Mach Kernel*



*Linux operating systems use the Linux Kernel*

Various GNU/Linux Distributions



Ubuntu



Red Hat



SUSE

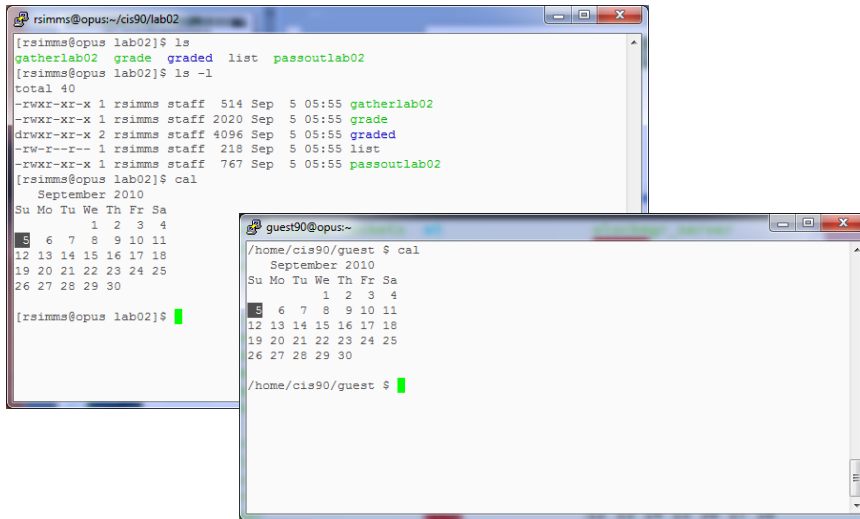


Debian

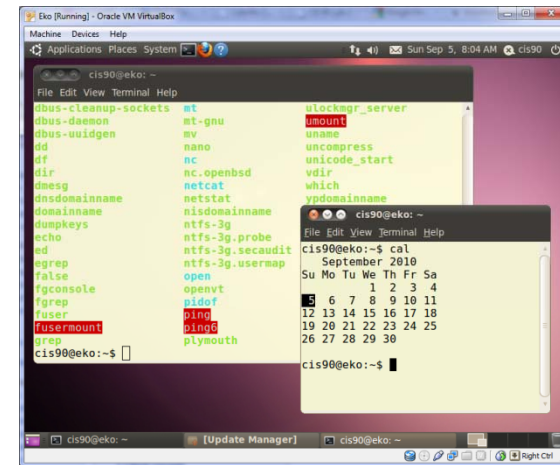
Embedded Linux



# Terminals



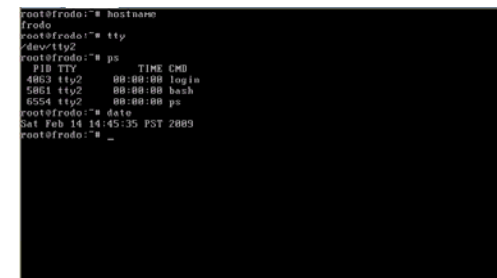
**Putty terminals** (with scroll bars, colors, customizable backgrounds, fonts and sizes) and runs on Windows



**Graphical terminals** (with scroll bars, colors, customizable backgrounds, fonts and sizes)

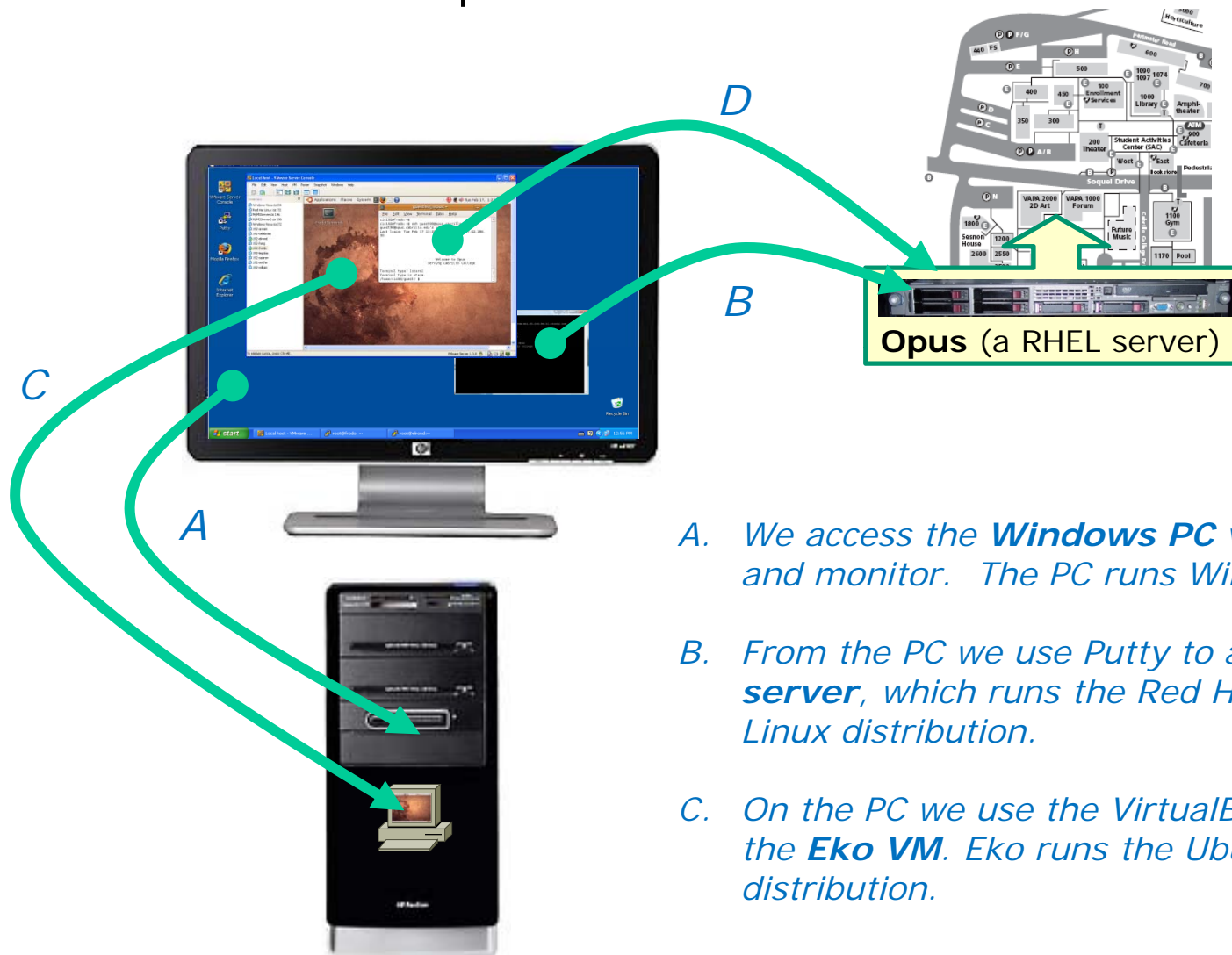


*Terminals were used in the old days to interact with computers. Today we use **terminal emulators** that are software programs.*



**Virtual terminals** (use ctrl-alt-fn) (no scroll bar, also called a console)

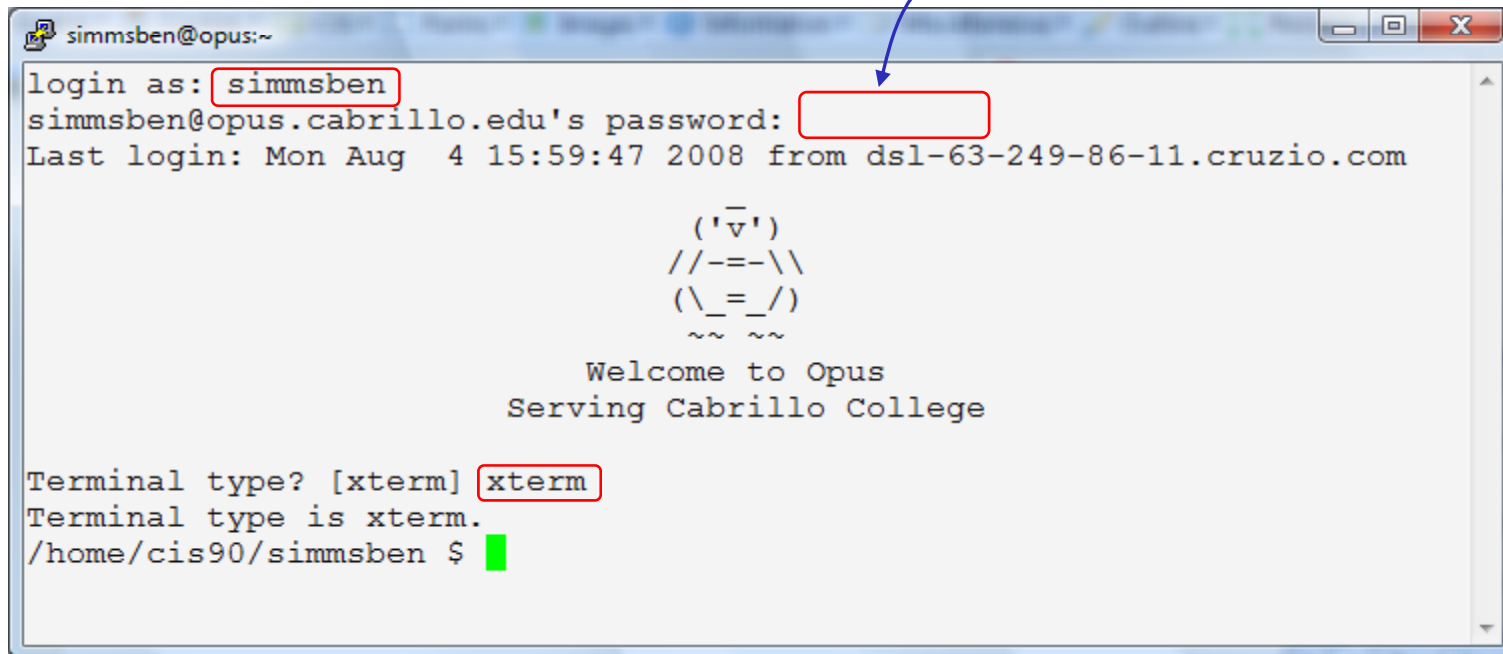
We used three computers for Lab 1 !!



- A. We access the **Windows PC** via its keyboard and monitor. The PC runs Windows XP.
- B. From the PC we use Putty to access the **Opus server**, which runs the Red Hat Enterprise Linux distribution.
- C. On the PC we use the VirtualBox to access the **Eko VM**. Eko runs the Ubuntu Linux distribution.
- D. From Eko VM, we SSH to access Opus (different session than B)

## Logging in

*Note: the password is never echoed for security reasons*



```
simmsben@opus:~  
login as: simmsben  
simmsben@opus.cabrillo.edu's password:  
Last login: Mon Aug  4 15:59:47 2008 from dsl-63-249-86-11.cruzio.com  
  
      _  
    ('v'  
  //---\ \  
  (\_=_/ )  
   ~ ~ ~  
  
  Welcome to Opus  
  Serving Cabrillo College  
  
Terminal type? [xterm] xterm  
Terminal type is xterm.  
/home/cis90/simmsben $
```

always requires:

**username + password + terminal type**

## The Putty program

The image shows two terminal windows side-by-side. The left window has a black background and shows the output of 'ls /bin' with a white cursor. The right window has a white background and shows the same output with a green cursor. In both, several words are highlighted in red.

```

[rsimms@server0-01 rsimms]$ ls /bin
arch      cut      fgrep    ls        pwd        sync
ash       date     gawk     mail      r           r
ash.static dd      grep     mkdir     r           r
awk       df       gtar     mknod     r           r
basename dmesg   gunzip   mktemp    r           r
bash      dnsdomainname gzip     more      r           r
bash2     doexec  hostname mount      r           r
bsh       domainname igawk    mt        s           s
cat       dumpkeys ipcalc   mv        s           s
chgrp     echo    kbd_mode netstat   s           s
chmod     ed       kill     nice      s           s
chown     egrep   link     nisdomainname s
cp        env      ln        pgawk     s           s
cpio      ex       loadkeys ping      s           s
csh       false   login    ps        s           s
[rsimms@server0-01 rsimms]$

[rsimms@nosmo src]$ ls /bin
alsanmute  dnsdomainname  kbd_mode  nisdomainname  sync
arch       doexec         keyctl    pgawk          tar
ash        domainname     kill      ping           tcsh
ash.static dumpkeys       ksh       ping6          touch
awk        echo           link      ps             tracepath
basename  ed             ln        pwd            tracepath6
bash      egrep         loadkeys  red            traceroute
bsh       env           login     rm             traceroute6
cat       ex            ls        rmdir          true
chgrp     false        mail      rpm            umount
chmod     fgrep        mailx     rvi            uname
chown     gawk         mkdir     rview         unicode_start
cp        gettext      mknod    sed            unicode_stop
cpio      grep         mktemp   setfont       unlink
csh       gtar         more     setserial     usleep
cut       gunzip       mount    sh             vi
date      gzip         mt       sleep         view
dd        hostname     mv       sort           ypdomainname
df        igawk        netstat  stty          zcat
dmesg     ipcalc       nice     su
[rsimms@nosmo src]$
  
```

*Why does Putty sometimes have a **black background** and sometimes a **white background**?*

**Rich's Cabrillo College CIS Classes Resources**

Home **Resources** Forums CIS Lab CTC

Login  
Flashcards  
Admin

CIS 90  
Previous Classes

102 days till term ends!

Cabrillo College  
Web Advisor  
CCC Confer  
Static IPs  
Quick Ref  
VM Repairs  
GAH!

**Links**

<p><b>Instructors</b></p> <ul style="list-style-type: none"> <li>Linux Master Jim</li> <li>Programming Master Ed</li> <li>Network Master Gerlinde</li> <li>Network Master Rick</li> <li>Web Master John</li> <li>Windows Master Gary</li> </ul> <p><b>Clubs</b></p> <ul style="list-style-type: none"> <li>GNU Linux Users Group</li> </ul> <p><b>Departments</b></p> <ul style="list-style-type: none"> <li>CNSA</li> <li>CIS</li> <li>CS</li> </ul> <p><b>Crib Sheets</b></p> <ul style="list-style-type: none"> <li>Ollie Wright (CIS 90)</li> </ul> <p><b>Documentation</b></p> <ul style="list-style-type: none"> <li>TLDP</li> <li>LINFO</li> </ul> <p><b>Animations</b></p> <ul style="list-style-type: none"> <li>Linux network technologies</li> </ul>	<p><b>Getting Linux</b></p> <ul style="list-style-type: none"> <li>Linux ISOs</li> <li>Kernels</li> <li>RPMs (rpmfind)</li> <li>RPMs (pbone)</li> </ul> <p><b>Tools and Software</b></p> <ul style="list-style-type: none"> <li>Apache</li> <li>Bastille</li> <li>cygwin</li> <li>DOS boot disks</li> <li>Dynamips/Dynagen</li> <li>John the Ripper</li> <li>MSDN Academic Alliance</li> <li>Netfilter</li> <li>Putty SSH Tools</li> <li>Quagga routing suite</li> <li>Tripwire</li> <li>VirtualBox</li> <li>VMware Server</li> <li>Wireshark</li> </ul> <p><b>Standards</b></p> <ul style="list-style-type: none"> <li>IETF (RFCs)</li> <li>IEEE</li> </ul> <p><b>Commands</b></p> <ul style="list-style-type: none"> <li>Practical</li> <li>Summary</li> <li>Useful</li> <li>vi summary</li> </ul>	<p><b>Howtos</b></p> <ul style="list-style-type: none"> <li>HowtoForge</li> <li>email</li> <li>DNS</li> <li>Ethernet</li> <li>NFS</li> <li>NIS</li> <li>PPP</li> <li>Putty SSH</li> <li>sed</li> </ul> <p><b>Student Help</b></p> <ul style="list-style-type: none"> <li>Making a backup by Mich</li> <li>Home V router by Marc</li> <li>Putty to by Marc</li> <li>Installing by Marc</li> <li>Linux Pe by Mich</li> <li>Guide to by Mich</li> </ul> <p><b>Linux New</b></p> <ul style="list-style-type: none"> <li>linuxtod</li> <li>LinuxWo</li> <li>Linux</li> <li>Linux W</li> <li>COMPU</li> </ul>
---	--	---

**Rich's Howtos**

**Putty**

- Installing PuTTY on Windows
- Configuring the appearance of PuTTY

**VirtualBox**

- Bringing the Eko VM home

There is a Howto on the Resource page to walk you through customizing Putty

**Linux Howtos**  
Configuring the appearance of PuTTY  
Fall, 2008

**Software used**

- PuTTY SSH client (download)

**Step 1 - Run PuTTY and login**

The default appearance is 10 point Courier New font with white text on a black background. The translation is ISO-8859-1 which may garble the ' displayed in "Linux User's Manual".

```

simmsben@opus:~
MSG (1) Linux User's Manual MSG (1)
NAME
  mesg - control write access to your terminal

SYNOPSIS
  mesg [y|n]

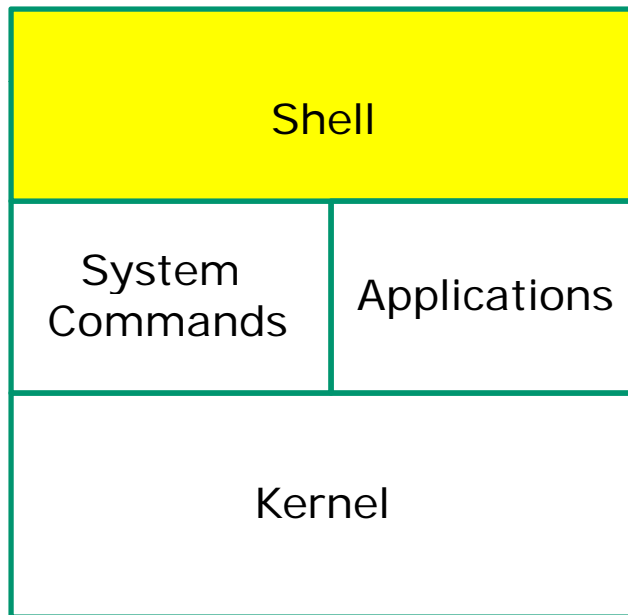
DESCRIPTION
  Mesg controls the access to your terminal by others. It's typically
  used to allow or disallow other users to write to your terminal (see
  write(1)).

OPTIONS
  :
  
```

Right click on the top of the window to get a menu.

