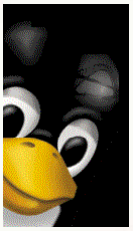
Vista Solutions

Enabling Specialized Computing Solutions

http://WWW.VistaSolutions.Net

Introduction to Linux for Unix System Administrators



Rob Lucke
Vista Solutions
Rob.Lucke@VistaSolutions.Net

Seminar Agenda

- **Introduction**
- **Section 1: System Installation**
  - Disk Partitioning Tutorial
  - System Installation
  - *Lab 1: Installation from CD-ROM/DVD*
  - Boot Managers: LILO and GRUB
  - Installing Linux from the Network
  - *Lab 2: System Installation from the Network*
- **Lunch**

Seminar Agenda (continued 1)

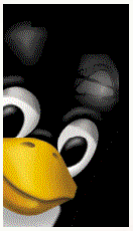
- **Section 2: System Administration**
  - Booting and Start-up
  - Hardware Configuration and Troubleshooting
  - Networking
  - File System Layout, What Goes Where
  - Software Installation and Update
  - Linux Security
  - *Lab 3: System Configuration*
  - Linux File Systems, Software RAID, and Quotas
  - DHCP, NFS, NIS, and Samba
  - Building The Linux Kernel and Modules
  - Dynamic Kernel Modules
  - *Lab 4: Kernel, Modules, Software, and Miscellaneous*
- **Questions and Wrap-Up (Graduation Time!)**

Warning!

This seminar is intended for Unix system administrators who have little or no exposure to Linux. If you just spent a week in Linux boot-camp, or if you have years of experience with Linux, then this seminar will be too basic for you.

Introduction

- What is Linux?
- What is a Linux Distribution?
- Linux Versioning
- What is “Open Source Software Development”?
- Getting Help and Software



What is Linux?

- **When we say "Linux", we are really talking about the combination of two things:**
  - An open source kernel, written by Linus Torvalds (Linux)
  - A set of open source tools from the "Free Software Foundation", the GNU tools
- **Linux is Unix-like, but written to published Unix (POSIX and other) specifications to avoid copyright and patent issues**
- **This is not an exhaustive tutorial, just some of the things that I have run into during my exposure to Linux**
- **This seminar is based on the RedHat Linux distribution version 9.0, there are many others (SuSE, Debian, Red Flag…)**

What is a Linux distribution?

- **A particular version of the kernel**
  - **RedHat 9.0 is based on the 2.4.20-13.9 kernel**
- **A particular set of packages and a package manager**
  - **May be in Redhat Package Manager (RPM) format**
  - **May be in Debian (Deb) format**
  - **May be tar-balls or source packages**
- **A particular system structure and "philosophy"**
  - **File system layout**
  - **Approach to system management**
- **A set of installation tools**
  - **Disk partitioning**
  - **System installation**
- **"Value Added"**
  - **Update tools**
  - **Support**
  - **Documentation**

Explaining Linux Version Numbers

- **The Linux kernel and most packages follow a similar (but not always identical) versioning scheme**
- **Kernel Version = MajorRelease.MinorRelease.Step**
  - **Odd-numbered minor-release kernels are "development"**
  - **Even-numberd minor-release kernels are "stable"**
  - **Minor release numbers are incremented with patches**
  - **Example "2.4.20-13" is a stable kernel**
- **Red Hat has started using Major.Minor.Step.Release, as in 2.4.20-13.9 for Redhat 9.0 and 2.4.20-13.8 for Redhat 8.0, etc.**
- **Packages use a similar scheme (more later on this)**
  - **red-carpet-1.3.3-4.ximian.1.i386.rpm**
  - **gdb-5.2-2.i386.rpm**
  - **ethereal-0.9.4-0.7.3.0.i386.rpm**

What is Open Source?

- **Software protected by the GNU General Public License (GPL) or similar license schemes**
- **There are multiple versions of the GPL and other "open-source" license schemes**
- **From */usr/src/linux2.4/Documentation/COPYING*:**

  "*Also note that the only valid version of the GPL as far as the kernel is concerned is _this_ particular version of the license (ie v2, not v2.2 or v3.x or whatever), unless explicitly otherwise stated.*"

  -- Linus Torvalds

What is Open Source? (continued 1)

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it. For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software. […]

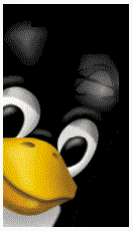
from: /usr/src/linux2.4/Documentation/COPYING © 1989, 1991 Free Software Foundation, Inc.

Getting Help and Software For Linux

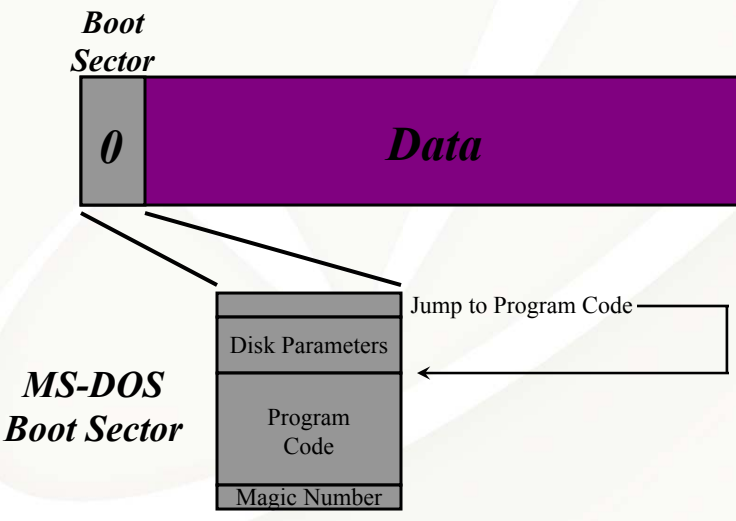
- **Open Source software is available from a *lot* of places. I tend to frequent:**
  - http://rpmfind.net
  - http://sourceforge.net
- **Documentation and HOWTOs can tend to be out of date. The Redhat site is good for Redhat specific documentation and has pointers to other sites. For example:**
  - http://tldp.org
- **There are lots of news groups and chat channels devoted to all aspects of Linux and GNU applications**
- **There are formal support channels available for Linux: HP, Redhat, etc.**

Disk Partitioning Tutorial

- Floppy Disk Format
- The Master Boot Record (MBR)
- Partition Layout for a Hard Disks
- Device files for Primary Partitions
- Devices for Extended Partitions

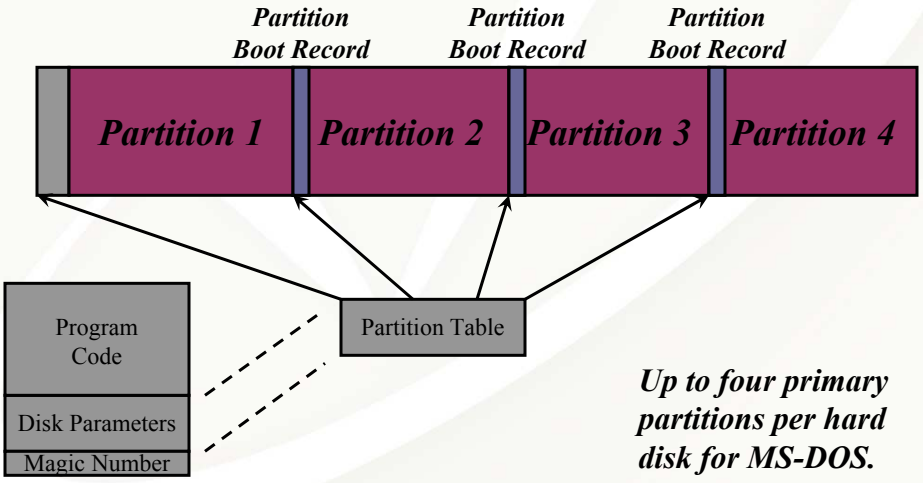


Data Layout for a Floppy Diskette



Data Layout for a Hard Disk (IDE or SCSI)

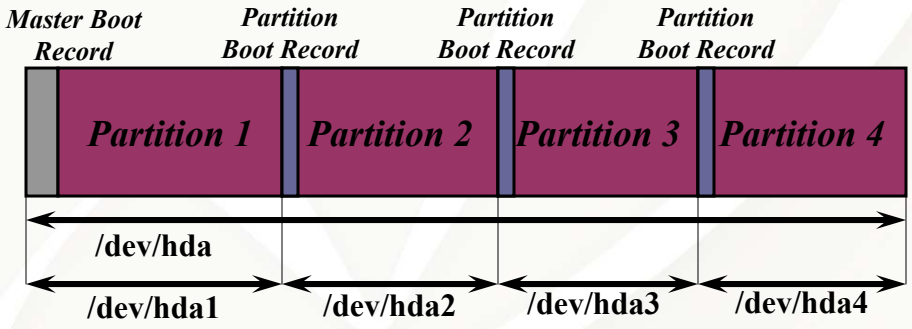
(without extended partitions)



Up to four primary partitions per hard disk for MS-DOS.

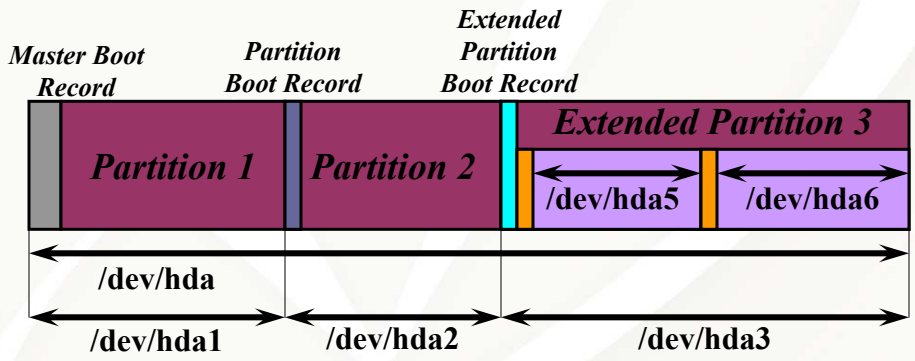
Data Layout and Devices for an IDE Hard Disk

(without extended partitions)



Note that each partition is accessible as if it were an independent device with its own boot sector. Linux creates devices for this configuration as shown.

Data Layout for an IDE Hard Disk
(with extended partitions)



If more than four partitions are needed, one primary partition is divided into an extended partition containing several logical partitions. Note that the partition tables of the logical partitions are not accessible as the first block of some device.

Partition Table from a Real System Using "fdisk"

```
# fdisk /dev/hda

The number of cylinders for this disk is set to 1229.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
(e.g., DOS FDISK, OS/2 FDISK)

Command (m for help): p

Disk /dev/hda: 255 heads, 63 sectors, 1229 cylinders
Units = cylinders of 16065 * 512 bytes

Device Boot    Start       End    Blocks   Id  System
/dev/hda1   *      1        13    104391   83  Linux
/dev/hda2         14        78    522112+  82  Linux swap
/dev/hda3         79      1229   9245407+  83  Linux

Command (m for help): q
```

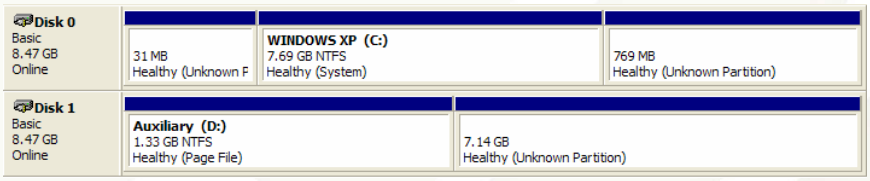
Partition from a Real System Using “parted"

```
# parted /dev/had

GNU Parted 1.4.24
Copyright (C) 1998, 1999, 2000, 2001, 2002 Free Software Foundation, Inc.
This program is free software, covered by the GNU General Public License.

This program is distributed in the hope that it will be useful, but WITHOUT ANY
WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A
PARTICULAR PURPOSE.  See the GNU General Public License for more details.

Using /dev/hda
Information: The operating system thinks the geometry on /dev/hda is
1229/255/63.  Therefore, cylinder 1024 ends at 8032.499M.

(parted) p

Disk geometry for /dev/hda: 0.000-9641.953 megabytes
Disk label type: msdos

Minor   Start     End     Type      Filesystem  Flags
 1        0.031   101.975  primary   ext3        boot
 2      101.975   611.850  primary   linux-swap
 3      611.851  9640.568  primary   ext3

(parted) q
```

Lessons Learned About Multi-boot Systems

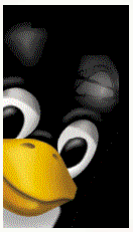
- If you have an NTFS file system, then it must be in the first partition on the disk (example order: Windows NT/XP, Windows 98, /boot, FAT16)
- All bootable partitions must be within 1023 cylinders of the start of the drive
- Windows will only boot if within the first 4 GB of the disk
- There can be only one active/visible partition unless you use a boot manager like PowerQuest BootMagic
- Bootable partitions must be on IDE disk 0 or SCSI disk 0 (!)
- Only four primary partitions per disk
- Windows NT does not know how to read FAT32 file systems, Windows XP does
- If your first drive is SCSI or RAID, Linux may not have the drivers built into the kernel, so you MUST configure an initial RAM disk image (initrd) containing the dynamically loadable kernel modules (more later)

A Dual-boot System, Linux and Windows XP



- 31 MB Partition is /boot, inside the 1023 cylinder limit
- /boot is the first physical partition, but the *second* partition table entry (I did this with "Partition Magic" from PowerQuest)
- The third partition on disk 0 is Linux swap
- The second partition on disk 1 is Linux /
- Windows XP lives in the second partition on disk 0 and uses the first partition on disk 1 as paging
- I installed Windows XP first, then "inserted" Linux and used the GRUB boot manager (more on GRUB later)

Installing Redhat Linux



- Booting the Install Media
- Partitioning the Disk
  - Normal
  - RAID
  - LVM
- Installing the Boot Loader
- Configuring the Network
- Configuring the Firewall
- Options
- Picking Packages
- Installing Packages
- Video Configuration
- Boot Disk Creation
- X-Windows Configuration
- Reboot

System Installation Comparison

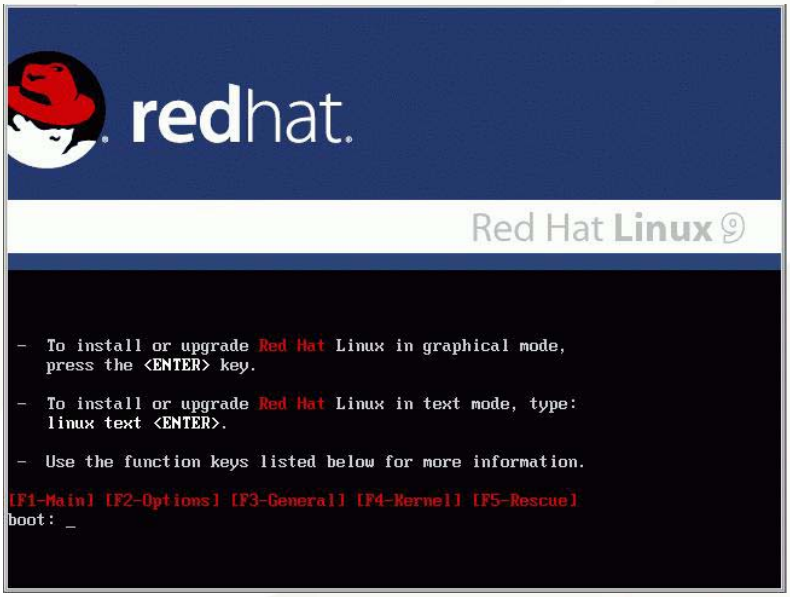
- **HP-UX**
  - CD-ROM or DVD
  - Ignite-UX (network)
- **Software Distribution Utilities (SDU)**
- **Graphical or text-based installation tool**
- **Three main phases:**
  - Configure disk layout
  - Install system filesets
  - Configure subsystems

- **Redhat Linux**
  - CD-ROM, DVD, or floppy
  - Kickstart (network)
  - Others (systemimager)
- **Redhat Package Manager (RPM)**
- **Graphical or text-based (VGA) installation tool**
- **Three main phases:**
  - Partition the disk
  - Install system packages
  - Configure subsystems

Installation Notes

- The Redhat Linux installation tool is called “anaconda”
- Because most graphics cards (all that will work with x86 hardware) have a VGA mode, the installer can work in VGA (80x24 color) mode
- The installer tries to start an X-server for a graphical user interface during install if you don’t select a text-based installation
- You can select which mode the install takes place in at the installation CD prompt with “linux text”
- Obviously, if you have only a serial port, then you are stuck with VGA mode
- If you can, install on a machine with a graphics-enabled card, then use an imaging technique like “systemimager” to “clone” the system image to a non-graphics environment (more on this later)
- Our hardware for this seminar supports bit-mapped graphics, so we will not concentrate on the VGA installation
- Most of the configuration that you do graphically is available after the system is installed in either VGA or X-windows mode.
- There are a number of tools named redhat-config-<something>” that do VGA, X-windows, or command-line configuration of the system. For example, “redhat-config-network” will set up the network parameters for your system

Initial Installation Boot Menu (VGA mode)

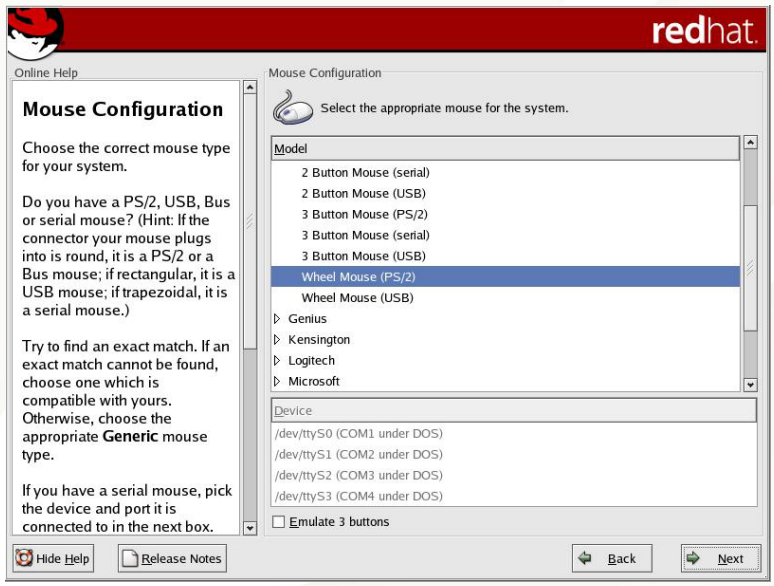


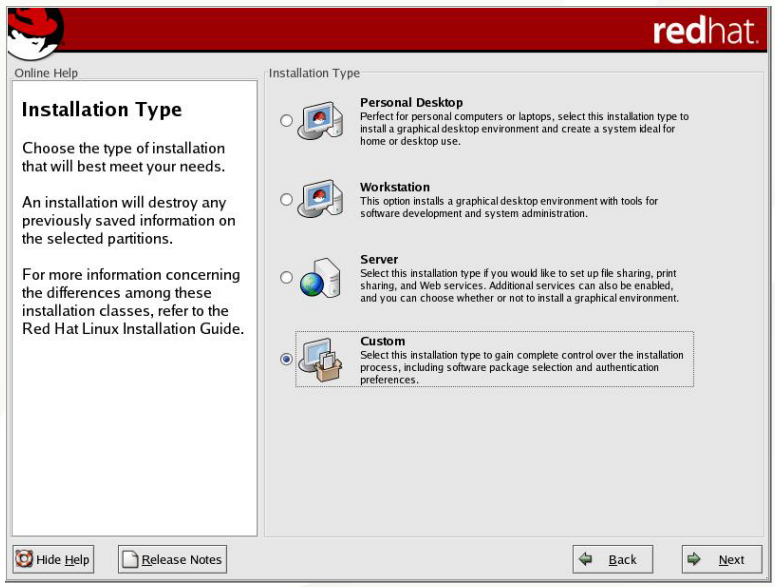
Installation Welcome

Installation Language Selection

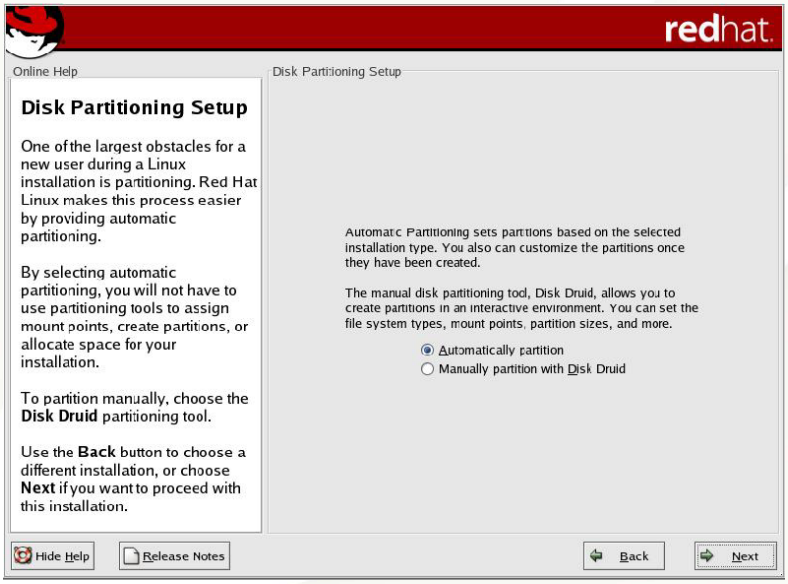
Configure Keyboard

Configure Mouse

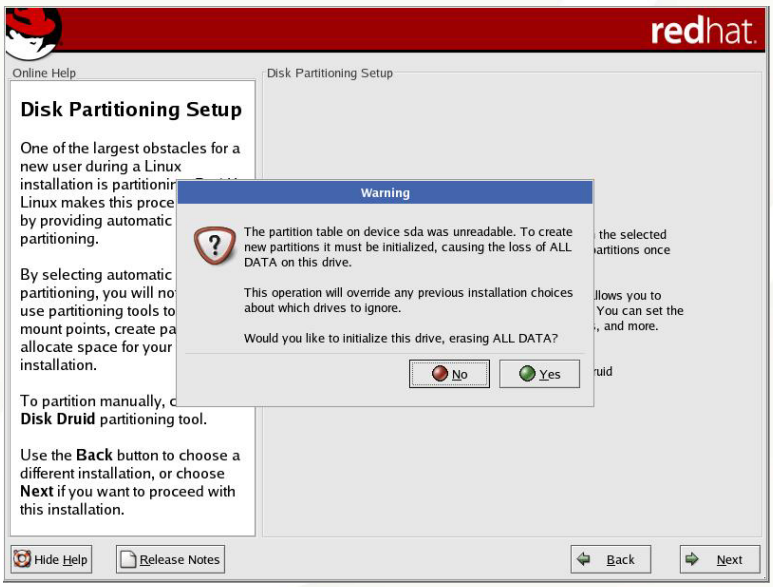




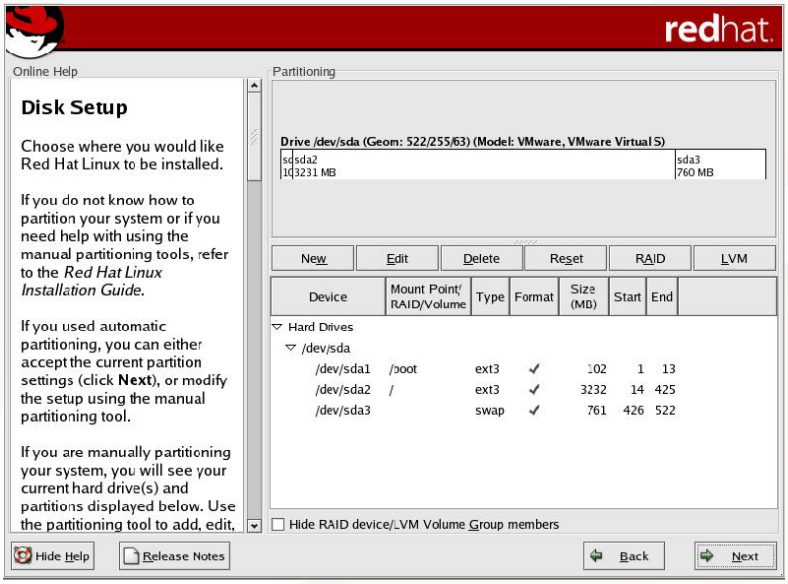
Automatic Partitioning



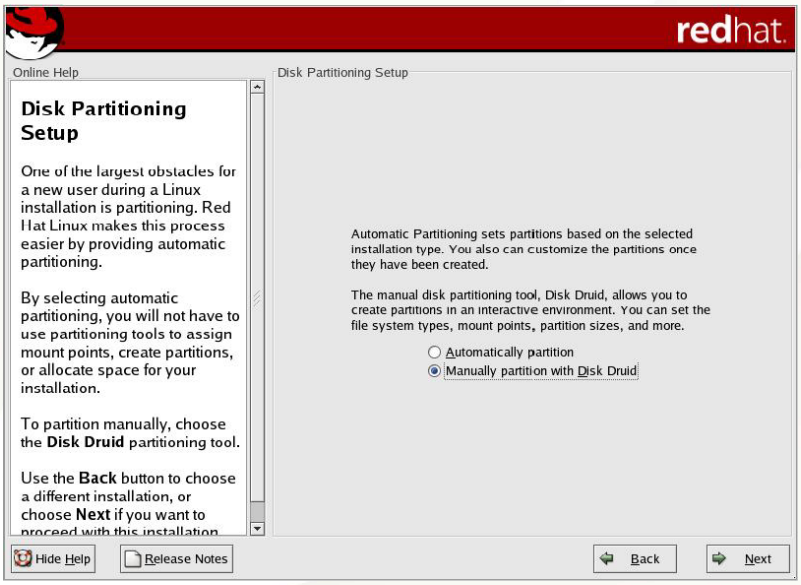
Automatic Partitioning – Format Partition Table



