CSE 265:
System and Network Administration

- Software Installation, Localization, and Maintenance
  - Installation, customization
  - Keeping your systems up to date
  - Package management: RPM
  - Automating downloading and installation: YUM
- Change Management
  - OS Upgrades
  - Maintenance windows
  - Service conversions

Software installation

- Linux is not pre-installed from most vendors

    - And even if it were, you'd want to re-install!

- A sysadmin must

    - Install Linux (or indeed, any OS)

    - Automate mass installations

    - Localize (customize) the systems

    - Keep the systems updated

    - Manage add-on software packages

Linux installation

- Basic Linux installation

  - From our first project, you now have some experience with this

  - It is usually easier with a CD-ROM or USB :-)

- Automating installation

  - Many packages:
    Solaris JumpStart,
    Red Hat KickStart,
    SUSE AutoYaST,
    Windows AutoLoad

KEEP CALM AND INSTALL LINUX

Localization / Customization

- A single common install is almost never enough
  - Different hardware, different service requirements
- Need to automate any customization!
- Track some limited set of common configurations
- Probably want some custom or additional non-OS-supplied software
  - e.g., GNU tools, graphics packages
  - Often goes in /usr/local or /opt
  - Sometimes want a more custom namespace
    - Permit installation of multiple versions of a package

Keeping your systems up to date

- Assuming you have only a few, centrally-managed OS configurations

- How do you keep all the systems up to date?

  - Copy files directly from master host

    - e.g., with rsync or rdist
    - difficult to use with core OS, OK for local filesystems

  - Use package management system built into distro

Package management

- Essentially all UNIX/Linux distributions use some kind of package management system
  - RPM for Red Hat, Fedora, SUSE
  - .deb for Debian, Ubuntu
- These packages can include applications, source code, configuration files, etc.
  - Usually can 'undo' the installation of a package too
  - Can run scripts to customize the installation
    - e.g., look in other config files for information

RPM

- **RPM Package Manager**
  - rpm
    - --install, --upgrade, --erase, --query
- **How to use**
  - Download updated package
  - rpm --upgrade openssh-2.9p2-12.i386.rpm
    - Get errors saying other packages depend on old one!
    - Download additional package updates
    - Upgrade all simultaneously

Using RPM

- Sometimes, we need to remove a package

[root@brian brian]# **rpm -q kernel**

kernel-2.6.12-1.1381_FC3

kernel-2.6.14-1.1644_FC4

kernel-2.6.14-1.1656_FC4

kernel-2.6.15-1.1831_FC4

kernel-2.6.15-1.1833_FC4

[root@brian brian]# **uname -a**

Linux brian.local.davison.net 2.6.14-1.1644_FC4 #1 Sun Nov 27 03:25:11 EST 2005 i686 i686 i386 GNU/Linux

[root@brian brian]# **rpm -e kernel-2.6.12-1.1381_FC3 kernel-2.6.14-1.1656_FC4 kernel-2.6.15-1.1831_FC4**
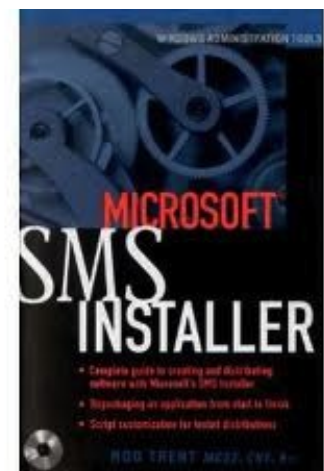
[root@brian brian]#

Automatic download & installation

- Sometimes you'll want to upgrade packages automatically (always have the latest updates)

  – Red Hat has commercial tools for this

- Can also use apt-get, apt-rpm, and yum for Linux; Solaris AutoPatch; Windows SMS

YUM

- Yellowdog Updater Modified
  - YUP: Yellowdog Updater
    - Provides updates across networks
  - YUM created by Duke sysadmins when trying to improve YUP
    - Separated headers from RPM files for dependency information

- Popular, powerful

- Can create your own YUM repositories

Using YUM

- yum install *packagename*

- yum search *string*

- yum provides *substring*

- yum update *packagename*

- yum update

Change management (1/3)

- Change management
  - Communication
    - Reduces errors – everyone thinks through proposed changes
  - Scheduling
    - Choose times to minimize impact

- Documented proc. for updating system config. files

- Revision history and locking
  - RCS, CVS, Subversion – useful for code development too!
  - Prevent simultaneous changes
  - Identify who made what changes, and why
  - Allow for recovery of old versions

Change management (2/3)

- Communicate changes to customers

- Scheduling depends on the kind of work

    - Routine updates

        • Happen all the time

        • Do not cause widespread problems when mistakes are made

    - Major updates

        • Affect a large number of systems

        • Require a significant outage

        • Relatively rare

    - Sensitive updates

        • Not large, but could cause significant outage

        • Reasonably common; scheduled for slow periods

Change management (3/3)

- Change proposal forms
  - Detail what changes to make
  - Systems and services affected
  - Reasons for change
  - Risks, test procedure
  - Time required

- Meetings to review proposed changes
  - Approve, reschedule
  - Examine plans, time-frames, back-out process

Server OS upgrades (1/3)

- Develop a service checklist
  - What services are provided?
  - What customers use the services?
  - Which software provides each service?
- Verify software compatibility with new OS
  - Contact vendors
  - Perhaps test on a separate machine
  - Some software may not work with new OS
    - Find software upgrades, or
    - Get different software, or
    - Drop the software entirely (assuming OS upgrade is non-negotiable)

OS Upgrades (2/3)

- Verification tests for each piece of software
  - Ideally, want a master script to say OK or FAIL
  - Some software may have a test script
  - Tests need to be tested and debugged before use in an upgrade!
- Write a back-out plan
  - Set a particular time at which the back-out plan is activated
  - Make backups before the upgrade
- Select a maintenance window
  - Decide when, and how long through agreement with customers

OS Upgrades (3/3)

- Announce the upgrade

- Execute the tests to make sure they are correct

  - And that errors don't exist before the upgrade (causing concern when failures are found afterwards!)