**Step 2 - Get to Reconfiguration window**

## The Shell



- Allows users to interact with the computer via a “**command line**”.
- **Prompts** for a command, parses the command, finds the right program and gets that program executed.
- Is called a “**shell**” because it hides the underlying operating system.
- Multiple shell programs are available: **sh** (Bourne shell), **bash** (born again shell), **csh** (C shell), **ksh** (Korn shell).
- The shell is a **user interface** and a **programming language** (scripts).
- GNOME and KDE desktops could be called **graphical shells**





## Shell Prompt

*The shell **prompt** is a string of text ending with a \$*

```
/home/cis90/guest $ date
Sun Sep  5 05:59:15 PDT 2010
/home/cis90/guest $ who
rsimms pts/1      2010-09-05 05:17 (dsl-88-222-111-200.dhcp.cruzio.com)
root   :0         2009-12-18 17:30
guest90 pts/2      2010-09-05 05:57 (dsl-88-222-111-200.dhcp.cruzio.com)
simmsben pts/3      2010-09-05 06:12 (dsl-88-222-111-200.dhcp.cruzio.com)
/home/cis90/guest $ whoami
guest90
/home/cis90/guest $ hostname
opus.cabrillo.edu
/home/cis90/guest $ id
uid=503(guest90) gid=90(cis90) groups=90(cis90),100(users)
context=user_u:system_r:unconfined_t
/home/cis90/guest $ ps
  PID TTY          TIME CMD
 8536 pts/2      00:00:00 bash
 8617 pts/2      00:00:00 ps
/home/cis90/guest $ tty
/dev/pts/2
/home/cis90/guest $
```

## Shell Commands

### *Entering various **shell commands** from Lab 1*

```
/home/cis90/guest $ date
Sun Sep  5 05:59:15 PDT 2010
/home/cis90/guest $ who
rsimms pts/1      2010-09-05 05:17 (dsl-88-222-111-200.dhcp.cruzio.com)
root   :0         2009-12-18 17:30
guest90 pts/2      2010-09-05 05:57 (dsl-88-222-111-200.dhcp.cruzio.com)
simmsben pts/3     2010-09-05 06:12 (dsl-88-222-111-200.dhcp.cruzio.com)
/home/cis90/guest $ whoami
guest90
/home/cis90/guest $ hostname
opus.cabrillo.edu
/home/cis90/guest $ id
uid=503(guest90) gid=90(cis90) groups=90(cis90),100(users)
context=user_u:system_r:unconfined_t
/home/cis90/guest $ ps
  PID TTY          TIME CMD
 8536 pts/2      00:00:00 bash
 8617 pts/2      00:00:00 ps
/home/cis90/guest $ tty
/dev/pts/2
/home/cis90/guest $
```

## Shell Commands

### *Output from the various shell commands from Lab 1*

```

/home/cis90/guest $ date
Sun Sep 5 05:59:15 PDT 2010
/home/cis90/guest $ who
rsimms pts/1 2010-09-05 05:17 (dsl-88-222-111-200.dhcp.cruzio.com)
root :0 2009-12-18 17:30
guest90 pts/2 2010-09-05 05:57 (dsl-88-222-111-200.dhcp.cruzio.com)
simmsben pts/3 2010-09-05 06:12 (dsl-88-222-111-200.dhcp.cruzio.com)
/home/cis90/guest $ whoami
guest90
/home/cis90/guest $ hostname
opus.cabrillo.edu
/home/cis90/guest $ id
uid=503(guest90) gid=90(cis90) groups=90(cis90),100(users)
context=user_u:system_r:unconfined_t
/home/cis90/guest $ ps
  PID TTY          TIME CMD
 8536 pts/2    00:00:00 bash
 8617 pts/2    00:00:00 ps
/home/cis90/guest $ tty
/dev/pts/2
/home/cis90/guest $

```

## Commands from last week's lesson and lab

<b>cal</b>	<i>Prints calendars</i>
<b>clear</b>	<i>Clears the screen</i>
<b>date</b>	<i>Shows the time and date</i>
<b>exit</b>	<i>Exits login session</i>
<b>history</b>	<i>Shows previous commands</i>
<b>hostname</b>	<i>Shows name of computer being interacted with</i>
<b>id</b>	<i>Shows UID's, GID's and SELinux information</i>
<b>ps</b>	<i>Shows process information</i>
<b>ssh</b>	<i>Initiates connection and login to remote computer</i>
<b>uname</b>	<i>Shows name of operating system</i>
<b>tty</b>	<i>Shows terminal device being used for session</i>
<b>who</b>	<i>Shows all users who are logged in</i>
<b>whoami</b>	<i>Like <b>who</b>, but only shows your login session</i>

*Note, each of these commands is actually a program residing in the /bin or /usr/bin directories.*

## Shell tty command

Running three Putty sessions at the same time to Opus. Note that each session is assigned a different terminal device.



Use the **tty** command to identify the **terminal device** being used for a session

## Lab 1 Questions

*Are there any questions on these question?*

- 1) On Opus, what was the prompt string?
- 2) What does the history command do?
- 3) On Opus, what was your uid (user id) number?
- 4) On Opus, what was the name of the shell program being run?
- 5) What terminal device did you use to access Opus?
- 6) On Eko, what is the output from the hostname command?
- 7) What command shows the other users that are logged in?
- 8) What command shows you the name of the computer you are interacting with?
- 9) On Eko, what three keys must be pressed locally to use terminal tty2?
- 10) On Eko, if you log off one session, do you get logged off all the other sessions?
- 11) On Eko, is your command history the same for all login sessions?
- 12) What command logs you off?

*You can resubmit Lab 1 as many times as you want till midnight*

# More on ssh

# ssh command

login to a remote system

Syntax: **ssh** *user@hostname*

Where

- user = the user login name
- hostname = the name or IP address of the remote computer

Examples:

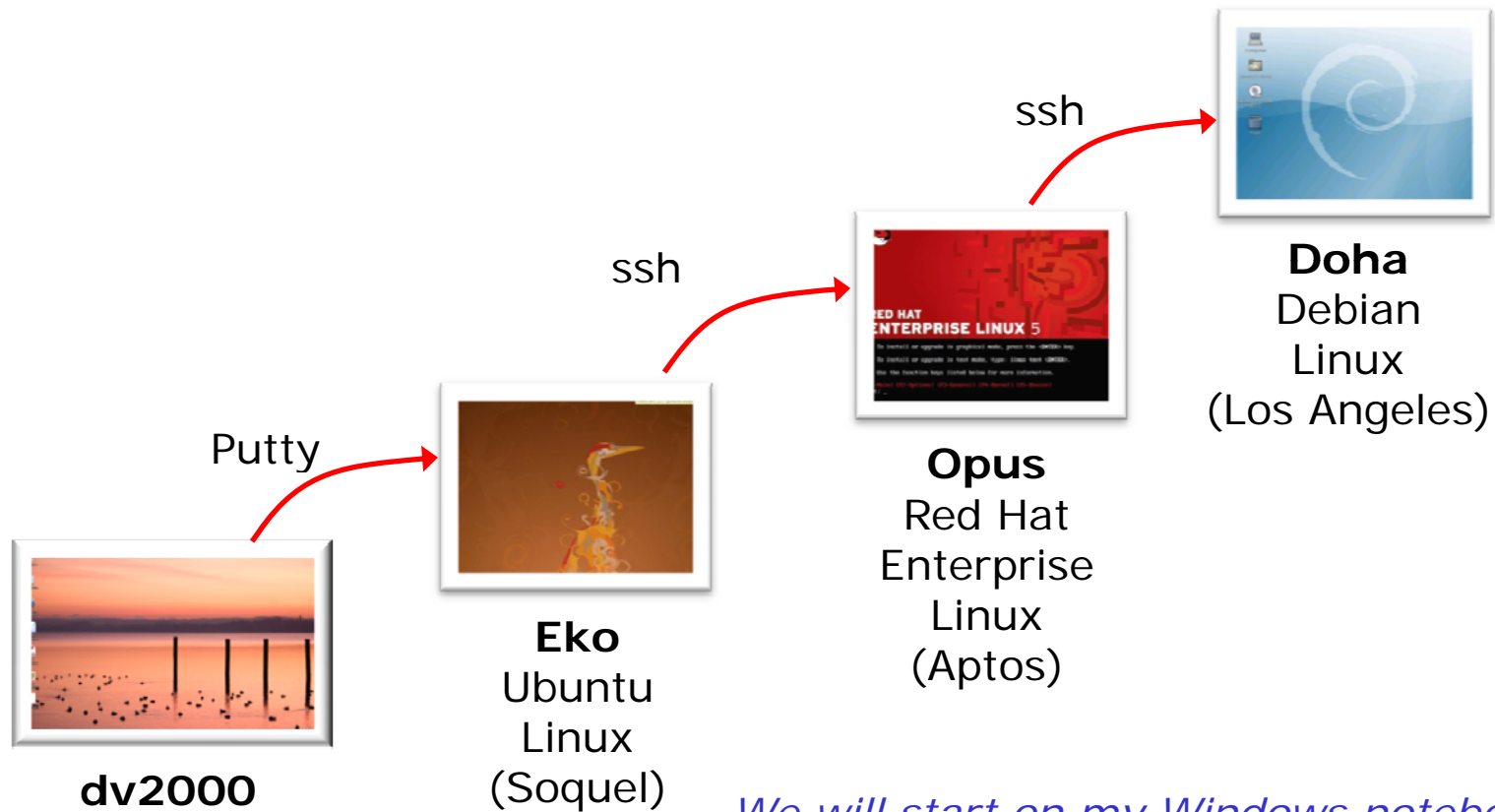
```
ssh guest90@opus.cabrillo.edu
```

```
ssh cis90@172.30.1.198
```

```
ssh root@frida
```



## "SSH Hopping"



*We will start on my Windows notebook computer dv2000. From dv2000 Putty into Eko. From Eko, ssh into Opus. From Opus ssh into simms-teach.com*

"SSH Hopping"

```

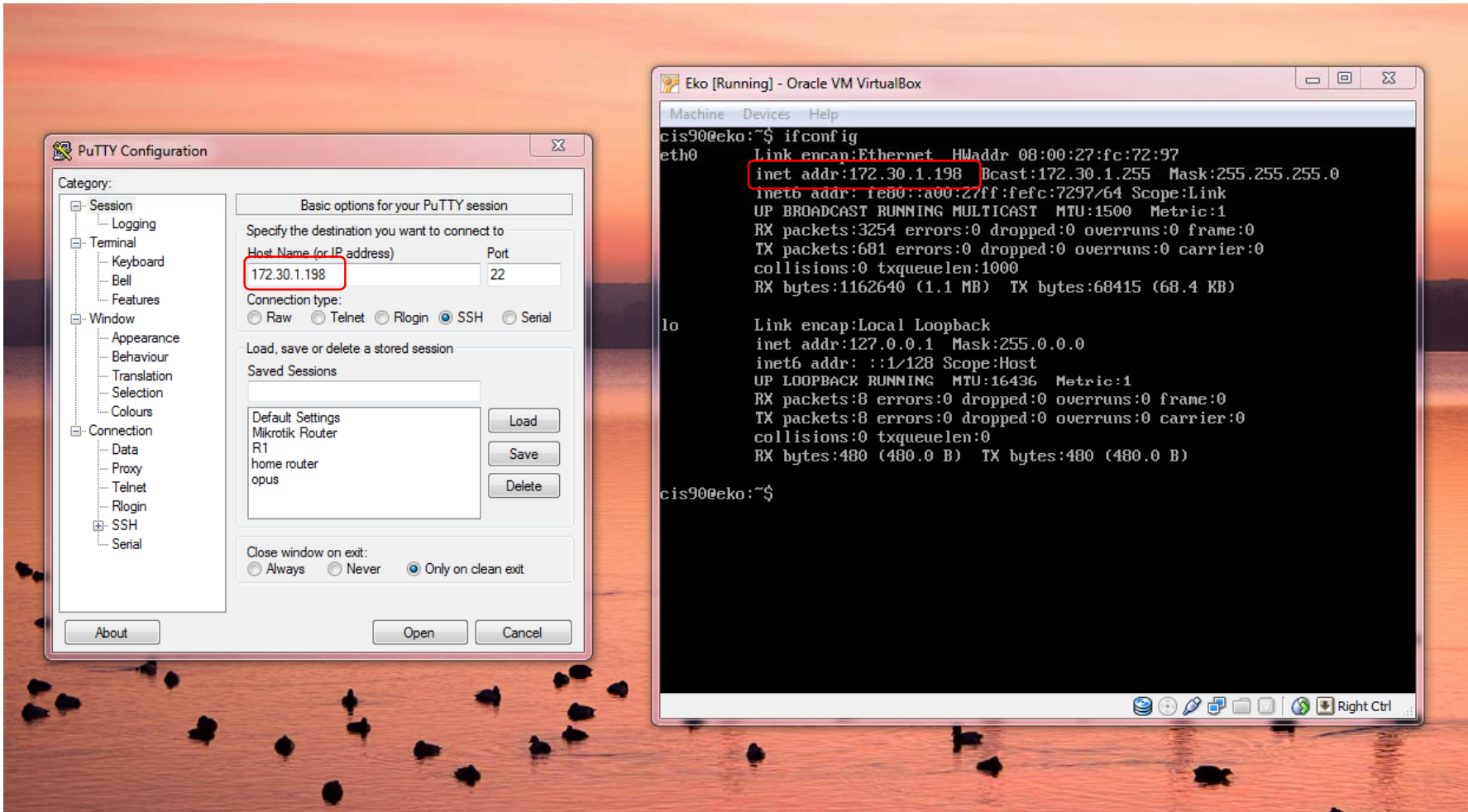
Eko [Running] - Oracle VM VirtualBox
Machine  Devices  Help
cis90@eko:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:fc:72:97
          inet addr:172.30.1.198  Bcast:172.30.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fefc:7297/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3254 errors:0 dropped:0 overruns:0 frame:0
          TX packets:681 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1162640 (1.1 MB)  TX bytes:68415 (68.4 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:480 (480.0 B)  TX bytes:480 (480.0 B)

cis90@eko:~$ _
  
```

*If you try this command on Opus you will get an error message. It has to do with your path and we will cover that later in this lesson.*

*First, lets use the **ifconfig** command to get the IP address of the Eko VM. It is 172.30.1.198 for the Eko I'm using at home.*



*Ok, lets begin. Lets Putty from Windows to the Eko VM using the IP address we just determined*

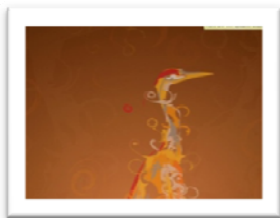
```
login as: cis90
Server refused our key
cis90@172.30.1.198's password:
Linux eko 2.6.32-24-generic #38-Ubuntu SMP Mon Jul 5 09:22:14 UTC 2010 i686 GNU/
Linux
Ubuntu 10.04.1 LTS

Welcome to Ubuntu!
 * Documentation: https://help.ubuntu.com/

59 packages can be updated.
15 updates are security updates.

Last login: Sat Sep  4 16:10:07 2010
cis90@eko:~$ hostname
eko
cis90@eko:~$ uname
Linux
cis90@eko:~$ tty
/dev/pts/2
cis90@eko:~$ cat /etc/*-release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=10.04
DISTRIB_CODENAME=lucid
DISTRIB_DESCRIPTION="Ubuntu 10.04.1 LTS"
cis90@eko:~$
```

*Eko is an Ubuntu Linux 10.04 system and we came in using the /dev/pts/2 terminal device*



*OK, now we have logged in to **Eko** from **dv2000***

```
guest90@opus:~  
cis90@eko:~$ ssh guest90@opus.cabrillo.edu  
guest90@opus.cabrillo.edu's password:  
Last login: Sun Sep  5 17:44:37 2010 from adsl-70-143-65-176.dsl.pltn13.sbcglo  
bal.net  
  
      _  
    ('v')  
  //--==\\  
  (\_=_/)  
  ~ ~ ~  
  
  Welcome to Opus  
  Serving Cabrillo College  
  
Terminal type? [xterm]  
Terminal type is xterm.  
/home/cis90/guest $ hostname  
opus.cabrillo.edu  
/home/cis90/guest $ tty  
/dev/pts/1  
/home/cis90/guest $ uname  
Linux  
/home/cis90/guest $ cat /etc/*-release  
Red Hat Enterprise Linux Server release 5.4 (Tikanga)  
/home/cis90/guest $ █
```

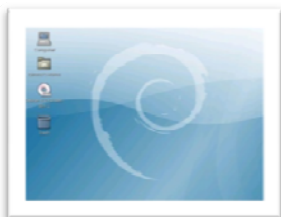
*Opus is a RHEL 5.4 system and we came in using the /dev/pts/1 terminal device*



*OK, we have logged in to **Opus** from **Eko***

```
guest90@opus:~  
/home/cis90/guest $ ssh rsimms@simms-teach.com  
The authenticity of host 'simms-teach.com (69.163.236.47)' can't be established.  
RSA key fingerprint is 0e:c2:f6:f4:d9:86:9d:4b:c4:3d:77:e7:a4:bb:59:14.  
Are you sure you want to continue connecting (yes/no)? yes  
Warning: Permanently added 'simms-teach.com,69.163.236.47' (RSA) to the list of  
known hosts.  
rsimms@simms-teach.com's password:  
  
doHa  
  
Welcome to doha.dreamhost.com  
  
Any malicious and/or unauthorized activity is strictly forbidden.  
All activity may be logged by DreamHost Web Hosting.  
  
Last login: Sun Sep  5 18:04:16 2010 from 207.62.186.9  
[doha]$ hostname  
doha  
[doha]$ tty  
/dev/pts/16  
[doha]$ uname  
Linux  
[doha]$ ls /etc/*_version  
/etc/debian version  
[doha]$ cat /etc/*_version  
5.0.5  
[doha]$
```