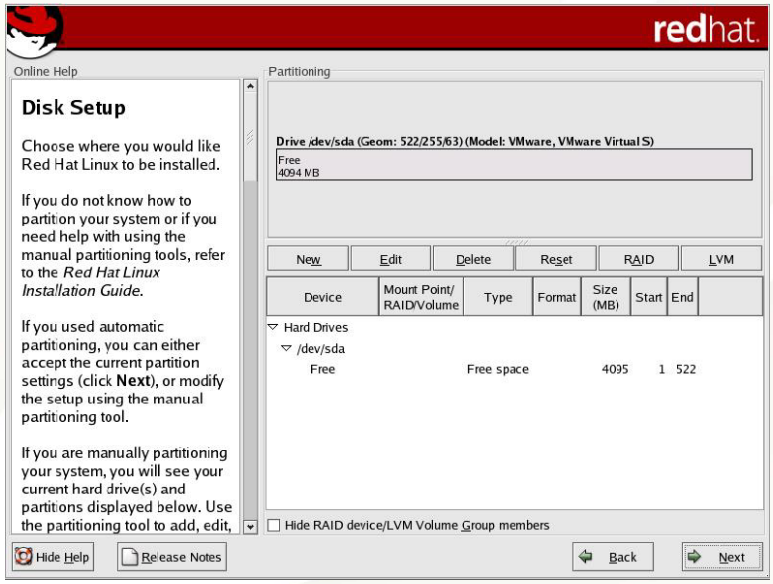
Automatic Partitioning – Resulting Disk Layout



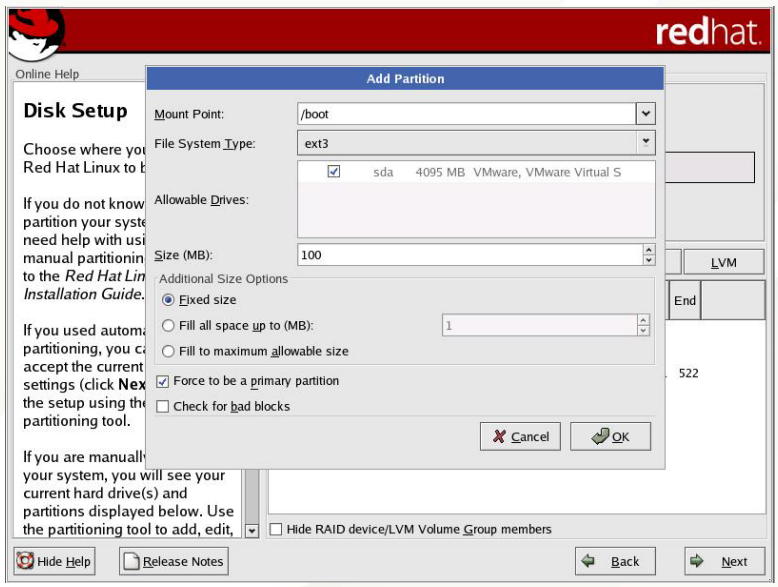
Manual Partitioning with Disk Druid



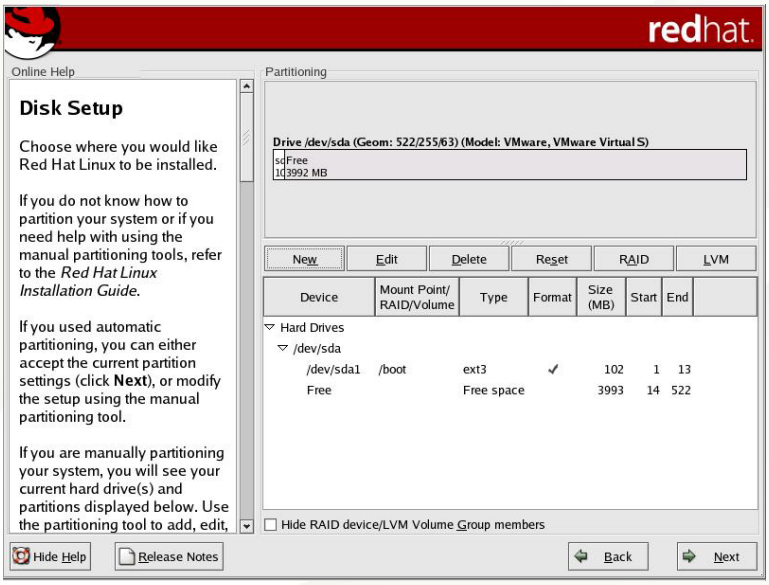
Manual Partitioning – Free Disk Space



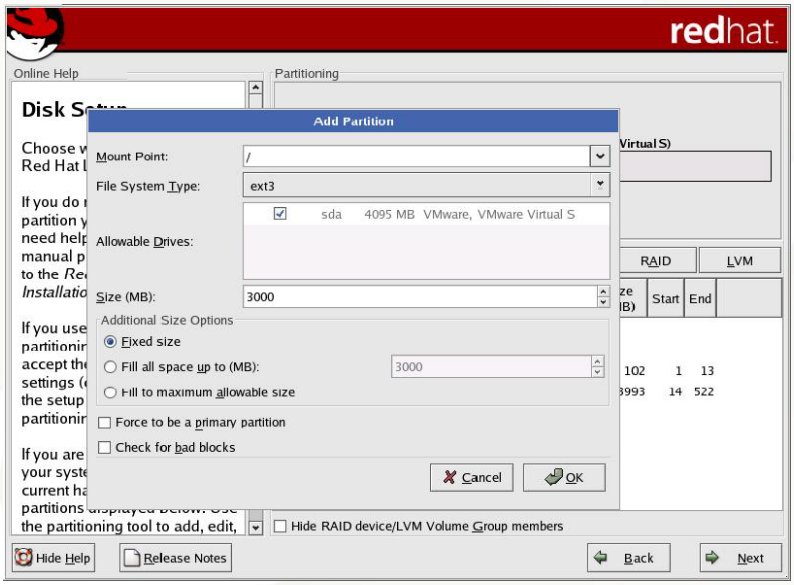
Manual Partitioning – Adding /boot



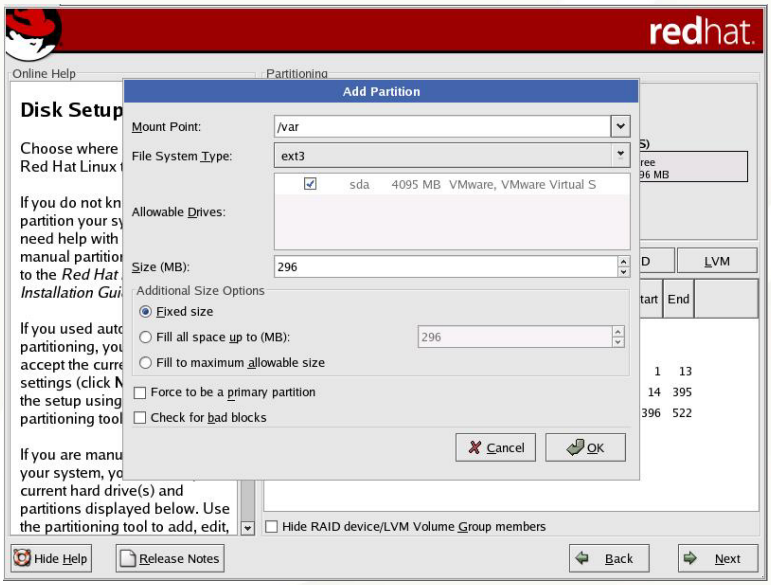
Manual Partitioning - /boot Added



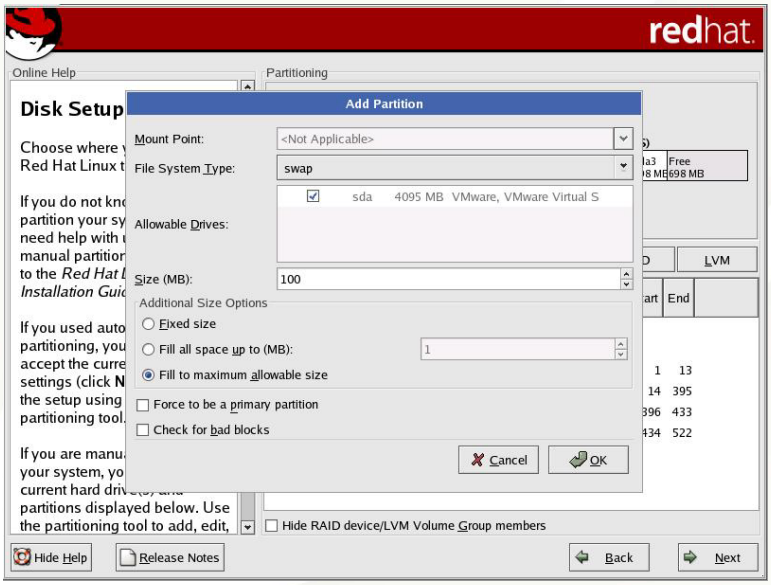
Manual Partitioning – Adding Root Partition



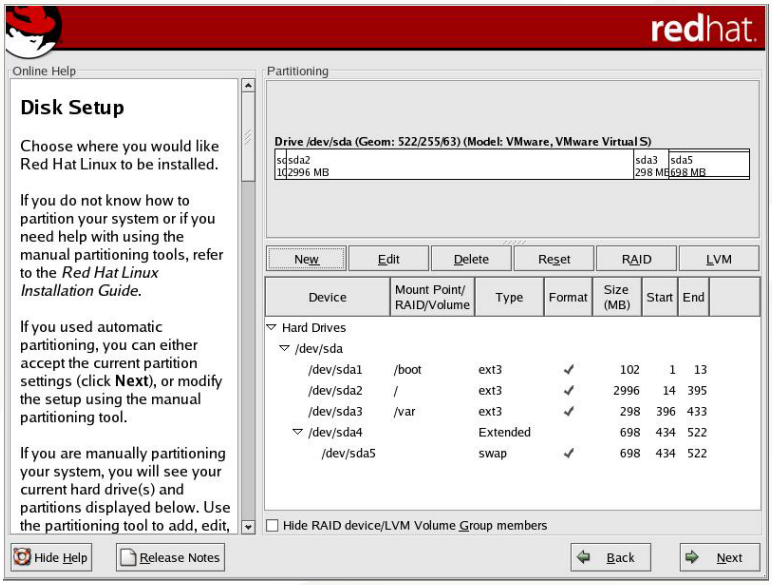
Manual Partitioning – Adding A Separate /var



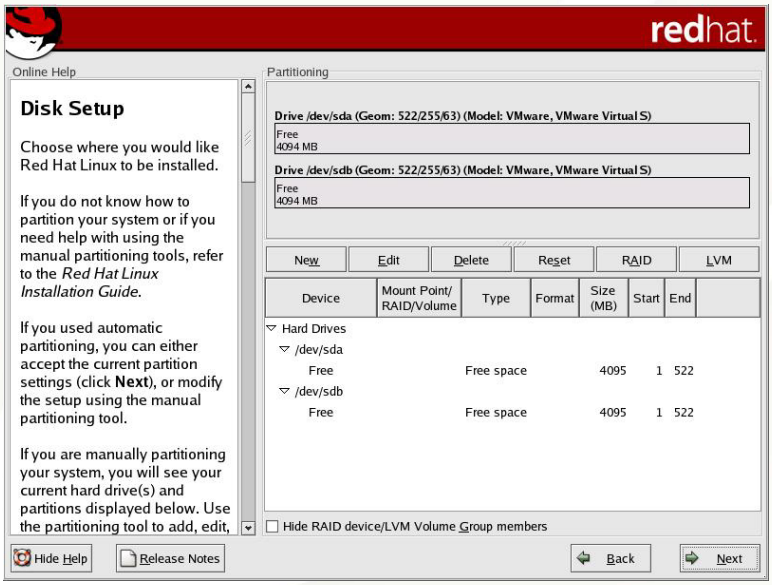
Manual Partitioning – Adding Swap



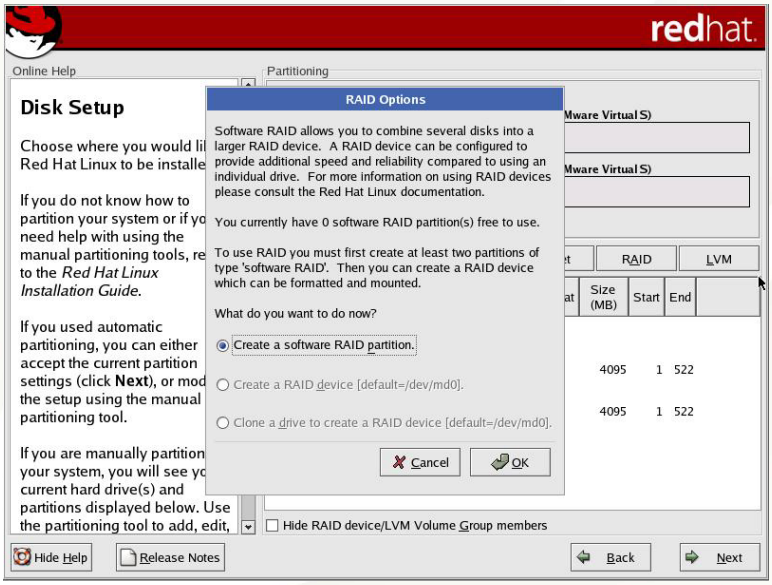
Manual Partitioning – Final Configuration



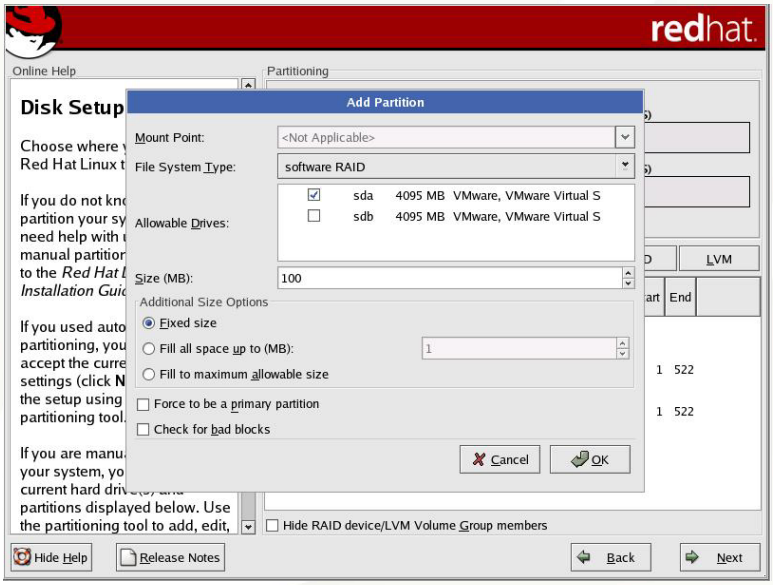
Manual RAID Partitioning – RAID 1 (mirroring) Example



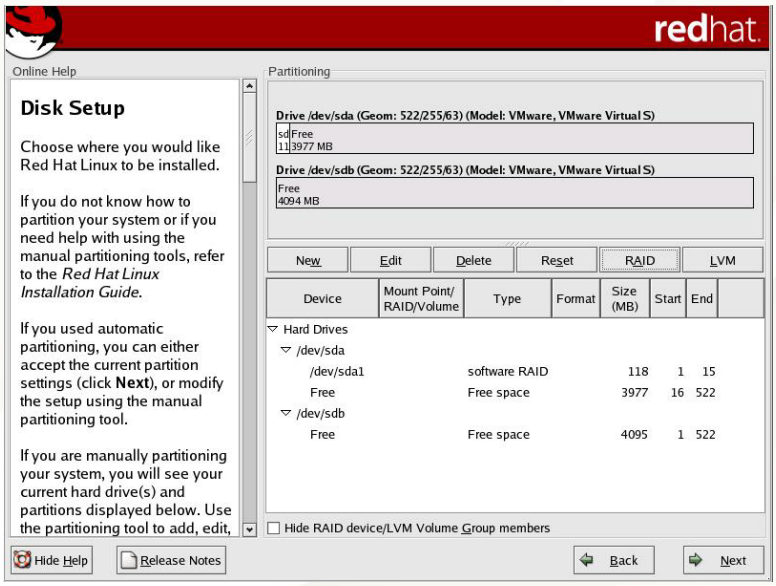
Manual RAID Partitioning – Adding A RAID Partition



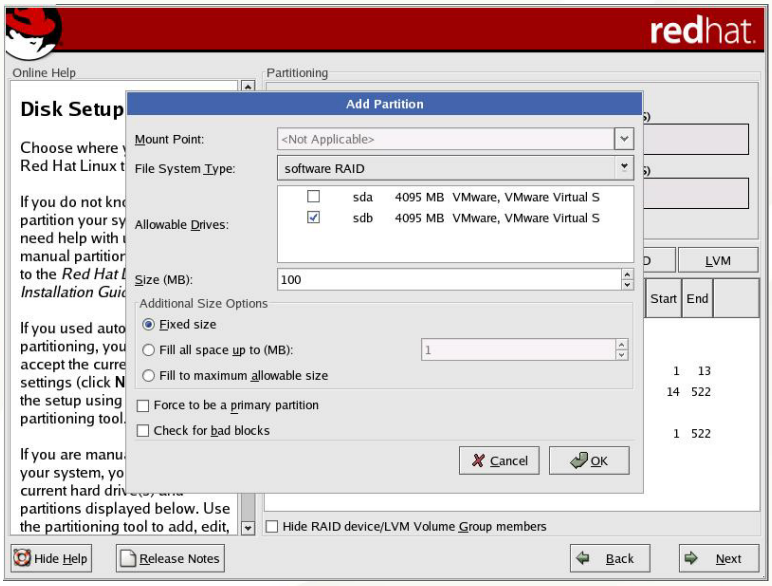
Manual RAID Partitioning – Selecting Disk



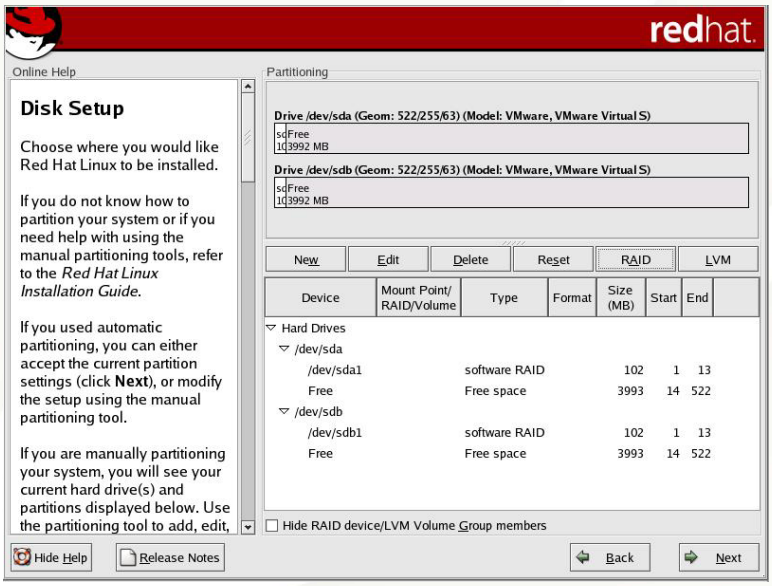
Manual RAID Partitioning – First Partition Added



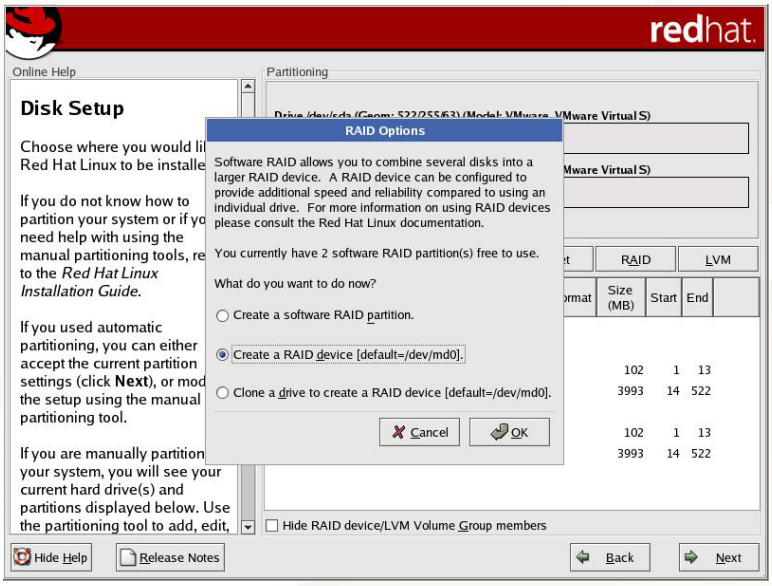
Manual RAID Partitioning – Adding Second Partition



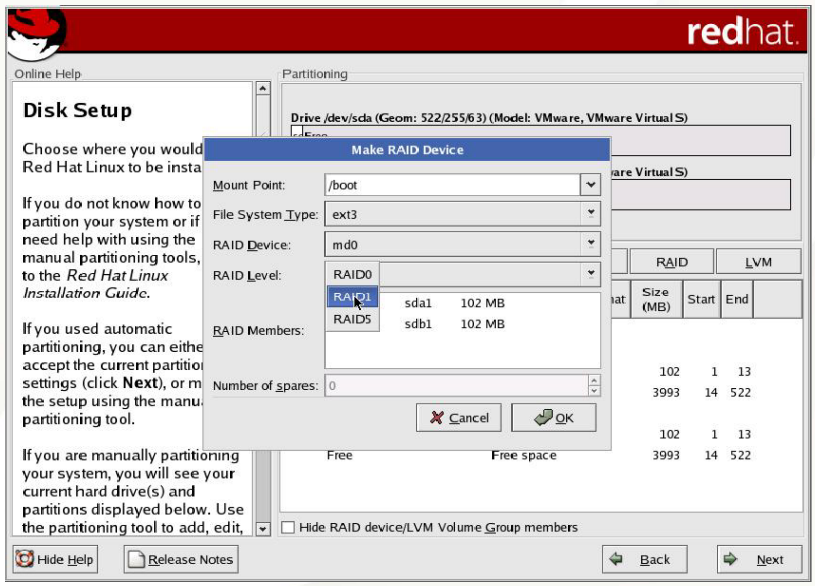
Manual RAID Partitioning – Second Partition Complete



