an overview of linux

http://LinuxMeister.net/overview-LINUX.pdf - *notes added 11/3/2014*

UNIX history

Ken Thompson developed the UNIX operating system in PDP-7 Assembly language and then developed the B language, a machine-independent language.

Dennis Ritchie developed the C language by modifying B, and with Thompson rewrote the UNIX system in C.

This happened around 1969. The date function in UNIX starts counting in 1970. It adds time from that point. That's why Y2K wasn't a problem for UNIX. However, the year 2038 is... unless a 64 bit system is used...

Kernighan is a co-developer of the C programming language,
Ritchie being the other.

Later ANSI C was developed, the original version often referred to as K&R C for Kernighan and Ritchie. Another researcher at Bell Labs, Bjarne Stroustrop, created the OOP (object oriented programming) version called C++, built up C.

See pages 9 and following in "A Practical Guide to the UNIX System",
Mark G. Sobell, isbn 0-8053-7565-1

GNU & Linux History

Programmers worldwide were greatly inspired
by the GNU project by **Richard Stallman**,
a software movement to provide free and quality software.

1985 Richard Stallman published his famous "GNU Manifesto"

In 1991, **Linus Benedict Torvalds**
was a 2nd year student of Computer Science
at the University of Helsinki

In September 1991 - Linux was released to the internet.

The Historic USENET post

In August 25, 1991 the historic post was sent to the MINIX news group by Linus .....

From: torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)
Newsgroups: comp.os.minix
Subject: What would you like to see most in minix?
Summary: small poll for my new operating system
Message-ID: <1991Aug25.205708.9541@klaava.Helsinki.FI>
Date: 25 Aug 91 20:57:08 GMT
Organization: University of Helsinki

Hello everybody out there using minix -
I'm doing a (free) operating system (just a hobby, won't be big and
professional like gnu) for 386(486) AT clones. This has been brewing
since april, and is starting to get ready. I'd like any feedback on
things people like/dislike in minix, as my OS resembles it somewhat
(same physical layout of the file-system (due to practical reasons)
among other things). I've currently ported bash(1.08) and gcc(1.40),and
things seem to work.This implies that I'll get something practical within a
few months, andI'd like to know what features most people would want. Any
suggestions are welcome, but I won't promise I'll implement them :-)
Linus (torvalds@kruuna.helsinki.fi)
PS. Yes - it's free of any minix code, and it has a multi-threaded fs.
It is NOT protable (uses 386 task switching etc), and it probably never
will support anything other than AT-harddisks, as that's
all I have :-(.

Linux version 0.01 was released mid September 1991, and was put on the net.

Linux / UNIX - Vocabulary

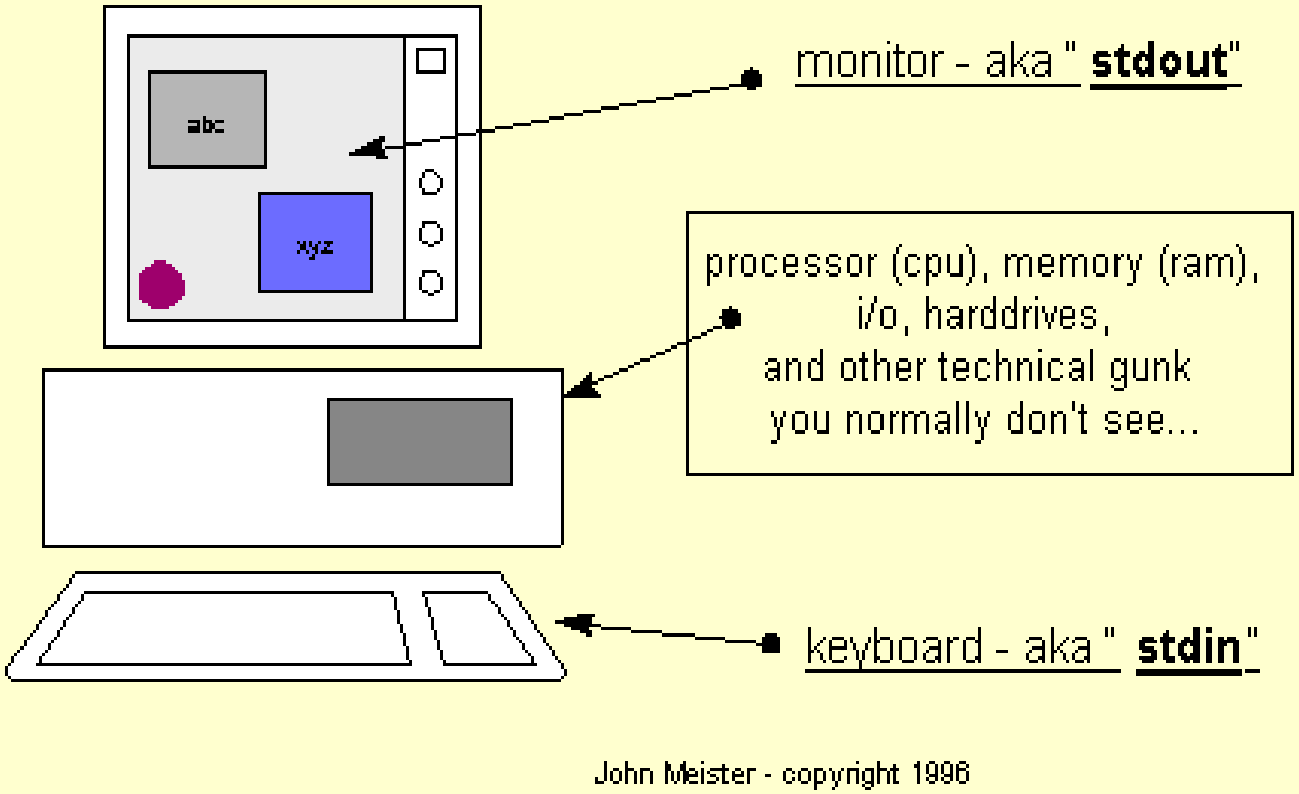
#	= pound, hash - (comment)
!	= bang
*****	= splat, star - (wildcard)
vi	= "v" "i" - (**the** Editor)
/	= slash, root (root filesystem)
/etc	= (slash) "et cee"
/bin	= (slash) bin
/usr	= (slash) "user", "us-r"
/opt	= (slash) "opt"
/sbin	= (slash) "s-bin"
\	= back slash (MicroSoft slash)

But first, more background...