*simms-teach.com is really named doha. It is a Debian Linux 5.0.5 system and we came in using the /dev/pts/16 terminal device.*



*OK, we have logged into **simms-teach.com** (really named **doha**) from **Opus***



# Opus Logins (A deep dive)

## Login and Passwords

- 1) *init* starts up the **mingetty** program for each terminal which then prompts for login username, gets it, then starts login.

```
CentOS release 4.6 (Final)
Kernel 2.6.9-67.ELsmp on an i686
nosmo login: _
```

```
[root@nosmo ~]# ps t tty1
  PID TTY          STAT       TIME COMMAND
 3545 tty1        Ss+        0:00 /sbin/mingetty tty1
```

- 2) *login* collects the password and checks it with */etc/passwd* and */etc/shadow*

```
CentOS release 4.6 (Final)
Kernel 2.6.9-67.ELsmp on an i686
nosmo login: rsimms
Password: _
```

```
[root@nosmo ~]# ps t tty1
  PID TTY          STAT       TIME COMMAND
 3545 tty1        Ss+        0:00 /bin/login -
```

- 3) If a match then *login* then starts up the shell specified in the */etc/passwd* file

```
CentOS release 4.6 (Final)
Kernel 2.6.9-67.ELsmp on an i686
nosmo login: rsimms
Password:
Last login: Mon Jul  7 14:25:17 on tty1
[rsimms@nosmo ~]$ _
```

```
[root@nosmo ~]# ps t tty1
  PID TTY          STAT       TIME COMMAND
 4917 tty1        Ss+        0:00 -bash
```



# /etc/passwd

```
root@nosmo:~  
[root@nosmo ~]# cat /etc/passwd  
root:x:0:0:root:/root:/bin/bash  
bin:x:1:1:bin:/bin:/sbin/nologin  
daemon:x:2:2:daemon:/sbin:/sbin/nologin  
adm:x:3:4:adm:/var/adm:/sbin/nologin  
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin  
sync:x:5:0:sync:/sbin:/bin/sync  
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown  
halt:x:7:0:halt:/sbin:/sbin/halt  
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin  
news:x:9:13:news:/etc/news:  
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin  
operator:x:11:0:operator:/root:/sbin/nologin  
games:x:12:100:games:/usr/games:/sbin/nologin  
gopher:x:13:30:gopher:/var/gopher:/sbin/nologin  
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin  
nobody:x:99:99:Nobody:/:/sbin/nologin  
dbus:x:81:81:System message bus:/:/sbin/nologin  
vcsa:x:69:69:virtual console memory owner:/dev:/sbin/nologin  
rpm:x:37:37:/:/var/lib/rpm:/sbin/nologin  
haldaemon:x:68:68:HAL daemon:/:/sbin/nologin  
netdump:x:34:34:Network Crash Dump user:/var/crash:/bin/bash  
nscd:x:28:28:NSCD Daemon:/:/sbin/nologin  
sshd:x:74:74:Privilege-separated SSH:/var/empty/ssh:/sbin/nologin  
rpc:x:32:32:Portmapper RPC user:/:/sbin/nologin  
mailnull:x:47:47:/:/var/spool/mqueue:/sbin/nologin  
smbx:x:51:51:/:/var/spool/mqueue:/sbin/nologin  
rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin  
nfsnobody:x:65534:65534:Anonymous NFS User:/var/lib/nfs:/sbin/nologin  
pcap:x:77:77:/:/var/arpwatch:/sbin/nologin  
xfs:x:43:43:X Font Server:/etc/X11/fs:/sbin/nologin  
ntp:x:38:38:/:etc/ntp:/sbin/nologin  
gdm:x:42:42:/:var/gdm:/sbin/nologin  
rsimms:x:500:500:Rich Simms:/home/rsimms:/bin/bash  
[root@nosmo ~]#
```

Fields f1:f2:f3:f4:f5:f6:f7

- f1 = User name
- f2 = Password
- f3 = User id (uid)
- f4 = Group id (gid)
- f5 = Comment
- f6 = Home directory
- f7 = Command/shell

*/etc/passwd has all the login accounts. It used to have passwords too but not anymore*

## /etc/shadow

*The passwords are now kept in /etc/shadow and they are encrypted*

```

root@nosmo:~
[root@nosmo ~]# cat /etc/shadow
root:$1$afIkPzQ1$S7ITNpPnKa0b6h3gI6qw8.:13994:0:99999:7:::
bin:*:13994:0:99999:7:::
daemon:*:13994:0:99999:7:::
adm:*:13994:0:99999:7:::
lp:*:13994:0:99999:7:::
sync:*:13994:0:99999:7:::
shutdown:*:13994:0:99999:7:::
halt:*:13994:0:99999:7:::
mail:*:13994:0:99999:7:::
news:*:13994:0:99999:7:::
uucp:*:13994:0:99999:7:::
operator:*:13994:0:99999:7:::
games:*:13994:0:99999:7:::
gopher:*:13994:0:99999:7:::
ftp:*:13994:0:99999:7:::
nobody:*:13994:0:99999:7:::
dbus:!!:13994:0:99999:7:::
vcsa:!!:13994:0:99999:7:::
rpm:!!:13994:0:99999:7:::
haldaemon:!!:13994:0:99999:7:::
netdump:!!:13994:0:99999:7:::
nscd:!!:13994:0:99999:7:::
sshd:!!:13994:0:99999:7:::
rpc:!!:13994:0:99999:7:::
mailnull:!!:13994:0:99999:7:::
smb:!!:13994:0:99999:7:::
rpcuser:!!:13994:0:99999:7:::
nfsnobody:!!:13994:0:99999:7:::
pcap:!!:13994:0:99999:7:::
xfs:!!:13994:0:99999:7:::
ntp:!!:13994:0:99999:7:::
gdm:!!:13994:0:99999:7:::
rsimms:$1$xvRe00gP$K4ZkBCCdK1KVAhTtud0Ir.:13994:0:99999:7:::
[root@nosmo ~]#

```

Fields f1:f2:f3:f4:f5:f6:f7:f8:f9

f1 = User name

f2 = Password

- \$1\$... (MD5 encrypted)
- \* (locked)
- !! (no password set)

f3 = Day last changed (since 1/1/70)

f4 = Days till change is allowed

f5 = Days till change is required

f6 = Days of warning before change

f7 = Days till account is disabled

f8 = Day account was disabled

f9 = Reserved

# Your Opus Account

(for those of you that submitted Lab 1 on Opus)

# Your new Opus user account

- Everyone who submits Lab 1 will get their own user account on Opus.
- The first time you log in with this account you will be prompted to change your password.
- Please make it a strong password!
- **Botnets** and **ne-er-do-wells** are constantly attempting to break into computers attached to the Internet! Even my little Frodo VM!

## They never stop trying

Monday:

Failed logins from:

122.249.183.95 (x183095.ppp.asahi-net.or.jp): 3 times

218.64.5.131 (131.5.64.218.broad.nc.jx.dynamic.163data.com.cn): 3 times

Illegal users from:

78.46.83.76 (static.76.83.46.78.clients.your-server.de): 3 times

218.4.157.178: 3 times

pam\_succeed\_if(sshd:auth): error retrieving information about user teamspeak : 1 time(s)

reverse mapping checking getaddrinfo for

131.5.64.218.broad.nc.jx.dynamic.163data.com.cn failed - POSSIBLE BREAK-IN ATTEMPT! : 3 time(s)

pam\_succeed\_if(sshd:auth): error retrieving information about user ts : 2 time(s)

pam\_succeed\_if(sshd:auth): error retrieving information about user plcmsspip : 2 time(s)

pam\_succeed\_if(sshd:auth): error retrieving information about user PlcmSpIp : 1 time(s)

*We used to get up thousands of attempts every day until we made some changes to the firewall on Opus. Attacks always coming from different computers around the world.*

# logwatch

*the ne'er-do-wells trying to break in ...  
this is why you need strong passwords*

----- SSHD Begin -----

```
SSHD Killed: 1 Time(s)
SSHD Started: 1 Time(s)
Disconnecting after too many authentication failures for user:
guest90 : 1 Time(s)
```

## Failed logins from:

```
76.254.22.196 (adsl-76-254-22-196.dsl.pltn13.sbcglobal.net): 2 times
201.7.115.194 (201-7-115-194.spopa302.ipd.brasiltelecom.net.br): 2135 times
210.240.12.14: 20 times
```

## Illegal users from:

```
201.7.115.194 (201-7-115-194.spopa302.ipd.brasiltelecom.net.br): 564 times
210.240.12.14: 42 times
```

```
Users logging in through sshd:
guest:
 76.254.22.196 (adsl-76-254-22-196.dsl.pltn13.sbcglobal.net): 2 times
jimg:
 70.132.20.25 (adsl-70-132-20-25.dsl.snfc21.sbcglobal.net): 7 times
ordazedw:
 76.254.22.196 (adsl-76-254-22-196.dsl.pltn13.sbcglobal.net): 1 time
root:
 63.249.86.11 (dsl-63-249-86-11.cruzio.com): 3 times
 70.132.20.25 (adsl-70-132-20-25.dsl.snfc21.sbcglobal.net): 1 time
rsimms:
 63.249.86.11 (dsl-63-249-86-11.cruzio.com): 2 times
```

## /var/log/wtmp and var/log/btmp

```
[root@opus log]# lastb | sort | cut -f1 -d' ' | grep -v ^$ | uniq -c > bad
[root@opus log]# sort -g bad > bad.sort
[root@opus log]# cat bad.sort | tail -50
 471 ftp
 472 public
 490 test
 490 tomcat
 498 user
 506 service
 508 mike
 508 username
 524 cyrus
 530 pgsq1
 532 test1
 544 master
 554 linux
 554 toor
 576 paul
 584 support
 590 testuser
 604 irc
 610 test
 656 noc
 686 www
 690 postfix
 723 john
 734 testing
 738 adam
 746 alex
 754 info
 798 tester
 832 library
 935 guest
 990 admin
1002 office
1022 temp
1070 ftpuser
1138 webadmin
1298 nagios
1332 web
1374 a
1384 student
1416 postgres
1690 user
1858 oracle
1944 mysql
2086 webmaste
5324 test
10803 root
10824 admin
18679 root
24064 root
[root@opus log]#
```

*Top 50 usernames used by the ne'er-do-wells*

## How to make a strong password

- The longer the better (8 or more characters)
- Not in any dictionary
- Use upper case, lowercase, punctuation, digits
- Something you can remember
- Keep it secret
- Change when compromised

Wh01e#!!

(Whole sh'bang)

KuKu4(co)2

(Cuckoo for Cocoa Puffs)

#0p&s@ve

(shop and save)

Idl02\$da

(I do laundry on Tuesday)



# passwd command

## change password

*Use the **passwd** command to change your password*

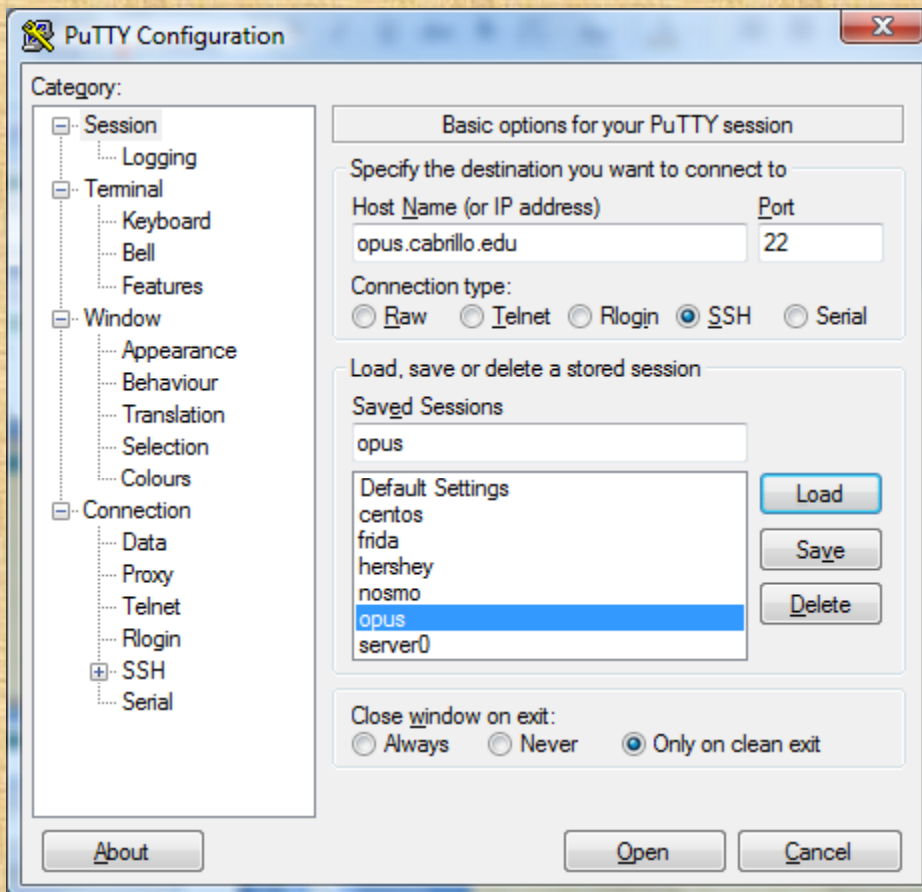
```
/home/cis90/simmsben $ passwd
Changing password for user simmsben.
Changing password for simmsben
(current) UNIX password:
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
/home/cis90/simmsben $
```

*Note, the passwords  
are not echoed as  
you type them.*

Turn OFF the recording

## Class Exercise

### Login to Opus and change passwords



Login to Opus:

1. Use new student accounts.
2. Change your password with the **passwd** command.

*First 5 letters  
of last name*

*First 3 letters  
of first name*

username: 11111fff  
password: 11111fff1

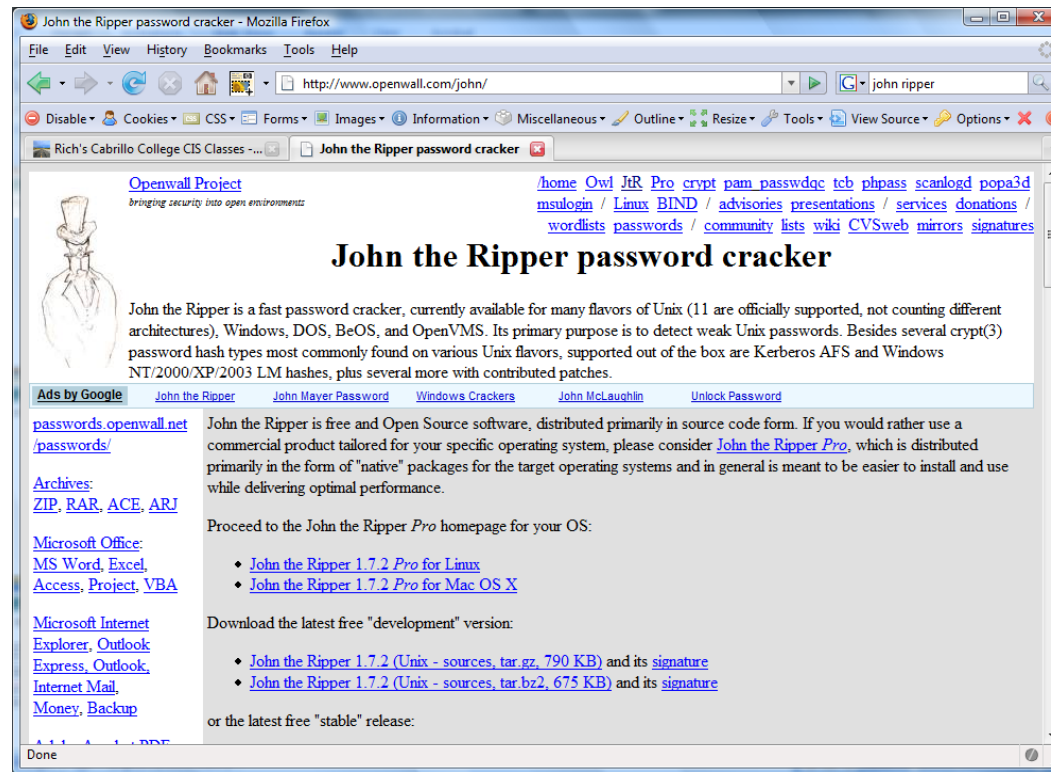
*The digit 1 (one)*

*Don't have an account yet? They get created (manually) after you submit Lab 1 which is due tonight.*

Turn recording back ON

# John the Ripper

*An open source cracker that tries common passwords first followed by a brute force dictionary attack*



*john-1.7.2/run/password.lst has most popular passwords to try first*

# Housekeeping

## Can I add this class?

- 8 add codes given out during the first week (only 5 used)
- If a student receives, but does not use an add code, **the add code will be given to the next student** on the wait list. Add codes can only be used once.
- Enrolled and wait-listed students that don't show up for class **will be dropped or lose their space on the wait list** unless they have made prior arrangements with the instructor.
- Between 9/1 and 9/10, the instructor will email add codes to students on the waitlist as spaces become available. The last day for students to add CIS 90 is 9/10.
- CIS 90 will be offered again next term, so students may have to wait.

Turn OFF the recording



# Roll Call

Enrolled and wait-listed students that don't show up for class **will be dropped or lose their space on the wait list** unless they have made prior arrangements with the instructor.