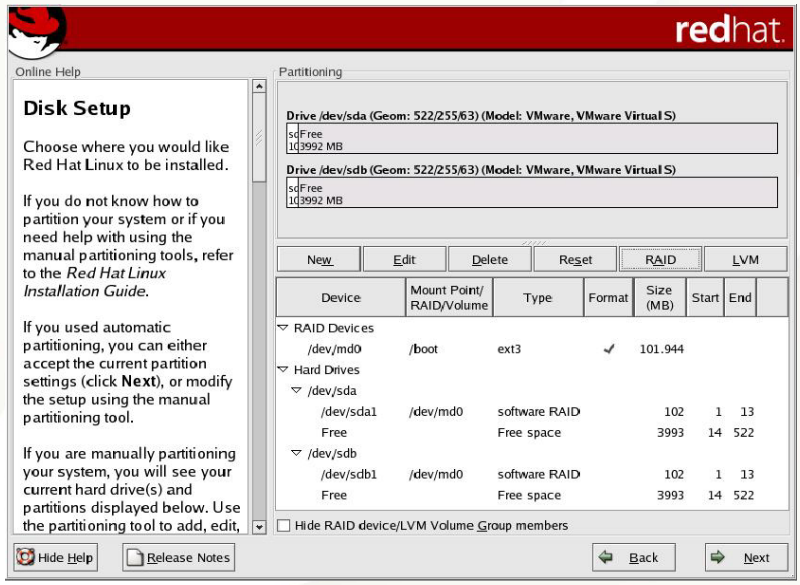
Manual RAID Partitioning – Create RAID Device



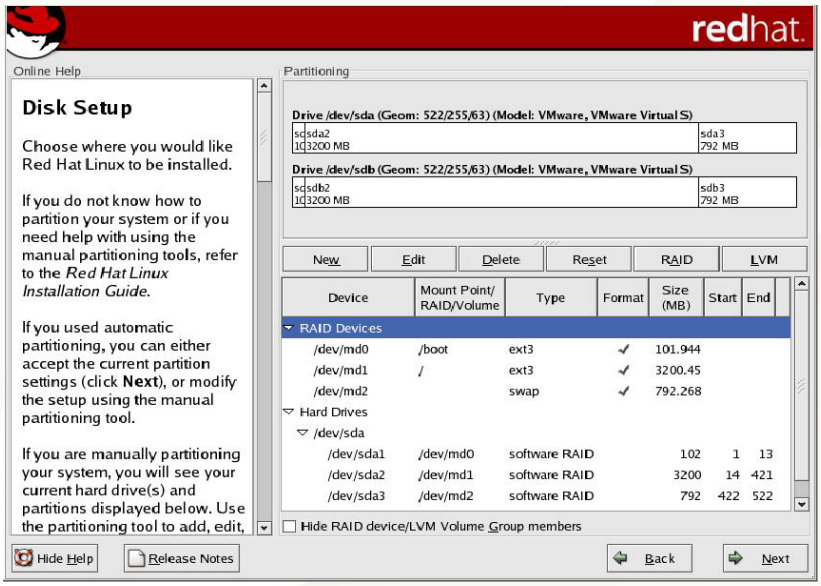
Manual RAID Partitioning – Create RAID /boot



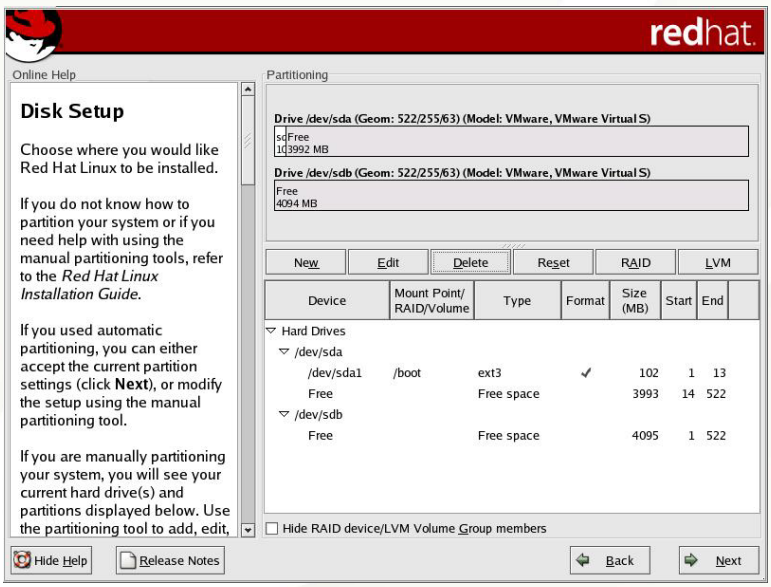
Manual RAID Partitioning – First RAID Created



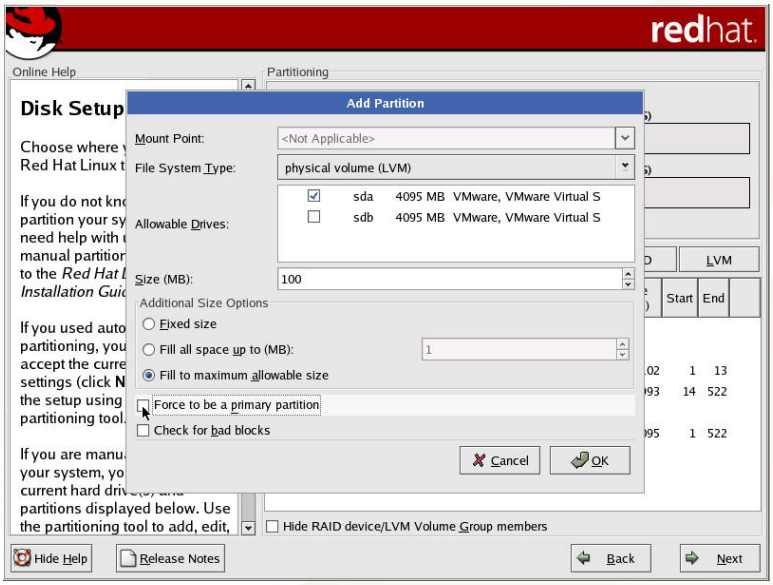
Manual RAID Partitioning – All RAID Devices Complete



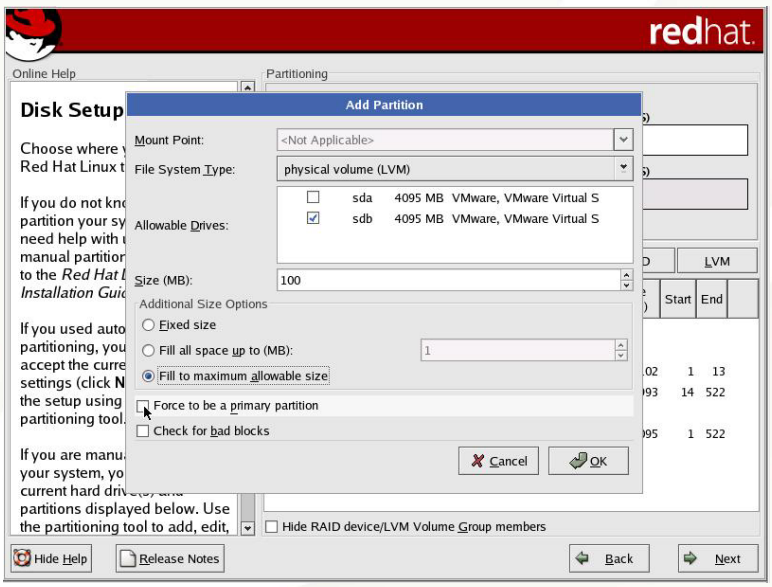
Manual LVM Partitioning – Adding /boot



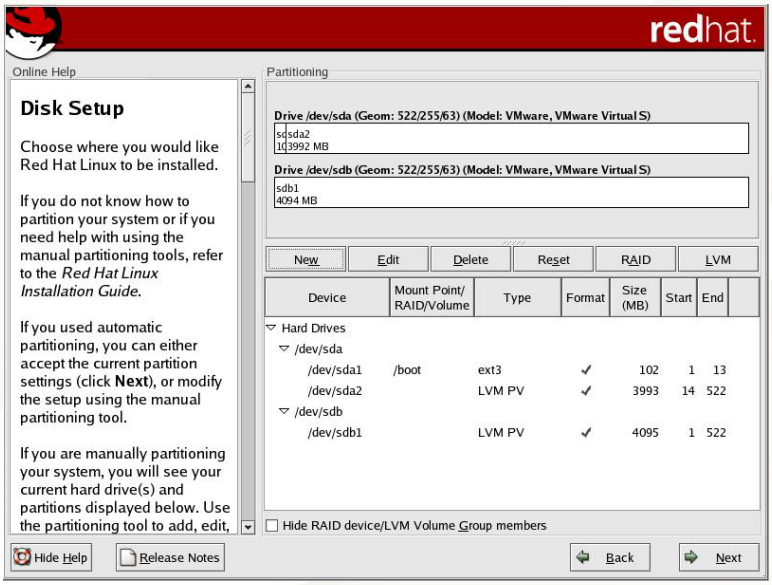
Manual LVM Partitioning – First Physical Volume



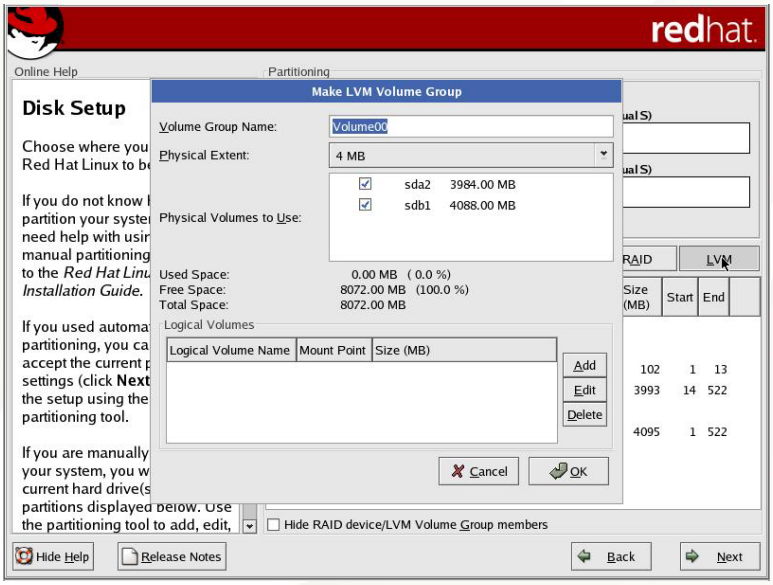
Manual LVM Partitioning – Second Physical Volume



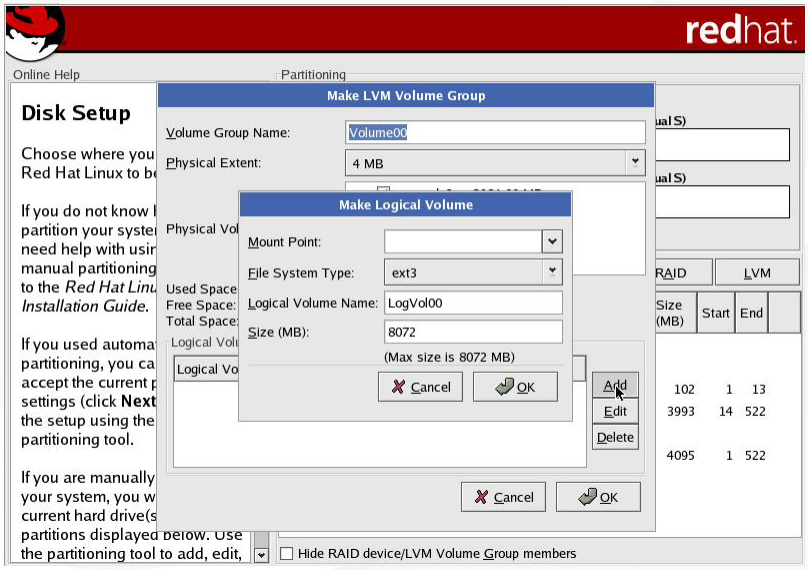
Manual LVM Partitioning – Physical Layout



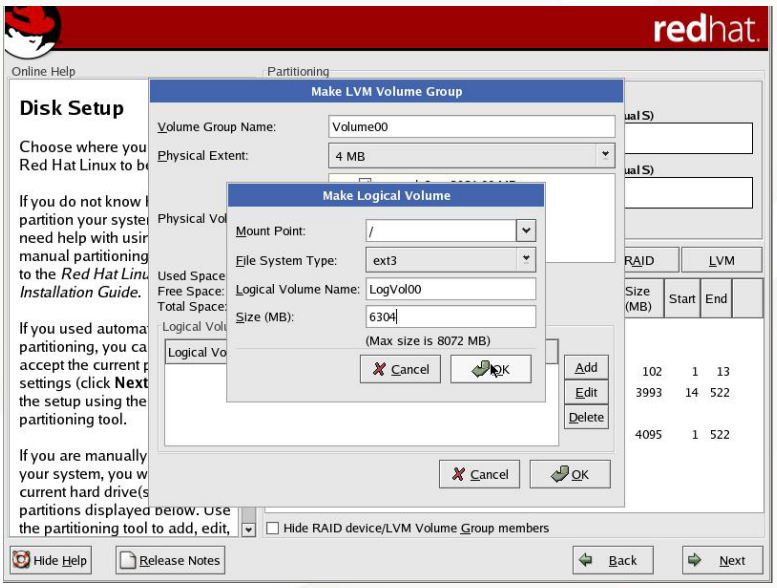
Manual LVM Partitioning – Volume Group Information



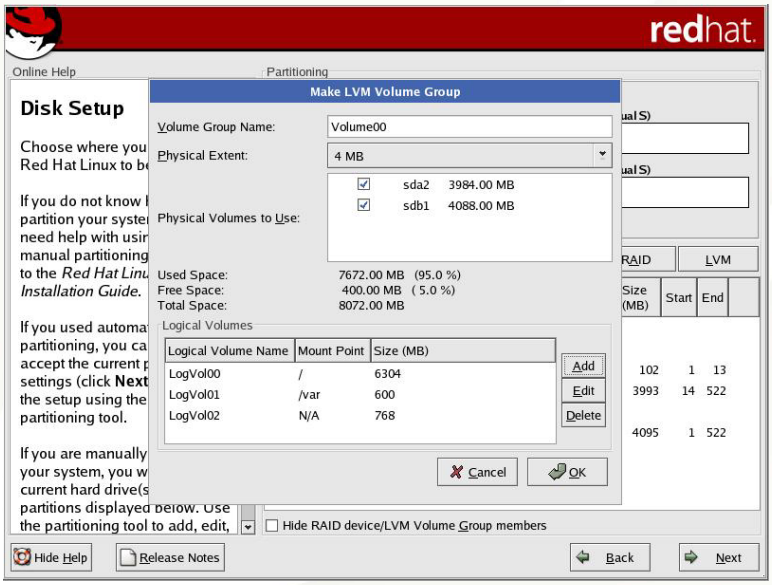
Manual LVM Partitioning – Make Logical Volume



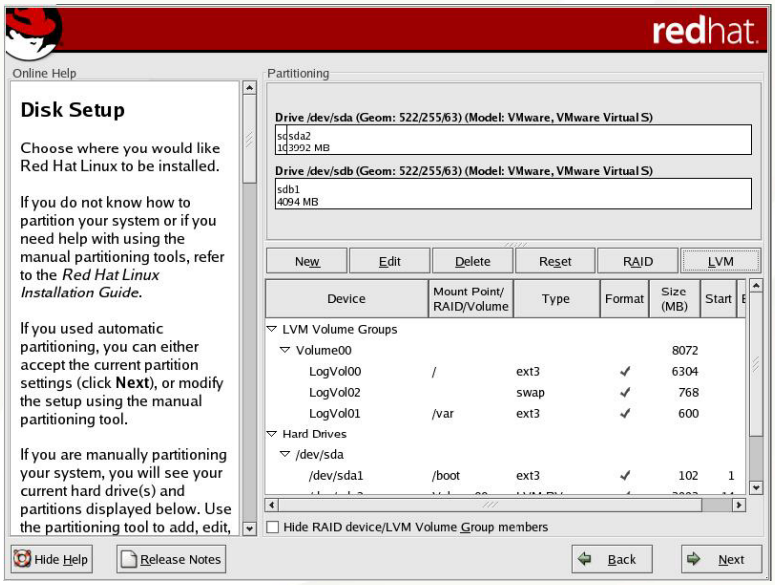
Manual LVM Partitioning – Making Root Volume



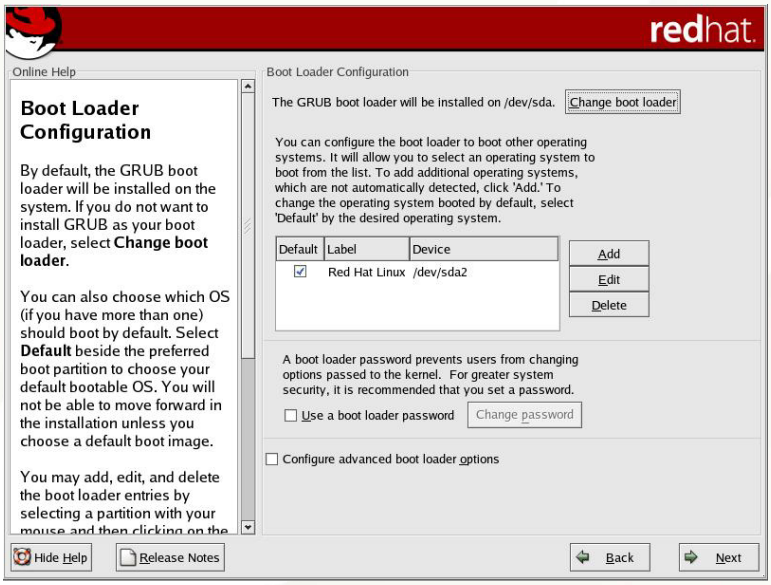
Manual LVM Partitioning – All Logical Volumes Created



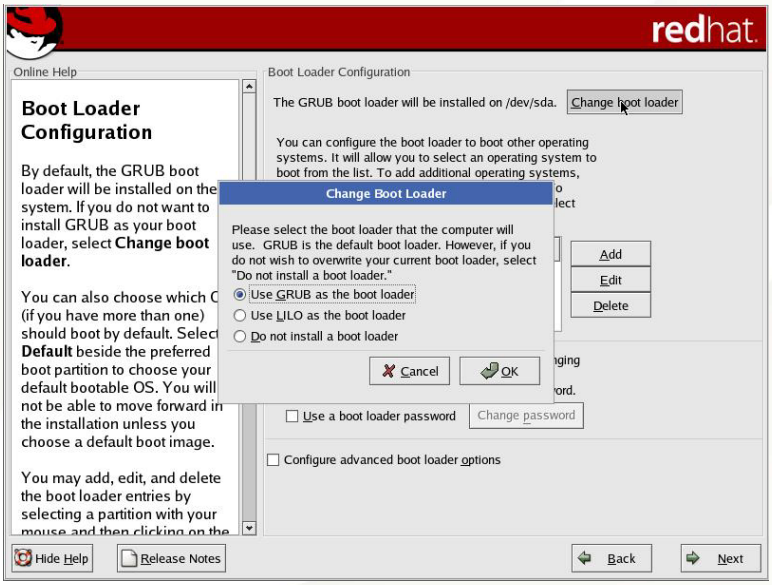
Manual LVM Partitioning – Physical Layout



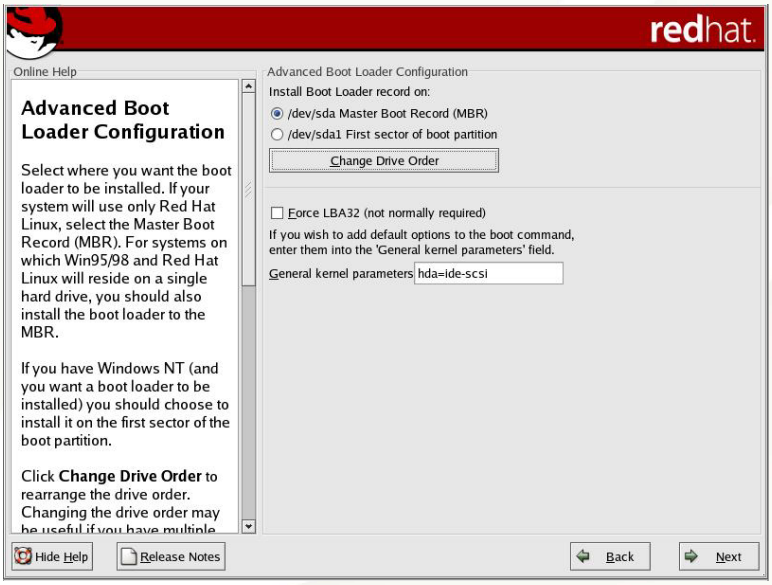
Configuring the Boot Loader – GRUB or LILO



