Introduction to Unix

Jayaprakash Pisharath

Computer Engineering Seminar
22 October 2003
Outline

- Unix Core
  - Unix shell
  - Startup and scripts
  - Tips for everyday usage
- Networking
  - Networking basics
  - Security/Network issues
  - NFS basics
  - NIS basics
- Wrap up
About this talk

- We are all here to learn
- I’m not presenting my research paper
- Some of the opinions/styles of usage are just mine. Yours may differ
UNIX Core
Unix

Why Unix?
- Foundations are very strong.
  - Developed by, for the research people
  - First kernel ~ 16K
- Lot of improvements over the years. Lot of variants: SCO, BSD, Sun, HP, SGI, IBM... Linux followed suit.

Objective of this talk. To make people aware of the fact that Unix is not scary and that it is very powerful (esp. for doing research).
- Simple to use. Learn it once – then on, you will find it simple.
Shell

- Shell is
  - A command interpreter
  - A fundamental tool not only for running programs but also for writing them
  - The key to customization of a system as per the user needs.
What is it, what can it do

- Command line interpreter
  - Identifies your command, runs them for you
  - Easiest way to work with commands – man
- Types of shell
  - Bourne shell – sh
  - C shell – csh (“C” type of programming)
  - Korn shell – ksh (an improved version of ones above)
  - GNU shell – bash (sh bourne again)
  - Others – tcsh, zsh, rc, and so on
- Each shell has its own advantages and disadvantages. Try to stick to one (sh, ksh, bash are kind of better) as syntaxes are different across shells.
Shell and more

- Ask your sys admin if you want to change your default shell
- Even you can spawn it by issuing commands (sh, ksh, bash, tcsh)
- Setting the SHELL variable

- How are files arranged and what can I customize?
File Organization

- `/` - root
- `/bin, /sbin` - has most of the programs
- `/usr` – has user programs. Typical places to look for
  - `/usr/bin`
  - `/usr/sbin`
  - `/usr/X11/bin` (XWindows)
- `/var` – workspace for most of the programs
  (mail spool, print spool)
- `/tmp, /var/tmp` – temp storage (locks, data). **NEVER** delete anything here.
- `/home` – home directories
- `/etc` – don’t bother unless you are root (config)
- `/include, /lib` – includes, libraries (also in `/usr`). More later.
Going back to shell

- Environment variables
  - Variables that are used globally by the Operating System (OS)
  - OS uses this to track programs, find your home directories, set your machine’s screen display, etc.
  - Helps you work with a shell w/o having to type long names
  - Commonly used ones
    - HOME – home dir
    - PATH – path for various programs (/bin, /sbin, etc)
    - MANPATH – path for man
    - DISPLAY – bring remote display to your machine
- echo $<variable> gives the current values
  - echo $HOME
Setting an Environment Variable

Find out which shell you are running
- Easiest way is to do “ps”

setenv – for C based shells (csh, tcsh)
- setenv <VARIABLE> <VALUE>
  setenv DISPLAY “mymachine.domain.com:0.0”
  setenv PATH $PATH:"/home/JP/bin"
  setenv PATH $PATH:"$HOME/bin"
  setenv PATH “$HOME/bin”:$PATH

export – for ksh, bash
- export DISPLAY="mymachine.domain.com:0.0"
- export PATH=$PATH:$HOME/bin

You can also just type setenv without any input and see. If it says “command not found”, then use export!

Many more – EDITOR, SHELL, USER, PRINTER

Unset the variable using “unsetenv”
Shell Variables

- Shell variables – specific to shell. You might not use it.
- Shell variables are both set and displayed using the set command. They can be unset by using the unset command
  - cwd: your current working directory
  - home: the path name of your home directory
  - path: the list of directories to search for programs
  - prompt: the text string used in the prompt for input
  - shell: your login shell
  - term: the type of 'terminal' you are using
  - argv: of significance to shell programmers
  - status: a code indicating the result of previous command
Login Process

- Environment variables, your other settings (available printers, mount points, etc)
- You can easily customize most of it
- Basics of script
  - Script is simply a collection of operating system commands put into a text file in their order of execution
  - Any file can be changed to a script by `chmod +x <filename>`
  - Example script (example)
    ```
    # Example Code
    echo "My home directory is $HOME"
    echo "The time is [`date +%h%m%s%p`] by my clock"
    ```
“dot” files

- “dot” files
  - They are hidden files (ls –a will list them). Mostly they are stand-alone scripts, meaning, anyone can run them.
  - Meant to help you customize your setup as per your needs
- Two types – normal scripts, X scripts
  - Normal scripts – login, logout, profile scripts
  - X scripts – meant for X server to setup your desktop
Initialization

- **.login**
  - Runs during each login. System looks for this file in user's home directory ($HOME). If not found, uses default values (from /etc).
  - Create a .login file in your $HOME
  - You can add stuff that you want to do during login. Example: set your wallpaper, start a calendar, list today’s appointments.
    - xv -root -max <file> -quit
    - $HOME/example
  - You can even start your own script from this. A script that monitors network traffic/lists who else is logged on!
  - Caveat: don’t do anything stupid. You might not be able to login.