Turn recording back ON

CIS 90 – Code Names  
Lord of the Rings Characters

Current Progress					
Code Name	Grading Choice	Q1	Q2	Q3	Q4
Max Points		3	3	3	3
aragorn	Grade				
arwen	Grade				
balrog	Grade				
boromir	Grade				
denethor	Grade				
dwalin	Grade				
elrond	Grade				
eomer	Grade				
eowyn	Grade				
faramir	Grade				
frodo	Grade				
galadriel	Grade				
gimli	Grade				
glorfindel	Grade				
ioreth	Grade				
legolas	Grade				
lobelia	Grade				
nazgul	Grade				
pippin	Grade				
saruman	Grade				
sauron	Grade				
theoden	Grade				
treebeard	Grade				

*Everyone who is enrolled for this course will be assigned a code name.*

*I will use your grading choice on the survey you send me (you can change your mind later)*

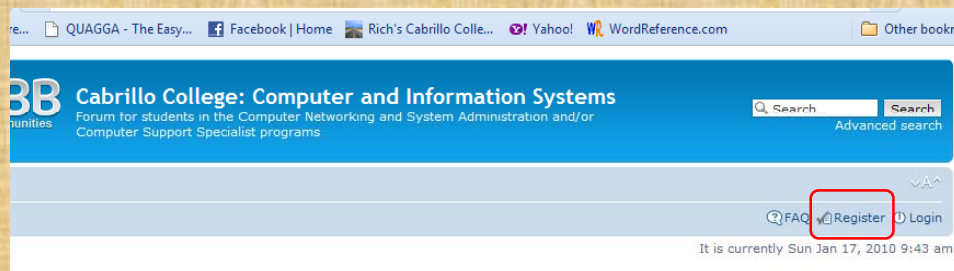
*Email me after Thursday for your code name.*

## Class Activity Forum Registration


There is a Forums link on [simms-teach.com](http://simms-teach.com)



Or browse to [opus.cabrillo.edu/forum](http://opus.cabrillo.edu/forum)



To Register:

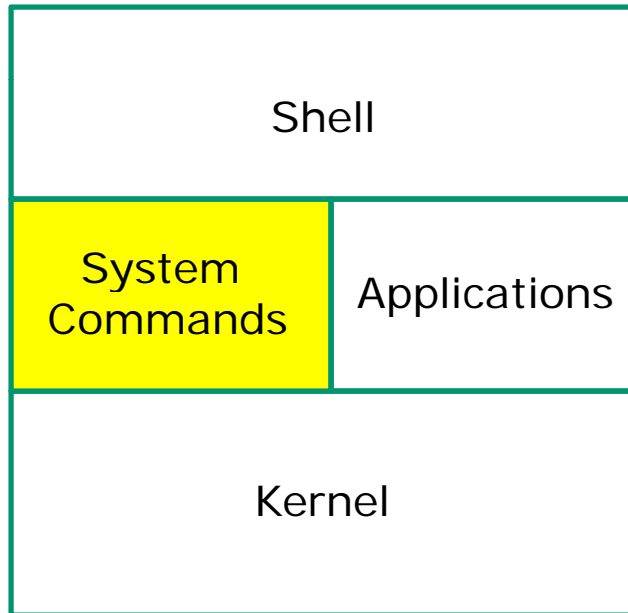
1. Browse to the forum
2. Click on  Register
3. Review and agree to terms
4. Your **Username** must be:
  - your **first and last name separated by a space** e.g. Rich Simms
  - match a name on the class roster

*Note: If you need your forum user name modified, please email the instructor.*

# Programs

# UNIX/Linux Architecture

## System Commands



- 100's of system commands and utilities .
- Commands like **ls** (list directories), **cat** (print a file), **rm** (remove a file), ... etc.
- Utilities like **vi** (text editor), **sort** (sorts file contents), **find** (searches), ... etc.
- Larger utilities like **sendmail** (email), **tar** (backup), **tcpdump** (sniffer), ... etc.
- Administrative utilities like **useradd**, **groupadd**, **passwd** (change password), ... etc.



## Introducing some new commands for this lesson

<b>apropos</b> <i>command</i>	<i>Looks up references in the whatis database</i>
<b>cat</b> <i>filename</i>	<i>print a file (from concatenate)</i>
<b>cd</b> <i>path</i>	<i>Change to a new directory</i>
<b>echo</b> <i>string</i>	<i>Print string (on screen)</i>
<b>file</b> <i>filename</i>	<i>Show file information</i>
<b>ls</b> <i>path</i>	<i>List files in a directory</i>
<b>type</b> <i>command</i>	<i>Shows where command resides on the path</i>
<b>bc</b>	<i>Binary calculator</i>

# Programs

## Executable binary code or scripts

Use *ls /bin* to show files in the /bin directory

```
rsimms@nosmo:~/depot/gcal-3.01/src
[rsimms@nosmo src]$ ls /bin
alsaunmute  dnsdomainname  kbd_mode  nisdomainname  sync
arch         doexec          keyctl     pgawk           tar
ash          domainname     kill       ping            tcsh
ash.static   dumpkeys       ksh        ping6           touch
awk          echo            link       ps              tracepath
basename    ed              ln         pwd             tracepath6
bash         egrep          loadkeys  red             traceroute
bsh         env            login     rm              traceroute6
cat          ex             ls         rmdir           true
chgrp       false          mail       rpm             umount
chmod       fgrep          mailx     rvi             uname
chown       gawk           mkdir     rview          unicode_start
cp          gettext        mknod     sed             unicode_stop
cpio        grep           mktemp    setfont        unlink
csh         gtar           more      setserial      usleep
cut         gunzip         mount     sh              vi
date        gzip           mt         sleep           view
dd          hostname      mv         sort            ypdomainname
df          igawk         netstat   stty            zcat
dmesg       ipcalc        nice      su
```

Note, all UNIX/Linux commands (programs) are stored as files.

Can you find the **date**, **hostname**, and **ps** commands we used in Lab 1?

Can you find the **bash** shell?

Which files are programs? The simple answer is "they are the green and bright red ones .. you can run them"



# Programs

## Executable binary code or scripts

Use *ls /usr/bin* to show files in the */usr/bin* directory

```

rsimms@opus:~$ ls /usr/bin
[
411toppm          htdbm             ppmforge
a2p               htdigest          ppmglobe
a2ps              htmlview          ppmhist
ab                httpasswd         ppmlabel
ac                i386              ppmmake
aclocal           i386-redhat-linux-c++  ppmmix
aclocal-1.4       i386-redhat-linux-g++ ppmnorm
aclocal-1.5       i386-redhat-linux-gcc ppmntsc
aclocal-1.6       icc2ps            ppmpat
aclocal-1.7       icclink           ppmquant
aclocal-1.9       icotrans          ppmquantall
aconnect          iceauth           ppmrainbow
acpi_listen       icontopbm        ppmrelief
activation-client iconv              ppmrough
addftinfo         id                ppmshadow
addr2line         ident             ppmshift
afs5log           identify          ppmspread
alacarte          idn               ppmtocad
alsamixer         iecset           ppmtoarbtxt
amidi             ifnames          ppmtobmp
amixer            ilbmtoppm        ppmtogyuv
amtu              imake             ppmtogif
amuFormat.sh     im-chooser       ppmtocr
animate           imgtoppm         ppmtoilbm
antlr             import            ppmtjpeg
antlr-java        includes          ppmtoleaf
anytopnm          indent            ppmtolj
aplay             indxib           ppmtolss16
aplaymidi        info              ppmtomap
apm               infocmp          ppmtomitsu
apmsleep          infokey          ppmtompeg
apropos          infotocap        ppmtoneo
ar                infotopam        ppmtopcx
ar86              innochecksum     ppmtopgm
arecord           install          ppmtopil
arecordmidi      install-catalog  ppmtopic
                  instmodsh        ppmtopj

```

There are a "ton" of commands (programs) in this directory.

You will need to scroll through a lot of pages to see them all!

Can you find the *id* command we used in Lab 1?

## Programs

Executable binary code or scripts

Lets take a deep dive on two random commands:

**apropos** and **cal**

**apropos** - searches the whatis database for a string of text

**cal** - prints a calendar

*We will be using the **ls**, **file**, **cat** and **type** commands to learn more about the **apropos** and **cal** commands*

*I'll be using this graphic to indicate a program that has been loaded into memory to be run*





## Programs

Executable binary code or scripts



`apropos`

`cal`

```
[rsimms@nosmo src]$ apropos uname
oldolduname [obsolete] (2) - obsolete system calls
olduname [obsolete] (2) - obsolete system calls
uname (1p) - return system name
uname (1) - print system information
uname (2) - get name and information about current kernel
uname (3p) - get the name of the current system
uuname (1) - List the names of the known remote UUCP sites
```

*Use **apropos** to look up a reference in the **whatis** database.*

```
[rsimms@nosmo src]$ cal
```

```
      July 2008
Su Mo Tu We Th Fr Sa
      1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31
```

*Use **cal** to print a calendar*

# Programs

Executable binary code or scripts



apropos



cal

*Note, the **ls** command shows both **apropos** and **cal** are in the **/usr/bin** directory. They show as green because they are programs.*

```

simmsben@opus:/usr/bin
/usr/bin $ ls apropos cal
apropos cal
/usr/bin $
/usr/bin $
/usr/bin $ ls -l apropos cal
-rwxr-xr-x 1 root root 1786 Jul 12 2006 apropos
-rwxr-xr-x 1 root root 18764 Mar  3 10:43 cal
/usr/bin $
/usr/bin $ █
    
```

*Using the **-l** option on the **ls** command prints a "long listing" that shows additional information. The **x**'s indicate the execute permission bits are set.*

## Programs

Executable binary code or scripts



apropos



cal

The **file** command shows that **apropos** is a shell script and **cal** is binary code (has been compiled)

```
simmsben@opus:/usr/bin
/usr/bin $ file apropos
apropos: Bourne shell script text executable
/usr/bin $
/usr/bin $
/usr/bin $
/usr/bin $ file cal
cal: 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
/usr/bin $ █
```

# Programs

Executable binary code or scripts



```

simmsben@opus:/usr/bin
/usr/bin $ cat apropos
#!/bin/sh
#
# apropos -- search the whatis database for keywords.
# whatis -- idem, but match only commands (as whole words).
#
# Copyright (c) 1990, 1991, John W. Eaton.
# Copyright (c) 1994-1999, Andries E. Brouwer.
#
# You may distribute under the terms of the GNU General Public
# License as specified in the README file that comes with the man
# distribution.
#
# apropos/whatis-1.5m aeb 2003-08-01 (from man-1.6d)
#
# keep old PATH - 000323 - Bryan Henderson
# also look in /var/cache/man - 030801 - aeb

program=`basename $0`

# When man pages in your favorite locale look to grep like binary files
# (and you use GNU grep) you may want to add the 'a' option to *grepopt1.
ap
ap
wh
wh
gr
gr
if
then
echo "usage: $program keyword ..."
exit 1
fi

manpath='man --path | tr : '\040''

if [ "$manpath" = "" ]
then
echo "$program: manpath is null"
exit 1

```

The **cat** command can print the **apropos** file because it is a readable **ASCII** script

```

simmsben@opus:/usr/bin
/usr/bin $ cat cal
ELF4tD4(4444440909090040%i9iiBDHhH PâtdE6ÊÊQ&td/lib/ld-linux.so.2GNU libn
curses.so.5_gmon_start__Jv_RegisterClassesgetent_fini_inittputstgetstrlib
c.so.6_IO_stdin_usedstrcpy_printf_chkexit_IO_putcsetlocaleoptindstrchr_sw
printf_chk_prognamecdgettextstrncpymbstowcs_stack_chk_failputcê0i3Â-EI*9"
IK'y^c"HU" dp8C28FîÁ'F*298Nôy°âÿiâÿ' 8`$*(CE0i4Pv4iE-KâA80qX'memcpy_strt
ifÿB@ternaln1_langinfogetenv_q ype_b_locstderr_snprintf_chklocaltime_vfpr
intf_chkwcstombs_sprintf_châO ndtextdomain_libc_start_main_edata_bss_star
t_endGLIBC_2.3GLIBC_2.3.4GâR C_2.4GLIBC_2.0libdl.so.2/lib/ld-linux.so.2qFXHÊ
ç^VSFXH QLû.SFXHRB]F9SFX'I ẽ:iÊ;Ûÿÿÿ; ;üÿÿÿ°;Ëÿÿÿ;DÿÿÿÛ; ;$48;Øÿÿÿ<;ØÿÿÿL;h
; ;øÿÿÿ-;âÿÿÿ';øÿÿÿÛ;ÿÿÿÛ;
$43*ID00Üâë 1-8°
6°
ø°
*°
$(,04»
<@D
é é
üh'
héâ
ÿÿ*
ÿÿÿ
Ph'
Ötê
9Ãv
ÿ
â;
4'Uâ;è°
Ât. t6
ÂtwèZÄ[]Ç$èÿDÉÁV$SLS
t$ü,1AöÄâp~kA4°`ÄHÄ9óuó[È^ÄöÄt"ø°
èQ=è0ÄöÄü)ÄiÄi090ÄèuqWVSi
èQ=è0ÄöÄü)Ä,kòd90uè%

```

The **cat** command "chokes" trying to print the **binary** cal file.  
That's because binary files contain unprintable characters.

# Programs

Executable binary code or scripts



From: gcal-3.01.tar.gz

```
rsimms@nosmo:~/depot/gcal-3.01/src
[rsimms@nosmo src]$ head -50 gcal.c
/*
 * gcal.c: Main part which controls the extended calendar program.
 *
 * Copyright (c) 1994, 95, 96, 1997, 2000 Thomas Esken
 *
 * This software doesn't claim completeness, correctness or usability.
 * On principle I will not be liable for ANY damages or losses (implicit
 * or explicit), which result from using or handling my software.
 * If you use this software, you agree without any exception to this
 * agreement, which binds you LEGALLY !!
 *
 * This program is free software; you can redistribute it and/or modify
 * it under the terms of the 'GNU General Public License' as published by
 * the 'Free Software Foundation'; either version 2, or (at your option)
 * any later version.
 *
 * You should have received a copy of the 'GNU General Public License'
 * along with this program; if not, write to the:
 *
 * Free Software Foundation, Inc.
 * 59 Temple Place - Suite 330
 * Boston, MA 02111-1307, USA
 */
static char rcsid[]="$Id: gcal.c,v 1.1 1997/01/01 00:00:00; $"

/*
 * Include header files.
 */
#include "tailor.h"
#ifdef HAVE_ASSERT_H
#include <assert.h>
#endif
#endif
```

*Note: The **cal** binary code resulted from compiling the original **gcal.c** source code.*

```
rsimms@nosmo:~/depot/gcal-3.01/src
[rsimms@nosmo src]$ file /usr/bin/cal
/usr/bin/cal: ELF 32-bit LSB executable, Intel 80386, version 1
(SYSV), for GNU/Linux 2.2.5, dynamically linked (uses shared lib
s), stripped
[rsimms@nosmo src]$ █
```

*Because GNU Linux software is licensed under the GPL you can make your own custom version of the commands or the kernel!*

cal



## Programs

Executable binary code or scripts

*Use the type command to locate where a command resides*

```
[rsimms@opus run]$ type uname  
uname is /bin/uname  
[rsimms@opus run]$ type cal  
cal is /usr/bin/cal  
[rsimms@opus run]$ type uname cal  
uname is /bin/uname  
cal is /usr/bin/cal
```

*name of the file (command/program)*

*name of the directory where file is found*





## Class Exercise Programs

```
/home/cis90/guest $ apropos uname  
/home/cis90/guest $ cal  
/home/cis90/guest $ type uname cal
```

*Issue these commands and compare what you get with the previous slides.*

```
/home/cis90/guest $ cd /usr/bin  
/usr/bin $ ls apropos cal  
/usr/bin $ ls -l apropos cal  
/usr/bin $ file apropos cal  
/usr/bin $ cat apropos  
/usr/bin $ cat cal  
/usr/bin $ reset
```

*Do you know the name of the directory where the cal and apropos commands are kept?*

*Do you know which program is a ASCII text script and which is a binary executable?*

# Inputs to commmands (programs)

*You will get these questions when you submit Lab 2*

Name a UNIX command that gets its input only from the command line?

Name an interactive command that reads its input from the keyboard?

Name a UNIX command that gets its input from the Operating System?