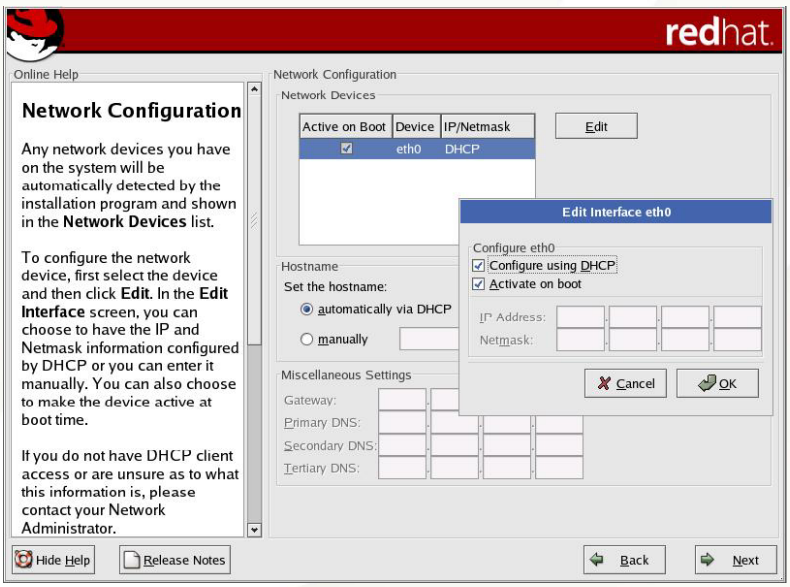
Selecting the Boot Loader



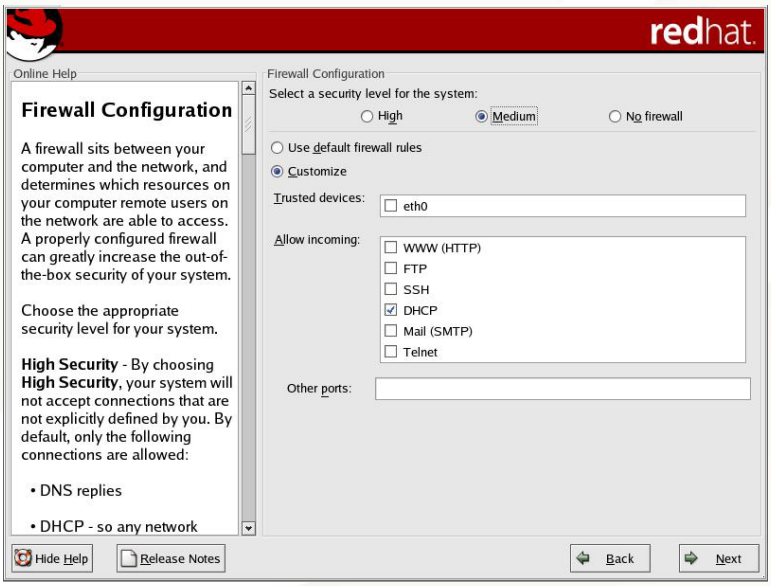
Advance Boot Loader Configuration



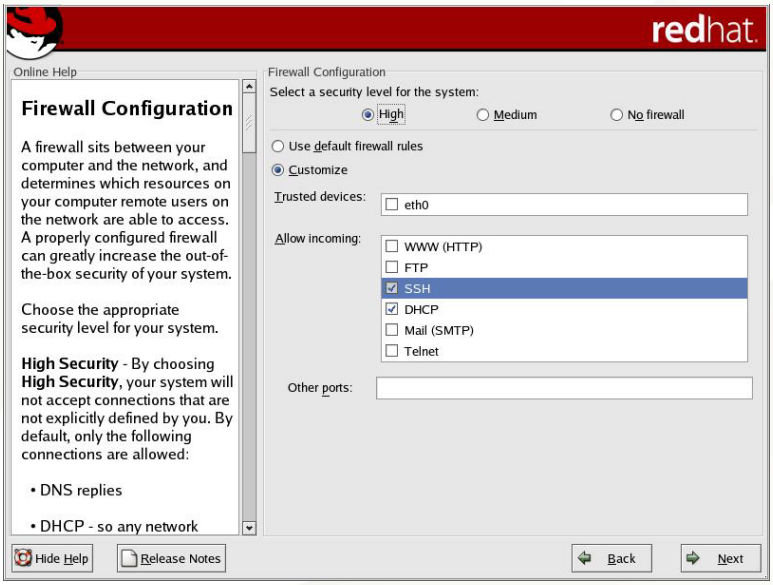
Configuring the Network Interfaces



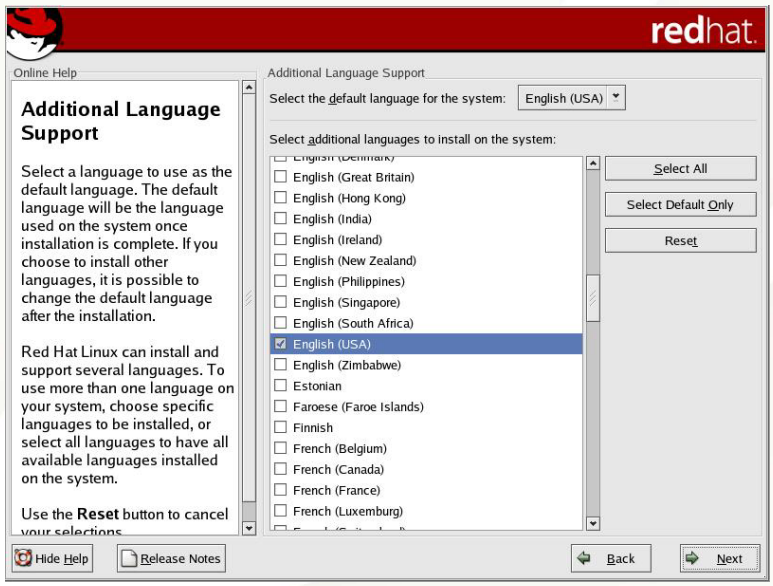
Firewall Default Settings



Firewall Example Settings



Adding System Language Support

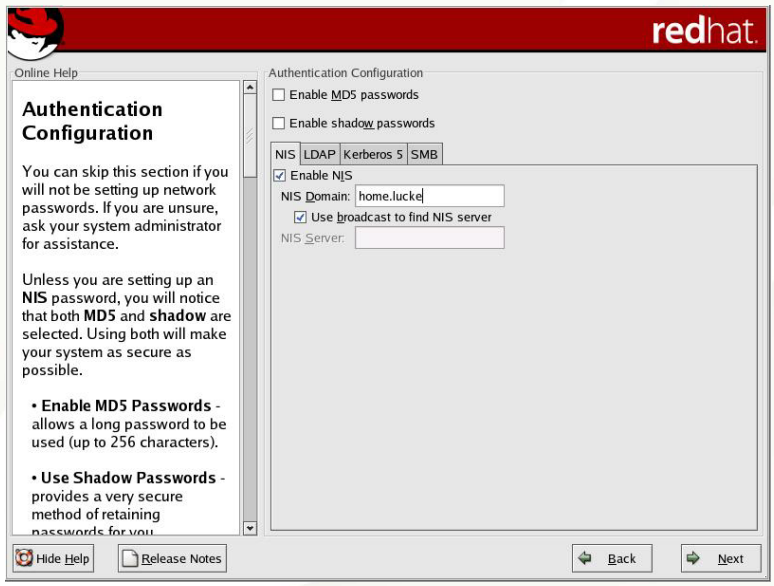


Selecting the Time Zone

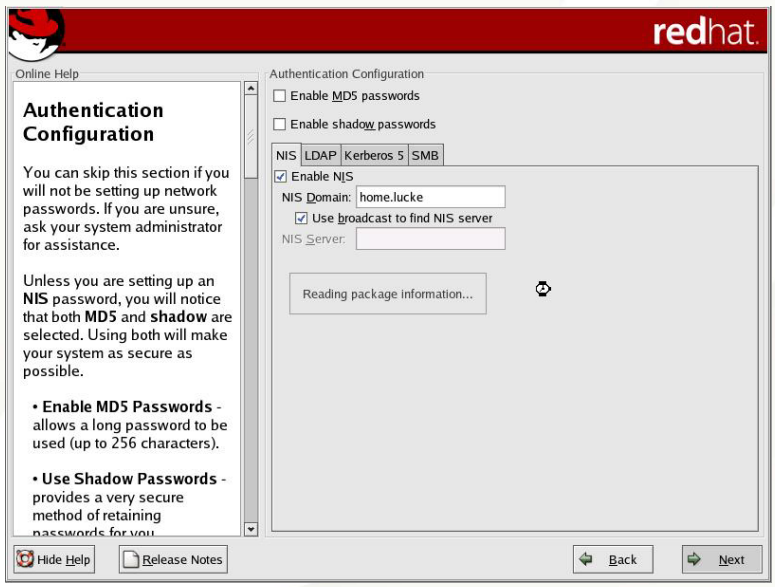
Setting the Root Password

Choosing System Authorization Method

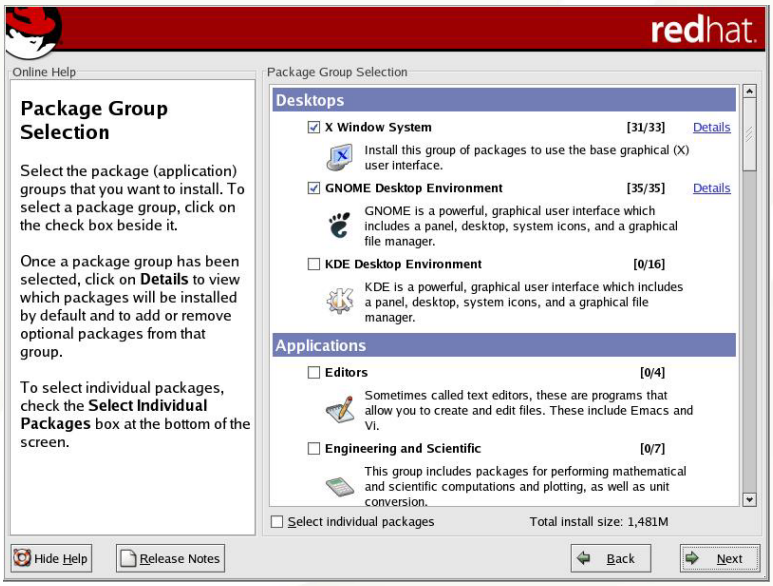
Using NIS as the Authentication Mechanism



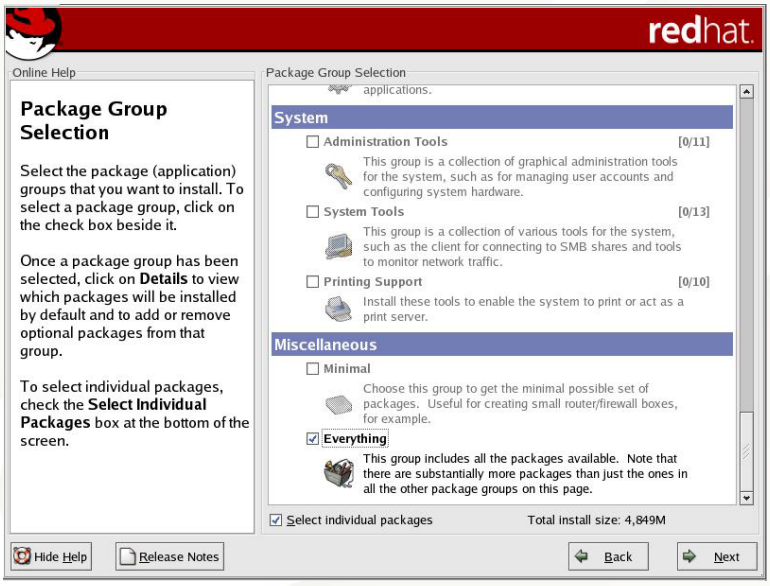
On to Loading Packages Now …



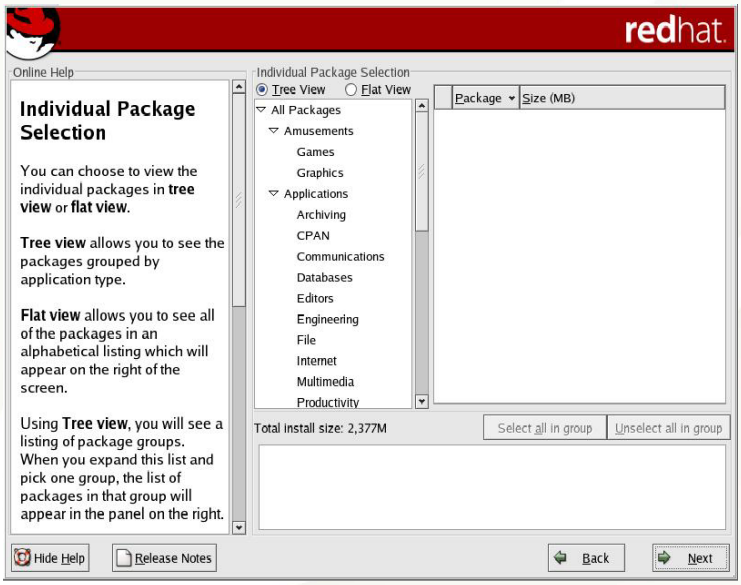
Default Package Group Selection



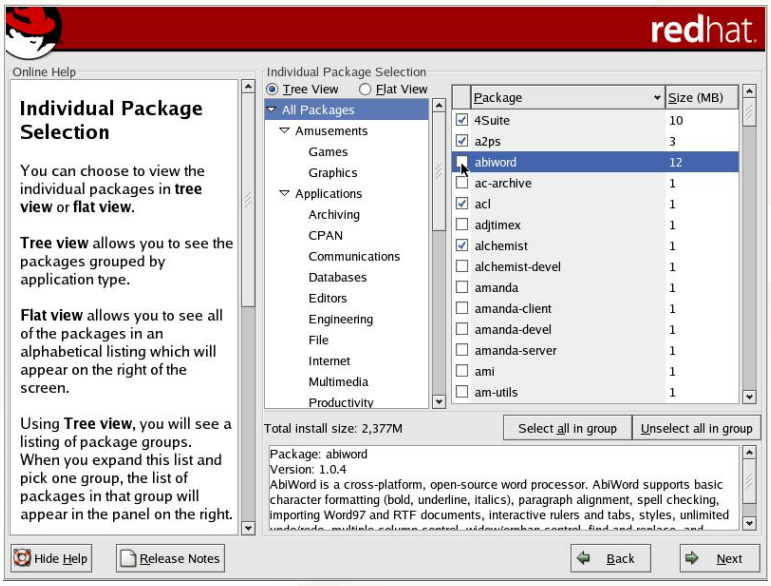
And Down at the Bottom of the List …



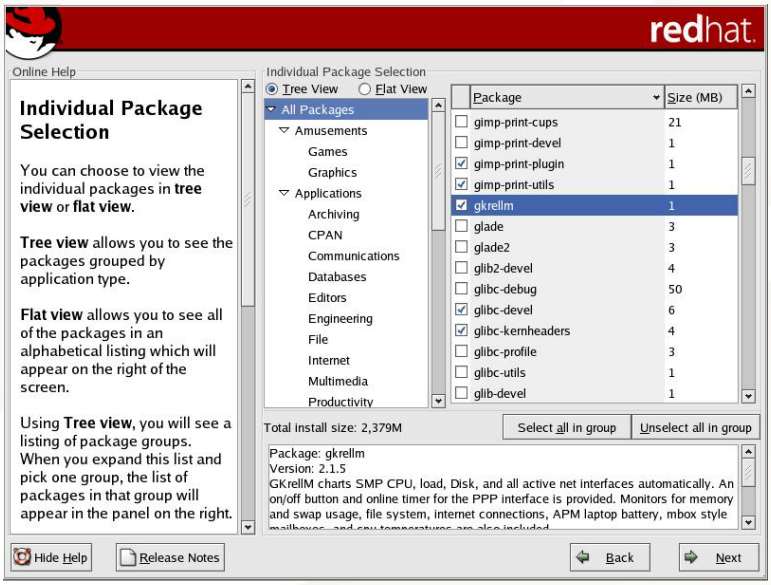
Let's Pick Individual Packages to Add



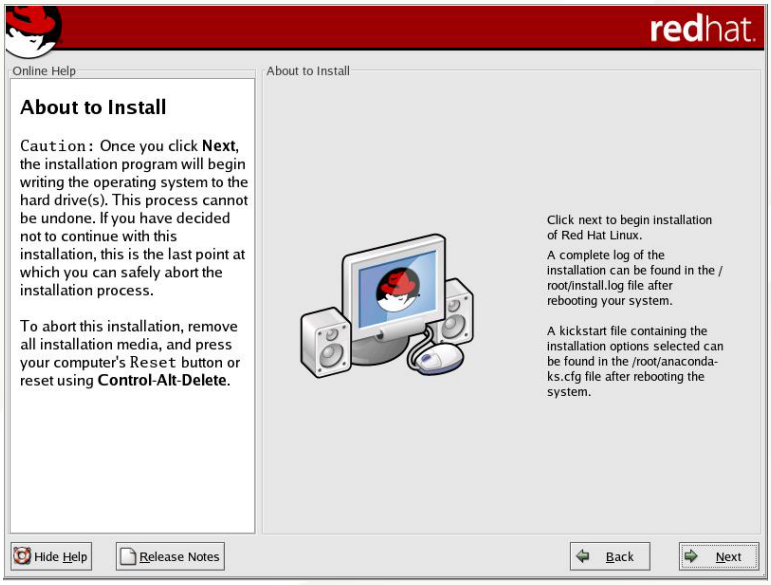
So Many Packages, So Little Time …



Adding a Package to the List

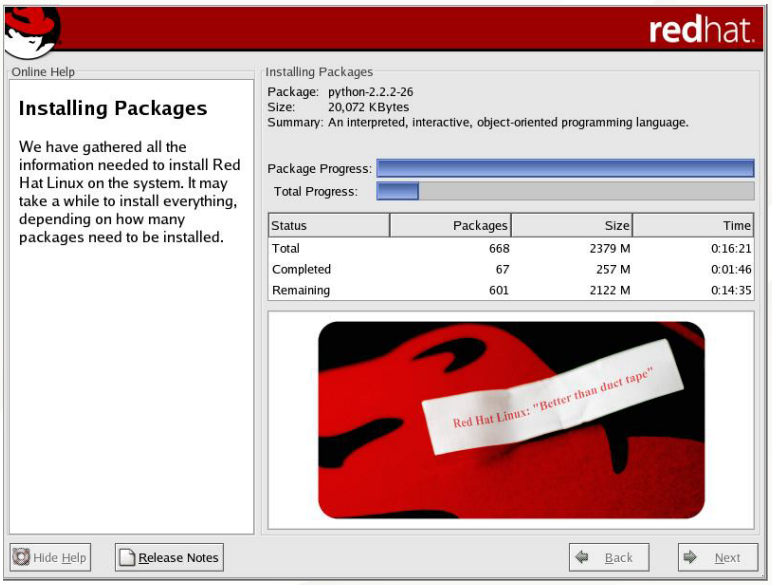


Ready to Install?



File System Formatting

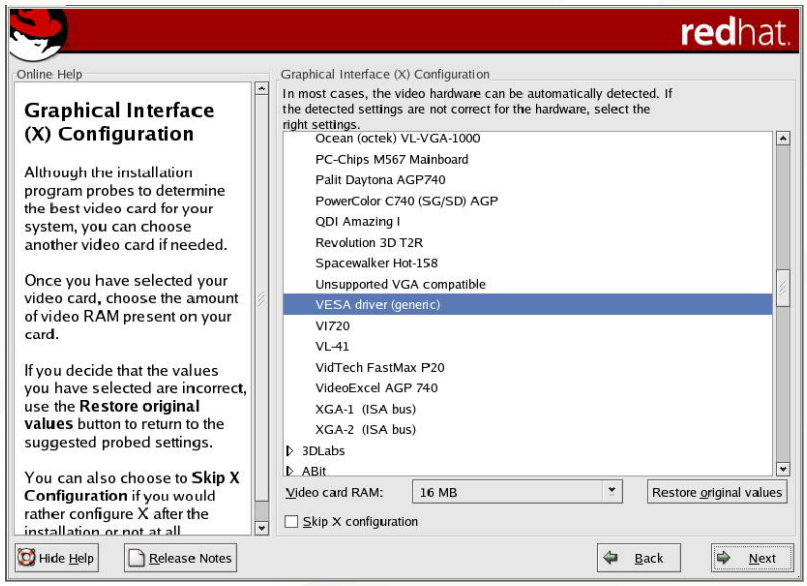
Installing: Don't Believe the Estimated Time …



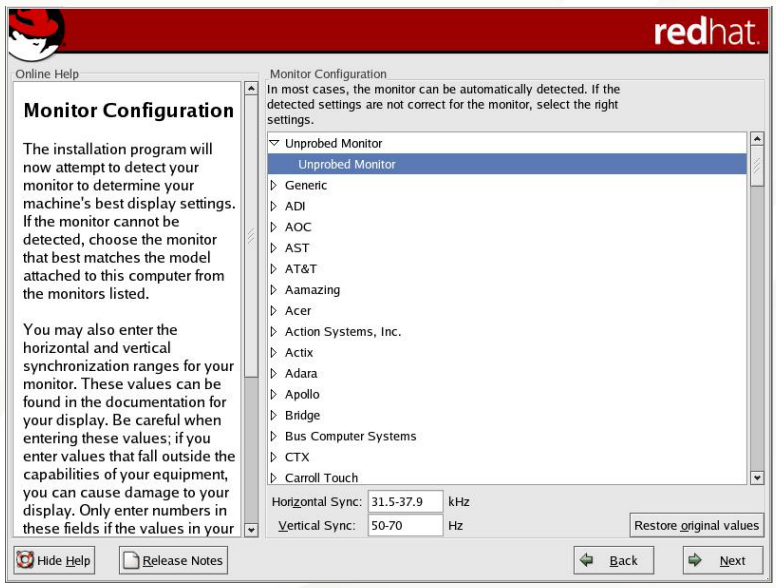
Creating Emergency Boot Disk

Always Create a Boot Disk, If Possible

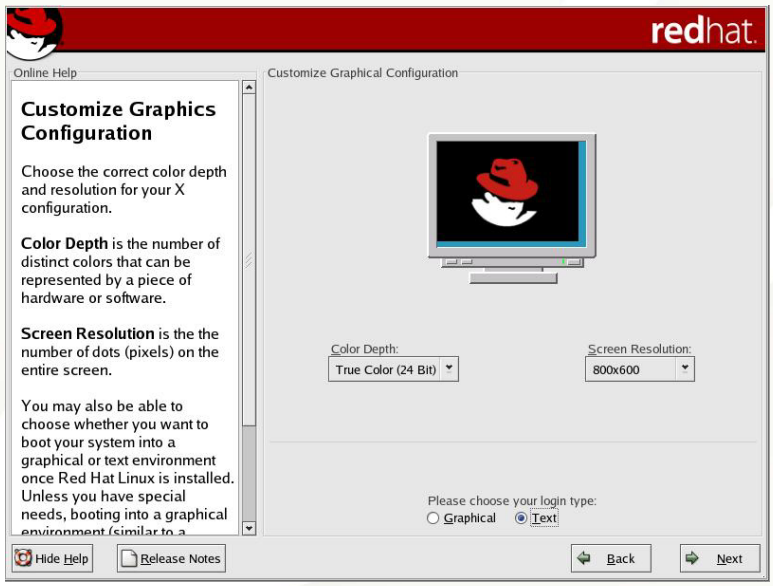
Setting Video Hardware Information



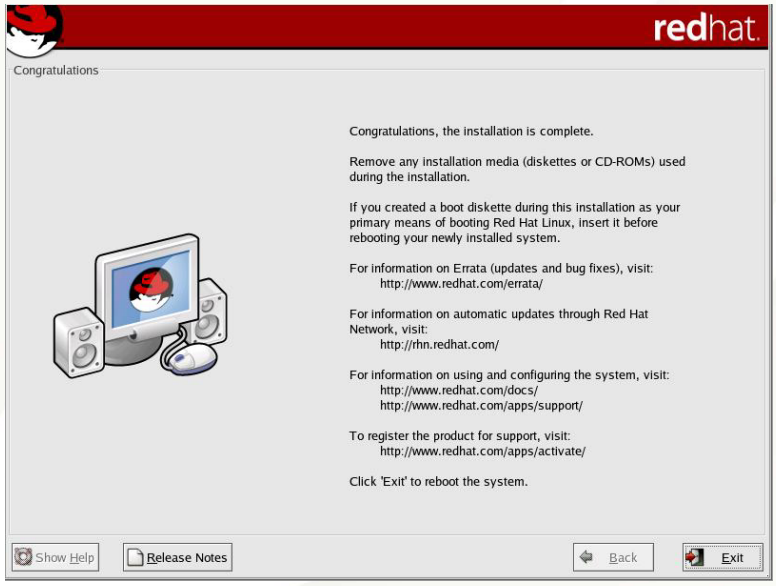
Choosing Your Monitor



Setting Login Type, Color Depth, and Screen Resolution



Whew! You're Finished – Well, Almost …



Lessons Learned About System Installation

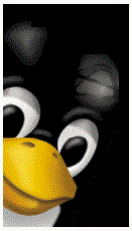
- Always make a boot disk!
- GRUB is a good boot manager and replaces LILO …
- GRUB understands file systems structure and is not dependent on offset like LILO (don't have to re-run it every change you make)
- Make a GRUB boot/installation disk, too!
- Always make a boot disk!
- You can place the /boot partition into a small partition on the first IDE/SCSI drive and still mount the Linux root partition from elsewhere
- Note that SCSI disk 0 has the LOWEST priority in the SCSI chain, so the PC BIOS has things a little backwards …
- Oh, and -- ALWAYS MAKE A BOOT DISK!