Before we begin the install we will discuss:

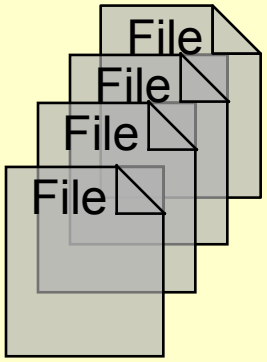
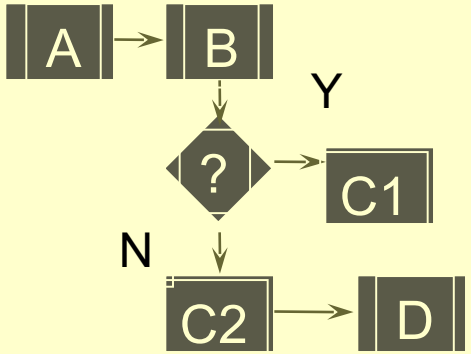
- STDIN, STDOUT ???
- What does Linux/UNIX consist of?
- Basic File types
- File Systems and Directory Structures
- logging in / gaining access
- passwords
- the “essential commands”
- Shells
- … and several basic commands

a starting point

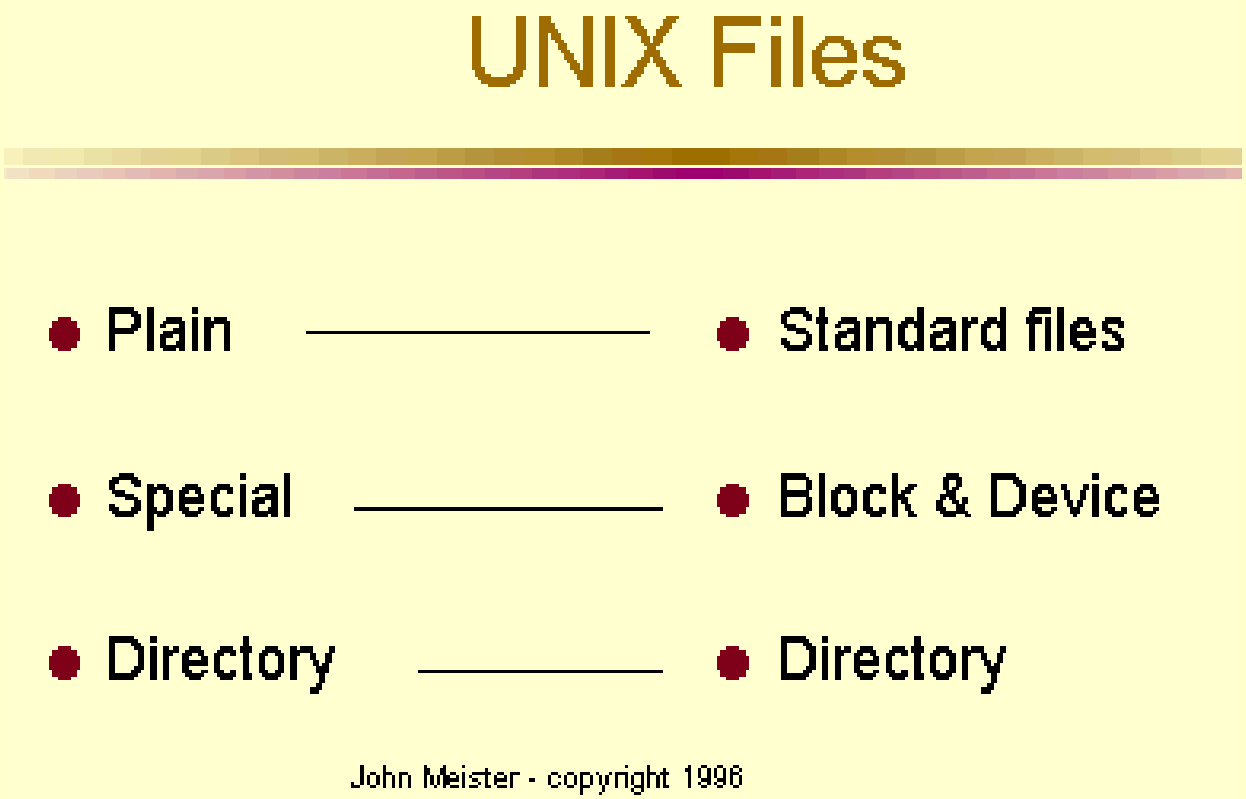


a starting point

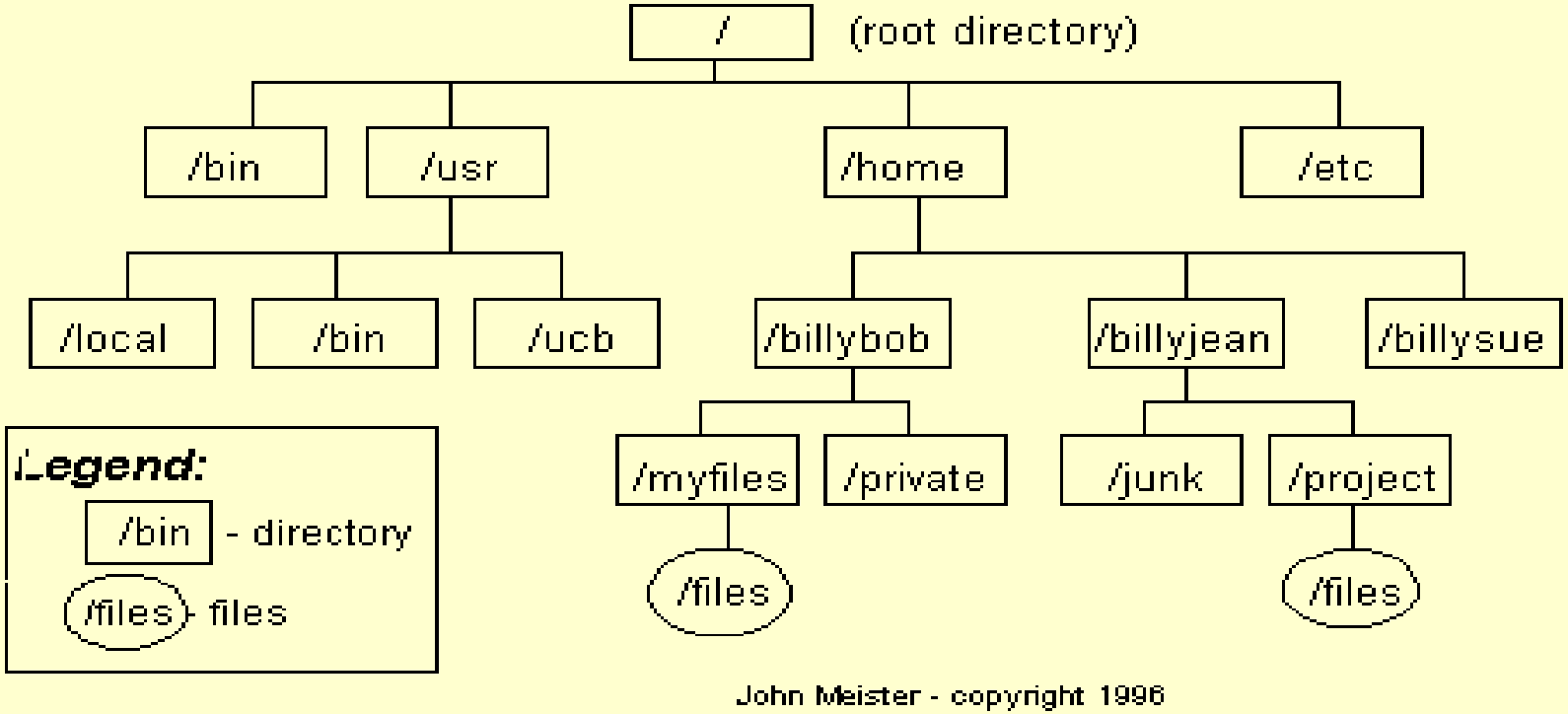
Linux/ UNIX consists of 2 things:

Files	Processes
	
Files contain information	Processes use files.