Name a UNIX command that gets its input only from the command line?

```
/home/cis90/simmsben $ echo hello world  
hello world
```

*The **echo** command is an example of a command that gets its input from the command line*

Name a UNIX command that gets its input only from the command line?

```

/home/cis90/simmsben $ banner hello world
#           # ##### #           #           #####
#           # #           #           #           # #
#           # #           #           #           # #
##### #           #           #           #           #
#           # #           #           #           # #
#           # #           #           #           # #
#           # ##### ##### ##### #####
#           # ##### #           #           #####
# # # #           # #           # #           # #
# # # #           # #           # #           # #
# # # #           # ##### #           #           #
# # # #           # #           #           #           #
# # # #           # #           #           #           #
## ## ##### #           # ##### #####

```

The **banner** command is an example of a command that gets its input from the command line

Name an interactive command that reads its input from the keyboard?

```
/home/cis90/simmsben $ bc
bc 1.06
Copyright 1991-1994, 1997, 1998, 2000 Free Software Foundation, Inc.
This is free software with ABSOLUTELY NO WARRANTY.
For details type `warranty'.
2+2
4
500-200+3
303
sqrt(64)
8
quit
```

*The **bc** (binary calculator) command is an example of an interactive command that reads its input from the keyboard*

Name an interactive command that reads its input from the keyboard?

```
/home/cis90/simmsben $ passwd
Changing password for user simmsben.
Changing password for simmsben
(current) UNIX password:
New UNIX password:
BAD PASSWORD: is too similar to the old one
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
/home/cis90/simmsben $
```

*The **passwd** command is an example of an interactive command that reads its input from the keyboard*



Name a UNIX command that gets its input from the Operating System?

```

/home/cis90/simmsben $ who
dycktim pts/1      2010-09-07 17:07 (nosmo-nat.cabrillo.edu)
root    :0          2009-12-18 17:30
velasoli pts/2      2010-09-07 17:08 (adsl-75-41-114-88.dsl.pltn13.sbcglobal.net)
guest90 pts/3      2010-09-07 16:56 (nosmo-nat.cabrillo.edu)
rsimms  pts/4      2010-09-07 15:54 (dsl-63-249-103-107.dhcp.cruzio.com)
guest90 pts/5      2010-09-07 16:59 (nosmo-nat.cabrillo.edu)
watsohar pts/6      2010-09-07 17:03 (nosmo-nat.cabrillo.edu)
swansgre pts/7      2010-09-07 17:10 (nosmo-nat.cabrillo.edu)
guest90 pts/8      2010-09-07 17:10 (nosmo-nat.cabrillo.edu)
abbenste pts/9      2010-09-07 17:11 (nosmo-nat.cabrillo.edu)
/home/cis90/simmsben $

```

*The **who** command is an example of a command that gets its input from the Operating System*

Name a UNIX command that gets its input from the Operating System?

```
/home/cis90/simmsben $ uname  
Linux  
/home/cis90/simmsben $
```

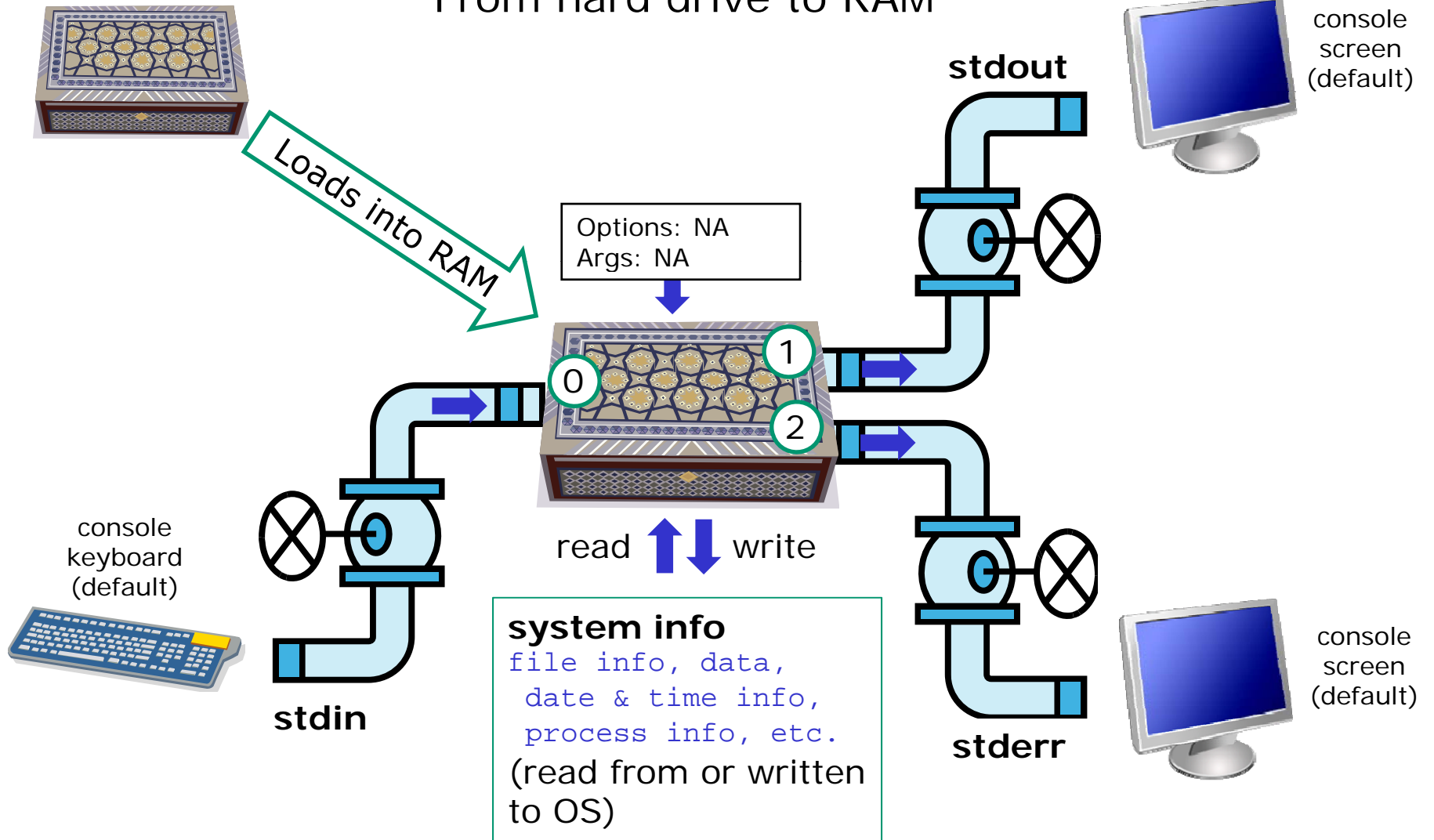
*The **uname** command is an example of a command that gets its input from the Operating System*

# Program to Process

The next slides are a preview of future lessons on processes ... for now just you don't need to understand all the ins and outs of how this works.

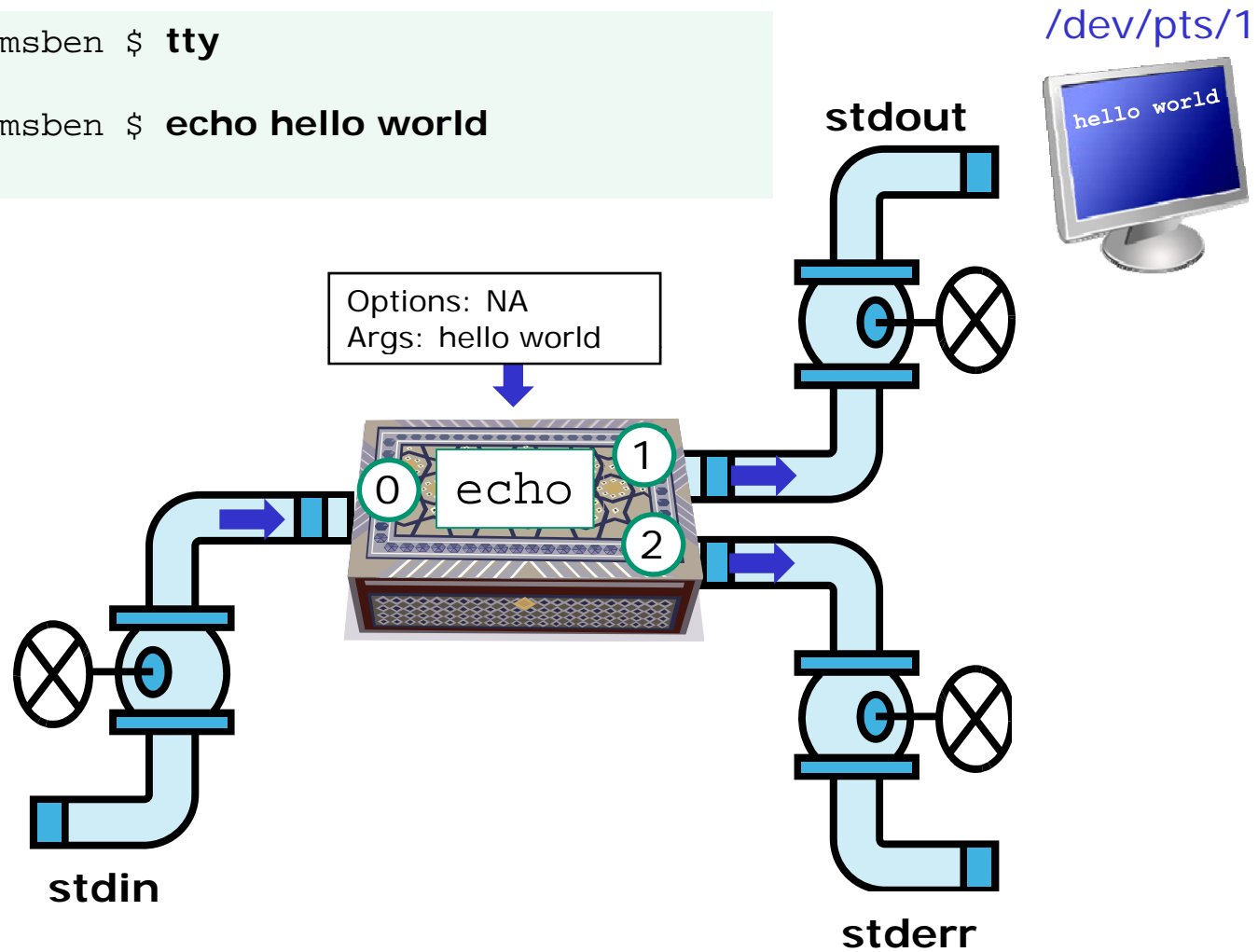
Program  
(a file on drive)

Program to Process  
From hard drive to RAM



### Example program to process: Is command

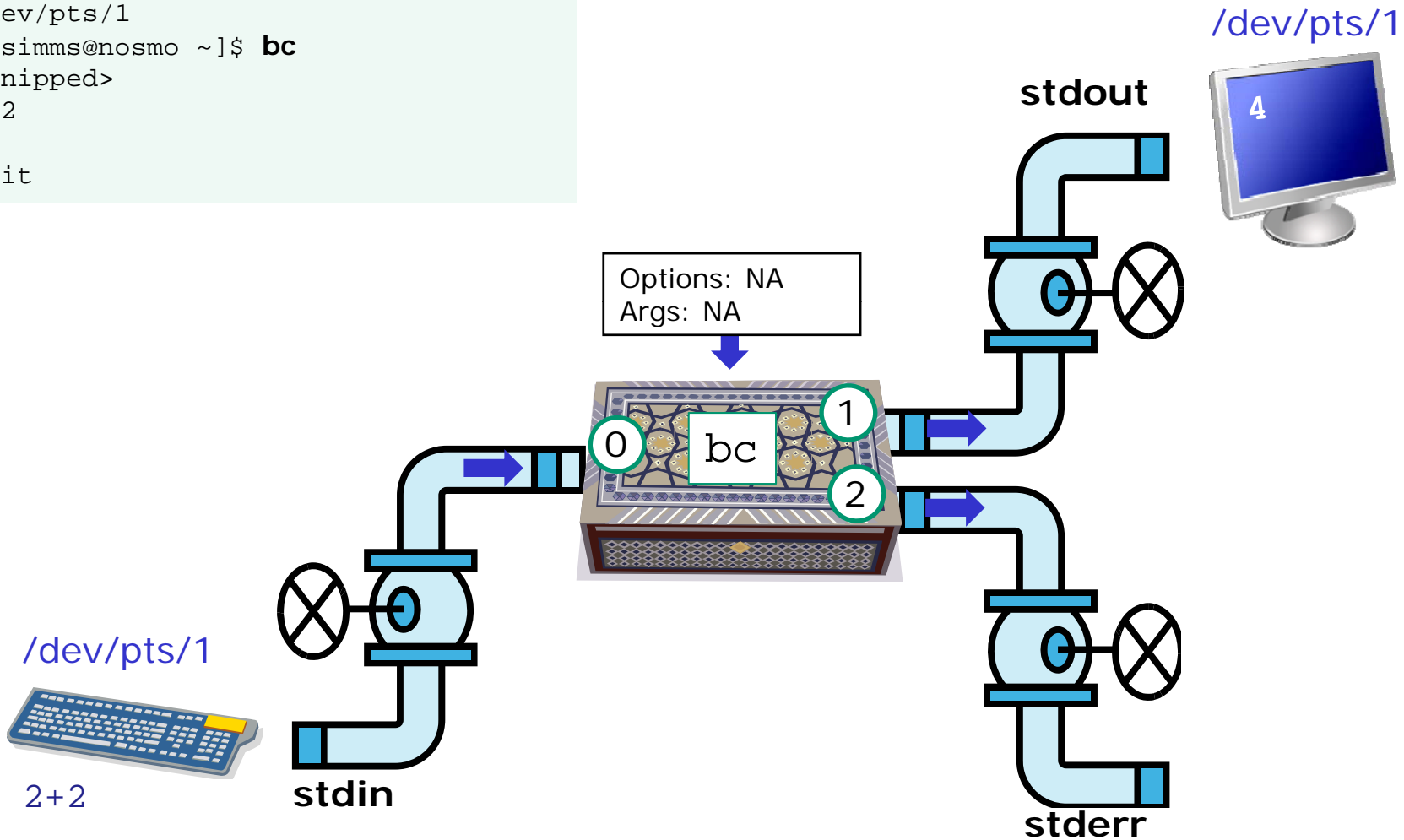
```
/home/cis90/simmsben $ tty  
/dev/pts/1  
/home/cis90/simmsben $ echo hello world  
hello world
```



*The **echo** command is an example of a command that gets its input from the command line*

Example program to process: bc command

```
[rsimms@nosmo ~]$ tty
/dev/pts/1
[rsimms@nosmo ~]$ bc
<snipped>
2+2
4
quit
```



*The **bc** (binary calculator) command is an example of an interactive command that reads its input from the keyboard*

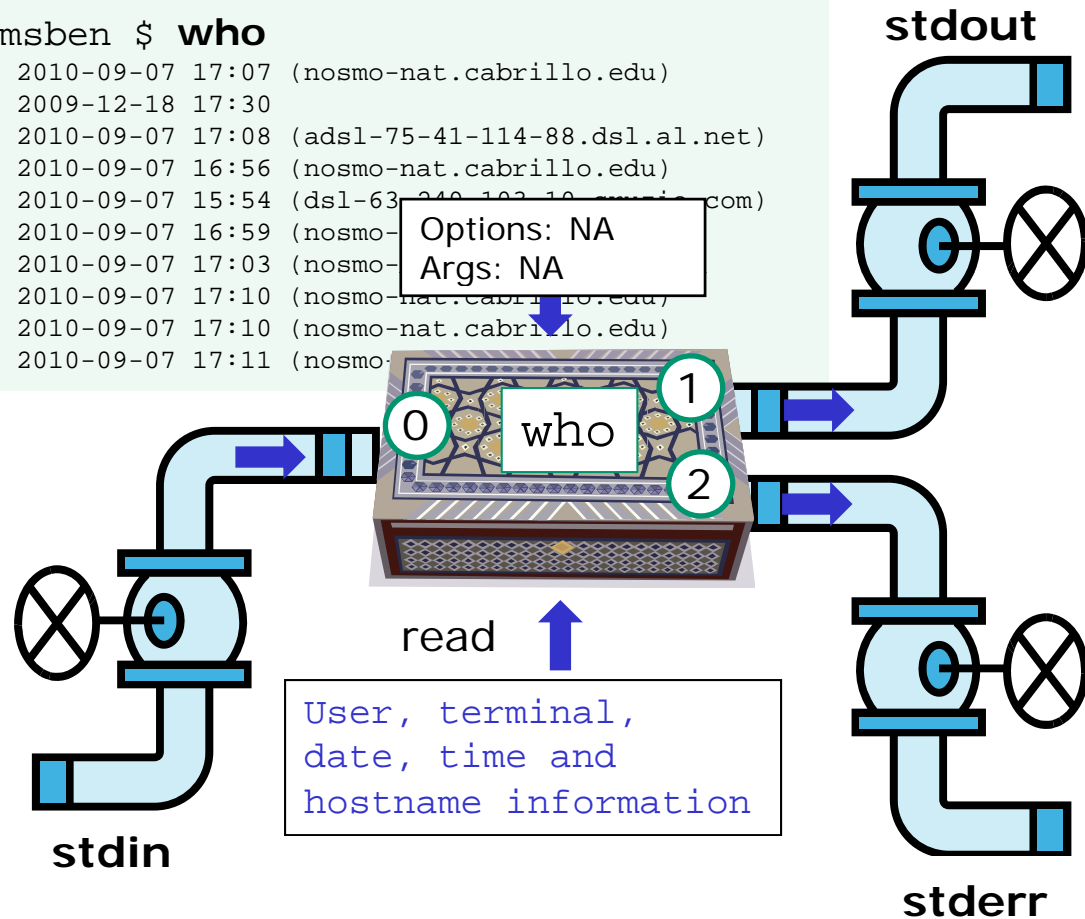
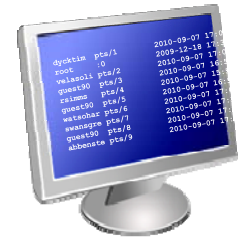
Example program to process: Is command

```
/home/cis90/simmsben $ tty
/dev/pts/1
```

```
/home/cis90/simmsben $ who
```

```
dycktim pts/1      2010-09-07 17:07 (nosmo-nat.cabrillo.edu)
root    :0          2009-12-18 17:30
velasoli pts/2     2010-09-07 17:08 (adsl-75-41-114-88.dsl.al.net)
guest90 pts/3     2010-09-07 16:56 (nosmo-nat.cabrillo.edu)
rsimms  pts/4     2010-09-07 15:54 (dsl-63-240-102-10.cable.comcast.com)
guest90 pts/5     2010-09-07 16:59 (nosmo-nat.cabrillo.edu)
watsohar pts/6    2010-09-07 17:03 (nosmo-nat.cabrillo.edu)
swansgre pts/7    2010-09-07 17:10 (nosmo-nat.cabrillo.edu)
guest90 pts/8    2010-09-07 17:10 (nosmo-nat.cabrillo.edu)
abbenste pts/9    2010-09-07 17:11 (nosmo-nat.cabrillo.edu)
```

/dev/pts/1



The **who** command is an example of a command that gets its input from the Operating System



## Class Exercise Running Programs

1. What console device are you on? (use **tty**)
2. List the files in your current directory (use **ls** command). Where did the **ls** process get this file information?
3. Run the calculator program (the **bc** command).
  - Add  $2 + 2$
  - Multiply  $5 * 7$
  - Divide  $5 / 0$
  - Quit

Where does the **bc** program get its input from?

# Command Syntax

## Command Syntax

Command

Options

Arguments

Redirection

**Command** – is the name of an executable program file.

**Options** – various options which control how the program will operate.

**Arguments** – the objects the command is directed to work upon. Multiple arguments are separated by spaces.

**Redirection** – The default input stream (stdin) is from the console keyboard, the default output (stdout) and error (stderr) streams go to the console screen. Redirection can modify these streams to other files or devices.