Lab #1: Installing Redhat Linux from CD-ROM

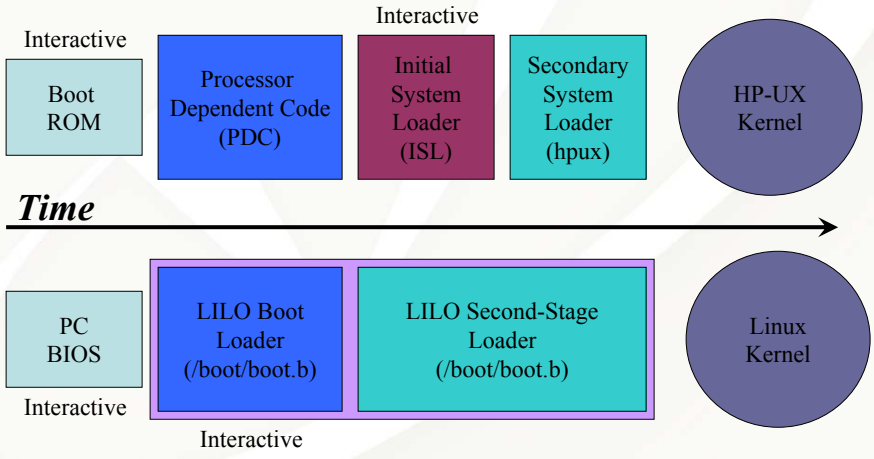
See Lab #1 Handout for details

Linux Boot Loaders

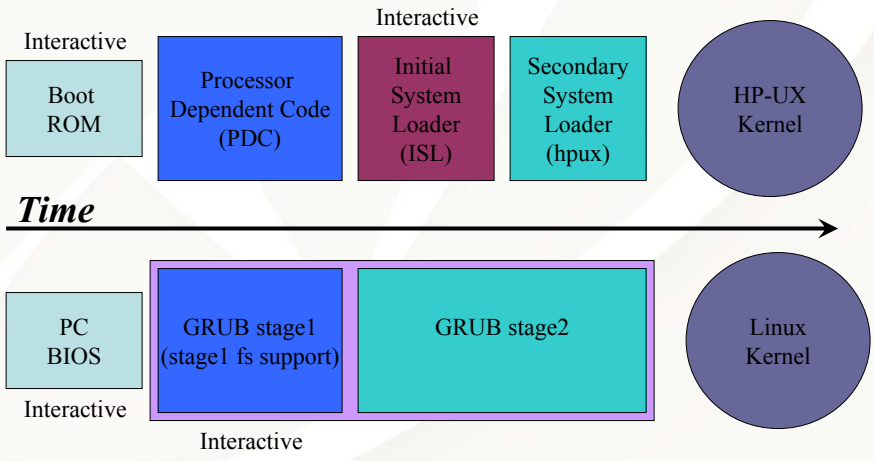
- LILO
- GRUB



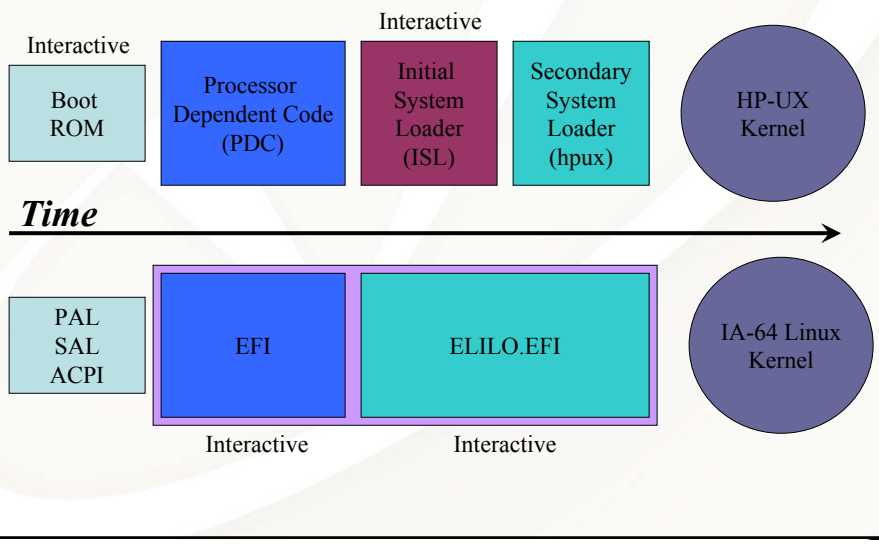
Comparing LILO Boot Sequence to HP-UX



Comparing GRUB Boot Sequence to HP-UX



Comparing ELILO.EFI Boot Sequence to HP-UX



Linux LILO Tips

- LILO is not the only way to boot a Linux system (thank goodness)
  a) Other Linux boot managers (GRUB, bootactv, loadlin, etc.)
  b) Commercially available boot managers (i.e. BootMagic, System Commander, etc.)
  c) Windows NT/XP boot manager
     1. "dd if=/dev/hda of=/tmp/linux.bs bs=512 count=1"
     2. Move "linux.bs" to Windows NT "root" and configure into "boot.ini" file
- The important thing to remember: whatever boot manager is in control must be able to find and access the boot sector containing LILO's loader
- If your system's hard drive configuration does not meet the Linux booting requirements, then you can boot from a floppy disk
- If your system is ONLY Linux, then things are much simpler!

Example LILO configuration file

```
boot          = /dev/fd0        # Specify boot device
delay         = 10              # Wait 10 seconds
message = bootmessage           # Text prompt
read-only                       # Mount root RO

label =  linux_up               # Uniprocessor
         image   = vmlinuz-2.2.12-20
         initrd  = initrd-2.2.12-20.img
         root    = /dev/hda5
label = linux_smp               # Multi-processor
         image   = vmlinuz-2.2.12-20smp
         initrd  = initrd-2.2.12-20smp.img
         root    = /dev/hda5
label = linux_old               # Last week's kernel
         image   = vmlinuz_old
         initrd  = initrd-2.2.12-20old.img
         root    = /dev/hda5
```

Booting the Kernel With LILO

- If properly configured, LILO will present you with a menu of kernels to boot.
- While booting, LILO outputs "L…I…L…O" if working properly. Where it stops is the only diagnostic you will get if something goes wrong. See documentation for LILO to determine where in the boot process things have gone awry.
- Type Ctrl-X to get a text-mode boot prompt
- In text-mode, the TAB key will display possible boot targets
- LILO can boot Windows systems as well as Linux systems
- Run LILO "lilo –v –v –t " to see all output and TEST the configuration, if something is wrong your system may not boot (time for the boot floppy you made…)
- Be CAREFUL when running LILO, you can inadvertently overwrite the MBR on multi-boot systems when all you want to do is change the BR in the /boot partition!

Practical Experiences with LILO

- When you update a kernel, you must re-run LILO
- You should uninstall LILO before "decomissioning" a system, especially if LILO is installed in the MBR
- The default configuration file for LILO is /etc/lilo.conf
- You can build a "mini" boot partition on a floppy, either by hand or using the *"/sbin/mkbootdisk"* command
- The LILO command may be run on a mounted disk by specifying the "-r <directory>" option, which will do a "chroot" to that directory
- Examine the "mkbootdisk" script for examples of what is needed to build a bootable floppy disk
- With the proper symbolic links and naming conventions, you can have a very flexible way to try new kernel configurations

Personal Experiences with LILO

- I personally prefer to avoid LILO at all cost
- LILO stores file system offset information about the kernel and configuration files in the boot sector, which means that *any* time you make a change to LILO or system configurations you must re-run LILO to update the boot sector information
- Forget to re-run LILO after changes and your system will be unbootable
- LILO also seems to be “resistant” to uninstalling itself from the MBR if you want to replace it …
- MAKE A COPY OF YOUR MASTER BOOT RECORD BEFORE INSTALLING LILO (if multi-boot)!
    - dd if=/dev/hda of=/tmp/boots.orig bs=512 count=1

GRand Unified Bootloader (GRUB) Tips

- /usr/share/grub/i386-redhat contains the “raw” GRUB installation files
- /boot/grub contains grub boot files, including grub.conf and the files that provide file-system support
- > “info grub”
  will invoke complete documentation for grub, including installation instructions and examples (using “info” takes some getting used to, it is EMACS in disguise)
- GRUB supports ext2, ext3, xfs, reiserfs, fat, minix, jfs, vstafs, ffs
- More on GRUB coming up …

Making a GRUB Boot Disk

- You can make a GRUB boot disk that will allow you to boot any supported system from the floppy in the event of an emergency
- The GRUB files are in /usr/share/grub/redhat-i386
- To make a boot disk:

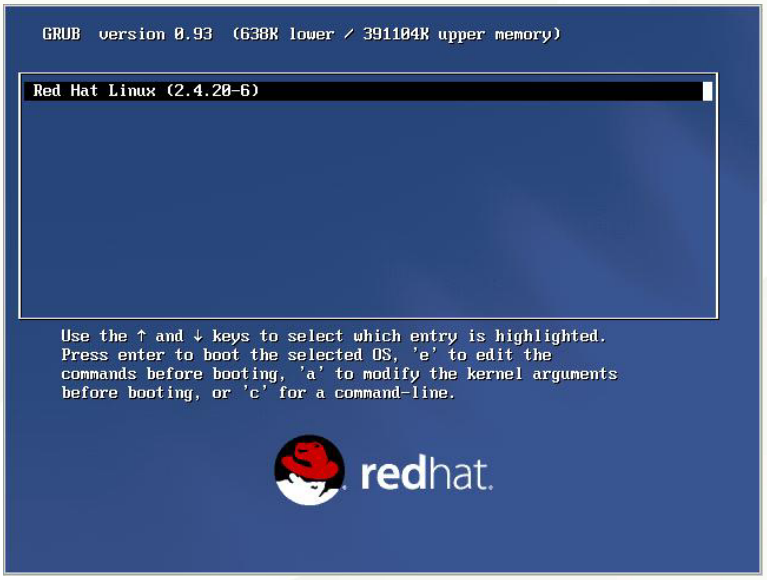
a) Make sure the floppy is not write-protected
b) All data on the floppy will be destroyed by this process!
c) cd /usr/share/grub/redhat-i386
d) dd if=stage1 of=/dev/fd0 bs=512 count=1
e) dd if=stage2 of=/dev/fd0 bs=512 seek=1

An Example GRUB Configuration File

- /boot/grub/grub.conf is the configuration file location
- An example from one of my systems:

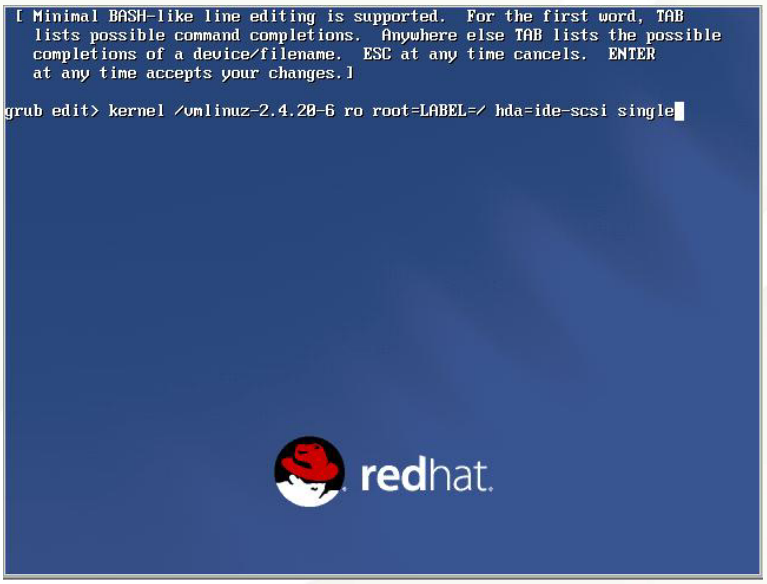
```
# grub.conf generated by anaconda
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.

#boot=/dev/hdc
default=1
timeout=10
splashimage=(hd0,0)/grub/splash.xpm.gz
title Red Hat Linux (2.4.20-18.9)
        root (hd0,0)
        kernel /vmlinuz-2.4.20-18.9 ro root=LABEL=/ hda=ide-scsi
        initrd /initrd-2.4.20-18.9.img
title Red Hat Linux (2.4.20-13.9)
        root (hd0,0)
        kernel /vmlinuz-2.4.20-13.9 ro root=LABEL=/ hda=ide-scsi
        initrd /initrd-2.4.20-13.9.img
```

Grub Boot Screen



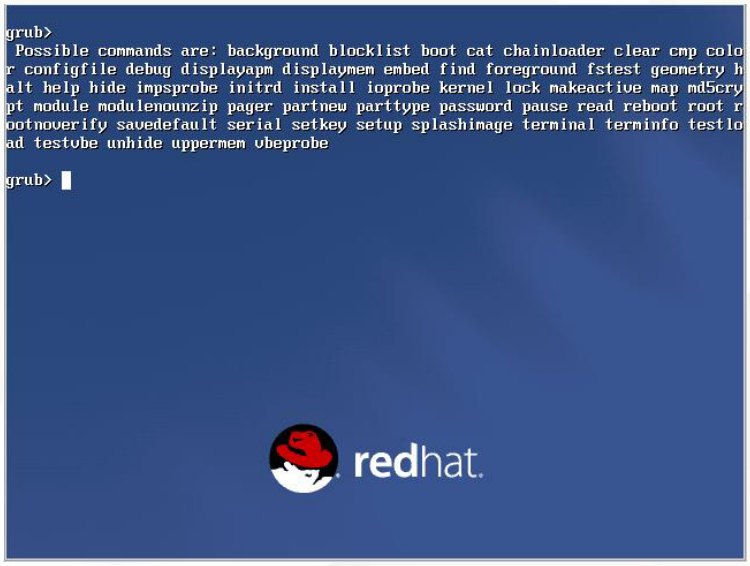
Grub Edit Screen (type “e” to enter)

Editing GRUB Kernel Definition for Single-User Mode

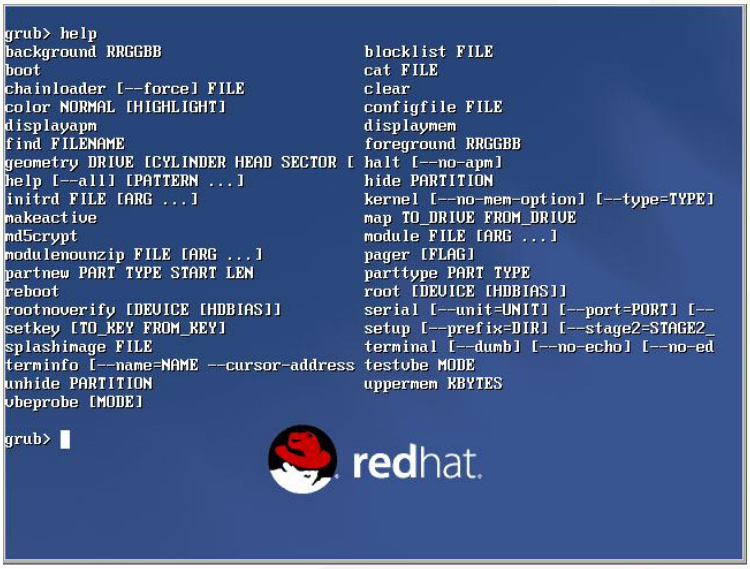


Finishing the GRUB Edit (type <CR> to return)

Possible GRUB Commands (type <TAB> to list)



Getting GRUB Help



Using GRUB From A Boot Disk

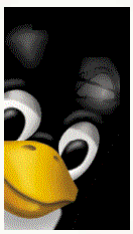
Ready To Boot (type “b”)

Sample grub.conf File from Dual-boot System

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.

#boot=/dev/sda
default=2
timeout=10
splashimage=(hd0,2)/grub/splash.xpm.gz
title Red Hat Linux (2.4.20-18.9smp)
        root (hd0,2)
        kernel /vmlinuz-2.4.20-18.9smp ro root=LABEL=/ hda=ide-scsi hdb=ide-scsi
        initrd /initrd-2.4.20-18.9smp.img
title Red Hat Linux (2.4.20-18.9)
        root (hd0,2)
        kernel /vmlinuz-2.4.20-18.9 ro root=LABEL=/ hda=ide-scsi hdb=ide-scsi
        initrd /initrd-2.4.20-18.9.img
title WindowsXP
        rootnoverify (hd0,0)
        chainloader +1
```

Installing Redhat Linux from the Network

- Using Kickstart
- Introduction to Systemimager



Redhat's Kickstart Installation Tool

- May be used for either installations or upgrades
- Driven by a kickstart configuration file, accessed from floppy or over the network
- Accesses rpm packages in an install tree from the network via NFS, FTP, or HTTP
- There are two places to get initial .ks files:
  - From the examples on the Redhat Documentation CD
  - Edit /root/anaconda-ks.cfg (results of installation process)
- You may also create a kickstart file from scratch with the "redhat-config-kickstart" command, or with your favorite text editor
- Installations may be text or graphics
- Installations may be interactive or automatic

The Kickstart File Contents

- The kickstart file contains sections, each must be specified in the order expected by the kickstart tool
- The sections contain configuration items which (usually) may be declared in any order
- You can leave out items that are not "required"
- Any required item that you leave out will cause a prompt for the information
- You can place comments in the kickstart file by starting a line with "#"

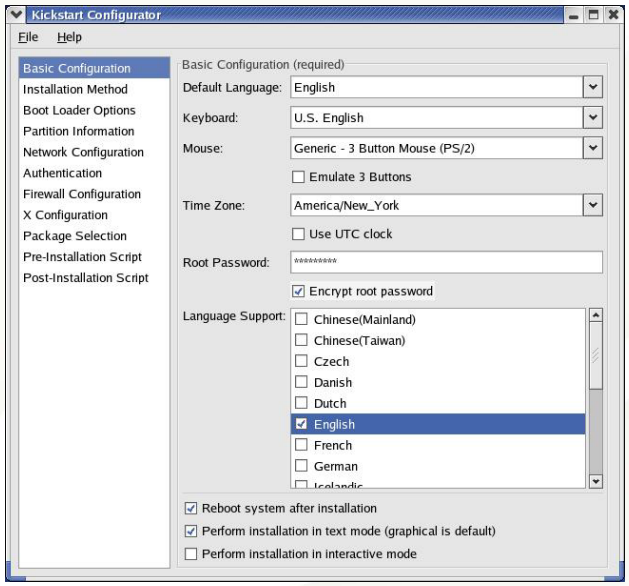
An Example Kickstart from Anaconda

```
# Kickstart file automatically generated by anaconda.
install
lang              en_US.UTF-8
langsupport       --default en_US.UTF-8 en_US.UTF-8
keyboard           us
mouse             generic3ps/2 --device psaux

xconfig --card "Intel 810" --videoram 16384 --hsync 30-80 --vsync
      56-85 --resolution 1024x768 --depth 24 --startxonboot      \
       --defaultdesktop gnome

network --device eth0 --bootproto static --ip 192.168.0.103 --
      netmask 255.255.255.0 --gateway 192.168.0.1                \
      --nameserver 192.168.0.1 --hostname hppav1i

rootpw                    --iscrypted faW/KOgOe6Tis
firewall                  --disabled
authconfig        --enablenis --nisdomain home.dom
```

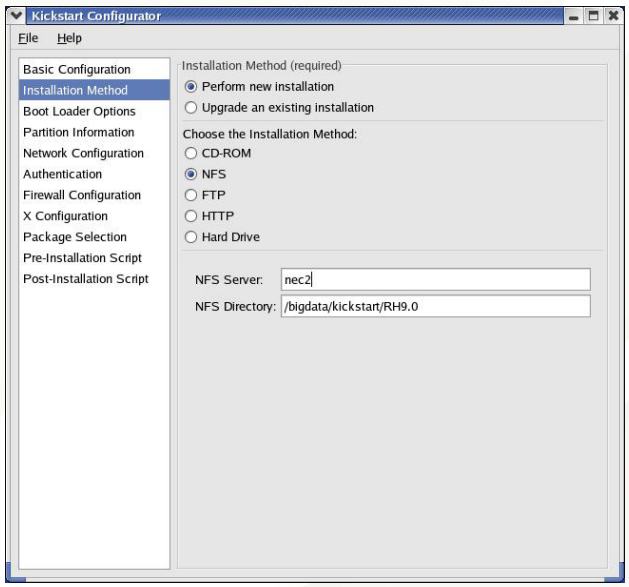
An Example Kickstart from Anaconda (continued 1)

```
timezone          America/Los_Angeles
bootloader        --location=mbr --append hda=ide-scsi

# The following is the partition information you requested
# Note that any partitions you deleted are not expressed
# here so unless you clear all partitions first, this is
# not guaranteed to work

#clearpart --linux
#part /boot --fstype ext3 --onpart hdc1
#part / --fstype ext3 --onpart hdc2
#part swap --onpart hdc3