UNIX files - 4 types



UNIX filesystem tree



Linux Filesystems

The root Linux filesystem is made up of 12 filesystems by default:

- /bin - "binaries" - commands
- /dev - "devices" - special files
- /etc - "et cetera" - config files
- /sbin - "secure bin" - complete commands
- /home - "home" - user files
- /opt - "optional" - added programs
- /lost+found - "lost and found" - system use
- /mnt - "mount" - special mount point
- /proc - "process" - unique process id info
- /usr - "user lib" - libraries
- /var - "var" - system logs
- /usr/local - "user local" - local programs

File directories

```
drwxr-xr-x   2 root     bin          2048 Sep 27  2001 bin
drwxr-xr-x   2 root     root         1024 Dec 30  1999 boot
drwxr-xr-x   2 root     root         1024 Oct  6  1997 cdrom
drwxr-xr-x   2 root     root        26624 Feb 27 13:02 dev
drwxr-xr-x  19 root     root         4096 Mar 31 10:44 etc
drwxr-xr-x  46 root     root         1024 Feb 10 14:00 home
drwxr-xr-x   3 root     root         3072 Nov 10  2000 lib
drwxr-xr-x   3 root     root         1024 Nov 13  1998 local
drwxr-xr-x   2 root     root        12288 Nov  9  1998 lost+found
drwxr-xr-x   4 root     root         1024 Apr  6  2000 mnt
dr-xr-xr-x   5 root     root            0 Feb 27 05:01 proc
drwx--x---  30 root     root         4096 Apr  2 11:38 root
drwxr-xr-x   2 root     bin          2048 Dec 15  1998 sbin
drwxrwxrwt   2 root     root         9216 Apr  2 22:38 tmp
drwxr-xr-x  21 root     root         1024 Sep 15  1999 usr
drwxr-xr-x  18 root     root         1024 Feb 19 18:17 var
-r--------   1 root     root       464979 Nov  9  1998 vmlinuz
-rw-r--r--   1 root     root       492075 Sep 10  1999 vmlinuz-scsi-2.0.37
drwxr-xr-x   4 john     jeep         1024 Jun  9  2000 wagoneers
drwxr-xr-x  12 root     root         1024 Jul 19  1999 www
```

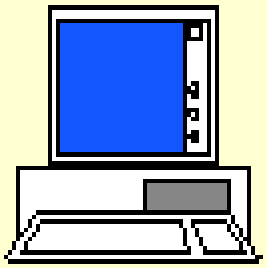
Linux File Directory (ls -al)

```
drwxr-xr-x   8 john     jeep         1024 Apr  2 21:31 .
drwxr-xr-x  30 john     jeep         2048 Jan  2 13:11 ..
-rw-r--r--   1 john     jeep            0 Apr  2 21:31 .bashrc
-rw-r--r--   1 john     jeep            0 Apr  2 21:31 .dtprofile
-rw-r--r--   1 john     jeep            0 Apr  2 21:31 .exrc
-rw-r--r--   1 john     jeep            0 Apr  2 21:31 .forward
-rw-r--r--   1 john     jeep            0 Apr  2 21:31 .kshrc
-rw-r--r--   1 john     jeep            0 Apr  2 21:29 .profile
-rw-r--r--   1 john     jeep            0 Apr  2 21:31 .sh_history
-rw-r--r--   1 john     jeep            0 Apr  2 21:31 .signature
drwxr-xr-x   7 john     jeep         1024 Aug 25  2002 CS540-summer-2002
-rw-r--r--   1 john     jeep        12146 Jul 28  2002 IT-cost-factors.html
-rw-r--r--   1 john     jeep            0 Apr  2 21:28 Linux-rules-Microsoft-drools
drwxr-xr-x   3 john     jeep         1024 Feb 17 14:08 MG-416
-rw-r--r--   1 john     jeep       127696 May  5  2002 Microsoft-DLLs.html
drwxr-xr-x   3 john     jeep         1024 May 10  2002 UNIX-short-ovrvw
-rw-r--r--   1 john     jeep         2057 Dec  5  2001 Unix-executables-bin.html
-rw-r--r--   1 john     jeep         3457 Dec  5  2001 Unix-libraries-lib.html
-rw-r--r--   1 john     jeep         8643 Sep  1  2002 bnc-connector.html
-rw-r--r--   1 john     jeep         1759 Dec  4  2001 code-red-clean-perl-sh.txt
lrwxrwxrwx   1 john     jeep           25 May  5  2002
                                computing-history.html -> ../computing-history.html
drwxr-xr-x   3 john     jeep         1024 Jul 24  2002 cs470
-rw-r--r--   1 john     jeep       165888 Oct  3  2002 cs470-syllabus.doc
-rw-r--r--   1 john     jeep         9211 Nov 18 19:57 file-systems.html
-rw-r--r--   1 john     jeep        13918 Apr 20  2002 some-links.html
```

How to access the system...

Login

the ways to access a UNIX system:



- telnet "*hostname*"
- su - "*username*" (from shell)
- login (from console or shell)
- rlogin "*hostname*"

NOTE: telnet and rlogin not used now, use: ssh <hostname>

Passwords

- use the “passwd” command to change
- should be at least 7 characters long
- should include numbers, punctuation, or mixed case
- passwords should NOT include dictionary words
- passwords should NOT include personal data
- passwords should NOT contain profanity
- passwords should NOT be reused
- passwords should NOT be shared with others
- passwords should be different for each account or system
- passwords should be changed often
- passwords should NOT be written down and left near system

Linux Shells

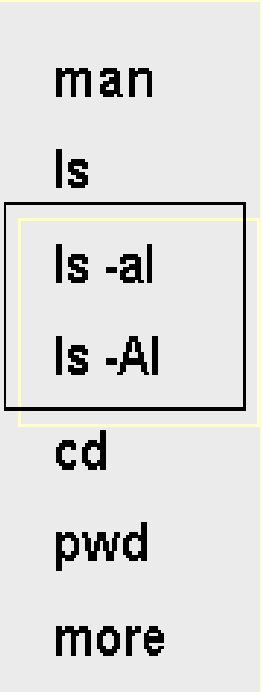
more /etc/shells

/bin/sh - Bourne Shell
/bin/ash - similar to sh
/bin/bash - bash - GNU Bourne-Again Shell

/bin/ksh - **Korn** shell (similar to POSIX shell) - RECOMMENED
/bin/zsh - zsh most closely resembles ksh but includes many enhancements

/bin/csh - C shell (*not recommended*)
/bin/tcsh - C shell with file name completion and command line editing

The Essential UNIX Commands



the ESSENTIAL commands

man

man(1) man(1)

NAME

man - format and display the on-line manual pages

manpath - determine user's search path for man pages

SYNOPSIS

man [-acdfhkKtwW] [-m system] [-p string] [-C config_file] [-M path] [-P pager] [-S section_list] [section] name ...

DESCRIPTION

man formats and displays the on-line manual pages. This version knows about the MANPATH and (MAN)PAGER environment variables, so you can have your own set(s) of personal man pages and choose whatever program you like to display the formatted pages. If section is specified, man only looks in that section of the manual. You may also specify the order to search the sections for entries and which preprocessors to run on the source files via command line options or environment variables. If name contains a / then it is first tried as a filename, so that you can do man ./foo.5 or even man /cd/foo/bar.1.gz.

OPTIONS

-C config_file

Specify the man.conf file to use; the default is /usr/lib/man.conf. (See man.conf(5).)