## Command Syntax

Command	Options	Arguments	Redirection
clear			
hostname			
hostname	-i		
hostname	-s		
ps			
ps	-e		
ps	-F		
ps	-e -F		
ps	-eF		
ls			
ls	-l		
ls		/Blake	
ls	-l	/Blake	
ls	-l	/Blake	> blakepoems
echo		Red white Blue	

# Command Syntax



```
[root@opus ~]# hostname  
opus.cabrillo.edu
```

```
[root@opus ~]# hostname -i  
127.0.0.1
```

```
[root@opus ~]# hostname -s  
opus
```

# Command Syntax

Command

Options

Arguments

Redirection

[root@opus ~]# **ps** *Using the ps command with no options*

PID	TTY	TIME	CMD
14801	pts/0	00:00:00	bash
15728	pts/0	00:00:00	ps

[rsimms@opus ~]\$ **ps -F** *Using the ps command with the -F (extra full format) option*

UID	PID	PPID	C	SZ	RSS	PSR	STIME	TTY	TIME	CMD
rsimms	14801	14800	0	1165	1452	0	06:50	pts/0	00:00:00	-bash
rsimms	15729	14801	0	1061	928	1	13:47	pts/0	00:00:00	ps -F

[rsimms@opus ~]\$ **ps -e** *Using the ps command with the -e (all processes) option*

PID	TTY	TIME	CMD
1	?	00:00:05	init
2	?	00:00:00	migration/0
3	?	00:00:00	ksoftirqd/0
4	?	00:00:00	watchdog/0
5	?	00:00:00	migration/1
6	?	00:00:00	ksoftirqd/1
7	?	00:00:00	watchdog/1
8	?	00:00:00	events/0

< snipped >

# Command Syntax

Command

Options

Arguments

Redirection

```
[rsimms@opus ~]$ ps -e -F Using the ps command with 2 options (separated)
UID      PID  PPID  C   SZ  RSS  PSR  STIME  TTY      TIME  CMD
root      1    0    0  515 628   0   2008 ?        00:00:07  init [3]
root      2    1    0    0    0    0   2008 ?        00:00:00  [migration/0]
root      3    1    0    0    0    0   2008 ?        00:00:00  [ksoftirqd/0]
root      4    1    0    0    0    0   2008 ?        00:00:00  [watchdog/0]
root      5    1    0    0    0    1   2008 ?        00:00:00  [migration/1]
root      6    1    0    0    0    1   2008 ?        00:00:00  [ksoftirqd/1]
< snipped >
```

```
[rsimms@opus ~]$ ps -eF Using the ps command with 2 options (combined)
UID      PID  PPID  C   SZ  RSS  PSR  STIME  TTY      TIME  CMD
root      1    0    0  515 628   0   2008 ?        00:00:07  init [3]
root      2    1    0    0    0    0   2008 ?        00:00:00  [migration/0]
root      3    1    0    0    0    0   2008 ?        00:00:00  [ksoftirqd/0]
root      4    1    0    0    0    0   2008 ?        00:00:00  [watchdog/0]
root      5    1    0    0    0    1   2008 ?        00:00:00  [migration/1]
root      6    1    0    0    0    1   2008 ?        00:00:00  [ksoftirqd/1]
< snipped >
```

*Note: options can be combined to save a little typing*

## Command Syntax

Command

Options

Arguments

Redirection

```
/home/cis90/simmsben/Poems $ ls no options or arguments  
ant Blake nursery Shakespeare twister Yeats
```

```
/home/cis90/simmsben/Poems $ ls -l 1 option and no arguments  
total 48  
-rw-r--r-- 1 simmsben cis90 237 Aug 26 2003 ant  
drwxr-xr-x 2 simmsben cis90 4096 Jul 20 2001 Blake  
-rw-r--r-- 1 simmsben cis90 779 Oct 12 2003 nursery  
drwxr-xr-x 2 simmsben cis90 4096 Oct 31 2004 Shakespeare  
-rw-r--r-- 1 simmsben cis90 151 Jul 20 2001 twister  
drwxr-xr-x 2 simmsben cis90 4096 Jul 20 2001 Yeats  
/home/cis90/simmsben/Poems $
```

*The -l option on ls provides a "long listing" showing additional file information*



## Command Syntax

Command

Options

Arguments

Redirection

```
/home/cis90/simmsben/Poems $ ls Blake/ no options and 1 argument
jerusalem tiger
```

```
/home/cis90/simmsben/Poems $ ls -l Blake/ 1 option and 1 argument
total 16
-rw-r--r-- 1 simmsben cis90 582 Jul 20 2001 jerusalem
-rw-r--r-- 1 simmsben cis90 115 Jul 20 2001 tiger
```

```
/home/cis90/simmsben/poems $ ls -l Blake Yeats 1 option and 2 arguments
Blake:
total 16
-r--r--r-- 1 guest90 cis90 582 Jul 20 2001 jerusalem
-r--r--r-- 1 guest90 cis90 115 Jul 20 2001 tiger
```

```
Yeats:
total 24
-r--r--r-- 1 guest90 cis90 855 Jul 20 2001 mooncat
-r--r--r-- 1 guest90 cis90 520 Jul 20 2001 old
-r--r--r-- 1 guest90 cis90 863 Jul 20 2001 whitebirds
```

*Note: Multiple arguments are separated by spaces*

## Command Syntax

Command

Options

Arguments

Redirection

```
/home/cis90/simmsben/Poems $ ls -l Blake/ > blakepoems
```

*1 options, 1 argument  
and some redirection*

```
/home/cis90/simmsben/Poems $ cat blakepoems
```

*0 options, 1 argument*

```
total 16
```

```
-rw-r--r-- 1 simmsben cis90 582 Jul 20 2001 jerusalem
```

```
-rw-r--r-- 1 simmsben cis90 115 Jul 20 2001 tiger
```

## Class Exercise Command Line

```
clear
```

```
hostname
```

```
hostname -i
```

```
hostname -s
```

```
ps
```

```
ps -e
```

```
ps -F
```

```
ps -e -F
```

```
ps -eF
```

```
ls
```

```
ls -l
```

```
ls /bin
```

```
ls -l /bin
```

```
ls -lS /bin
```

```
ls -ls /bin > yourlastname
```

```
cat yourlastname
```

Try these commands  
out on your computer

# Environment Variables

## echo command

*echo prints the arguments supplied on the command line*

```
[rsimms@opus run]$ echo hello
hello
[rsimms@opus run]$ echo "My name is Rich"
My name is Rich
[rsimms@opus run]$ echo LOGNAME
LOGNAME
[rsimms@opus run]$ echo $LOGNAME
rsimms
```

*What is the deal with \$LOGNAME ???  
... it is something called a **variable***

## variables

**\$LOGNAME**

*LOGNAME is a predefined variable that is set by the system to be your username*

*The \$ is special metacharacter and it means "the value of"*

## Variables

A little tiny bit of "programming" now

*Think of variables as named boxes and the \$ in front of a variable name means "the contents of"*

```
$ echo $LOGNAME
simmsben
```

```
$ echo $HOSTNAME
opus.cabrillo.edu
```

```
$ echo $HOME
/home/cis90/simmsben
```

```
$ echo $SHELL
/bin/bash
```



## Shell (Environment) Variables

common environment variables

Shell Variable	Description
HOME	Users home directory (starts here after logging in and returns with a <code>cd</code> command (with no arguments))
LOGNAME	User's username for logging in with.
PATH	List of directories, separated by ':'s, for the Shell to search for commands (which are program files) .
PS1	The prompt string.
PWD	Current working directory
SHELL	Name of the Shell program being used.
TERM	Type of terminal device , e.g. dumb, vt100, xterm, ansi, etc.



# Shell (Environment) Variables

## common environment variables

Shell Variable	Description
TERM	Type of terminal device , e.g. dumb, vt100, xterm, ansi, etc.

```

guest90@opus:~/poems
login as: guest90
guest90@opus.cabrillo.edu's password:
Last login: Wed Sep  8 06:56:57 2010 from adsl-71-146-19-45.dsl.pltn13.sbcgloba
.net

      _
     ('v')
    //--\\
   (\_=_/)
    ~ ~

  Welcome to Opus
  Serving Cabrillo College

Terminal type? [xterm]
Terminal type is xterm.
/home/cis90/guest $ ls

```

*Note the TERM variable gets set every time we log into Opus*

# Shell (Environment) Variables

## env command

```
/home/cis90/simmsben/Poems $env
```

```
HOSTNAME=opus.cabrillo.edu  
SHELL=/bin/bash  
TERM=xterm
```

```
HISTSIZE=1000
```

```
USER=simmsben
```

```
LS_COLORS=no=00:fi=00:di=00;34:ln=00;36:pi=40;33:so=00;35:bd=40;33;01:cd=40;33;01:or=01;05;37;41:mi  
=01;05;37;41:ex=00;32:*.cmd=00;32:*.exe=00;32:*.com=00;32:*.btm=00;32:*.bat=00;32:*.sh=00;32:*.csh=  
00;32:*.tar=00;31:*.tgz=00;31:*.arj=00;31:*.taz=00;31:*.lzh=00;31:*.zip=00;31:*.z=00;31:*.Z=00;31:*.  
.gz=00;31:*.bz2=00;31:*.bz=00;31:*.tz=00;31:*.rpm=00;31:*.cpio=00;31:*.jpg=00;35:*.gif=00;35:*.bmp=  
00;35:*.xbm=00;35:*.xpm=00;35:*.png=00;35:*.tif=00;35:
```

```
USERNAME=
```

```
MAIL=/var/spool/mail/simmsben
```

```
PATH=/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/home/cis90/simmsben/./bin:/home/cis90/simmsbe  
n/bin:.
```

```
INPUTRC=/etc/inputrc
```

```
PWD=/home/cis90/simmsben/Poems
```

```
LANG=en_US.UTF-8
```

```
SSH_ASKPASS=/usr/libexec/openssh/gnome-ssh-askpass
```

```
SHLVL=1
```

```
HOME=/home/cis90/simmsben
```

```
BASH_ENV=/home/cis90/simmsben/.bashrc
```

```
LOGNAME=simmsben
```

```
CVS_RSH=ssh
```

```
LESSOPEN=|/usr/bin/lesspipe.sh %s
```

```
G_BROKEN_FILENAMES=1
```

```
_/bin/env
```

```
OLDPWD=/home/cis90/simmsben
```

```
/home/cis90/simmsben/Poems $
```

*The **env** command shows all the environment variables used by the shell*

# Shell Variables

## set command

```
/home/cis90/simmsben/Poems $set
```

```
BASH=/bin/bash
BASH_ARGC=( )
BASH_ARGV=( )
BASH_ENV=/home/cis90/simmsben/.bashrc
BASH_LINENO=( )
BASH_SOURCE=( )
BASH_VERSINFO=( [0]="3" [1]="2" [2]="25" [3]="1"
[4]="release" [5]="i686-redhat-linux-gnu" )
BASH_VERSION='3.2.25(1)-release'
COLORS=/etc/DIR_COLORS.xterm
COLUMNS=80
CVS_RSH=ssh
DIRSTACK=( )
EUID=1160
GROUPS=( )
G_BROKEN_FILENAMES=1
HISTFILE=/home/cis90/simmsben/.bash_history
HISTFILESIZE=1000
HISTSIZ=1000
HOME=/home/cis90/simmsben
HOSTNAME=opus.cabrillo.edu
HOSTTYPE=i686
IFS=$' \t\n'
IGNOREEOF=10
INPUTRC=/etc/inputrc
LANG=en_US.UTF-8
LESSOPEN='|/usr/bin/lesspipe.sh %s'
LINES=24
LOGNAME=simmsben
```

```
LS_COLORS='no=00:fi=00:di=00;34:ln=00;36:pi=40;33:so=00;35
:bd=40;33;01:cd=40;33;01:or=01;05;37;41:mi=01;05;37;41:ex=
00;32:*.cmd=00;32:*.exe=00;32:*.com=00;32:*.bat=00;32:*.ba
t=00;32:*.sh=00;32:*.csh=00;32:*.tar=00;31:*.tgz=00;31:*.a
rj=00;31:*.taz=00;31:*.lzh=00;31:*.zip=00;31:*.z=00;31:*.Z
=00;31:*.gz=00;31:*.bz2=00;31:*.bz=00;31:*.tz=00;31:*.rpm=
00;31:*.cpio=00;31:*.jpg=00;35:*.gif=00;35:*.bmp=00;35:*.x
bm=00;35:*.xpm=00;35:*.png=00;35:*.tif=00;35:'
MACHTYPE=i686-redhat-linux-gnu
MAIL=/var/spool/mail/simmsben
MAILCHECK=60
OLDPWD=/home/cis90/simmsben
OPTERR=1
OPTIND=1
OSTYPE=linux-gnu
PATH=/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/home/
cis90/simmsben/./bin:/home/cis90/simmsben/bin:
PIPESTATUS=( [0]="0" )
PPID=26514
PROMPT_COMMAND='echo -ne
"\033]0;${USER}@${HOSTNAME%%.*}:${PWD/#$HOME/~}"; echo -ne
"\007"'
PS1='$PWD $'
PS2='> '
PS4='+ '
PWD=/home/cis90/simmsben/Poems
SHELL=/bin/bash
SHELLOPTS=braceexpand:emacs:hashall:histexpand:ignoreeof:i
nteractive-comments:monitor
SHLV=1
SSH_ASKPASS=/usr/libexec/openssh/gnome-ssh-askpass
TERM=xterm
UID=1160
USER=simmsben
USERNAME=
_=env
consoletype=pty
```

*The **set** command shows all the variables used by the shell and by the user*

## Environment variables

PATH, TERM, PS1, HOME

Use echo \$\_\_\_\_\_ to show value

```
[rsimms@nosmo ~]$ echo $PATH
/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin:/home/rsimms/bin

[rsimms@nosmo ~]$ echo $TERM
xterm

[rsimms@nosmo ~]$ echo $HOME
/home/rsimms

[rsimms@nosmo ~]$ echo $PS1
[\u@\h \W]\$
```

Use = (no spaces, no \$ sign) to change value

```
[rsimms@nosmo ~]$ PS1="By your command >"
By your command >
By your command >PS1="What can I do for you $LOGNAME? "
What can I do for you rsimms?
What can I do for you rsimms? PS1="[\u@\h \W]\$ "
[rsimms@nosmo ~]$
```

user name

hostname

working directory

# bash shell tip

## changing the prompt

Prompt Code	Meaning
\!	history command number
\#	session command number
\d	date
\h	hostname
\n	new line
\s	shell name
\t	time
\u	user name
\w	entire path of working directory
\W	only working directory
\\$	\$ or # (for root user)

The prompt string can have any combination of text, variables and these special codes.

# bash shell tip

## changing the prompt

Prompt string	Result
<code>PS1='\$PWD \$'</code>	<code>/home/cis90/simmsben/Poems \$</code>
<code>PS1="\w \$"</code>	<code>~/Poems \$</code>
<code>PS1="\W \$"</code>	<code>Poems \$</code>
<code>PS1="\u@\h \$"</code>	<code>simmsben@opus \$</code>
<code>PS1='\u@\h \$PWD \$'</code>	<code>simmsben@opus /home/cis90/simmsben/Poems \$</code>
<code>PS1='\u@\\$HOSTNAME \$PWD \$'</code>	<code>simmsben@opus.cabrillo.edu /home/cis90/simmsben/Poems \$</code>
<code>PS1='\u \! \$PWD \$'</code>	<code>simmsben 825 /home/cis90/simmsben/Poems \$</code>
<code>PS1="[\u@\h \W/\\$"</code>	<code>[simmsben@opus Poems/\$</code>
<code>PS1='\$PWD \$'</code>	<code>/home/cis90/simmsben/Poems \$</code>

*Important: Use single quotes around variables that change. For example if you use \$PWD with double quotes, the prompt will not changes as you change directories! More on this later ...*

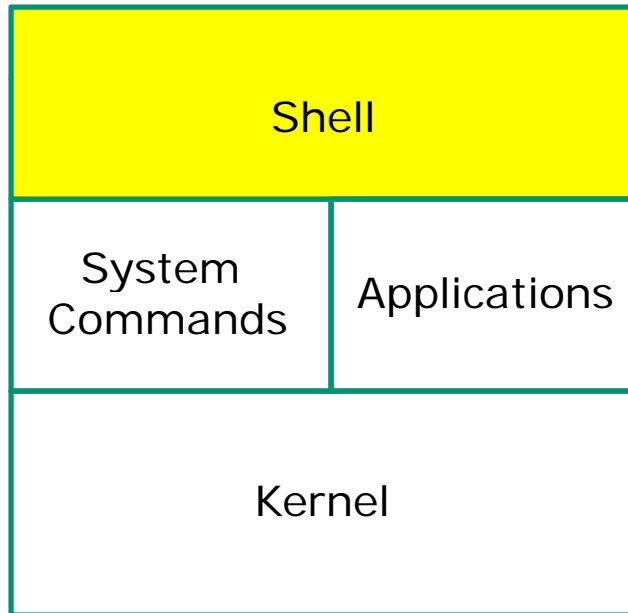
## Class Exercise Environment Variables

1. Print the name of your shell (hint **echo \$SHELL**)
2. Print your path (hint: the PATH variable)
3. Print your username (hint: the LOGNAME variable)
4. Change your prompt to "What is your command master? "
5. Change your prompt to "[\u@\h \W]\\$"
6. Print all of your environment variables.
7. What kind of terminal device are you using? (hint: the TERM variable)

# Shell



## The Shell

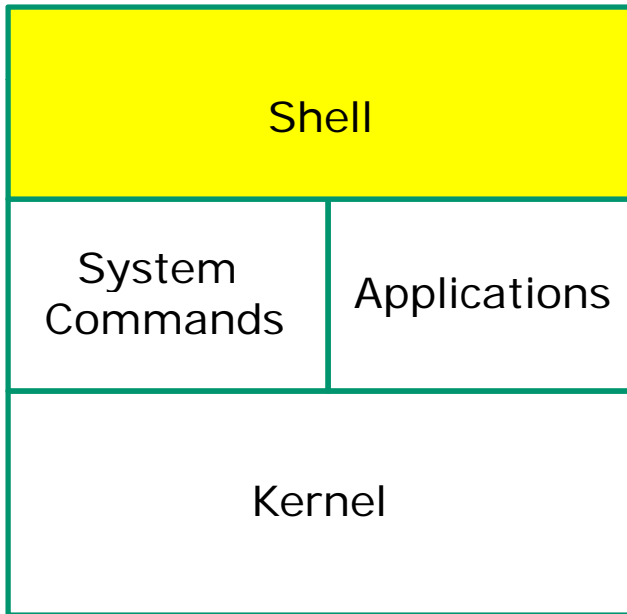


- Allows users to interact with the computer via a “**command line**”.
- **Prompts** for a command, parses the command, finds the right program and gets that program executed.
- Is called a “**shell**” because it hides the underlying operating system.
- Multiple shell programs are available: **sh** (Bourne shell), **bash** (born again shell), **csh** (C shell), **ksh** (Korn shell).
- The shell is a **user interface** and a **programming language** (scripts).
- GNOME and KDE desktops could be called **graphical shells**