- Do the upgrade (with someone watching)

- Repeat tests with newer OS; debug if needed

- If all else fails, rely on the back-out plan

  - Test again to make sure back to starting state

- Communicate completion/back-out to customers

WE'RE UPGRADING OUR SERVERS!
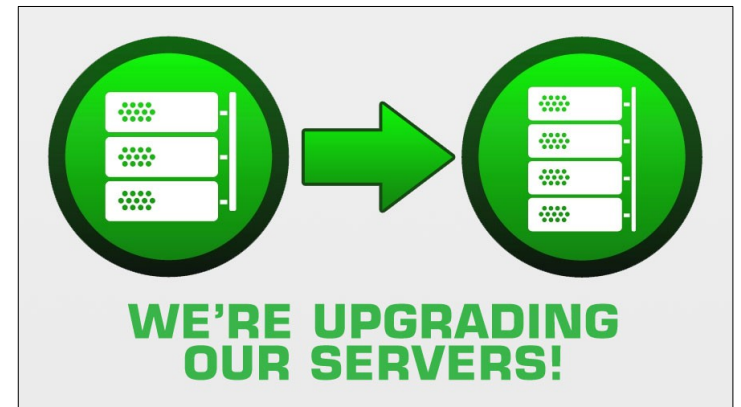
Maintenance windows (1/8)

- Maintenance windows

  - Time to make many changes, across multiple systems

  - Scheduled service interruptions

    - Disruptive cleaning

  - May stop all services/systems

    - Can reduce complexity, make testing easier

- Scheduling

  - Need to coordinate with rest of organization

  - Avoid end of month, quarter, or year

  - Announce early, perhaps more than a year in advance

Maintenance windows (2/8)

- Planning
  - All tasks need to be thought out in advance
  - Actual work during outage is (should be) just to follow the plan
- Flight director (as in NASA)
  - One person responsible
  - Sends out announcements
  - Scheduling/rejecting the submitted work proposals
  - Monitors progress, verifies that testing is completed
  - Decides when to back-out of a proposed change

Maintenance windows (3/8)

- Change proposals – what needs to be included?

Maintenance windows (4/8)

- – Change proposals
    - • What changes are going to be made?
    - • What machines will you be working on?
    - • What are the pre-maintenance window dependencies and due dates?
    - • What services need to be up for the change to happen?
    - • What will be affected by the change?
    - • Who is performing the work?
    - • How long for change – in active time and elapsed time, including testing, and how many people needed?
    - • What are the test procedures? What equipment is required?
    - • What is the back-out procedure and how long will it take?

Maintenance windows (5/8)

- Master plan

  - Considers

    - resource allocations (people, equipment, time)

    - dependencies (services, people, equipment)

  - Need slack in schedule to allow for things to go wrong!

- Disabling access

  - First step in maintenance window is to disable (or discourage) system access

    - Place notices with window times clearly visible

    - Disable remote access to site (dial-in, LAN, wireless)

    - Make announcements as window begins, set voicemail

Maintenance windows (6/8)

- Shutdown/boot sequence
  - Proper sequence is required for many systems that would otherwise hang indefinitely, waiting for a non-existent service

  - Might otherwise need to bring machines back up so that others can shut down (or start) cleanly
  - Incorrect sequence can also cause hard-to-debug failures

- Deadlines for change completion
  - Flight director decides when changes are taking too long and need to be aborted (use back-out plan)

Maintenance windows (7/8)

- Comprehensive system testing
  - Work incomplete unless fully tested
  - Often includes a system-wide shutdown and re-start
  - Generally includes visiting clients and testing desktops (might include rebooting every desktop)

- Post-maintenance communication
  - Let organization know that the system should be fully restored
  - Tell of main successes, and any continuing service outages (along with expected time to repair)
  - Write in advance for long outages

Maintenance windows (8/8)

- Re-enable remote access
  - Can't forget!
  - Also, reset voicemail
- Visible presence the next morning
  - Put flight director and other senior staff in helpdesk area to monitor calls and listen for problems regrd. completed work
  - Make visible customer concern
- Postmortem
  - Review what went wrong
  - Discuss what should be done differently

Service conversions (1/2)

- *Removing one service and replacing it with another*

- Small groups first, then expand

  - Minimize impact of any failures

- Communication

  - Alert customers to changes and how it will affect them in advance

- Minimize intrusiveness/layers vs. pillars

  - Better to make all customer-visible changes at once (per-customer)

  - "Rioting Mob" technique

Service conversions (2/2)

- Avoid flash-cuts!

    - Find bugs with small sets of users

    - May require extra resources (duplicate hardware, etc.) to provide redundant services

- Want successful flash-cuts (when unavoidable)

    - More communication, user training needed (British Telecom)

- Back-out plan

    - Must be able to go back to prior config in case of problems

        - Perhaps not noticed immediately

        - Need to decide in advance when back-out plan will be implemented

            - e.g., if conversion can't be completed within two hours