Unix System Administration

Chris Schenk
Lecture 01 – Tuesday Jan 15

CSCI 4113, Spring 2008

Course Information

- http://www.cs.colorado.edu/~schenkc/courses/csci4113-sp08
- Use above page for all materials
  - Available from my home page
- Course Text
  - Linux Administration Handbook
- Others (all O'Reilly, go figure)
  - Running Linux, 4th Edition
  - Unix in a Nutshell, 4th Edition
  - Essential System Administration, 3rd Edition

Course Information (cont)

- Logistics
  - TR, ECCR 105, 9:30 - 10:45am
  - Final Time Saturday, May 3rd 7:30am-10:00am
- Office Hours
  - ECCS 128 (CSEL), TW 5:00pm-6:00pm
  - More as needed by appointment
  - ***schenkc@colorado.edu*** (better than stopping by unannounced)
    - Make sure '**4113**' shows up somewhere in the subject

Course Information (cont)

- Grading
  - 70% -- Labs (10 or 11 of them)
  - 15% -- Quizzes (3 of them)
  - 15% -- Final exam
- Take-home style
  - Everything is due one week after it's assigned
  - Quizzes will be like labs, but covering other details
  - Final exam will be cumulative and due at the exam time
  - Everything will be on the course syllabus page

Topics

- Basic Unix navigation
  - Directories, files, links, modes, ownership, networking
- Building a machine
  - distributions, partitioning, packages, workstations, servers
- Servers/Services
  - SSH, web, mail, DNS, MySQL, PHP,
- Security
  - Passwords, certificates, firewalls, logs

Labs

- Build a machine from scratch
  - Work in teams of two!
- Ubuntu 7.10 server install
  - Minimal install with 2GB of disk space
  - 128MB ram available per machine
- Virtual machine run through VMWare
  - Vmware-server-console program in the CSEL
  - Hosted on nsa.cs.colorado.edu
  - Login with your CSEL password

Other

- Class Format: Informal
  - Large class
  - Ask lots of questions
- Slides
  - Available in OpenOffice 2.0 (it's free!) and PDF format on the web page
- Schedule
  - Not yet finished, but will be updated weekly
  - Syllabus page will have most information

CSEL Account Creation

- Login to CUConnect -> PLUS
- Email & Passwords
- Account Activation and Passwords
- Specialty Accounts
- Login with studentID & pin
- Select 'Create account for csel.cs'
- Enter new password twice, updates within 5 minutes
- **% ssh csel.cs.colorado.edu**

File Conventions

- /etc – config files
  - /etc/init.d – startup scripts for everything
- /tmp – temp files
- /var – variable files
  - Logs, packages, email inboxes, process Ids
- /usr – Things for the users
- /sbin, /usr/sbin – Super-user binaries
- /bin, /usr/bin – Regular user binaries
- /boot – kernel and boot loader

Startup Scripts

- `/etc/init.d`
- Controls the startup and shutdown of services
  - Controlled by runlevels
  - Default runlevel configured in `/etc/inittab`
- Runlevel scripts symlinked to actual scripts
  - `/etc/rc0.d`, `/etc/rc2.d`
- Scripts take different commands
  - Mainly 'start' and 'stop', but others as well (restart)

Editing Important Files

- Let's get used to 'vi'
  - `visudo`, `vipw`, `vigr`
- Safely edits the key files on the computer
  - /etc/passwd, /etc/group, /etc/sudoers
- Protects against two things
  - Bad syntax – if it's not readable, bad things happen!
  - Race conditions – atomic swap of the files
- Doesn't actually invoke 'vi'
  - Invokes `/usr/bin/editor` or what's in `$EDITOR`

Shells and Environments

- Look at available shells – `/etc/shells`
- Simply programs executed by the OS
  - Starts with mingetty, spawned by init
  - Executes `/bin/login`
  - `/bin/login` checks password and executes shell listed in `/etc/passwd`
  - Executed as a **login** shell
    - Privileges changed to your user
- Variables are created
  - `printenv` and `$echo $VARNAME` to see them

Shells and Environments (cont)

- `$PAGER` is like `$EDITOR`
  - `man` invokes `$PAGER`
  - `update-alternatives` changes both
- Forgot to type sudo before your command?
  - `% sudo !!`
- Want to execute the last 'vim' command?
  - `% !vim`
- Screen makes your life easier
  - % screen
  - `<ctrl>-a, d` and `screen -r`

Filesystems

- Devices
  - all behave like files, permissions like any other file
- Partitions and Types
  - physical separation of hard drive space
  - fdisk - ext2, ext3, vfat, swap, reiserfs, xfs
- Mounts
  - Access to partitions on the system
  - /etc/fstab
- Features
  - File types, modes, ownership