# Life of the Shell



- 1) Prompt
- 2) Parse
- 3) Search
- 4) Execute
- 5) Nap
- 6) Repeat





# Life of the Shell

## 1) Prompt user for a command (uses the PS1 environment variable)

Examples:

```
[rsimms@opus work]$ echo $PS1  
[\u@\h \W]\$  
[rsimms@opus work]$
```

*Regular Opus prompt  
for non CIS 90 classes*

```
[root@nosmo ~]# echo $PS1  
[\u@\h \W]\$  
[root@nosmo ~]#
```

*Note the change to #  
when logged on as root*

```
/usr/bin $ echo $PS1  
$PWD $  
/usr/bin $
```

*We use this prompt in CIS  
90 to show current path*



# Life of the Shell

## 2) Parse command user typed (analyze and dissect text string into tokens)

*This is the command which needs to match a program file or script to run.*

*This is an argument which is passed to the program when it is run*

*This indicates stdout will be redirected*

```
[rsimms@opus work]$ ls -lR /bin/p* > pcommands
```

*These are options which are passed to the program when it is run*

*This is a filename expansion metacharacter*

*This is the file that output from stdout is redirected to*



# Life of the Shell

## 3) Search for the program file to run (only look in directories on the PATH)

*/bin directory is on the path*

```
[rsimms@opus work]$ echo $PATH
/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/home/rsimms/bin
```

```
[rsimms@opus work]$ type -a ls
ls is aliased to `ls --color=tty'
ls is /bin/ls
[rsimms@opus work]$
```

*type command shows that **ls** is in the /bin directory*

```
[rsimms@opus work]$ ls /bin/ls
/bin/ls
```

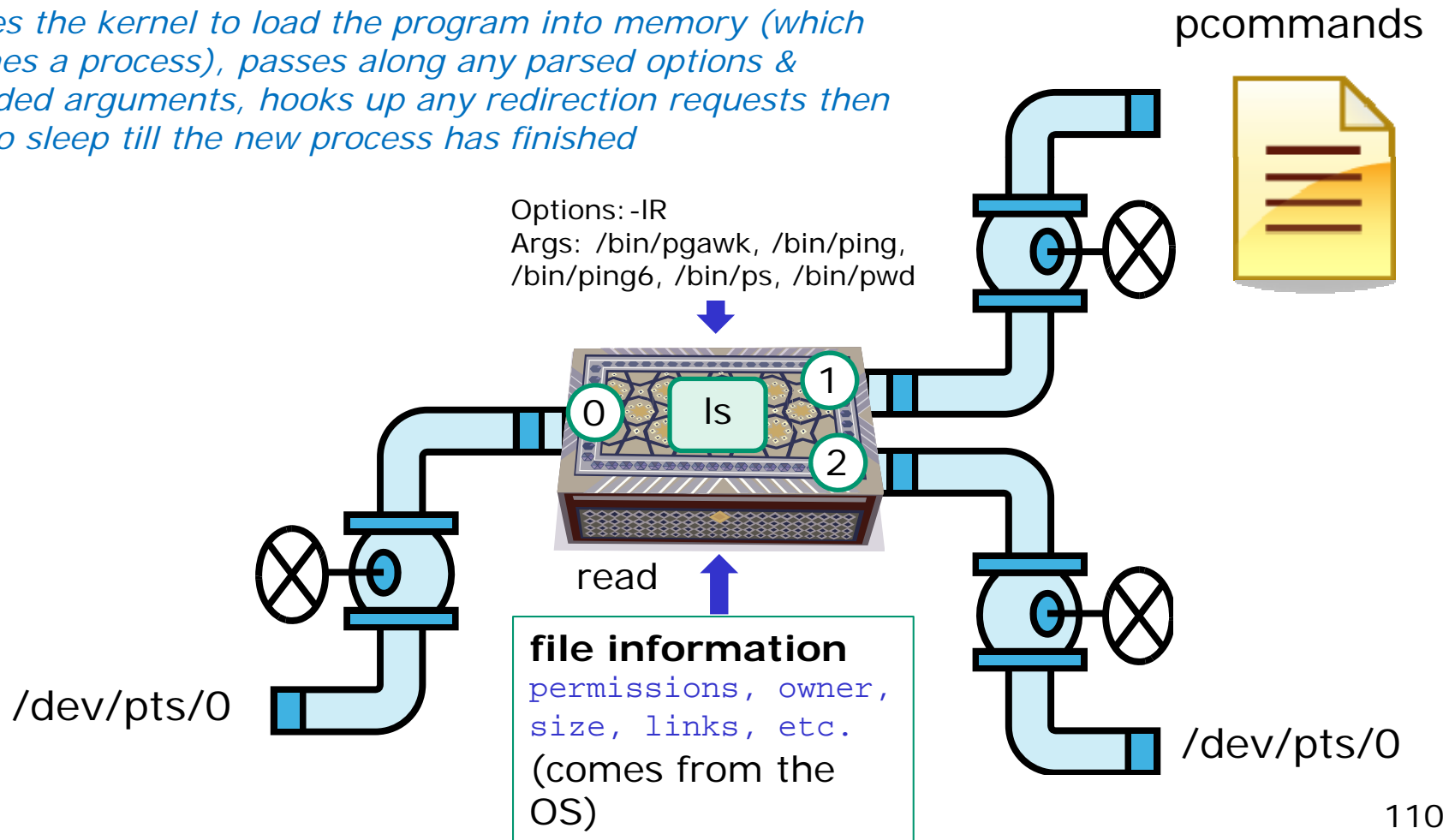
*ls command lists the **ls** file and it is executable (green)*



# Life of the Shell

## 4) Execute the command

*Invokes the kernel to load the program into memory (which becomes a process), passes along any parsed options & expanded arguments, hooks up any redirection requests then goes to sleep till the new process has finished*





## Life of the Shell

### 5) Nap while the command (process) runs to completion

(The shell (itself a loaded process) goes into the sleep state and waits till the command process is finished)

```
[rsimms@opus work]$ ls -lR /bin/p* > pcommands
```

```
[rsimms@opus work]$ cat pcommands
```

```
-rwxr-xr-x 1 root root 321216 Jan 15 2007 /bin/pgawk
-rwsr-xr-x 1 root root 35864 Dec 21 2006 /bin/ping
-rwsr-xr-x 1 root root 31244 Dec 21 2006 /bin/ping6
-r-xr-xr-x 1 root root 79068 Jan 2 2008 /bin/ps
-rwxr-xr-x 1 root root 22980 Nov 30 2007 /bin/pwd
[rsimms@opus work]$
```



## Life of the Shell

6) And do it all over again ... go to step 1





# What the heck !!@@##

Four commands: **hostname**, **ps**, **iptables** and **ifconfig**

```
[rsimms@opus ~]$ ls /bin/hostname /bin/ps
/bin/hostname /bin/ps
[rsimms@opus ~]$ ls /sbin/iptables /sbin/ifconfig
/sbin/ifconfig /sbin/iptables
```

Two work and two don't:

[rsimms@opus ~]\$ hostname  
opus.cabrillo.edu

[rsimms@opus ~]\$ ps

PID	TTY	TIME	CMD
14801	pts/0	00:00:00	bash
14902	pts/0	00:00:00	ps

[rsimms@opus ~]\$ iptables -L  
-bash: iptables: command not found

[rsimms@opus ~]\$ ifconfig  
-bash: ifconfig: command not found

*The **hostname** and **ps** commands work fine. Why do the **iptables** and **ifconfig** commands get the "not found" error?*

*... because they are not on the path*

**!!@@##**

**!!@@##**



# What the heck !!@@##

## The Shell and the PATH

- The shell will only search for commands on the “path”
- The path is determined by the environment variable PATH
- Use **echo \$PATH** to see your current path

*echo command  
prints a text string*

*The \$ means “the  
value of”*

```

cisco@localhost:~
File Edit View Terminal Go Help
[cisco@localhost cisco]$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin:/home/cisco/bin
[cisco@localhost cisco]$
    
```

*This user's path has the  
following directories:*

1. `/usr/local/bin`
2. `/usr/bin`
3. `/bin`
4. `/usr/X11R6/bin`
5. `/home/cisco/bin`

*The order is important as it determines the order in which the  
directories are searched by the shell for a command*



# What the heck !!@@## The Shell and the PATH

```
cisco@localhost:~  
File Edit View Terminal Go Help  
[cisco@localhost cisco]$ echo $PATH  
/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin:/home/cisco/bin  
[cisco@localhost cisco]$
```



*Here is the path ... well not the actual path but the analogy works!*



# What the heck !!@@##

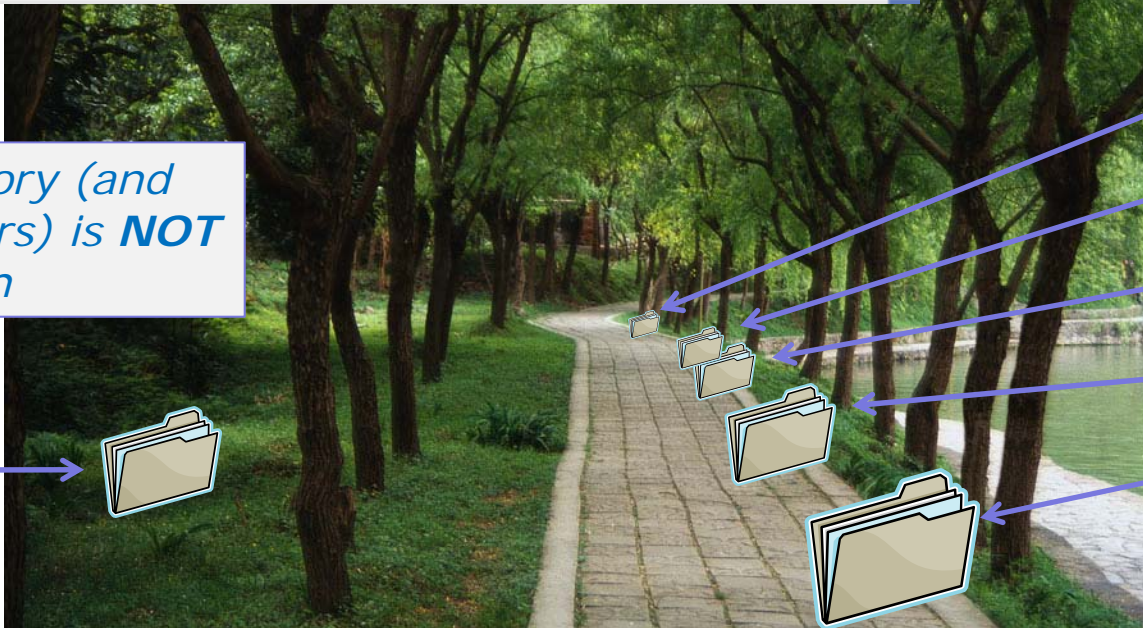
## The Shell and the PATH

```
cisco@localhost:~  
File Edit View Terminal Go Help  
[cisco@localhost cisco]$ echo $PATH  
/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin:/home/cisco/bin  
[cisco@localhost cisco]$
```

*These directories are **on** the path*

*This directory (and many others) is **NOT** on the path*

/sbin



/home/cisco/bin

/usr/X11R6/bin

/bin

/usr/bin

/usr/local/bin



# The Shell and the PATH

```

cisco@localhost/bin
File Edit View Terminal Go Help
[cisco@localhost bin]$ cd /bin
[cisco@localhost bin]$ ls [cdhiptuw]*
cat  cp  date  dnsdomainname  hostname  ping  tcsh  uname  usleep
chgrp  cpio  dd  doexec  igawk  ps  touch  unicode_start
chmod  csh  df  domainname  ipcalc  pwd  true  unicode_stop
chown  cut  dmesg  dumpkeys  pgawk  tar  umount  unlink
[cisco@localhost bin]$
    
```

*The **cat**, **hostname**, **ps** and **uname** commands are in the /bin directory*



*The /bin directory is on the path*

```

[rsimms@opus ~]$ hostname
opus.cabrillo.edu
[rsimms@opus ~]$ ps
  PID TTY          TIME CMD
 14801 pts/0    00:00:00 bash
 14902 pts/0    00:00:00 ps
    
```



*These commands work fine*



# The Shell and the PATH

```

cisco@localhost:/sbin
File Edit View Terminal Go Help
[cisco@localhost sbin]$ cd /sbin
[cisco@localhost sbin]$ ls i*
ibod      ifport
icntrl    ifup
ide_info  ifuser
ifcfg     init
ifconfig  initlog
ifdown    insmod
ifenslave insmod_ksymlinks_clean
install-info iptables
installkernel iptables-restore
ip         iptables-save
ipaddr    iptunnel
ippd      isdnctrl
ippstats  isdnlog
    
```

The *ifconfig* and *iptables* commands are in the */sbin* directory



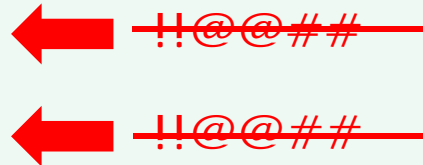
The */sbin* directory is **NOT** on the path



These commands don't work because they were **not found** on the path.

```

[rsimms@opus ~]$ iptables -L
-bash: iptables: command not found
[rsimms@opus ~]$ ifconfig
-bash: ifconfig: command not found
    
```



OK, makes sense now

## Class Exercise

### Life of the Shell

1. Issue a **uname** command and a **type uname** command. What happened?
2. Issue a **iptables -L** command and a **type iptables** command. What happened?
3. Try **ls -lR /bin/p\* > yourlastname** and **cat yourlastname**. What did the \* do?
4. Show your path (hint use echo \$PATH).
5. Show your prompt string (hint use echo \$PS1)
6. Can you find iptables? hint use:

```
find / -name iptables 2> /dev/null
```

# Meta- characters



## Metacharacters

<cr> (carriage return)

The unprintable carriage return <cr> marks the end of a command and lets the shell know to start processing it.

```
[rsimms@opus ~]$ ps
  PID TTY          TIME CMD
 19015 pts/0    00:00:00 bash
 19378 pts/0    00:00:00 ps

[rsimms@opus ~]$ hostname
opus.cabrillo.edu

[rsimms@opus ~]$ echo "Use <cr> to end the command"
Use <cr> to end the command
```

*Pressing the Enter key here generates an invisible <cr>*

# Metacharacters

## \$ (the value of)

Use \$ for the "value" of a variable

Analogy: Each variable is a named location. The contents of any location is the "value" of that variable.

```
$ echo $LOGNAME
simmsben
```

```
$ echo HOME
HOME
```

```
$ echo $HOME
/home/cis90/simmsben
```

```
$ echo $SHELL
/bin/bash
```

```
$ echo $HOSTNAME
opus.cabrillo.edu
```



## Metacharacters

' " (single and double quotes)

- *One or more blanks between arguments is treated as a single blank*
- *Use " (double) or ' (single) quotes for preserving blanks*

```
[rsimms@opus ~]$ echo 1 2 3
1 2 3
```

*The blanks in these commands are used to separate the arguments*

```
[rsimms@opus ~]$ echo 1
1 2 3
```

```
[rsimms@opus ~]$ echo "1
1      2
          3
```

```
[rsimms@opus ~]$ echo "1 2 3"
1 2 3
```

```
[rsimms@opus ~]$ echo '"1 2 3"'
"1 2 3"
```

*Use single and double quotes together to show quote marks*

*Use quotes if blanks are important and are needed for spacing*

# Metacharacters

' " (quotes)

*Note that strings in " (double) quotes allow the \$ metacharacter to be interpreted by the shell*

```
[simmsben@opus Poems]$ echo Hello $LOGNAME  
Hello simmsben
```

```
[simmsben@opus Poems]$ echo "Hello $LOGNAME"  
Hello simmsben
```

```
[simmsben@opus Poems]$ echo 'Hello $LOGNAME'  
Hello $LOGNAME
```

*Not so with strings in ' (single) quotes*

*The use of a single quote will prevent the shell from interpreting the \$ metacharacter*

# Metacharacters

\ (don't interpret next metacharacter)

Use \ (back slash) to not interpret the next metacharacter

```
[rsimms@opus ~]$ echo a b c
a b c
```

```
[rsimms@opus ~]$ echo a b c \
> d e f
a b c d e f
```

Do not interpret the invisible <cr> at the end of the line (from the Enter key)

```
[rsimms@opus ~]$ echo $PS1
[\\u@\\h \\W]\\$
```

Do not interpret the \$ (which shows the value of the variable)

```
[rsimms@opus ~]$ echo \\$PS1
$PS1
```

```
[rsimms@opus ~]$ echo "Hello World"
Hello World
```

Do not interpret the double quote marks

```
[rsimms@opus ~]$ echo \\\"Hello World\\\"
\"Hello World\"
```

## Metacharacters ; (command separator)

*Use ; to put multiple commands on one line*

```
[simmsben@opus Poems]$ hostname; uname; echo $LOGNAME; ls  
opus.cabrillo.edu  
Linux  
simmsben  
ant Blake nursery Shakespeare twister Yeats
```

## More on the Command Line

### Handy Shortcuts

- Use up and down arrows to “retype” previous commands
- Left and right arrow for editing current command
- Use <tab> to complete filenames automatically

```

😊 [simmsben@opus Poems]$ hostname; name; echo $LOGNAME; ls Blake/
😊 opus.cabrillo.edu
😞 bash: name: command not found
😊 simmsben
😊 jerusalem tiger
    
```

*Press <tab> after the B and the shell fills in the remaining “lake/”*

```

[simmsben@opus Poems]$ hostname; uname; echo $LOGNAME; ls
Blake/
opus.cabrillo.edu
Linux
simmsben
jerusalem tiger
    
```

*Press up arrow and the shell retypes the previous command*

*Use the left arrow to backup and fix the typo (uname instead of name)*

## Class Exercise Metacharacters

```
echo a b          c  
echo "a b          c"
```

```
echo a b c \  
>d e f
```

```
echo $PS1  
echo \  
$PS1
```

```
echo "Hello $USERNAME"  
echo 'Hello $USERNAME'
```

```
echo ' "Hello World" '  
echo \  
"Hello World\  
'
```

```
hostname; uname; echo $LOGNAME; ls
```

Try these commands  
out on your computer



# DOCS

## Using man (manual) pages

Type the **man** command followed by the name of the command you want documentation on.