%packages
@ Everything
%post
```

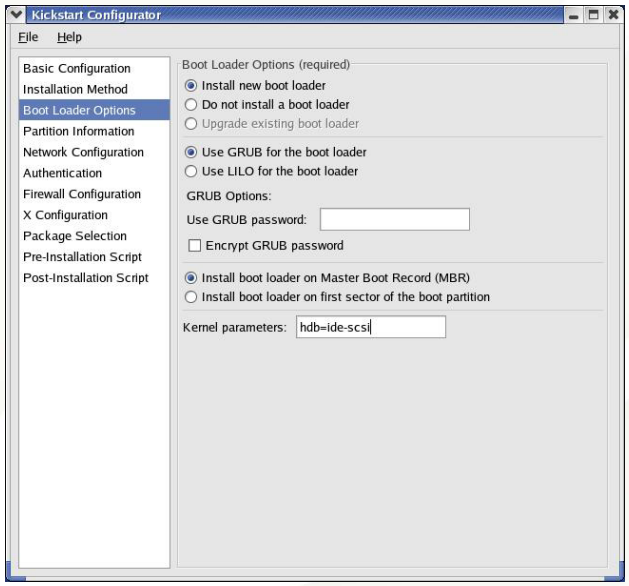
Using “redhat-config-kickstart” – Basic Configuration



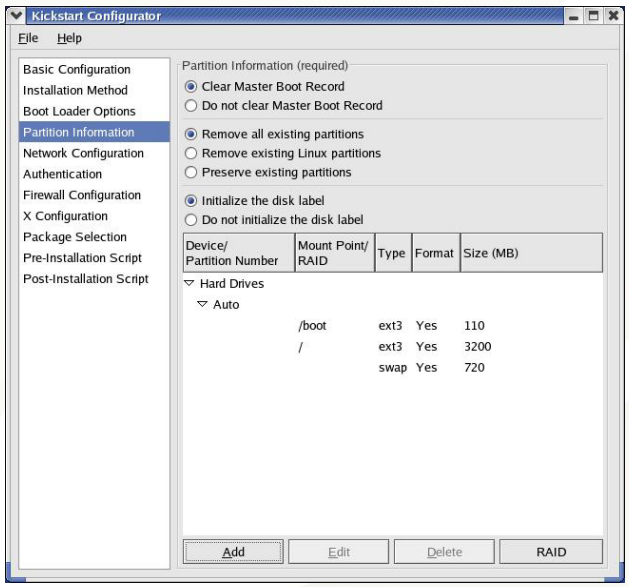
Using “redhat-config-kickstart” – Installation Method



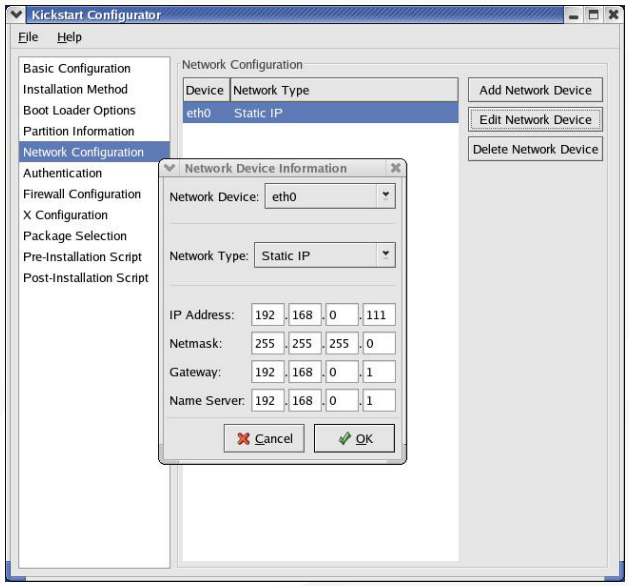
Using “redhat-config-kickstart” – Boot Loader



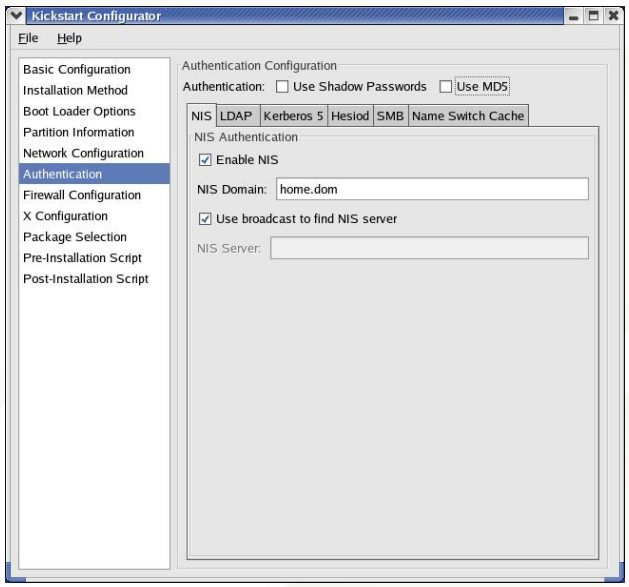
Using “redhat-config-kickstart” – Partition Information



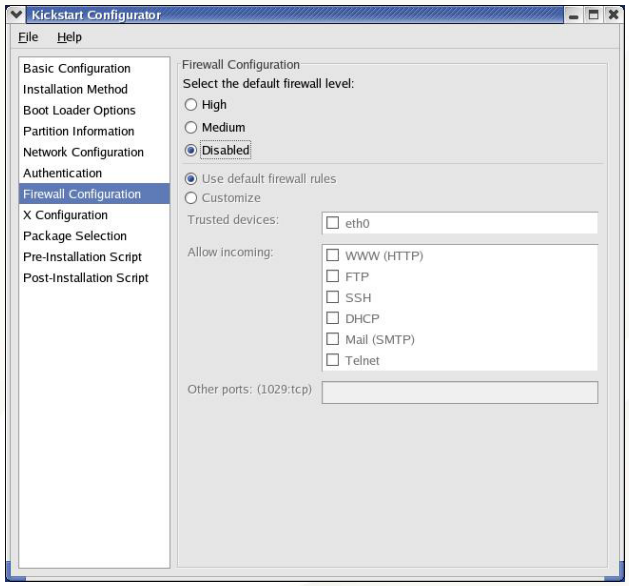
Using “redhat-config-kickstart” – Network Configuration



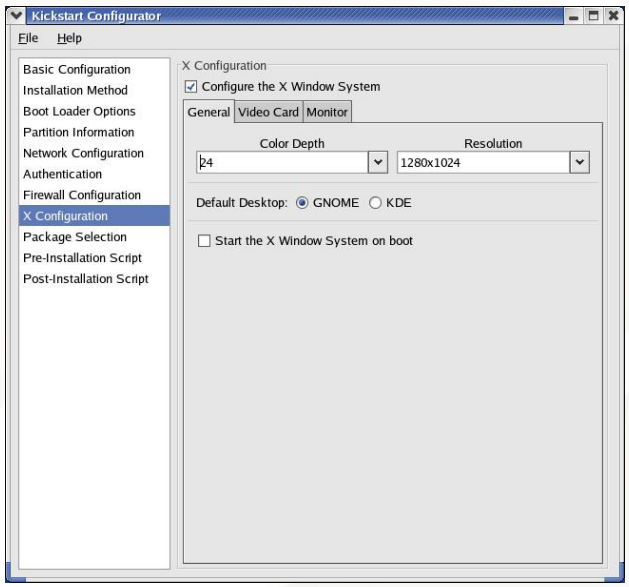
Using “redhat-config-kickstart” – Authentication



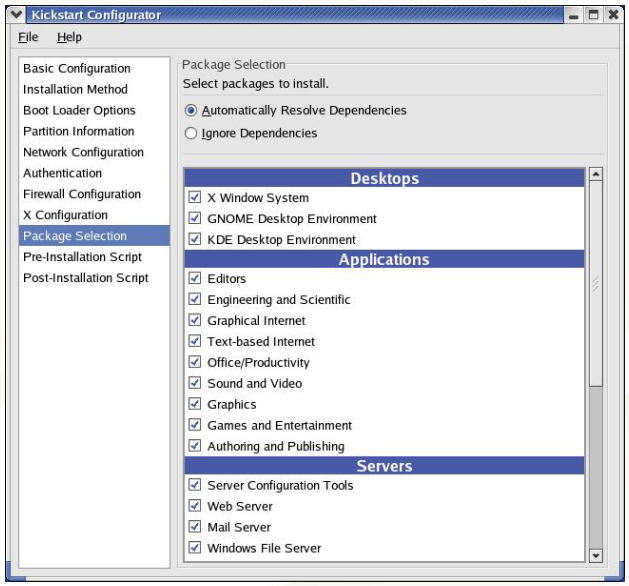
Using “redhat-config-kickstart” – Firewall Configuration



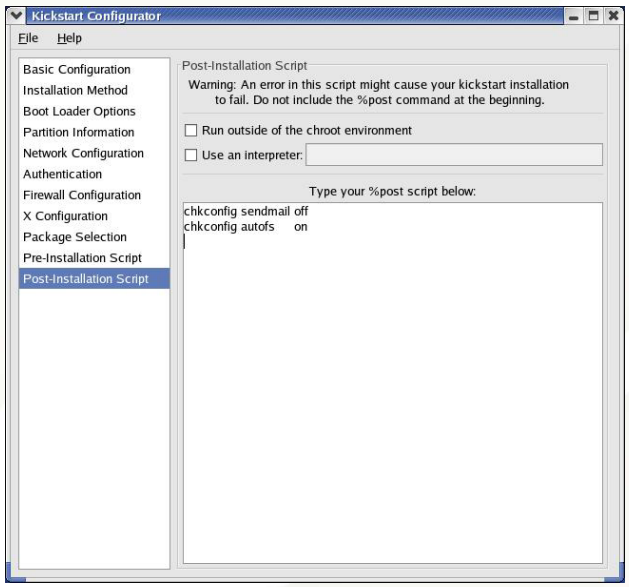
Using “redhat-config-kickstart” – X Configuration



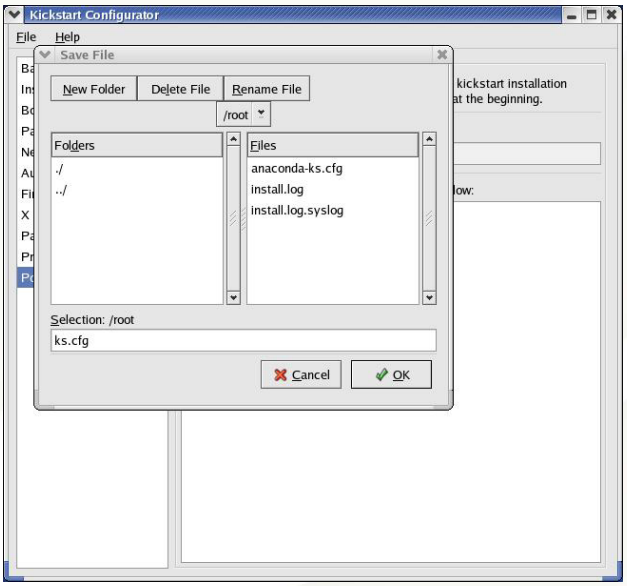
Using “redhat-config-kickstart” – Package Group Select



Using “redhat-config-kickstart” – Pre and Post Scripts



Using “redhat-config-kickstart” – Saving the File



Using “redhat-config-kickstart” – The Results (1)

```
#Generated by Kickstart Configurator
#System  language
lang en_US
#Language modules to install
langsupport  --default=en_US
#System keyboard
keyboard us
#System mouse
mouse generic3ps/2
#Sytem timezone
timezone America/New_York
#Root password
rootpw --iscrypted $1$zNCHLuyK$1FBPyan9usDvkDqNgVB7C0
#Reboot after installation
reboot
#Use text mode install
text
#Install Red Hat Linux instead of upgrade
install
#Use NFS installation Media
nfs --server=nec2 --dir=/bigdata/kickstart/RH9.0
```

Using “redhat-config-kickstart” – The Results (2)

```
#System bootloader configuration
bootloader --location=mbr --append hdb=ide-scsi
#Clear the Master Boot Record
zerombr yes
#Partition clearing information
clearpart --all --initlabel
#Disk partitioning information
part /boot --fstype ext3 --size 110 --asprimary
part / --fstype ext3 --size 3200 --asprimary
part swap --size 720
#System authorization infomation
auth  --enablenis --nisdomain home.dom
#Network information
network --bootproto=static --ip=192.168.0.111 --netmask=255.255.255.0 --gateway=192.168.0.1
        --nameserver=192.168.0.1 --device=eth0
firewall --disabled
xconfig --depth=24 --resolution=1280x1024 --defaultdesktop=GNOME
#Package install information
%packages --resolvedeps
@ X Window System
@ GNOME Desktop Environment
@ KDE Desktop Environment
@ Editors
```

Using “redhat-config-kickstart” – The Results (3)

```
@ Engineering and Scientific
@ Graphical Internet
@ Text-based Internet
@ Office/Productivity
@ Sound and Video
@ Graphics
@ Games and Entertainment
@ Authoring and Publishing
@ Server Configuration Tools
@ Web Server
@ Mail Server
@ Windows File Server
@ DNS Name Server
@ FTP Server
@ SQL Database Server
@ News Server
@ Network Servers
@ Development Tools
@ Kernel Development
@ Administration Tools
@ System Tools
%post
/sbin/chkconfig sendmail off
/sbin/chkconfig autofs     on
```

Kickstart – Okay, Now What?

- **We have our kickstart file saved for this system**
- **We need to make the kickstart file available to the system(s) that will be installed**
- **The installation tree containing the RPM files must be available via NFS, FTP, or HTTP**
- **Making the kickstart file available:**
  - **Bootable floppy**
  - **Bootable CD-ROM**
  - **Via DHCP**
- **Each method has its advantages and disadvantages**

Create A Bootable Floppy

- **The install DVD or the first install CD-ROM has the /images directory**
- **The file *bootdisk.img* is a complete image of a floppy drive**
- **Copying the image to the floppy will destroy any existing data!**
- ***dd if=bootdisk.img of=/dev/fd0 bs=1440k* (linux)**
- ***mount –t vfat /dev/fd0 /mnt/floppy***
- ***cp /root/ks.cfg /mnt/floppy***
- ***umount /mnt/floppy***
- **You can copy the file to the floppy without mounting it with *"mcopy ks.cfg a:"***

Create A Bootable CD-ROM

- **The installation DVD or the first CD-ROM has the directory *isolinux* on it**

- ***cp –r /mnt/cdrom/isolinux /tmp***
- ***cd /tmp***
- ***chmod u+w /tmp/isolinux/****
- ***mkisofs –o cd.iso –b isolinux.bin –c boot.cat ***
  ***-no-emul-boot -boot-load-size 4 –boot-info-table ***
  -R –J –v –T isolinux/

- **This will create a file, *tmp/cd.iso*, that contains an image of an ISO file system**

- **You can use *cdrecord* or *xcdroast* to burn the ISO image onto a CD**

Use DHCP to Specify Kickstart Information

- **Set up DHCP information for the client**

- **The DHCP server's */etc/dhcpd.conf* contains client's MAC address, host name, IP address, gateways, etc.**

- **"next-server" DHCPD parameter specifies NFS server name**

- **"filename" DHCPD parameter for client specifies either the absolute path and file or only the path (terminated in "/")**

- **If the "filename" parameter ends in a "/", then "<client-ip>-kickstart" is used as the kickstart file name**

- **Booting the client and selecting a DHCP boot will cause a *broadcast* DHCP request and an answer from the DHCP server with the kickstart information**

Now We Boot and Install …

- **Boot from the media and point the installer to the kickstart file by typing into the boot prompt:**
  - From floppy: `linux ks=floppy`
  - From RH CD #1 with floppy `linux ks=hd:fd0:/ks.cfg`
  - From boot CD `linux ks=cdrom:/ks.cfg`
  - From NFS `linux ks=<s>:/<path>`
  - From DHCP `linux ks`
- **Other variations:**
  - From NFS with interface
    ```
    linux ks=<s>:/<path> \
          ksdefvice=eth1
    ```
  - From web server `linux ks=http://foo.com/ks.cfg`

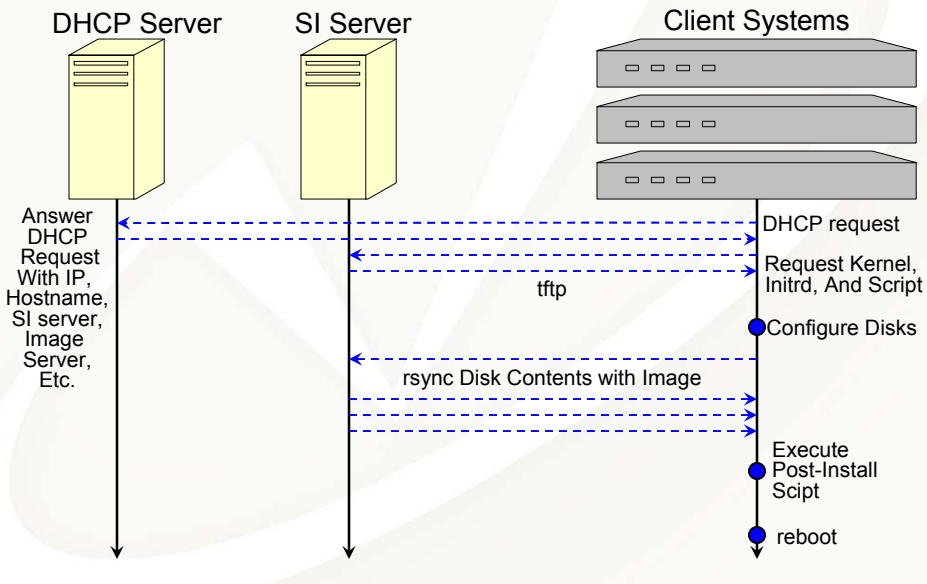
Introduction to “systemimager”

- The “systemimager” tool allows you to capture “golden client” images on a server and reinstall them or update them over the network
- http://WWW.Systemimager.Org –or– http://SourceForge.Net/projects/systemimager
- This tool is part of the “System Installation Suite” from http://SISuite.org
- Changes to the image on the server can be pushed to the clients without a re-install
- IA-32 and IA-64 partitioning schemes and systems are supported
- Network, CD, and DVD-based installations are supported
- Installing images instead of packages eliminates the configuration script overhead and the clean-up

Advantages of “systemimager” Approach

- **Deals with files that are not part of a package**
- **Scales better than kickstart installations**
- **An image-based installation takes less time than a package-based method**
- **Changes may be propagated without requiring a re-installation**
- **Does not depend on a particular distribution or package format**

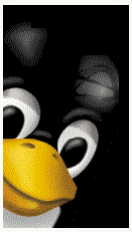
Overview of “systemimager” Operation



Lab #2: Installing Redhat Linux from the Network

See Lab #2 Handout for details

Linux Booting and System Start-up



- Passing Kernel Parameters
- Modifying Kernel Settings
- The /boot Directory
- What's In An initrd.img?
- Run-levels and Start-up
- Enabling Services
- Starting and Stopping Services

Passing Parameters to the Kernel at Boot Time

- The Linux kernel (and modules), just like HP-UX, may be passed “command-line” parameters when they are started (remember the “hpux -is disk(;0)vmunix” command?)
- LILO or GRUB can pass the following parameters (and many more) from the command-line or configuration file:
  - read-only — mount the root read-only
  - vga=mode — normal(80x25), extended(80x50), ask
  - ramdisk=<size> — set size of initial RAM disk
  - Root=<root_device> — device name or “current”
  - append=<string> — append <string> to options
  - literal=<string> — override ALL options with <string>
- LILO input: “<name> single” will boot into single-user mode
- See the file */usr/src/linux-2.4/Documentation/kernel-parameters.txt* for a list of all (most?) kernel parameters

Examples of Passing the Kernel Parameters

- kernel /vmlinuz-2.4.20-18.9 ro root=LABEL=/ hda=ide-scsi hdb=ide-scsi
- linux ks=ks.cfg
- linux console=ttyS00,9600
- linux mem=128M
- linux nfsroot=192.168.0.102:/shared-roots
- linux nosmp
- linux maxcpus=4

Modifying Kernel or Module Settings Automatically

- The "*sysctl*" command may be used to set or examine the parameters that affect system behavior, either at boot or "real-time"
- *"sysctl –a"* — List all parameters
- *"sysctl –w <parameter>=<value>"* — Set a parameter
- *"sysctl –p [<file>]"* — Load from file or /etc/sysctl.conf
- Many drivers and modules also allow setting values through the */proc* file system:

  echo "213458" > /proc/sys/net/core/rmem_default
  echo "546789" > /proc/sys/net/core/rmem_max

  changes the default socket buffer memory pool size and maximum size, respectively, for all sockets on the system.

Listing Parameters with the "sysctl" Command

```
> sysctl –a
[…]
    net.core.rmem_default = 65535
    net.core.wmem_default = 65535
    net.core.rmem_max = 65535
    net.core.wmem_max = 65535
    vm.max-readahead = 127
    vm.min-readahead = 3
    vm.max_map_count = 65536
    vm.page-cluster = 3
    vm.pagetable_cache = 25 50
    vm.kswapd = 512  32        8
    vm.overcommit_memory = 0
    vm.bdflush = 40    0         0         0   500  3000     60        0
        0
    vm.freepages = 1055        2304     3072
    kernel.overflowgid = 65534
    kernel.overflowuid = 65534
[…]
```

The /boot Directory Contents

boot.b	←	Boot sector
chain.b	←	Chain loader
config-2.4.20-18.9	←	Kernel configuration answers
grub	←	Grub boot loader directory
initrd-2.4.20-18.9.img	←	Kernel initial RAM disk image
kernel.h	←	Kernel header file (made at boot)
message	←	Boot message (English)
message.ja	←	Boot message (Japanese)
module-info	←	Link to current module-info
module-info-2.4.20-18.9	←	Kernel module loading info
os2_d.b	←	Boot for OS2
System.map	←	Link to current system map
System.map-2.4.20-18.9	←	Kernel symbols and addresses
vmlinuz	←	Link to current kernel
vmlinuz-2.4.20-18.9	←	Compressed Linux kernel

The /boot/grub Directory Contents

device.map	←	Grub to linux device map (hd0->hdc)
e2fs_stage1_5	←	Stage 1 loader for EFS
fat_stage1_5	←	Stage 1 loader for FAT
ffs_stage1_5	←	Stage 1 loader for FFS
grub.conf	←	Grub configuration file
jfs_stage1_5	←	Stage 1 loader for JFS
menu.lst	←	Link to grub.conf
minix_stage1_5	←	Stage 1 loader for Minix file system
reiserfs_stage1_5	←	Stage 1 loader for ReiserFs
splash.xpm.gz	←	Compressed bitmap background
stage1	←	Grub stage 1
stage2	←	Grub stage 2
vstafs_stage1_5	←	Stage 1 loader for VstaFs
xfs_stage1_5	←	Stage 1 loader for XFS

What's In An "initrd.img" File?

- **The initial RAM disk image file contains startup information for the kernel, which may include the dynamic modules needed to access the local hard disk, or ethernet drivers for a diskless system**
- **This situation occurs when the necessary drivers are not built into the kernel, instead they must be dynamically loaded from the disk**
- **As you may see, it is a classic "Chicken or egg" problem or Catch-22: You cannot access the disk until the drivers that are on the disk are loaded**
- **So, instead of building all potentially necessary drivers into the kernel and making it huge, the initial RAM disk allows the kernel to get to the subset of dynamic modules that it needs during boot**
- **After the initial startup is complete, the kernel unmounts the initial RAM disk and "switches" the root file system to the hard disk.**

Taking the Initial RAM Disk Image Apart

- ***cp initrd.2.4.20-18.9.img /tmp***
- ***gunzip < /tmp/initrd.2.4.20.18.9 > /tmp/initrd***
- ***mkdir /tmp/image***
- ***losetup /dev/loop0 /tmp/initrd***
- ***mount –o loop /tmp/initrd /tmp/image***
- ***cd /tmp/image***
- **<Take a look around, particularly at "linuxrc">**
- ***cd /tmp***
- ***umount /tmp/image***
- ***losetup –d /dev/loop0***
- ***rmdir /tmp/image; rm /tmp/initrd.2.4.20-18.9.img /tmp/initrd***

- **If you look carefully, you can see the point in the boot process where the kernel unmounts the initrd and switches to the hard drive (see "dmesg" or */var/log/messages* or console output)**

Comparison of HP-UX and Linux Run-level Startup

HP-UX Startup

- The world starts with “init”
- */etc/rc.config.d* contains startup data
- */etc/rc* script performs system startup
- /etc contains startup directories init.d and rc*.d
- */sbin/init.d* contains scripts that are linked into /sbin/rc*.d
- Each run level has an associated directory that contains startup and shutdown links for each subsystem
- The directory for each intervening run-level is “executed” on run-level change

Linux Startup

- The world starts with “init”
- */etc/sysconfig* and scripts contain startup data
- */etc/rc.sysinit, /etc/rc,* and */etc/rc.localperform* startup
- /etc/init.d and /etc/rc*.d are linked into /etc/rc.d/
- */etc/rc.d/init.d* contains scripts that are linked into /etc/rc*.d
- Each run level has an associated directory that contains startup and shutdown links for each subsystem
- Only the directory for the current run-level is “executed” when run-levels are changed

Comparison of HP-UX and Redhat Linux Run-levels

HP-UX Init run-levels:

0	halt
1	Single-user mode
2	Full multi-user
3	X11
4	Unused

- The “init:3:initdefault:” line in /etc/inittab controls the default level
- The “who –r” command returns the current value of init’s run-level
- No shell information about run-level is available by default
- “init <level” will change the run level

Linux Init run-levels:

0	halt
1	Single-user mode
2	Multi-user without NFS
3	Full multi-user
4	Unused
5	X11
6	Reboot

- The “id:5:initdefault:” line in /etc/inittab controls default level
- The “runlevel” command returns previous level and current level (N=none) “N 5”
- Run-level information is available in two shell environment variables: ${RUNLEVEL} and ${PREVLEVEL}
- “init <level>” will change the run level

The Linux /etc/inittab File

```
id:5:initdefault:

# System initialization.
si::sysinit:/etc/rc.d/rc.sysinit

l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6

# Trap CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 –r    \
        now
pf::powerfail:/sbin/shutdown -f -h +2 \
 "Power Failure; System Shutting       \
        Down"
```

```
# If power was restored before the
# shutdown kicked in, cancel it.
pr:12345:powerokwait:\
        /sbin/shutdown                  \
  -c "Power Restored; Shutdown  \
        Cancelled"

# Run gettys in standard runlevels
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6

# Run xdm in runlevel 5
x:5:respawn:/etc/X11/prefdm       \
        -nodaemon
```

The Redhat “chkconfig” Command

- Each file in */etc/init.d* may contain a special comment line that resembles:

  # chkconfig: 2345 80 30

  like this one from the */etc/init.d/sendmail* file.
- This comment tells the “*chkconfig*” command how to enable the associated service.
- “*chkconfig*” will create links in runlevels 2, 3, 4, and 5 with S80sendmail and K30sendmail (Start and Kill)
- *“chkconfig sendmail on”* will create the links *but not start* the service
- *“chkconfig sendmail off”* will remove the links but not stop the service
- *“chkconfig --list”* will show all services and their status, including services handled by xinetd
- If you don’t see a service listed, but the chkconfig information is in the */etc/init.d/<service>* file, try *“chkconfig --add <service>”*
- “*chkconfig --list sendmail*” will show a status line like (on, off):