- **.logout**
  - Runs during logout
  - You can do the same things as in .login
Initialization

- `.bashrc, .tcshrc, .kshrc`
  - These are files that customize the shell.
  - Different from `.login`. This runs every time you invoke a new shell.

- What can you put in here?
  - Anything that is useful for your daily use
  - Aliases
    - `alias rm="rm –i"` - this will make sure you don’t accidentally delete files
  - `PATH` settings
PATH settings

- You can have a local archive of your own programs. Programs that you compile, programs that you download off the net.
  - I use $HOME/bin as my local bin directory
- Basically, creating a shortcut for this is easy - useful for tab completion.
  - Open your dot file (eg. .cshrc, .tcshrc) file. Put the following:
    setenv PATH "$HOME/bin":$PATH
    (use export accordingly for bash, ksh)
X initialization

- When X server (your desktop) comes up
  - It looks for certain files in user’s home directory ($HOME).
  - If not found, uses default values (from /etc/X11)
- This is the set of dot files in your home directory that the X server looks for:
  - .xinitrc (x server generic)
  - .xsession (x server generic)
  - .twmrc (window manager specific)
X initialization

- .xinitrc: decides which window manager to use (ex. CDE, gnome, kde)
- .xsessions: programs that start in each X session (xclock, xterm)
- .twmrc: specifies the attributes of X (window size, color).
- There are lot of other files too – don’t worry too much (Xauthority, Xdefaults)
Everyday Tasks

- Check disk quota
  - “quota” specifies how much disk space a user is using up and how much is left. Works only if there is a quota limit set by your sys admin

- Check system disk configuration
  - `df -k`: Lists disk map (in KB).

- How much am I consuming?
  - `du -sk /home/username`
    - Gives how much a user is consuming in KB
  - `du -sk <dir>`
    - Gives how much `<dir>` is consuming in KB, for each file in dir, use
  - `du -k <dir>`
Everyday Tasks

- Want to clean up every time you login?
  - Put the following in your .login
    ```bash
    rm `find $HOME/* -name core`
    ```
- You can do a routine cleanup too. Clean up your cache every 10 days. Can use cron job for that (man cron). That’s how sys admins do lot of tasks.
- Perhaps, Chirayu can write a Perl script and show you all in next week’s session!!
Terminals

- Default in older machines: vt100 (DEC)
- New machines carry xterm as default
- Clearing a terminal if you see garbled output
  - “reset”: resets the terminal screen
  - “man reset” for cool examples about terminal test and set
- dtterm, xterm-color, linux – are some TERMS that support color. Useful if you have colored ls, vim, etc.
  - can also use setenv to set terminal (use environment variable TERM) dynamically in a given window
Networking Basics
IP address

- Every machine on the Internet has a unique identifying number, called an IP Address. A typical IP address looks like this: 216.27.61.137
- Interpreted in binary – the first 4 bits determine what class the address falls into
- 4 classes of IP addresses
  - Class A: 1.0.0.0 through 127.0.0.0 (big corporations)
  - Class B: 128.0.0.0 through 191.255.0.0 (educational, other)
  - Class C: 192.0.0.0 through 223.255.255.0 (smaller institutions)
  - Classes D, E, and F: 224.0.0.0 through 254.0.0.0. They are either experimental or are reserved for special purposes and don't specify any network.
- Finding your IP address: ifconfig (-a on some machines)
- You will see your machine’s IP address and a local 127.0.0.1
Subnet and Netmask

- Subnet: A way of creating more sub-networks within each network

- Within each network (found using netID), there can be multiple subnetworks.