Example: **man bc**

```

simmsben@opus:~
/home/cis90/simmsben $
/home/cis90/simmsben $ man bc
bc(1)                                     bc(1)

NAME
    bc - An arbitrary precision calculator language

SYNTAX
    bc [ -hlwsqv ] [long-options] [ file ... ]

VERSION
    This man page documents GNU bc version 1.06.

DESCRIPTION
    bc is a language that supports arbitrary precision numbers with inter-
    active execution of statements. There are some similarities in the
    syntax to the C programming language. A standard math library is
    available by command line option. If requested, the math library is
    defined before processing any files. bc starts by processing code from
    all the files listed on the command line in the order listed. After
    all files have been processed, bc reads from the standard input. All
    code is executed as it is read. (If a file contains a command to halt
    the processor, bc will never read from the standard input.)
  
```



Use these keys to scroll



Use q key to quit

# Using Google

*Do a Google search on "linux xxx command" where xxx is the command you want documentation for.*

Example: google linux bc command

The image shows two overlapping browser windows. The left window is a Google search for "linux bc command". The search results include:

- bc - Linux Command - Unix**: Linux / Unix Command Library: bc examples. [linux.about.com/od/commands/](http://linux.about.com/od/commands/)
- Linux and UNIX bc command**: linking you to information about the [www.computerhope.com/unix/ubc](http://www.computerhope.com/unix/ubc)
- command-line calculations u**: bc is included with (almost?) all Li math library functions in the bc co [www.basicallytech.com/blog/index](http://www.basicallytech.com/blog/index)
- Command line calculator, bc**: How to do calculation if I only have very complicated calculation. To pe [linux.byexamples.com/archives/](http://linux.byexamples.com/archives/)
- Linux bc Command- Basic**: What is Linux bc Command? ... above command displays the sum [www.hscrepts.com/tutorials/linux-4](http://www.hscrepts.com/tutorials/linux-4)
- bc: A Handy Utility | Linux Jo**: Mr. McAndrew shows us how the algorithms. Linux, as with almost [www.linuxjournal.com/article/2544](http://www.linuxjournal.com/article/2544)

The right window shows the "bc - Linux Command - Unix" page from [linux.about.com](http://linux.about.com/od/commands/l/bcmd1_bc.htm). The page includes an advertisement for PayPal, a search bar with "linux bc command" entered, and a "Free Linux Newsletter!" sign-up form. The main content area contains:

**Linux / Unix Command: bc**  
Command Library

**Sponsored Links**

- Google Apps for Business**: Get email for your domain & reduce IT costs. Start your free trial! [www.google.com/apps/business](http://www.google.com/apps/business)
- California Online Classes**: Discover Free Public School at Home with Award-Winning Curriculum! [www.K12.com](http://www.K12.com)
- Unix Commands Tutorial**: Find Unix education & training Providers in our Business Directory [www.business.com](http://www.business.com)

**NAME**  
bc - An arbitrary precision calculator language

**SYNTAX**  
bc [ -hlwsvq ] [long-options] [ file ... ]

**DESCRIPTION**  
bc is a language that supports arbitrary precision numbers with interactive execution of statements. There are some similarities in the syntax to the C programming language. A standard math library is available by command line option. If requested, the math library is defined before processing any files. bc starts by processing code from all the files listed

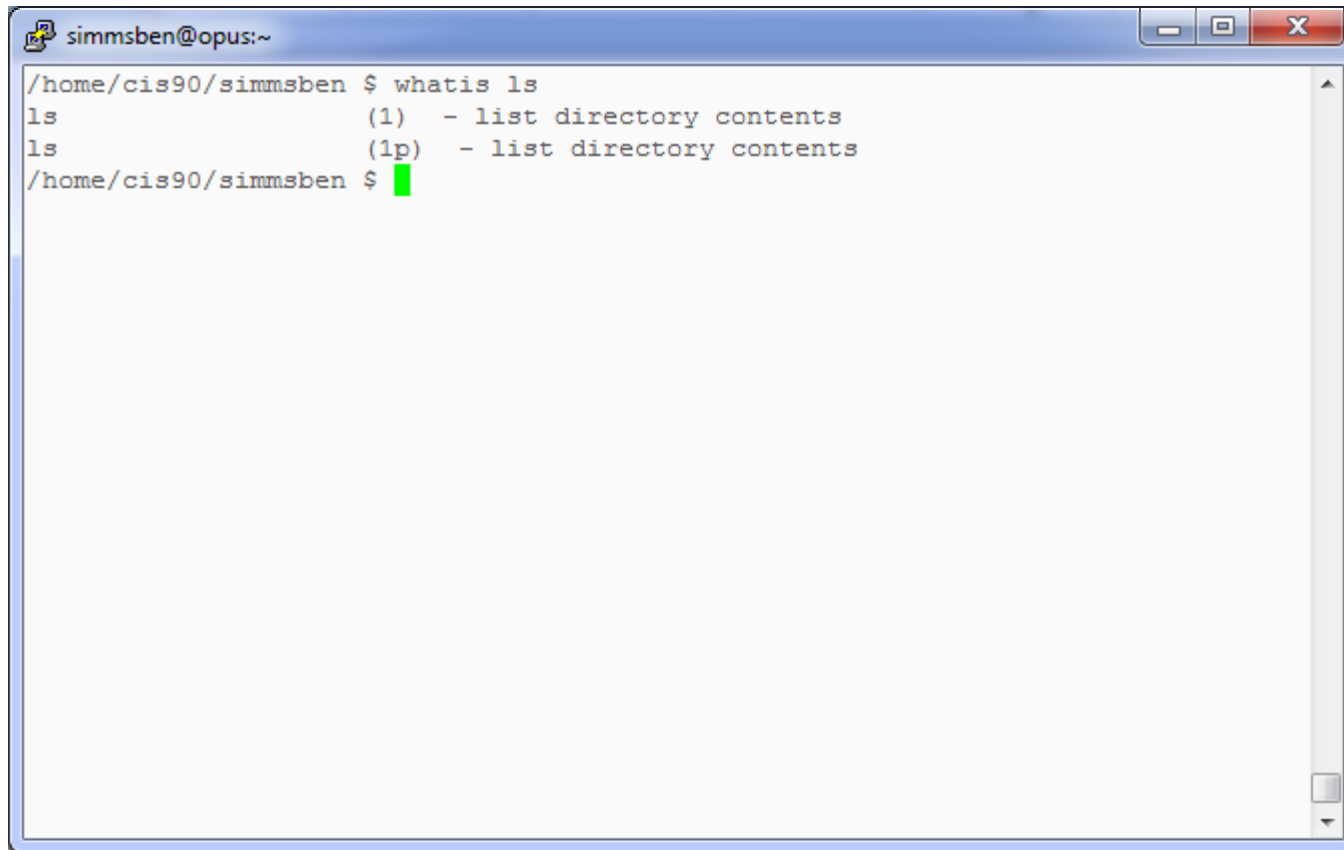
**Advertisement**  
PayPal  
MERCHANT SERVICES  
DID YOU KNOW...  
The world's largest retailers use PayPal payment processing  
Walmart  
lenovo. eToys.com

## Other Documentation

- **whatis** *command*     *same as the **man -f** command*
- **apropos** *command*     *same as the **man -k** command*
- **info** *command*

## Documentation examples

Example: **whatis ls**

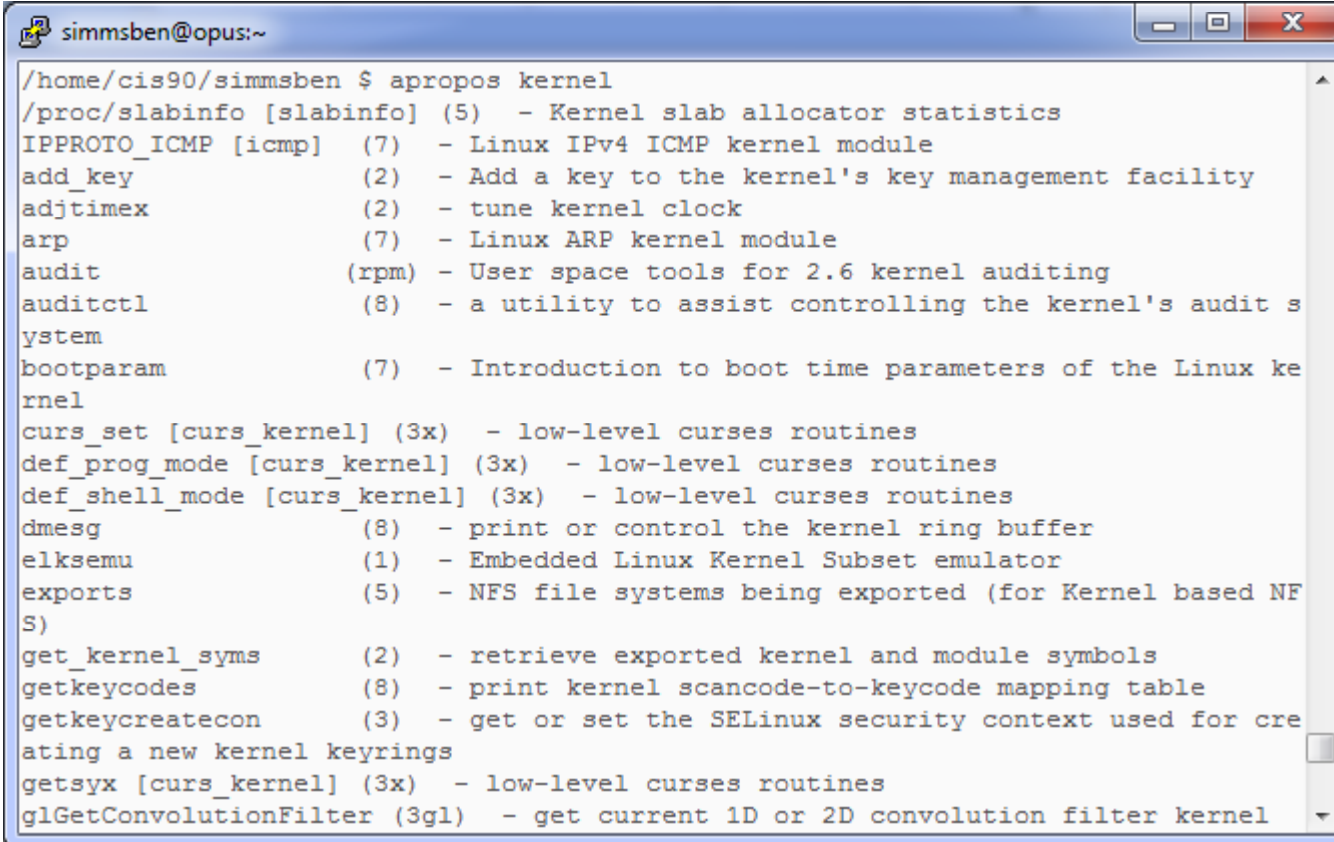


```
simmsben@opus:~  
/home/cis90/simmsben $ whatis ls  
ls          (1)  - list directory contents  
ls          (1p) - list directory contents  
/home/cis90/simmsben $ █
```

*whatis* searches the *whatis* database for a complete word. Same as the *man -f* command .

## Documentation examples

Example: **apropos kernel**



```

simmsben@opus:~
/home/cis90/simmsben $ apropos kernel
/proc/slabinfo [slabinfo] (5) - Kernel slab allocator statistics
IPPROTO_ICMP [icmp] (7) - Linux IPv4 ICMP kernel module
add_key (2) - Add a key to the kernel's key management facility
adjtimex (2) - tune kernel clock
arp (7) - Linux ARP kernel module
audit (rpm) - User space tools for 2.6 kernel auditing
auditctl (8) - a utility to assist controlling the kernel's audit system
bootparam (7) - Introduction to boot time parameters of the Linux kernel
curs_set [curs_kernel] (3x) - low-level curses routines
def_prog_mode [curs_kernel] (3x) - low-level curses routines
def_shell_mode [curs_kernel] (3x) - low-level curses routines
dmesg (8) - print or control the kernel ring buffer
elksemu (1) - Embedded Linux Kernel Subset emulator
exports (5) - NFS file systems being exported (for Kernel based NFS)
get_kernel_syms (2) - retrieve exported kernel and module symbols
getkeycodes (8) - print kernel scancode-to-keycode mapping table
getkeycreatecon (3) - get or set the SELinux security context used for creating a new kernel keyrings
getsyx [curs_kernel] (3x) - low-level curses routines
glGetConvolutionFilter (3gl) - get current 1D or 2D convolution filter kernel

```

*apropos* searches the *whatis* database for a string of text. Same as the *man -k* command .

## Documentation examples

### Example: **info ls**

```

simmsben@opus:~
file: coreutils.info, Node: ls invocation, Next: dir invocation, Up: Directo\
ry listing

10.1 `ls': List directory contents
=====

The `ls' program lists information about files (of any type, including
directories). Options and file arguments can be intermixed
arbitrarily, as usual.

For non-option command-line arguments that are direc
default `ls' lists the contents of directories, not rec
omitting files with names beginning with `.'. For othe
arguments, by default `ls' lists just the file name. I
argument is specified, `ls' operates on the current dir
as if it had been invoked with a single argument of `.'

By default, the output is sorted alphabetically, acc
locale settings in effect. (1) If standard output is a t
output is in columns (sorted vertically) and control ch
output as question marks; otherwise, the output is list
and control characters are output as-is.

--zz-Info: (coreutils.info.gz)ls invocation, 54 lines --Top-----
Welcome to Info version 4.8. Type ? for help, m for menu item.

```

- Navigating info pages:**
- Enter to follow links (\*'s)
  - n or <space> for next page
  - p for previous page
  - u for up tree
  - l for last page
  - q to quit

# Documentation

Two of my favorite documentation links

**Rich's Cabrillo College CIS Classes Resources**

Home **Resources** Forums CIS Lab CTC

Login  
Flashcards  
Admin

CIS 90  
Previous Classes

103 days till term ends!

Cabrillo College  
Web Advisor  
CCC Confer  
Static IPs  
Quick Ref  
VM Repairs  
GAH!

**Links**

- Instructors**
  - Linux Master Jim
  - Programming Master Ed
  - Network Master Gerlinde
  - Network Master Rick
  - Web Master John
  - Windows Master Gary
- Getting Linux**
  - Linux ISOs
  - Kernels
  - RPMS (rpmfind)
  - RPMS (pbone)
- Tools and Software**
  - Apache
  - Bastille
  - cygwin
  - DOS boot disks
  - Dyng
  - Job
  - MS
  - All
  - Net
  - Put
  - Qu
  - SU
  - Tri
  - Vir
  - VM
  - Wi
- Clubs**
  - GNU Linux Users Group
- Departments**
  - CNSA
  - CIS
  - CS
- Crib Sheets**
  - Ollie Wright (CIS 90)
- Documentation**
  - TLDP
  - LINFO
- Animations**
  - Linux network technologies

**The Linux Documentation Project**

2010-09-06

Español  
Français  
Italian  
Korean  
Português do Brasil

**LDP Worldwide**

- Mirrors
- Non-English info
- Translation effort
- Translated Guides
- Translated HOWTOs
- Printed books
- Main site

**LDP Information**

- FAQ
- Manifesto / license
- History
- Volunteers/Staff
- Job Descriptions
- Mailing lists
- LDP Weekly News
- Archives / RSS feed
- IRC
- Feedback
- Apparel

**Workshop**

**LDP Wiki:** The LDP Wiki is the entry point for any work in progress  
Members | Authors | Visitors

**Documents**

**HOWTOs:** subject-specific help  
latest updates | main index | browse by category

**Guides:** longer, in-depth books  
latest updates / main index

**FAQs:** Frequently Asked Questions  
latest updates / main index

**man pages:** help on individual commands (20060810)

**Search / Resources**

Links  
OMF search

**The Linux Information Project**

Welcome to The Linux Information Project (LINFO). This project is dedicated to providing high quality, comprehensive and easily accessible information about Linux and other free software. (New to Linux? Start here.)

**New on This Site:**

- October 27: root Definition page updated.
- October 19: Hard Link Definition page added.
- October 12: Characters: A Brief Introduction page updated.
- October 03: Byte Definition page updated.
- September 27: PDP-7 Definition page updated.
- September 24: The umount Command page added.
- September 20: The head Command page updated.

**Site Contents:**

The Linux Documentation and Information Projects



## Class Exercise Documentation

Use the man command on itself:

- **man man**

Research the **ls** command using:

- The **whatis** command
- The **man** command
- The **info** command
- Google

# Wrap up

New commands:

- |         |  |
|---------|--|
| apropos | - search for string in whatis database |
| bc      | - binary calculator                    |
| cat     | - print file(s)                        |
| cd      | - change directory                     |
| echo    | - print text                           |
| env     | - show shell environment variables     |
| info    | - online documentation with hot links  |
| file    | - show file information                |
| ls      | - show directory contents              |
| passwd  | - change password                      |
| set     | - show (or set) shell variables        |
| type    | - show command location in path        |
| man     | - manual page for a command            |
| whatis  | - command summary                      |

New Files and Directories:

- |             |  |
|-------------|--|
| /etc/passwd | - user accounts  |
| /etc/shadow | - encrypted passwords                                  |
| /bin        | - directory of commands                                |
| /sbin       | - directory of superuser commands                      |
| /usr/bin    | - directory of commands, tools and utilities           |
| /usr/sbin   | - directory of superuser commands, tools and utilities |

## Next Class

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

Quiz questions for next class:

- Name four directories where one can find commands?
- How do you show your path?
- What is the command to print the manual page for a command?

# Backup

Example program to process: ls command

```

/home/cis90/simmsben/Poems $ tty
/dev/pts/1
/home/cis90/simmsben/Poems $ ls
ant Blake nursery Shakespeare twister Yeats
/home/cis90/simmsben/Poems $
    
```