...

ls

LS(1L) LS(1L)

NAME
ls, dir, vdir - list contents of directories

SYNOPSIS
ls [-abcdefgiklmnopqrstuxABCFGLNQRSUX178] [-w cols] [-T cols] [-I pattern] [--all] [--escape] [--directory] [--inode] [--kilobytes] [--numeric-uid-gid] [--no-group] [--hide-control-chars] [--reverse] [--size] [--width=cols] [--tabsize=cols] [--almost-all] [--ignore-backups] [--classify] [--file-type] [--full-time] [--ignore=pattern] [--dereference] [--literal] [--quote-name] [--recursive] [--sort={none,time,size,extension}] [--format={long,verbose,commas,across,vertical,single-column}] [--time={atime,access,use,ctime,status}] [--color[={always,never,auto}]] [--help] [--version] [name...]

DESCRIPTION
This manual page documents the GNU version of ls, with color extensions. dir and vdir are versions of ls with different default output formats. These programs list each given file or directory name. Directory contents are sorted alphabetically. For ls, files are by default listed in columns, sorted vertically, if the standard output is a terminal; otherwise they are listed one per line. For dir, files are by default listed in columns, sorted vertically. For vdir, files are by default listed in long format.

OPTIONS
-a, --all
List all files in directories, including all files that start with `.'.

...

cd

```
--> man cd
<standard input>:11: can't open `man.macros': No such file or directory

cd(n)              Tcl Built-In Commands              cd(n)

NAME
      cd - Change working directory

SYNOPSIS
      cd ?dirName?

DESCRIPTION
      Change the current working directory to dirName, or to the home directory (as specified in the
      HOME environment variable) if dirName is not given.  Returns an empty string.

KEYWORDS
      working directory


Tcl                                                1


...
```

pwd

PWD(1) PWD(1)

NAME
pwd - print name of current/working directory

SYNOPSIS
pwd
pwd {--help,--version}

DESCRIPTION
This documentation is no longer being maintained and may be inaccurate or incomplete. The Texinfo documentation is now the authoritative source.

This manual page documents the GNU version of pwd. pwd prints the fully resolved name of the current directory. That is, all components of the printed name will be actual directory names -- none will be symbolic links.

Note that most Unix shells provide a built-in pwd command with similar functionality so the unadorned, interactive pwd command will usually execute the built-in version and not this one.

OPTIONS
--help Print a usage message on standard output and exit successfully.

--version
Print version information on standard output then exit successfully.

FSF GNU Shell Utilities 1

…

more

MORE(1) UNIX Reference Manual MORE(1)

NAME
more - file perusal filter for crt viewing

SYNOPSIS
more [-dlfpcsu] [-num] [+/ pattern] [+ linenum] [file ...]

DESCRIPTION
More is a filter for paging through text one screenful at a time. This version is especially primitve. Users should realize that less(1) provides more(1) emulation and extensive enhancements.

OPTIONS
Command line options are described below. Options are also taken from the environment variable MORE (make sure to precede them with a dash (``-")) but command line options will override them.

-num This option specifies an integer which is the screen size (in lines).

-d more will

….
(see also less)

who - whoami - who am i

"**whoami", "who", "who am i"**

"**whoami**" identifies the user of a shell.

If you "**su**" (switch user) and are a different user you may see your original login userid by typing:

"**who am i**"

"**who**", "**who -r**" - lists who is on the system

file, cat, head and tail

concatenate and print files

“file” may be used to determine the type of file.

“more” (or less) is recommended for viewing files.

“cat” is useful for redirecting data.

“head” and “tail” offer viewing of parts of files.

mkdir

mkdir Newdir
mkdir -p Newdir/subdir

used to create a new directory, the -p option allows creation of the first directory if it doesn't exist.

HANDS ON:

cd
mkdir Subdirectory
cd Subdirectory
pwd

cp

cp - copy files

cp -r - copy directories (recursive)

note the use of the asterick (*) as a wildcard

Hands ON:

1) cp listing.txt dir_list.txt
2) mkdir Manpages
3) ls -al *man*
4) cp *man* Manpages
5) ls -al Manpages
6) cp -r Manpages Deleteme
7) ls -al Deleteme
8) df -k .
9) rm -rf Deleteme
10) df -k .

mv

mv - move files (rename)

Hands ON:

1) man mv | col -b > mv.man
2) touch 1234.file
3) ls -al
4) mv -i 1234.file file.1234
5) ls -al
6) mv file.1234 listing.txt
7) ls -al > listing
8) more listing
9) mv listing listing.txt

rm

rm - remove directory entries

man page SYNOPSIS: rm [-f | -i] [-r] file ...

rm -r = remove directory, or recursive

Hands ON:

1) cat touch_man | col -b > touch.man
2) rm -i touch_man (answer yes)
3) cat tee_man | col -b > tee.man
4) rm -f tee_man
5) mkdir -p Delete_this_dir/Another_dir
6) mkdir Delete_this_dir/Second_dir
7) rm -ir Delete_this_dir (answer no to Second_dir)
8) Did you see: "rm: Delete: Directory not empty"?
9) rm -r Delete_this_dir (did it let you?)
10) (if not, this will) rm -rf Delete_this_dir
11) ls -al
12) rm delete_this_file testfile

touch and tee

touch: create a file (empty), change file access and modification times
tee: replicates standard output into a file

HANDS ON:

1) man touch > touch_man (anything displayed?)
2) cat touch_man
3) man tee | tee tee_man
4) cat tee_man
5) touch testfile
6) touch delete_this_file
7) cat testfile (see anything?)

UNIX File Names

When creating UNIX files you should:

1) avoid special characters
2) avoid spaces in the name
3) select a name that makes sense
4) select a name you're willing to type
5) use of extension ".txt" a good idea
6) underscores and dashes are ok in name
7) avoid extensions like ".doc, .xls, .exe"

redirection and pipes

shell redirection and pipes

redirects output of a process to a file - >

pipe output of a process to another process - |

UNIX consists of processes and files…

pipes permit the transfer of process outputs to other processes or files…

SPECIAL CHARACTERS

---

the following are standard special characters:

& ; | * ? ' " ` [ ] ( ) $ < > { } ^ # / \ % ! ~ -

(spaces also have significance)

Be careful with the use of these characters at the command line or in file names. If you have unexpected results with a command or a script it may be related to one of these.

col

col - filter reverse line feeds from input
useful for cleaning up “man pages”

Hands ON:

1) pwd (are you in “Subdirectory”?)
2) man col | col -b > col-man.txt
3) man ls | col -b > ls-man.txt
4) man more > more-man
5) ls -alt (note the order, try ls -al)
6) cat col-man.txt
7) cat more-man
8) cat more-man | col -b > more-man.txt
9) cat more-man.txt (note the difference?)

df / du

df - display free disk space (512 Bytes)
df -k - show free disk space in Kbytes

use to check disk space: UNIX filesystems should be less than 70% full. The root (/) filesystem is very critical.

du - display disk usage (512 Bytes)
du -k - display disk usage in Kbytes
du -s - display disk usage SUMMARY
du -sk - display disk usage SUMMARY in Kbytes

Used to find large files or summarize directories
The command will identify the current directory
or a specified path. Oftentimes it is used with the ".”

kill

kill is used to terminate a PROCESS

in order to find the correct process one uses “ps”

Once the PID is determined then you may kill the process: kill PID

if the process persists, then try: kill -9 PID

“date”

- **date** - displays the date and time
- SYNOPSIS: /usr/bin/date [-u] [+ format]

an EXAMPLE in a script:

/usr/bin/echo "=====" | /usr/bin/tee -a `/usr/bin/hostname`.`/usr/bin/date +%o%b%d`.status

(this creates a file with the hostname.99Nov23.status)

extract:

/usr/bin/cp /etc/hosts /etc/hosts.`date +'%y%m%d-%H'`

extracting date information

`date | awk '{print $6$2$3}'`

or date +'%y%m%d.%h'

TESTING the addition of a suffix to a file on HP-UX 10.20

touch file.`date +%o%b%d`

-rw-r--r-- 1 jmeister tty 0 Nov 23 13:04 file.99Nov23

making a copy of a system file BEFORE editing

cp /etc/hosts /etc/hosts.`date +%o%b%d`

(note: in Linux, use %y instead of %o for the year)

grep – global regular expression

Extracting select data from a listing of mulitple files...

ls -al *0* | grep -v total | grep -v lrwxr | grep -v lost | grep -v '\.' | more

Another excellent use of "grep" is to locate content in a file:

grep john /web/*.html - this will find all occurrences of john in html files in /web

Date: Wed, 2 Dec 1998 14:37:13 -0800 script to kill off unattached processes left from failed connections.

```
#!/bin/sh
w | grep -v root | sort
ps -aux | grep -v root | grep -v bin | sort
while (ps -aux | grep -v root | grep -v bin | grep ?)
      do   (
        echo "   killing the unattached process(es)... "
      ps -aux | grep -v root | grep -v bin | grep ? | awk '{ print "/bin/kill -9 " $2 }' > /tmp/killit
       sh /tmp/killit
        /bin/rm /tmp/killit
        sleep 5
       w | grep -v root | sort
       ps -aux | grep -v root | grep -v bin | sort
        )
        done
```

awk

awk is not a sound...

…it is the compilation of the three authors of the tool:

A. V. Aho, B. W. Kernighan, P. J. Weinberger:

The AWK Programming Language, Addison-Wesley, 1988.

awk is a pattern scanning and processing language

things to do with awk

check user id's and home directories:

```
cat /etc/passwd | awk -F: '{print $3" " $6}' | sort > sorted.pass.3.6
```

sort a password file in NIS:

```
cat /var/yp/passwd | awk -F: '{print $3" "$6" "$1}' | sort
```

test login against current userid:

```
if [ `who am i | awk '{print $1}'`=`whoami`]
then
blah;blah
exit;exit
fi
```

test to force an su login:

```
case `/usr/bin/who am i | awk '{ print $1 }'` in
   userid)  if [ `/usr/bin/who am i | awk '{ print $1 }'` = `/usr/bin/whoami` ]
        then
        echo "################no#login#######################"
        exit;exit
        fi;;
esac
```

find

using find to clean /tmp of files older than 1 day

host.root-#-> **find . -atime +1 -name '*' -exec rm -f {} \;**
or
host.root-#-> **find /tmp -atime +1 -name '*' -exec rm -f {} \;**

also: **find /usr/tmp -atime +1 -name '*' -exec rm -f {} \;**
remove cores: **find / -name core -exec rm -f {} \;**

using find to compress overladen user directories

find . -type f -exec compress -f {} \;

(and to undo it should they complain:)
find . -type f -exec uncompress {} \;

sed

sed - stream editor

cool sed tricks

Remove Blank Lines from file

cat file-blanklines-2-be-deleted | sed -e '/^$/d' | tee newfile-no-blanklines

some "sed-like" tricks from within vi:

insert # at beginning of lines

:%s/.*/# &/g - inserts a # at the beginning of every line

:15,$s/.*/# &/g - inserts a # at the beginning of line 15, until the end of the file

:15,250s/.*/# &/g - inserts a # from line 15 through 250.

turn one or more blank spaces into one space

:%s/ */ /g

remove blank lines, or lines with only spaces or tabs

:g/^[t]*$/d --> WHERE t=tab

Comparison of Operating Systems

SEE: http://LinuxMeister.net/Microsoft/DOS-Linux-cmds.html

perl (command line use)

perl - Practical Extraction and Report Language
also refered to as - practical eclectic rubbish lister (by the author of the language)

perl may be used at the command line in conjunction with other UNIX tools
like find and sed:

find ./ -type f -name '*.htm*' -exec perl -pi -e 's/Galations/Galatians/g'
{} \;
find ./ -type f -name '*.txt' -exec perl -pi -e 's/Galations/Galatians/g' {} \;

Remember the exercise to remove the microsoft carriage return?

tr -d '\015' < original.file > new.file

Perl at the command line makes the task simpler:

perl -pi -e 's/\015//g' file

basic scripting

scripting is basically placing commands
in a file and executing them

template:

```
#!/bin/ksh
#  author:  john
#  date: 5/9/96
#  purpose: template
#  Korn shell script template
USAGE="command: options"
# define variables:
# CP=/bin/cp # express as $CP
# place korn shell commands starting here
#
# END of template
```

basic scripting – define variables

```
#!/bin/ksh
#  author:  john
#  date: 5/9/96    # note: export is a shell function, not a command
#################################################
#  define variables - use variables vs. hard coding paths
#################################################
CAT=/bin/cat
CP=/bin/cp
DATE=`/bin/date | /bin/cut -c 5-11` ; export DATE  # echo $DATE
FDATE=`/bin/date +%d%b%y-%H%M` ; export FDATE # echo $FDATE
GREP=/bin/grep
LS=/bin/ls
MKDIR=/bin/mkdir
MV=/bin/mv
PERL=/usr/bin/perl
SORT=/usr/bin/sort
TITLE="`echo "$1"`" ; export TITLE ; echo $TITLE
UNIQU="/usr/bin/uniq -u"
VI=/usr/bin/vi
WCL="/usr/bin/wc -l"
#################################################
# END of template
```