```
sendmail      0:off     1:off     2:on      3:off     4:off     5:off     6:off
sendmail      0:off     1:off     2:on      3:on      4:on      5:on      6:off
```

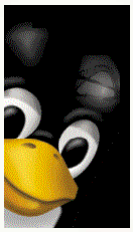
Starting, Stopping, and Restarting Services

- The "*service*" command will start, stop, or restart a service.
- *"service --status-all"* will return the current status of all services (may take a while …)
- *"service <service> stop"* will stop the service
- *"service <service> start"* will start the service
- Note: */var/run* has files containing PIDs for running processes, so commands like:

  kill –SIGHUP $(< /var/run/<service>.pid)

  will kill the service associated with the .pid file
- "*service <service> restart*" will stop and then start the service
- Neat, huh? Wish that HP-UX did this? 8^)

Hardware Configuration and Trouble-shooting



- Output From Dmesg
- Kudzu and Hardware Detection
- Listing PCI Devices
- The /proc Filesystem
- XFree86 Logs
- The System Log File
- Network Interface Status

Checking What Hardware Linux “Sees”

- **Linux supports quite a bit of common hardware and some off-brand chips, but not everything is -always- supported**
- **Whenever something does not get detected, the first step is to find out what is there**
- **For PCI cards and adapters, “*lspci*” will display devices**
- **For ISA cards, “*pnpdump*” will display plug-and-play devices on the system – Most systems today do not have ISA slots or cards**
- **The “dmesg” command and */var/log/messages* will display the hardware discovery process during boot**
- **The “kudzu” command is used at boot to discover new devices, you can also run it interactively (see */etc/sysconfig/hwconf*)**
- **The */proc* file system can also be a help in determining what Linux has and has not discovered (CPU, RAM, SCSI, USB, etc.)**

Output From “dmesg” Command (in /var/log/dmesg)

```
Linux version 2.4.20-13.9 (bhcompile@porky.devel.redhat.com) (gcc version 3.2.2 20030222 \
        (Red Hat Linux 3.2.2-5)) #1 Mon May 12 10:55:37 EDT 2003
BIOS-provided physical RAM map:
 BIOS-e820: 0000000000000000 - 000000000009f800 (usable)
 BIOS-e820: 000000000009f800 - 00000000000a0000 (reserved)
 BIOS-e820: 00000000000e6c00 - 0000000000100000 (reserved)
 BIOS-e820: 0000000000100000 - 000000001fef0000 (usable)
 BIOS-e820: 000000001fef0000 - 000000001feffc00 (ACPI data)
 BIOS-e820: 000000001feffc00 - 000000001ff00000 (ACPI NVS)
 BIOS-e820: 000000001ff00000 - 0000000020000000 (reserved)
 BIOS-e820: 00000000fff00000 - 0000000100000000 (reserved)
0MB HIGHMEM available.
510MB LOWMEM available.
On node 0 totalpages: 130800
zone(0): 4096 pages.
zone(1): 126704 pages.
zone(2): 0 pages.
Kernel command line: ro root=LABEL=/ hda=ide-scsi
ide_setup: hda=ide-scsi
Initializing CPU#0
Detected 731.117 MHz processor.
Console: colour VGA+ 80x25
Calibrating delay loop... 1458.17 BogoMIPS
Memory: 510204k/523200k available (1355k kernel code, 10432k reserved, 1004k data, 132k init,
        0k highmem)
[…]
```

Redhat Value Add: /var/log/dmesg

- **The */etc/rc.sysinit* file saves a copy of the dmesg output in */var/log/dmesg***
- **This can be very useful as the dmesg buffer is circular: as messages are added to it, the earliest messages disappear**
- **The */var/log/dmesg* file contains information that can help you troubleshoot the system startup:**
  - **Which drives and partitions are available**
  - **What disk (IDE or SCSI) interfaces were found**
  - **Information about what memory was found**
- **This is a feature that has been on Redhat Linux for quite a while**

The "kudzu" Configured Hardware Database

```
-
class: VIDEO
bus: PCI
detached: 0
driver: Card:Intel 810
desc: "Intel Corp.|82810 CGC [Chipset Graphics
        Controller]"
vendorId: 8086
deviceId: 7121
subVendorId: 8086
subDeviceId: 7121
pciType: 1
-
class: OTHER
bus: PCI
detached: 0
driver: agpgart
desc: "Intel Corp.|82810 GMCH [Graphics Memory
        Controller Hub]"
vendorId: 8086
deviceId: 7120
subVendorId: 0000
subDeviceId: 0000
pciType: 1
-
```

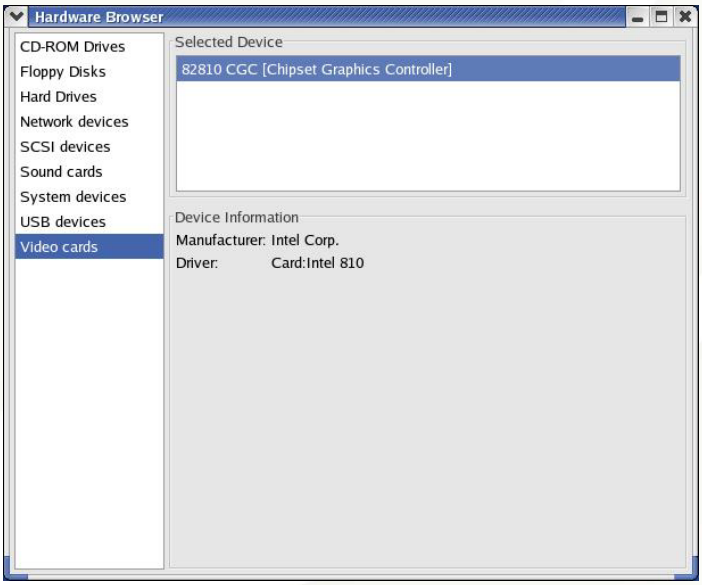
- **The information in the kudzu database, /usr/sysconfig/hwconf, can be helpful in tracking down issues**
- **The "vendorId" and "deviceId" values are used to identify hardware devices in */usr/share/hwdata/pcitable* entries**
- **Any device that is not found in the pcitable file will show up as "Unknown" and will not have a module loaded for it**
- **Example for Intel 810 Video:**

```
0x8086 0x7120 "agpgart" "Intel Corp.|82810 GMCH \
      [Graphics Memory Controller Hub]"
```

- **The state of the hardware scan is kept in */etc/sysconfig/hwconf*, */etc/modules.conf*, and in */etc/sysconfig/ifcfg-** files**
- **The */boot/module-info* file is also involved in matching the device to the module that drives it**
- **Example from module-info:**

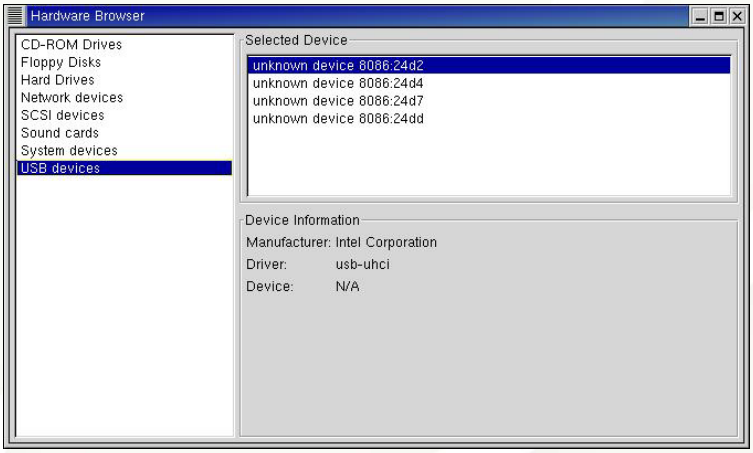
```
agpart
      video
       "Intel i810 Graphics Controller"
```

Using the “hwbrowser” Application



Here’s our Intel graphics controller showing up in the hardware browser

Now An Unknown Hardware Device …



We know this is an Intel device because the “vendorId” is 0x8086, but the system does not recognize it. This is a USB2.0 chipset running on a Redhat 7.2 version that does not support the device. At least we can figure out what is happening … eventually

The “lspci” Command

#lspci

00:00.0 Host bridge: Intel Corp. 82810 GMCH [Graphics Memory Controller Hub] (rev 03)
00:01.0 VGA compatible controller: Intel Corp. 82810 CGC [Chipset Graphics Controller] (rev 03)
00:1e.0 PCI bridge: Intel Corp. 82801AA PCI Bridge (rev 02)
00:1f.0 ISA bridge: Intel Corp. 82801AA ISA Bridge (LPC) (rev 02)
00:1f.1 IDE interface: Intel Corp. 82801AA IDE (rev 02)
00:1f.2 USB Controller: Intel Corp. 82801AA USB (rev 02)
00:1f.3 SMBus: Intel Corp. 82801AA SMBus (rev 02)
00:1f.5 Multimedia audio controller: Intel Corp. 82801AA AC'97 Audio (rev 02)
01:0b.0 SCSI storage controller: Adaptec AHA-7850 (rev 03)
01:0d.0 Ethernet controller: Accton Technology Corporation SMC2-1211TX (rev 10)
01:0e.0 Communication controller: Lucent Microelectronics LT WinModem

A Brief Introduction /proc

- **At first, the *proc* file system is a little, er, opaque and seems like magic**
- **It is a directory hierarchy that communicates with the kernel and modules**
- **This is not too much different than having a device file like */dev/ttyS0* invoke the serial port driver and communicate with the physical hardware device**
- **The kernel and modules will register with the */proc* file system if they support reads, writes, or both**
- ***/proc* is one of the most useful system administration tools when it comes to what the system is doing**
- **Many of the tools on Linux (i.e. top, gkrellm, etc.) use information from */proc***
- **When you access */proc* files with reads or writes, the kernel code or module “underneath” gets your request and can return or set data values in kernel memory**
- **We will return to */proc* in more detail a while**

Output From /proc/pci

```
# cat /proc/pci

PCI devices found:
  Bus  0, device   0, function  0:
    Host bridge: Intel Corp. 82810 GMCH
        [Graphics Memory Controller Hub]
        (rev 3).
  Bus  0, device   1, function  0:
    VGA compatible controller: Intel Corp.
        82810 CGC [Chipset Graphics
        Controller] (rev 3).
      IRQ 10.
      Prefetchable 32 bit memory at 0xf8000000
        [0xfbffffff].
      Non-prefetchable 32 bit memory at
        0xf4000000 [0xf407ffff].
  Bus  0, device  30, function  0:
    PCI bridge: Intel Corp. 82801AA PCI Bridge
        (rev 2).
      Master Capable.  No bursts.  Min Gnt=6.
  Bus  0, device  31, function  0:
    ISA bridge: Intel Corp. 82801AA ISA Bridge
        (LPC) (rev 2).
  Bus  0, device  31, function  1:
    IDE interface: Intel Corp. 82801AA IDE
        (rev 2).
      I/O at 0x1800 [0x180f].
  Bus  0, device  31, function  2:
    USB Controller: Intel Corp. 82801AA USB
        (rev 2).
      IRQ 11.
      I/O at 0x1820 [0x183f].
  Bus  0, device  31, function  3:
    SMBus: Intel Corp. 82801AA SMBus (rev 2).
      IRQ 9.
      I/O at 0x1810 [0x181f].
  Bus  0, device  31, function  5:
    Multimedia audio controller: Intel Corp.
        82801AA AC'97 Audio (rev 2).
      IRQ 9.
      I/O at 0x1200 [0x12ff].
      I/O at 0x1300 [0x133f].
  Bus  1, device  11, function  0:
    SCSI storage controller: Adaptec AHA-7850
        (rev 3).
      IRQ 9.
      Master Capable.  Latency=64.  Min
        Gnt=4.Max Lat=4.
      I/O at 0x3000 [0x30ff].
      Non-prefetchable 32 bit memory at
        0xf4100000 [0xf4100fff].
```

The Xfree86 Log File

- **Troubleshooting X-windows server and graphics card problems is easier if you use *`/var/log/XFree86.0.log`***

```
XFree86 Version 4.3.0 (Red Hat Linux release: 4.3.0-2)
Release Date: 27 February 2003
X Protocol Version 11, Revision 0, Release 6.6
Build Operating System: Linux 2.4.20-3bigmem i686 [ELF]
Build Date: 27 February 2003
[…]
(II) I810(0): Monitor0: Using hsync range of 30.00-80.00 kHz
(II) I810(0): Monitor0: Using vrefresh range of 56.00-85.00 Hz
(II) I810(0): Clock range:  12.00 to 136.00 MHz
[…]
(**) I810(0): DPI set to (72, 67)
[…]
(--) I810(0): Virtual size is 1024x768 (pitch 1024)
[…]
(==) Depth 24 pixmap format is 32 bpp
[…]
```

The System Log File: /var/log/messages

```
Jun 19 15:52:20 hppav1 kernel: ICH: IDE controller at PCI slot 00:1f.1
Jun 19 15:52:20 hppav1 kernel: ICH: chipset revision 2
Jun 19 15:52:20 hppav1 kernel: ICH: not 100%% native mode: will probe irqs later
Jun 19 15:52:20 hppav1 kernel:     ide0: BM-DMA at 0x1800-0x1807, BIOS settings:
      hda:pio, hdb:pio
Jun 19 15:52:20 hppav1 kernel:     ide1: BM-DMA at 0x1808-0x180f, BIOS settings:
      hdc:DMA, hdd:pio
Jun 19 15:52:20 hppav1 kernel: hda: LG CD-RW CED-8083B, ATAPI CD/DVD-ROM drive
Jun 19 15:52:20 hppav1 kernel: hdc: QUANTUM FIREBALLlct15 30, ATA DISK drive
Jun 19 15:52:20 hppav1 kernel: blk: queue c03cc404, I/O limit 4095Mb
Jun 19 15:52:20 hppav1 kernel: ide0 at 0x1f0-0x1f7,0x3f6 on irq 14
Jun 19 15:52:20 hppav1 apmd[1784]: Charge: * * * (-1% unknown)
Jun 19 15:52:20 hppav1 kernel: ide1 at 0x170-0x177,0x376 on irq 15
Jun 19 15:52:20 hppav1 kernel: hdc: attached ide-disk driver.
Jun 19 15:52:20 hppav1 kernel: hdc: host protected area => 1
Jun 19 15:52:20 hppav1 kernel: hdc: 58633344 sectors (30020 MB) w/418KiB Cache,
      CHS=58168/16/63, UDMA(66)
Jun 19 15:52:20 hppav1 kernel: Partition check:
Jun 19 15:52:21 hppav1 kernel:  hdc: [PTBL] [3877/240/63] hdc1 hdc2 hdc3
Jun 19 15:52:28 hppav1 kernel: eth0: SMC1211TX EZCard 10/100                \
      (RealTek RTL8139) at 0xe0955000, 00:10:b5:7c:70:42, IRQ 9
Jun 19 15:52:28 hppav1 kernel: eth0: Setting 100mbps full-duplex based on \
      auto-negotiated partner ability 45e1.
```

Checking the Status of Network Links

- One of the most frequent performance problems is the mis-negotiation of network link speeds and duplex: 100baseT-HD versus 100baseT-FD, etc.
- The “media independent interface tool”, “*mii-tool*” will tell you what your network interface has negotiated
- The *mii-tool* will also tell you what your network interface is “advertising” as far as capabilities and what its “link partner” is advertising
- This goes a long way towards troubleshooting “auto”-negotiation issues

Example of “mii-tool” Output

 # mii-tool -v eth0

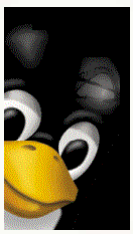
```
eth0: negotiated 100baseTx-FD, link ok
  product info: vendor 00:00:00, model 0 rev 0
  basic mode:   autonegotiation enabled
  basic status: autonegotiation complete, link ok
  capabilities: 100baseTx-FD 100baseTx-HD 10baseT-FD 10baseT-HD
  advertising:  100baseTx-FD 100baseTx-HD 10baseT-FD 10baseT-HD
  link partner: 100baseTx-FD 100baseTx-HD 10baseT-FD 10baseT-HD flow-control
```

 # mii-tool –v --force=100baseTx-FD eth0

- **Your interface hardware must support the MII (Media Independent Interface) standard**
- **Most modern network hardware appears to support MII operations**

Linux Networking

- **Initialization Scripts**
- **Administrative Commands**
- **Networking Tidbits**
- **Some Esoteric Examples**



Networking Initialization Scripts – Simple Example

 # cat /etc/sysconfig/network

NETWORKING=yes

 # cat /etc/sysconfig/network-scripts/ifcfg-eth0

DEVICE="eth0“
ONBOOT="yes“
BOOTPROTO="dhcp“

- **These settings are all you need if you have a DHCP server that is configured to return default gateway, hostname, IP address, netmask, NIS server, etc.**
- **The DHCP server matches the hardware ethernet address (Media Access Control or MAC) to the host-specific information and passes default parameters for the subnet**

Networking Initialization Scripts – More Realistic Example

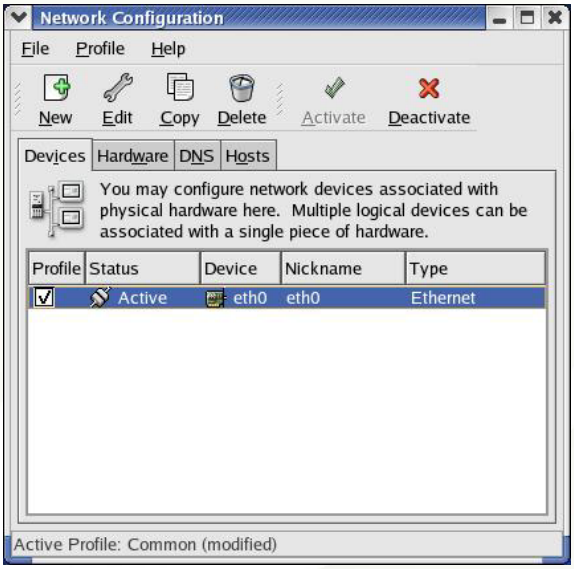
 # cat /etc/sysconfig/network

NETWORKING=yes
HOSTNAME=“hppav”
GATEWAY=192.168.0.1
NISDOMAIN=“home.domain”

 # cat /etc/sysconfig/network-scripts/ifcfg-eth0

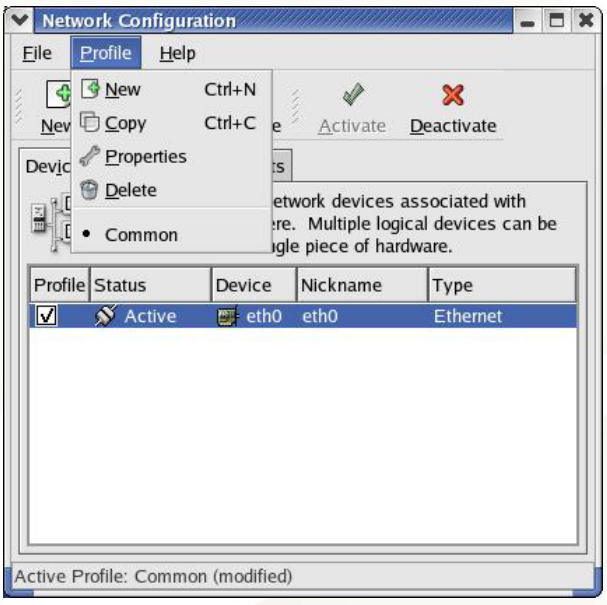
DEVICE="eth0“
ONBOOT="yes“
BOOTPROTO=static
NETWORK=192.168.0.0
IPADDR=192.168.0.101
NETMASK=255.255.255.0
BROADCAST=192.168.0.255

Using “redhat-config-network”



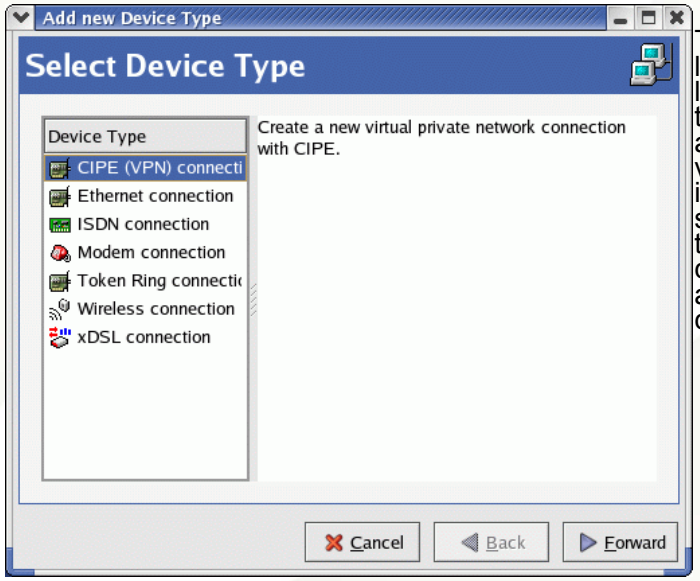
This tool is specifically aimed at configuring ethernet interfaces and the associated services like name lookup

Using “redhat-config-network” and Network Profiles



You may create network profiles that can be activated in different network situations. The default profile name is “Common” and will contain the configuration information that you create initially. You can create a new profile and save network configuration into it.

Using “redhat-config-network-druid”



This tool is more like a “wizard” that leads you through the steps of adding and configuring various network interface types. A similar interface for text-based configuration is available as redhat-config-network-tui

Miscellaneous Network Tidbits

- **Activating a network profile from the command line:**
  “redhat-config-network-cmd --profile <profile-name> --activate”
- **You can “channel bond” two interfaces with the “ifenslave” command. See the information in *`/usr/src/linux2.4/Documentation/networking/bonding.txt`* for details**
- **You can create aliases for your network device that have different network settings. A network alias has the format *device:0, device:1, etc.* For example the eth0:0 device is an alias for the eth0 interface and all routes will point to the eth0 device. See the information in *`/usr/src/linux2.4/Documentation/networking/alias.txt`* for details**
- **Linux “prefers” the use of the “*ip*” command in place of certain other networking commands. It needs a little help in the documentation department (like a man page, for instance), but it can do a *lot* there is documentation, but it is in postscript**
- **Linux supports IPv6, so you can start reading 128-bit MAC addresses**
- **Linux also supports traffic shaping on outgoing packets – this is left as an exercise to the motivated student**

Network Alias Example

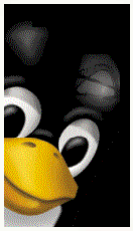
```
# ifconfig eth0:0 196.234.128.1
# ifconfig

eth0      Link encap:Ethernet  HWaddr 00:10:B5:7C:70:42
          inet addr:192.168.0.103  Bcast:192.168.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:253803 errors:0 dropped:0 overruns:0 frame:0
          TX packets:228867 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:30364299 (28.9 Mb)  TX bytes:24157683 (23.0 Mb)
          Interrupt:9 Base address:0x5000

eth0:0    Link encap:Ethernet  HWaddr 00:10:B5:7C:70:42
          inet addr:196.234.128.1  Bcast:196.234.128.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:253803 errors:0 dropped:0 overruns:0 frame:0
          TX packets:228867 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:30364299 (28.9 Mb)  TX bytes:24157683 (23.0 Mb)
          Interrupt:9 Base address:0x5000
```

Using the Linux “ip” Command

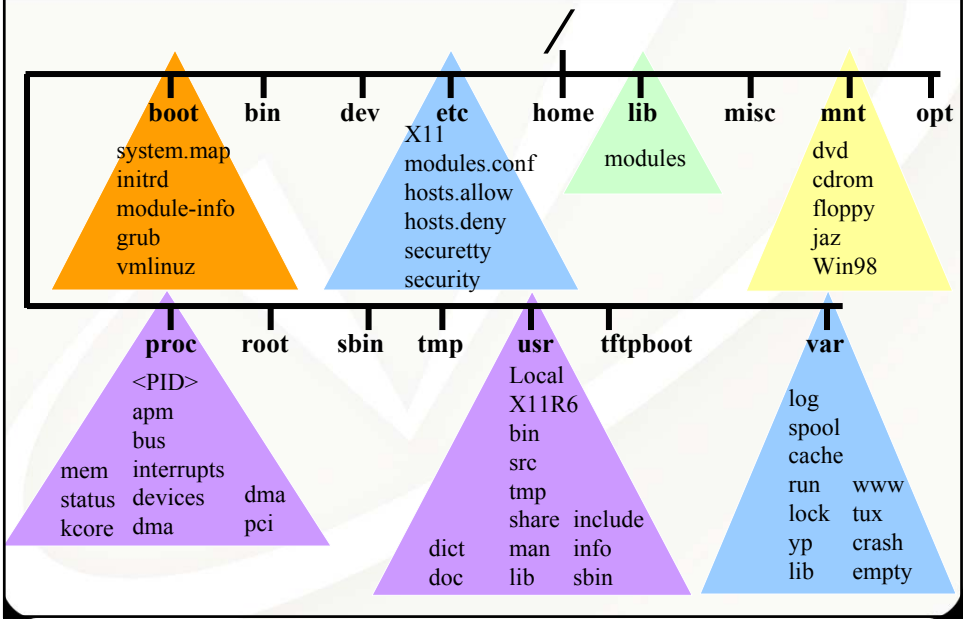
```
# ip route list
192.168.0.0/24 dev eth0  proto kernel  scope link  src 192.168.0.103
196.234.128.0/24 dev eth0  proto kernel  scope link  src 196.234.128.1
169.254.0.0/16 dev eth0  scope link
127.0.0.0/8 dev lo  scope link
default via 192.168.0.1 dev eth0