- Way of adding another hierarchy to the system

- So, subnetID can be common across multiple networks (their netIDs are still different)

- netmask is a mask to strip the initial bits and identify subnets within networks

- netmask are customizable (no of bits used)

---

Example:

```
16 bits 8 bits 8 bits
1 0 NetID subnetID hostID

edu  northwestern  ece  jaguar
edu  uiuc  ece  jaguar
```

jaguar.ece.northwestern.edu

---

22 October 2003

Computer Engineering Seminar

© Jayaprakash Pisharath
Static IP

- Admin manages everything.
- Admin has to manually assign you an IP (from a paper book!)
- Admin updates server files in /etc (hosts, resolv, sysconfig files) to register that you are part of his/her network.
- Restarts network daemon (“network”) to finalize the process
- Advantages: Very secure, easy to keep track of machines in the network
- Disadvantages: Very difficult job for sys admin to maintain, requires restart of network interface
Dynamic IP

- Worry-free life for a user/sysadmin
- Machine uses DHCP (Dynamic Host Configuration Protocol) to get an IP
- Remote server automates and manages the network configuration of desktop computers and other network devices that use the TCP/IP protocol
- Automatically assigns an IP every time a new machine comes to the network
- Advantages: Very easy to manage, no network restart
- Disadvantages: Tough to keep track of requests (log could get messy), security issues
Security issues
Problems

- Traffic analysis, eavesdropping, IP spoofing, DoS – don’t they sound too familiar?
- Unix is secure but can also be broken into!
- telnet allows you to pass some settings explicitly on to the server (such as environment variables).
- rlogin is a very unsafe way of logging into machines (esp. because you don’t have to type anything)
- ftp – you are transferring raw data
Secure Shell (SSH)

- Full secure replacement of ftp, telnet, rsh, rlogin
- Authentications
  - Password
  - Public-key encryption
  - Host-based (mutual handshaking)
SSH

- No clear-text passwords (multiple algorithms)
- Compression of data for speed
- Transparent, automatic tunneling
  - Caution: should use SSH2 or above
- Can run other applications (mail, graphic applications) through the same secure tunnel
Security issues

What can I as a user do?
- Use only ssh, sftp. Command line for Unix based systems (install it if not seen on your system). You can find it at www.openssh.org
- If you are trying to connect remotely from Windows, install ssh on the Windows box too. It is available at www.it.northwestern.edu (look in Software → Internet Software link) free for NU students.

What can I as a sys admin do?
- Disable all telnet, ftp, rlogin related services as the FIRST thing. Install and run SSH server.
- Do a ps to see if there are any unwanted/suspicious processes running.
- “netstat” can list open ports, services, etc. You can use this to detect vulnerable services – could also capture any malicious code trying to open ports
- Read /var/log files
- Make sure your system is updated regularly (for Unix* and windows)
NFS & NIS
NFS

- Network File System
  - Enables one machine to access another machine’s files (something like a “Network Neighborhood” for Unix)
  - NFS server, NFS client
  - Server just makes a directory shared by running certain daemons (nfsd)
  - Clients access files on the server by mounting (mount/automount) the server's shared file system.
NFS – A user perspective

Advantages
- Allows multiple computers to use the same files
- Reduces storage costs
- Provides data consistency and reliability
- Supports heterogeneous environments
- Reduces system administration overhead

Traffic is optimized using caches, bulk requests
Server can service even a disk-less client, but these days, most machines come with enough disk space

Disadvantages:
- Network is the bottleneck
- If you update your file very often, better avoid NFS directories.
- If $HOME is mounted, one can feel the bottleneck.
- Use a local directory and use mounted dir to do a single update, OR use nfs dir for storing large data that needs fewer updates
NIS – A user perspective

- Network Information Service
  - A distributed database service that allows a single set of configuration files to be maintained for an entire network of machines
- One master processor has global configurations
- Slaves copy the configurations.
- Updates (done on master/slaves) gets reflected everywhere
- NIS – complicated, could help only if you really need it (many machines, too many mounts, too many users)
- Unfortunately, it isn’t exaggeration to say that NIS is a security nightmare (ypbind, other issues)
Conclusions

- Unix is not that scary
- Learn it once, use it again and again
- Shells are not for geeks, it is for everyone. Use it for customization. It helps.
- How do you see your IP address in Unix?
- DHCP is better
- Say no to telnet, ftp, rlogin
- NFS and NIS – judge and decide
References

- Unix Shells:
  - http://www.kornshell.com/
  - http://www.tcsh.org/Welcome

- Scripting:
  - UNIX in a Nutshell (O'Reilly)
  - http://www.injunea.demon.co.uk/

- System administration:
  - Essential System Administration (O'Reilly)
  - Unix Guru Universe: http://www.ugu.com/

- NFS, NIS:
  - http://docs.sun.com
  - Search on www.google.com

More presentations are coming

- Perl (Chirayu – 29 Oct)
- Web/CGI (Sasha – 5 Nov)
- make/cook (Joseph – 19 Nov)
- latex (Robert – 26 Nov)
- MATLAB (Maged – 3 Dec)