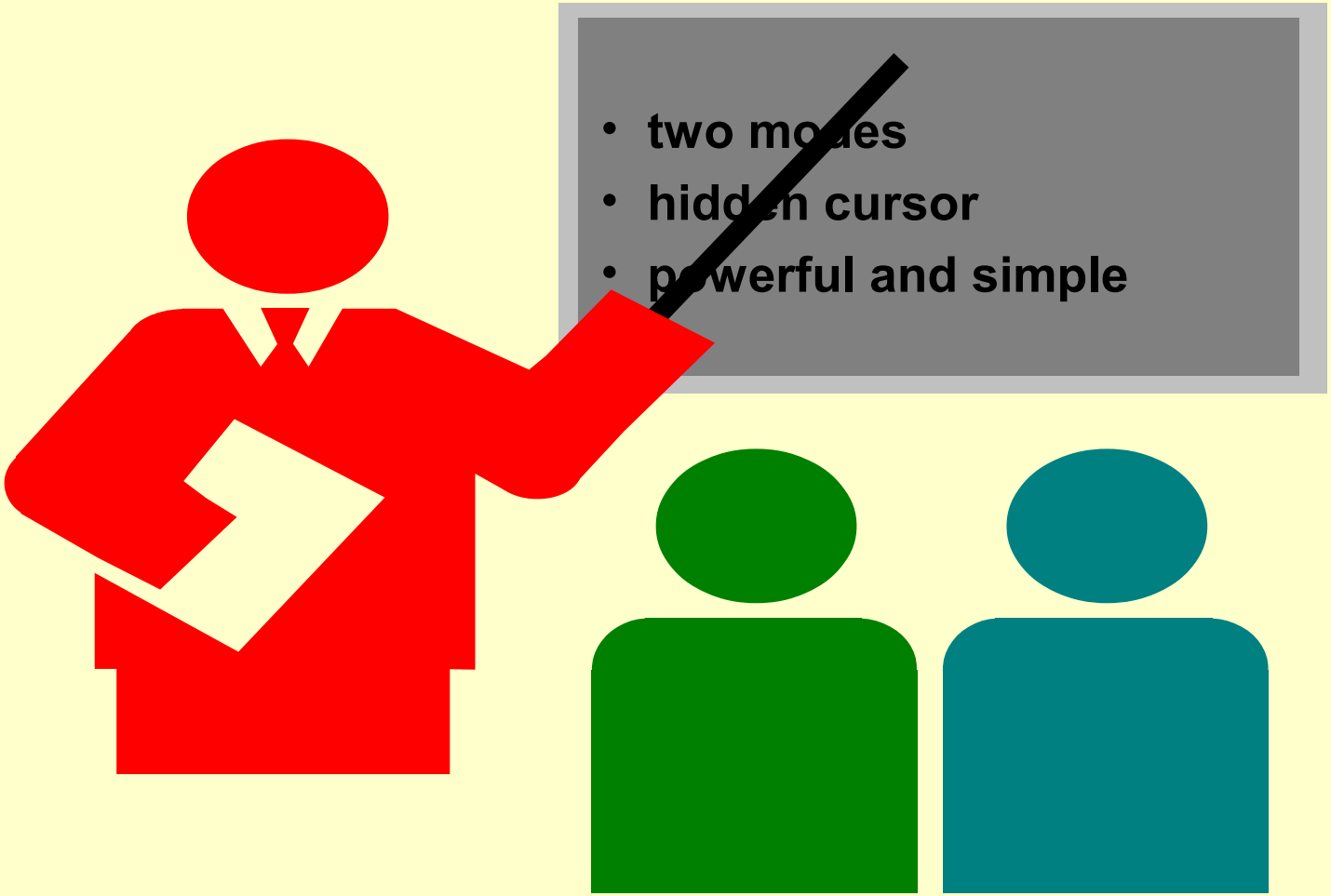
scripting example

```
#!/bin/sh
#  john meister September 20, 2000
#  PURPOSE: removes PC type carriage returns (^M) from ascii files
#  ARGS: $1 is filename passed with script
TEMPFILE=/tmp/temp_file$$
tr -d '\015' < $1 > $TEMPFILE
cmp $1 $TEMPFILE
  if [ $? -eq 0 ]
        then
          echo "    "
          echo "            no evil micr0s0ft influence detected,"
          echo "      file not edited...  carry on..."
          echo "    "
        else
          echo "    "
          echo "              purging file of evil micr0s0ft influence"
          echo "    "
          echo "  one moment please..."
          echo "    "
          echo "        file edited.      ########"
          echo "                    don't you feel better now?  "
          echo "                        ########"
          echo "    "
          cp $TEMPFILE $1
  fi
rm -f $TEMPFILE
```

Sample script - trim log file

```
#!/bin/sh
LOG=/root/logs/`date +'%y%b%d'-log.txt`
cd /var/logs/
/bin/ls -al *log*    |   tee -a $LOG
echo " ------- ” |   tee -a $LOG
/bin/ls -al wagoneers-access_log |   tee -a $LOG
echo "  " |   tee -a $LOG
if [ ! -f `date +'%y%b%d'-log.txt` ]
                then
                /bin/echo "         trimming log file” \
                                                >> $LOG
                echo " ------- ” >> $LOG
                /bin/cp access_log `date +'%y%b%d'-log.txt`
                /bin/cat /dev/null > access_log
    else echo "         existing log file - did not overwrite"
          echo " ------- ” >> $LOG
fi
echo " ------- ”  >> $LOG
/bin/ls -al wagoneers-access_log  | tee -a $LOG
echo " ------- ” |   tee -a $LOG
date |   tee -a $LOG
df . |   tee -a $LOG
```

intro to the vi editor



why vi?

The vi editor is available on **all** UNIX-like systems.

The vi editor does not require a GUI or X windows.

The vi editor is extremely powerful and fast.

The vi editor is used in the shell at the command line.

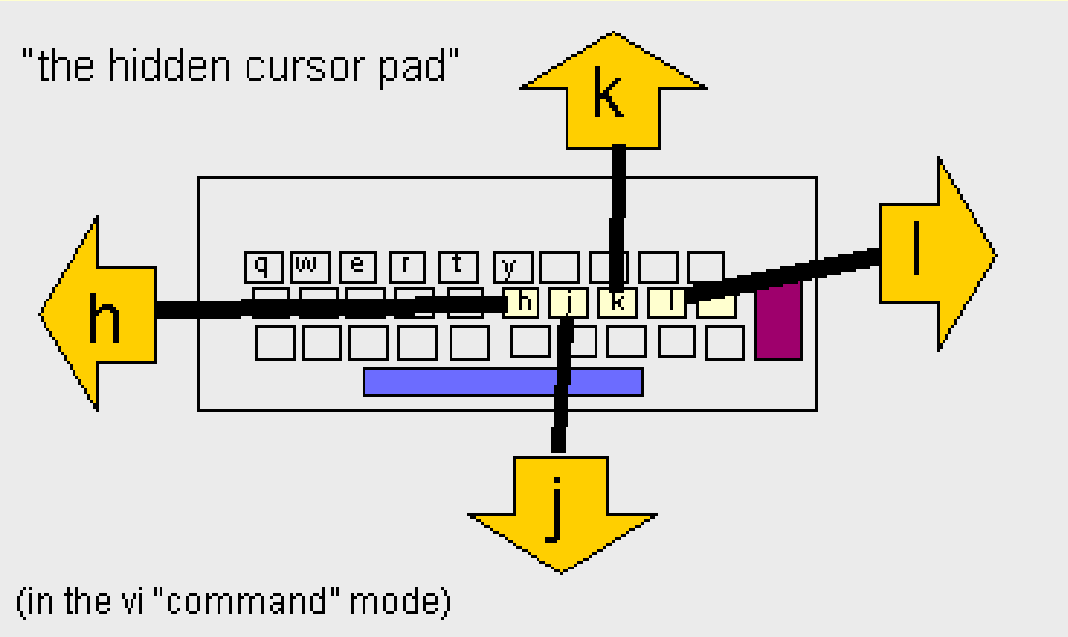
The vi editor can be used without a monitor...

vi - steep learning curve...