Features

- What most users see
  - I own my files (check with `stat` or `ls -l`)
  - Type: **-** (file), **b** (block dev), **c** (char dev), **d** (dir), **p** (pipe), **l** (symlink), **s** (socket)
  - **R**ead/**W**rite/e**X**ecute (rwx), **u**ser, **g**roup **o**ther
  - Size, Timestamps
  - Name!
- What users don't usually see
  - Reference count, INODE number
  - Access and Change time

Devices

- /dev directory describes all available devices
  - /dev/cdrom, /dev/sda*
- Different device types
  - block 'b', character stream 'c', pipe 'p'
  - TTYs, sound card, hard drives, cd drives
- Filesystems use block devices
  - /dev/hda, /dev/sda, /dev/scd0

Hard Drive Devices

- Labeling differs for IDE and SCSI devices
- IDE uses /dev/hd*
  - Primary IDE channel 1 - /dev/hda
  - Primary IDE channel 2 - /dev/hdc
- SCSI uses /dev/sd*
  - Based on SCSI channel ID
  - SCSI channel 1 - /dev/sda
  - SCSI channel 2 - /dev/sdb

Partitions

- A device is divided into physical extents
- Physical extents are raw
  - Filesystems built on top of partitions!
- Numbers are placed after device IDs
  - First partition on IDE device /dev/hda is /dev/hda1
  - Second partition on SCSI device /dev/sdb is /dev/sda2

fdisk – Partition Table Manipulator

- usage: `fdisk <device>`
  - `fdisk /dev/sda`
  - 'm' – print help
  - 'p' – print partition table
  - 'n' – add new partition
  - 't' – change partition's system ID
  - 'w' – write changes to disk

Partition Table

- Same sector as the Master Boot Record (MBR)
- 64-byte data structure, each entry 16 bytes long
- Maximum of 4 partitions!

- 10 fields in each entry in the table
  - Bootable flag
  - starting/ending head/sector/cylinder
  - System ID

System ID

- Describes the filesystem on each partition
  - independent to the OS, standard format
  - **one** bytes long, 256 total entries available
  - 0x06 (FAT16), 0x07 (NTFS), 0x05 (Extended), 0x82 (linux)
- Read by the OS at boot time
  - need to know which modules to load

Filesystems

- Partitions must be formatted to a specific filesystem
  - `mkfs` utility
  - different versions, mkfs.ext3, mkfs.reiser, etc
- Each filesystem requires a module in the kernel
  - `lsmod` (list modules) utility
  - 'used by' number of devices

Mounts

- Filesystem starts at / (root)
  - not to be confused with root user
  - tree structure starting a /, branching with directories
- A partition may be mounted to any point on the tree
  - partitions always mounted to directories:
    - /dev/sda1 - /
    - /dev/sda2 - /usr
    - /dev/sda5 - /var

Mounts (cont)

- How do you see your current mounts?
  - `mount` command
  - `df` command (with -h parameter)
- `mount` shows options on the filesystem
  - [no]exec, [no]atime, defaults, nosuid, ro/rw
- Partitions mounted at boot time read from a config file
  - /etc/fstab
  - Manually mounted partitions looked up in fstab
    - manual override of options

/etc/fstab

- Six fields for each line
  - device or label, mount point, filesystem type, options, dump enable, fsck priority
- Some devices are marked 'none'
  - filesystems used by the kernel for special tasks
  - process information, device addresses
- CDRoms list 'auto' for filesystem type
  - three types of cdrom filesystems: IS9660, Joilet, HFS
  - Data is the same, but the filesystem metadata is different

More on mount

- Manual mounting
  - `mount [-o <options>] <device> <mount point>`
- For entries in /etc/fstab only the device or mount point is needed
  - `mount /windows`
- Shows more information than simple devices?
  - NFS mounts

Mount Options

- `defaults` contains a number of options
  - rw, suid, dev, exec, auto, nouser, async
- `rw/ro` – Mount read/write, readonly
- `[no]exec` – Allow/disallow execution of binaries
- `[no]suid` – Allow/disallow setuid/setgid bits to work
- `[no]user` – Allow/disallow a user to mount the filesystem

umount

- Used to unmount a filesystem
- Almost identical in usage
  - `umount <device | mount point>`
- Problems may arise
  - `umount: /home: device is busy`
  - no file handles can be open if the device is to be unmounted
  - sitting in a directory on a device means files are open!
    - Your shell has handles open

lsof – List Open Files

- Great utility for debugging many types of problems
- Everything under linux is a file
  - so we can see almost anything with lsof!
- Usage
  - Open files: `lsof <file>`
  - Open ports: `lsof -i tcp:<port>`
  - Processes: `lsof -p <pid>`