The vi editor has a few limitations:

- very steep learning curve
- doesn’t use mouse or other graphical pointer

vi Editor: Simple Cursor Movement



vi Editor: Simple Cursor Movement

KEY	EFFECT
h	Move one character left
j	Move down one line
k	Move up one line
l	Move one character right
0	Move to beginning of current line
	(Note: this is "zero" key)
$	Move to end of current line

vi Editor: Movement

KEY	EFFECT – CURSOR MOVEMENT
w	Next beginning of word, treating punctuation as new word
b	Previous beginning of word, treating punctuation as new word

KEY	EFFECT – SCREEN MOVEMENT
^f	Scroll one screen forward
^b	Scroll one screen back

vi Editor: Editing

KEY	EFFECT
i	Insert text
o	Insert line below cursor
A	Append at end of line
esc	Command mode
:	Invoke “ex” command
r	Replace character
cw	Change word
x	Delete character
dw	Delete word
dd	Delete line

cont’d on next slide

vi Editor: Editing (cont'd)

KEY	EFFECT
p	Put (paste) contents of buffer
yw	Yank (copy) word
yy	Yank (copy) line
u	Undo last command
.	Repeat last command
U	Undo all changes to line
d$	Delete to end of line
C	Change text to end of line
J	Join lines

vi Editor: Movement by Searches

KEY	EFFECT
/ *pattern*	Search **forward** for *pattern*
? *pattern*	Search **backward** for *pattern*
n	Repeat search in same direction
N	Repeat search in **opposite** direction
/ <Ret>	Repeat previous forward search
? <Ret>	Repeat previous backward search

FILE PERMISSIONS

UNIX is a multi-user environment. Groups may share project files.
The permissions are broken into three groups:

OWNER GROUP OTHER

We control three aspects of the file:

READ (including copy) (value 4)
WRITE (including delete) (value 2)
EXECUTE (run as a program) (value 1)

A binary format is used to define file permissions.
The format is MSB (most significant bit) to LSB.

4 2 1 - 4 2 1 - 4 2 1
r w x r w x r w x

EXAMPLE:

```
total 58
drwxr-xr-x   6 john   sysadmin      1024 May 20  1999 .
drwxrwxr-x   5 john   sysadmin      1024 Aug  3 09:23 ..
-rw-rw-rw-   1 john   sysadmin        94 May 20  1999 aliases
-rw-r--r--   1 john   sysadmin      1512 May 20  1999 checklist.workstation
-rw-r--r--   1 john   sysadmin      1368 May 20  1999 hosts.980223
drwxr-xr-x   2 john   sysadmin      1024 May 20  1999 sun-base
-rwxr--r--   1 john   sysadmin      7985 May 20  1999 sun-config
-rw-r--r--   1 john   sysadmin      1021 May 20  1999 profile
-rw-r--r--   1 john   sysadmin      8511 May 20  1999 profile.980406
```

chmod / chown

chmod - change file permissions

More than one way to change permissions.

Example: change a file to user write/read

chmod 644 filename

or chmod u+w

chown - change file ownership

Example: change a files ownership

(note: you will need to be the owner of the file or directory)

chown john filename

or chown john:sysadmin filename

ksh / CDE setup

```
#########################################################################
#  ksh environment for Solaris 2.x  john jan 11, 2001
#########################################################################
 umask 022
#    basic aliases for ease of use and to protect system
alias ll="/usr/bin/ls -al”  ;  alias mv="/usr/bin/mv -i” ;  alias cp="/usr/bin/cp -i"
HISTFILE=~/History/$$sh_history; export HISTFILE
HISTSIZE=512; export HISTSIZE
EDITOR=/usr/bin/vi; export EDITOR
#########################################################################
#    PS1 prompt
#########################################################################
export WHMI=$(/usr/ucb/whoami)
export WHST=$(/usr/bin/hostname)
export PS1="$(print '\001\015\001\0033]2; $WHMI  ---> \"$WHST\"   ---> $PWD \007
 ------------------------------------------------
$WHMI@$WHST [$PWD]

>-->\001 ')"
#########################################################################
PS2="*--> additional input required:  > "; export PS2
#########################################################################
stty erase "^H" ; stty kill "^U" ; stty intr "^C" ; stty eof "^D"
resize
```

tr

tr is the translate program

try this command:

echo njdsptpgu mbdlt tfdvsjuz | tr b-z a-y

---

it should produce: microsoft lacks security

---

now try this one: echo Jftvt mpwft zpv! | tr b-z a-y

tar, compress and gzip

To back up or move data TAR is used.

tar - tape archive is the most common.

to create a tar file: **tar cvf file.tar file**

to untar a file: **tar xvf file.tar**

see man tar for more information.

to minimize the size of the tar file:

compress file.tar (it'll produce file.tar.Z)

gzip file.tar (it'll produced file.tar.gz)

to uncompress:

uncompress file.tar.Z

gunzip file.tar.gz

ftp - file transfer protocol

To move files between systems **ftp** is used.

to use:

ftp [-i] hostname

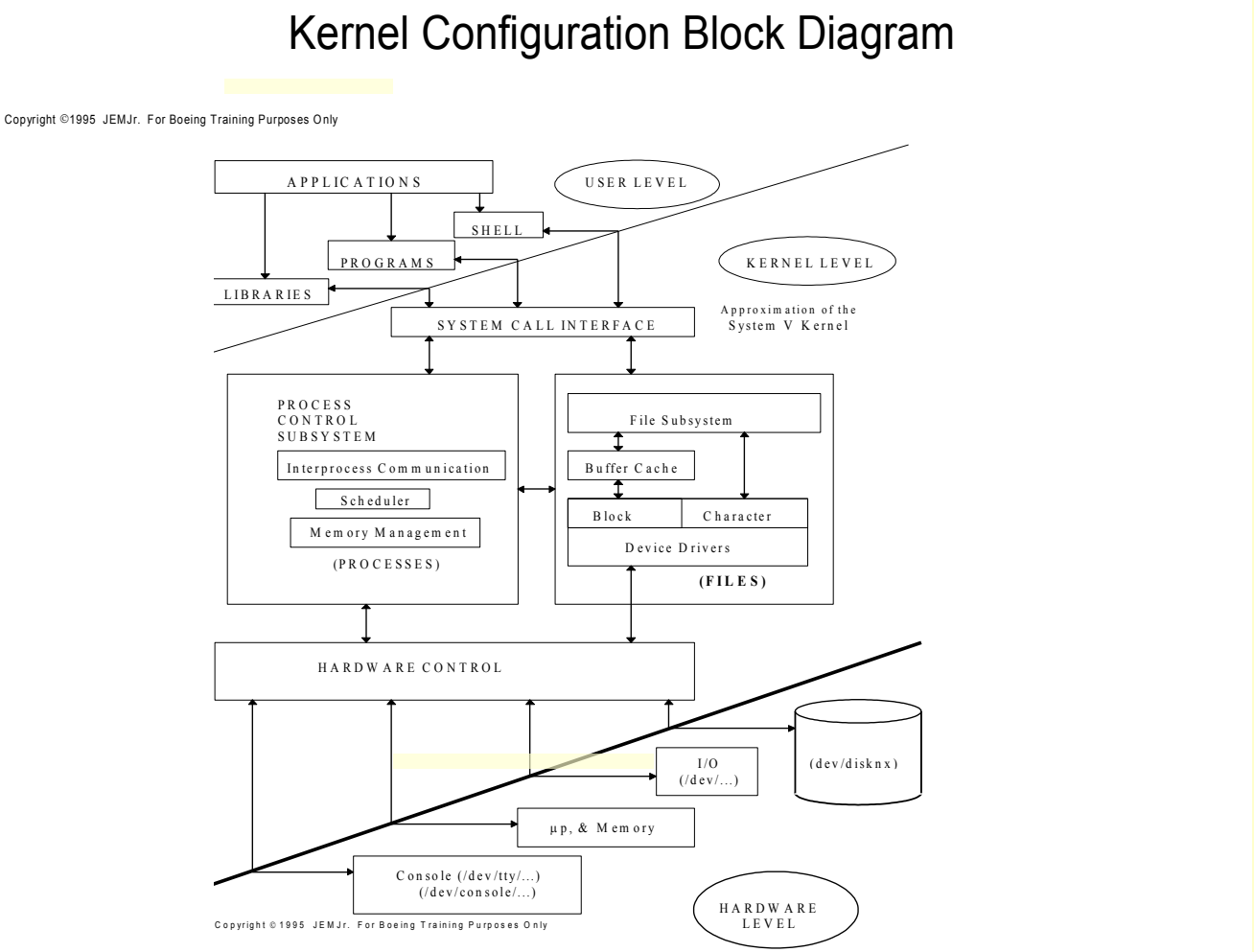
provide login id & password

commands used: **dir, cd, get, put, del, mput, mget, quit**

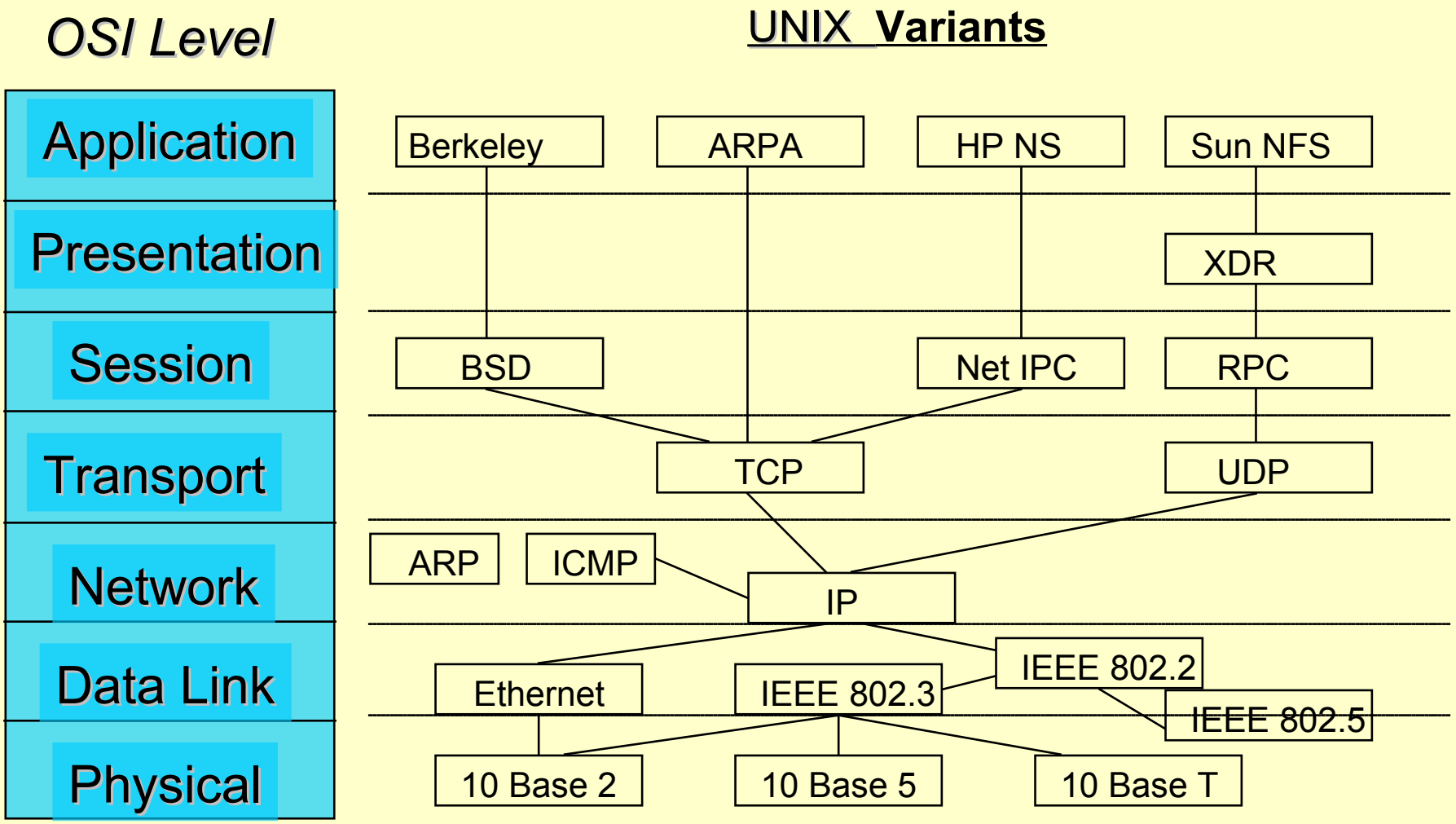
to get help type "?"

NOTE: ftp is rarely used now – use "scp"
"ssh" is used in place of telnet or rlogin

UNIX kernel block diagram



Internet Protocols - UNIX Variants



Programmers Do Not Throw Sausage Pizza Away

network abbreviations

Abbreviation	Stands for:
FTP	File Transfer Protocol
SMTP	Simple Mail Transfer Protocol
UDP	User Datagram Protocol
IP	Internet Protocol
TCP	Transmission Control Protocol
NFS	Network File System
DNS	Domain Name Service
SNMP	Simple Network Management Protocol
ICMP	Internet Control Message Protocol
MIB	Management Information Base
PING	Packet InterNet Groper

inetd daemon: provides Internet service management for a network

- (configuration: **/etc/inetd.conf** and **/etc/services files**)
- **comsat** daemon - biff's buddy (incoming mail...)
- **ftpd** daemon - File Transfer Protocol (TCP)
- **fingerd** daemon - finger services
- **rlogind** daemon - remote login services
- **rexecd** daemon - execute commands on remote host
- **rshd** daemon - restricted Bourne shell
- **talkd** daemon - talk function
- **telnetd** daemon - telnet services
- **tftpd** daemon - Trivial File Transfer Protocol (UDP)
- **uucpd** daemon - UNIX-to-UNIX copy protocol

  NOTE: **sshd** daemon – secure shell

network configuration files

The following assume TCP/IP and the use of
Domain Name Service (DNS):

- /etc/hosts - hostname & ip address and valid loopback address

---

- /etc/resolv.conf - domain, nameservers, search path

- /etc/inetd.conf - internet server configuration database

- /etc/services - identifies network commands, port #'s, etc.
- /etc/sendmail.cf - relay host, host name, etc.
- NOTE: /etc/postfix/main.cf

Typical Roles of a System Administrator

SUPPORT THE USER

Keep the systems running
Keep the systems "patched" (security, reliability, performance)
Create user accounts
Manage security and monitor
Provide System Documentation
Assist Users in the access and configuration of their accounts
Create tools and scripts to improve system usage
Backup and Restore Systems
Monitor system usage
Deal with printing issues
Troubleshoot Problems
Remove user accounts
Maintain system performance and security
Install computer systems & accessories
Install software onto system
Evaluate new products & tools

reading list

TITLE	isbn		Author(s)
UNIX for Dummies	1-878058-58-4		Levine & Young
UNIX, A Practical Guide	0-8053-7565-1		Sobell, Mark
UNIX Power Tools	0-679-79073-X		Peek, Loukides, et al
Unix System Administration Handbook	0-13-151051-7		Nemeth, Snyder, Seebass, Hein
Linux, A practical Guide			Sobell, Mark
The New Kornshell	0-13-182700-6		Korn, Bolsky

http://LinuxMeister.net/overview-LINUX.pdf - *notes added 11/3/2014*

for Linux or UNIX training contact john@johnmeister.com