# ip addr list
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
   inet 127.0.0.1/8 brd 127.255.255.255 scope host lo
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
   link/ether 00:10:b5:7c:70:42 brd ff:ff:ff:ff:ff:ff
   inet 192.168.0.103/24 brd 192.168.0.255 scope global eth0
   inet 196.234.128.1/24 brd 196.234.128.255 scope global eth0:0

# ip link list
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
   link/ether 00:10:b5:7c:70:42 brd ff:ff:ff:ff:ff:ff
```

The Linux Filesystem Layout

- A Simple Roadmap
- Device Files
- A Real /etc/fstab
- Interesting Side-Trips
- An Important Redhat-ism
- Introduction to /proc



A Simple Roadmap of the Linux Filesystem



Some Device Names from a Real System

Description	*Use*	*Device Name*
IDE Drive 0 /dev/hda	**WinXP/Linux**	
Windows partition	**Windows XP**	**/dev/hda1**
Linux partition	**/boot**	**/dev/hda2**
CD-ROM stacker		**/dev/hdc**
HP R/W CD-ROM		**/dev/hdd**
Floppy drive		**/dev/fd0**
SCSI address 0	**Linux swap**	**/dev/sda1**
SCSI address 0	**Linux /**	**/dev/sda5**
SCSI address 1	**Linux /vmdata**	**/dev/sdb1**
SCSI address 1	**Linux swap**	**/dev/sdb5**

Some Device Names from a Real System (continued 1)

Description	*Use*	*Device Name*
SCSI address 2	**Linux /vmdata1**	**/dev/sdc1**
SCSI address 2	**Linux swap**	**/dev/sdc5**
SCSI address 3	**Linux /vmdata2**	**/dev/sdd1**
SCSI address 3	**Linux swap**	**/dev/sdd5**
SCSI address 4	**HP Photo scanner**	**/dev/sge**
SCSI address 5	**HP 6200C scanner**	**/dev/sgf**
SCSI address 6	**JAZ 1 GB drive**	**/dev/sde4**
COM1	**V.90 modem**	**/dev/ttyS0**
Parallel port	**HP P1000 printer**	**/dev/lp0**

Example /etc/fstab File

```
/dev/sda5   /           ext3     defaults          1 1
/dev/hda2   /boot       ext3     defaults          1 2
/dev/sdb1   /vmdata     ext3     defaults          1 2
/dev/sdc1   /vmdata1    ext3     defaults          1 2
/dev/sdd1   /vmdata2    ext3     defaults          1 2
/dev/sda1   swap        swap     pri=1             0 0
/dev/sdb5   swap        swap     pri=1             0 0
/dev/sdc5   swap        swap     pri=1             0 0
/dev/sdd5   swap        swap     pri=1             0 0
/dev/fd0    /mnt/floppy ext2     owner,noauto      0 0
/dev/cdrom  /mnt/cdrom  iso9660  owner,noauto,ro   0 0
/dev/hdd    /mnt/cd-rw  iso9660  noauto,ro         0 0
/dev/sde4   /mnt/jaz    vfat     fat=16            0 0
none        /proc       proc     defaults          0 0
none        /dev/pts    devpts   gid=5,mode=620    0 0
```

Interesting File System Side-trips

- */etc/pam.d* directory containing PAM config
- */etc/profile.d* directory containing system-wide shell profiles
- */etc/logrotate.d* directory controlling log rotation
  - */usr/sbin/logrotate* command
  - */etc/logrotate.conf*
- */etc/rc.d* directory containing startup scripts
- */proc* directory containing system information
  - */proc/bus/usb* directory containing usb device information
  - */proc/bus/pci* directory containing pci device information

An Interesting Redhat-ism

- *`/etc/pam.d`* directory containing PAM config
  `/etc/profile.d` directory containing shell profiles
  `/etc/logrotate.d` directory controlling log rotation
  `/etc/rc.d` directory containing run-level definitions
  `/etc/init.d` directory containing service scripts
  `/etc/xinetd.d` directory containing inetd service definitions
- These ".d" directories make it easy for packages to install and remove themselves from the system without affecting other packages or risking damage to configuration files
- Typical service installation might:
  - Add service script to */etc/init.d*
  - Run chkconfig to create links in */etc/rc*.d*
  - Add service information to */etc/xinetd.d*
  - Add log processing to */etc/logrotate.d*
- Because logrotate is setup to "*include /etc/logrotate.d*", all files become part of the configuration without grepping, awking, or sedding.
- This is *very* manageable once you catch the paradigm!

Example /proc File System Contents

```
1/     1402/  1475/  1587/  1756/  1781/  1792/  4/    763/  bus/          ide/          mdstat
      slabinfo
1014/  1438/  1488/  1589/  1757/  1782/  1793/  5/    8/    cmdline       interrupts    meminfo
      stat
1056/  1461/  1543/  1591/  1758/  1783/  1794/  536/  846/  cpuinfo       iomem         misc
      swaps
1078/  1462/  1557/  1592/  1760/  1784/  1825/  541/  879/  devices       ioports       modules
      sys/
1099/  1463/  1569/  1594/  1762/  1785/  1828/  561/  90/   dma           irq/          mounts@
      sysvipc/
1167/  1464/  1571/  1595/  1764/  1786/  183/   590/  900/  driver/       kcore         mtrr
      tty/
12/    1465/  1579/  1598/  1766/  1787/  1971/  6/    950/  execdomains   kmsg          net/
      uptime
1241/  1466/  1581/  1599/  1768/  1789/  2/     7/    968/  fb            ksyms
      partitions  version
1354/  1467/  1583/  1749/  1770/  1790/  2014/  702/  986/  filesystems   loadavg       pci
      vmnet/
1371/  1474/  1585/  1755/  1772/  1791/  3/     722/  apm   fs/           locks         self@
```

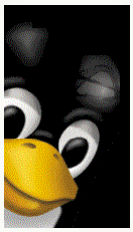
```
    PROC(5)                                Linux Programmer's Manual
NAME
       proc - process information pseudo-filesystem
DESCRIPTION
       /proc  is a pseudo-filesystem which is used as an interface to
    kernel data structures rather than reading and interpreting
    /dev/kmem.  Most of it is read-only, but some files allow kernel
    variables to be changed.
```

Example /proc/1 (PID 1 – Init) Directory Contents

```
-r--r--r--    1 root     root        0 Jul 29 22:46 cmdline
lrwxrwxrwx    1 root     root        0 Jul 29 22:46 cwd -> /
-r--------    1 root     root        0 Jul 29 22:46 environ
rwxrwxrwx     1 root     root        0 Jul 29 22:46 exe ->
    /sbin/init
dr-x------    2 root     root        0 Jul 29 22:46 fd
-r--r--r--    1 root     root        0 Jul 29 22:46 maps
-rw-------    1 root     root        0 Jul 29 22:46 mem
-r--r--r--    1 root     root        0 Jul 29 22:46 mounts
lrwxrwxrwx    1 root     root        0 Jul 29 22:46 root -> /
-r--r--r--    1 root     root        0 Jul 29 22:46 stat
-r--r--r--    1 root     root        0 Jul 29 22:46 statm
-r--r--r--    1 root     root        0 Jul 29 22:46 status
```

Software Installation and Update

- The Redhat Package Manager (RPM)
- Useful Commands
- Verifying Packages
- Using Redhat's "up2date"



The Redhat Package Manager – RPM

- **Installation and update of software on Linux (at least the RedHat distributions and several others) is done via the "rpm" command**
- **Packages may contain sources, binaries, configuration information and scripts, and PGP signatures for verification**
- **The system keeps a database of all installed packages, their revisions, and their dependencies**
- **To find out what packages are installed, "rpm –qa"**
- **To install a package,**
  "rpm –ivh <package>"
- **To update a package,**
  "*rpm –Uvh <package>*"
- **The kernel, libraries, and applications may be updated LIVE! (including glibc)**
- **Source RPMs (SRPMs) may be installed and used to build software and create regular RPMs (all SRPMs are on the Redhat distribution CD-ROMs)**

Some Useful RPM Commands

 # rpm -q --whatprovides /bin/ls

fileutils-4.1-10

 # rpm -qa | grep real

ethereal-0.9.4-0.7.3.0
ethereal-gnome-0.9.4-0.7.3.0

 # rpm --checksig RealPlayer-8.0-1.i386.rpm

RealPlayer-8.0-1.i386.rpm: md5 OK

 # rpm -q --filesbypkg ethtool

ethtool	**/usr/sbin/ethtool**
ethtool	**/usr/share/doc/ethtool-1.5**
ethtool	**/usr/share/doc/ethtool-1.5/AUTHORS**
ethtool	**/usr/share/doc/ethtool-1.5/COPYING**
ethtool	**/usr/share/doc/ethtool-1.5/ChangeLog**
ethtool	**/usr/share/doc/ethtool-1.5/INSTALL**
ethtool	**/usr/share/doc/ethtool-1.5/NEWS**
ethtool	**/usr/share/doc/ethtool-1.5/README**
ethtool	**/usr/share/man/man8/ethtool.8.gz**

Verifying Packages with RPM

- **Downloading binaries from the Internet is fraught with danger, but open-source also means that anyone can modify the source and add trojans, back-doors, etc. and make untrustworthy packages available**
- **First, download packages from trusted sources, and even then you should verify all packages that you download**
- **There are two basic levels of verification:**
  - **Ensuring that the package has not been corrupted**
  - **Ensuring that the package is signed by someone you trust**
- **An MD5 checksum is provided within RPM packages to ensure that the package has not been corrupted**
- **All packages from Redhat are signed with the Redhat GNU Privacy Guard (GPG) key**

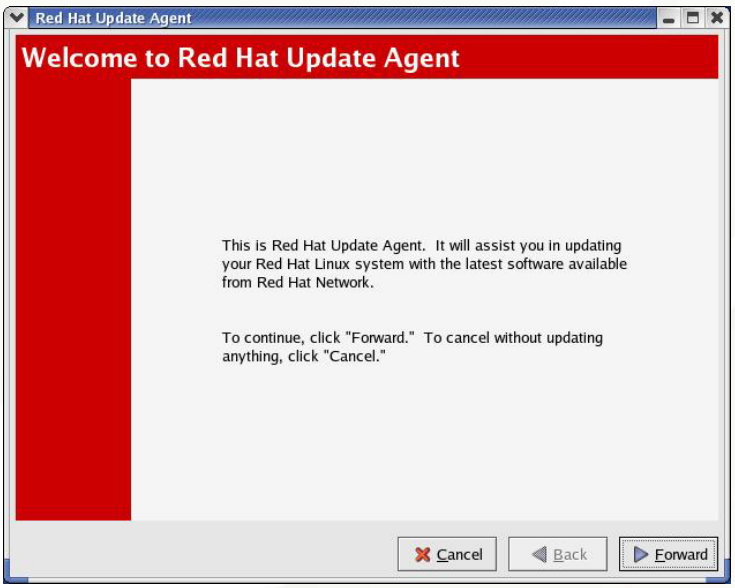
Importing the Redhat GPG Key

- **Since you will be doing your installation and updates as the "root" user, importing the Redhat GPG key is an important step**
- **This may have been done for you during system installation, you can check by executing *"rpm -qa gpg-pubkey*"*, which should respond with *"gpg-pubkey-db42a60e-37ea5438"***
- **If you need to import the key, then execute "*rpm --import /usr/share/rhn/RPM-GPG-KEY*" as the "root" user**
- **The key is also available on the Redhat distribution CDs**
- **You can check the MD5 checksum of the package with *"rpm –K --nosignature <rpm-file>"* you should see *"<rpm-file>: md5 OK"***
- **You can verify that the package was signed by Redhat with *"rpm –K <rpm-file>"* you should see the message *"md5 gpg OK"***

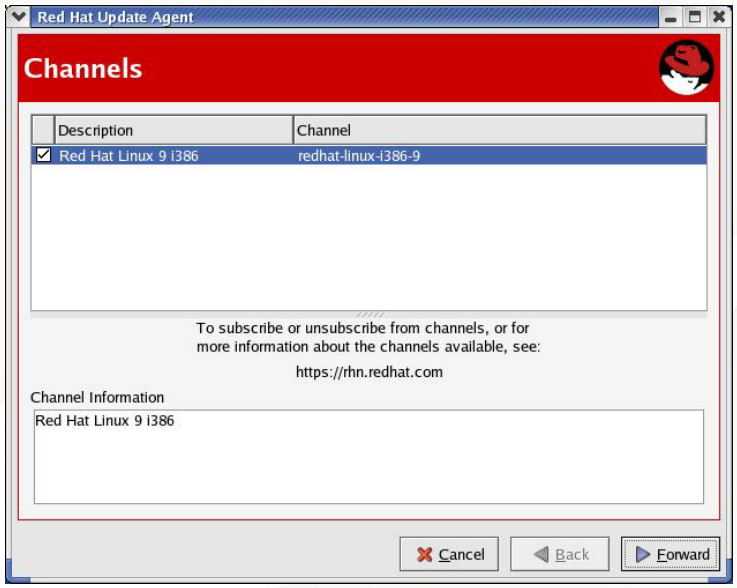
Redhat's Update Manager, "up2date"

- **RedHat provides an automated way of keeping your system packages up to date, the "up2date" command**
- **For 90 days after purchase, or if you buy support, you may connect to the RedHat FTP server with this tool**
- **All updated packages are presented in a graphical interface, and you may select the ones you want**
- **The packages are downloaded and installed**
- **Packages are stored in */var/spool/up2date* if you select the "keep after installation" option and are removed otherwise**
- **"*up2date –update*" will download all matching packages and update your system**

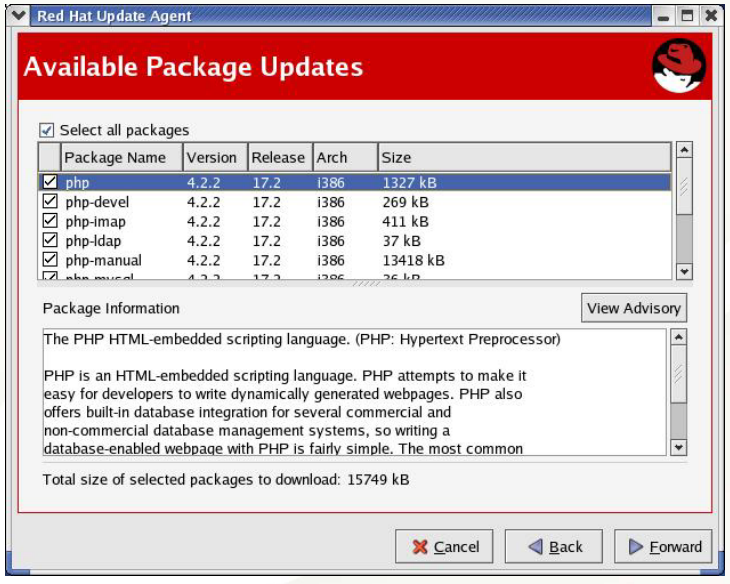
Starting "up2date"



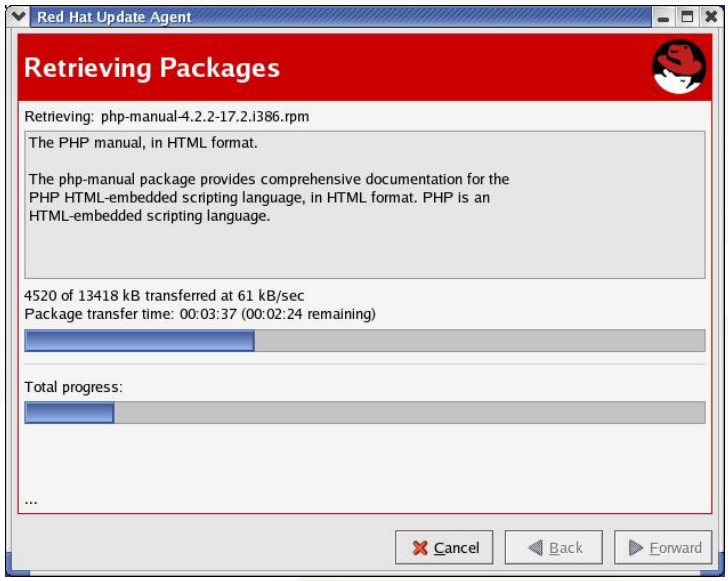
Update Channels in “up2date”



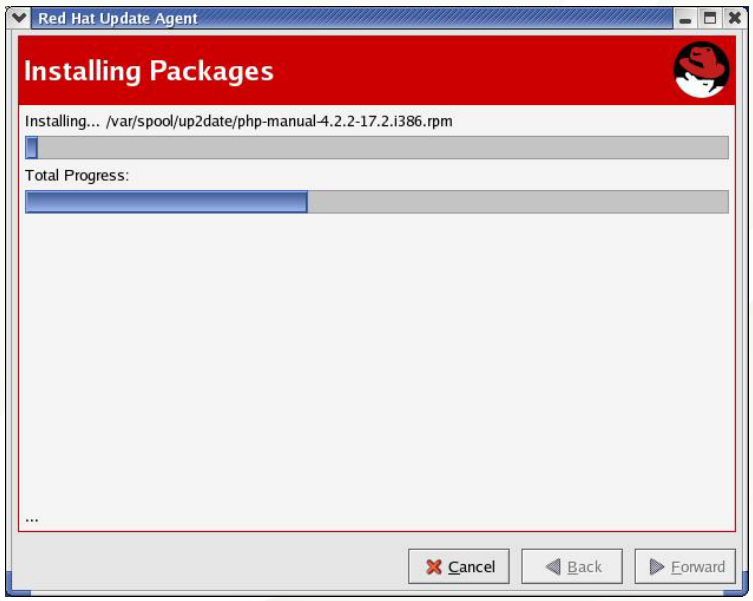
Selecting Packages from “up2date”



Downloading “up2date” Packages



Installing “up2date” Packages



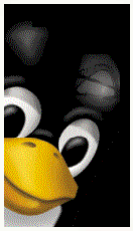
Finished

Other Software for Linux Systems

- JAVA at http://www.blackdown.org
- StarOffice from Sun Microsystems
- Open Office from http://www.OpenOffice.org
- Real Player from http://www.Real.com
- Vmware at http://www.VMware.com
- Mozilla browser from http://www.Mozilla.org
- XV X-windows image viewer/editor at http://www.trilon.com/xv/xv.html
- Other applications at http://FreshMeat.net or http://SourceForge.net (make absolutely sure that you type Freshmeat.*NET* or you will visit a site that your employer might not like)
- RPMs from ftp://Updates.Redhat.com or http://RpmFind.net

Linux Security

- Controlling System Access
- Pluggable Authentication Modules (PAM)
- Network Security
- Overview of "*netfilter*"
- Setting Up the Firewall



Linux System Security Defaults After Installation

- You will not be able to log in as "root" except at the console
- Login in as a "normal" user, then "su -" to "root", but you must HAVE a user configured besides "root" … Remember this at system installation time!
- Check out */etc/hosts.allow* and */etc/hosts.deny* to configure machines that can access your Linux box
- Check out */etc/xinetd.d* for configuration files for FTP, telnet, and other services
- For NFS to function, any firewall must permit access to the portmap service
- FTP, telnet, rlogin, etc. are disabled by default
- Use SSH wherever clear-text passwords are a problem!

Linux Security Overview

- There are a number of ways that you can control user access to a system with Linux (ignoring the “lock it in a room with no network” approach
- Linux allows using MD5 passwords along with shadow password and group files, and password strength checks and aging, all of which make it harder for crackers to break into your system
- This section is not intended to be an exhaustive security tutorial, only to point out some of the features of Linux that you can use to secure access to your systems
- A good book, though a little dated is “Linux System Security: The Administrator’s Guide to Open Source Security Tools” by Scott Mann and Ellen L. Mitchell, ISBN # 0-13-015807-0
- Linux, “out-of-the-box” is more secure than HP-UX, many services are disabled by default
- The default strategy is “deny all but what I explicitly allow”

Linux Password Management

- **Shadow password file is the default, no passwords are available to casual users**
  - Too much password data kept in /etc/shadow to go into here …
  - rob:x:1000:100:/home/rob:/bin/bash in /etc/passwd
  - chage Change user password expiration information
  - pwck Check password integrity
  - grpck Check group integrity
- */etc/passwd* */etc/passwd-* */etc/group* */etc/group-*
- */etc/shadow* (mode 400, root:root)
- */etc/gshadow* (mode 400, root:root)
- *passwd* -> pwconv -> *shadow*
- *shadow* -> pwunconv -> *passwd*
- *group* -> grconv -> *gshadow*
- *gshadow* -> grunconv -> *group*

User and Group Permission Tidbits

- Users on Redhat are created with their own group name that matches the user name
- This is so the default user and group permissions are more secure (default file permissions "rob rob : rwx r-x ---") for example
- If the SGID bit is set on a directory, then files and directories will inherit the GID of the parent directory instead of the GID from the program
- How would you use this fact?
  - Set up a project directory owned by the project manager
  - Set up a group that all members of the project belong to
  - Set the directory's group ownership to the project group
  - Set the SGID bit on the project directory
  - Set the project members' UMASK to 0007
  - Files will be created with the user's UID, but the project group ID, and all project members can manipulate the files without opening the directory permissions up for the world
- If the sticky bit is set on a directory, only the file's owner (or root) can delete it – this is another solution to for directories like */tmp* which would otherwise need to be rwx for "world"

Controlling System Shutdown and Reboot

- If you want to allow users other than root to shut the system down, then creating the */etc/shutdown.allow* file is useful
- The file contains the user names of logins that are allowed to shut the system down
- You should modify the line in */etc/inittab* that traps Ctrl-Alt-Del by adding the "-a" option to the shutdown command line to enable /etc/shutdown.allow
- If the –a option is used with shutdown and the */etc/shutdown.allow* file exists, then the file contents are used, otherwise it is ignored

Linux Pluggable Authentication Modules (PAM)

- The authentication process on Linux is controlled by PAM
- The directory */etc/pam.d* contains configuration files for applications that are "PAM aware" – I counted 91 entries in the directory on my system
- The directory */lib/security* contains the shared libraries that are loaded under the control of the PAM configuration files
- The directory */etc/security* contains files that also direct the behavior of PAM modules:
  - access.conf
  - chroot.conf
  - console.apps
  - pam_env.conf
  - console.perms
  - group.conf
  - limits.conf
  - time.conf
- PAM keeps changing so rapidly that it is very difficult to keep up with it – and as an added benefit, it is fairly complex

Example PAM Config File: /etc/pam.d/rlogin

```
#%PAM-1.0
# For root login to succeed here with pam_securetty, "rlogin" must be
# listed in /etc/securetty.
auth            required        pam_nologin.so
auth            required        pam_securetty.so
auth            required        pam_env.so
auth            sufficient      pam_rhosts_auth.so
auth            required        pam_stack.so service=system-
    auth
account         required        pam_stack.so service=system-
    auth
password        required        pam_stack.so service=system-
    auth
session         required        pam_stack.so service=system-
    auth
```

Module must return success for access to be granted, execution continues no matter what. The application is not notified of failure point if module is unsuccessful.

If modules succeeds, all other modules are ignored and success is returned to the application

PAM Module Types

- **Module types and their function:**
  - ***auth*** **Instructs application to prompt user for identification may set credentials or**
    grant **privileges**
  - ***account*** **Checks user's account attributes**
  - ***session*** **Provides functions before and after a user session is created**
  - ***password*** **Responsible for updating a user authentication token, stacked with auth module which provides prompting**

Linux Network Security

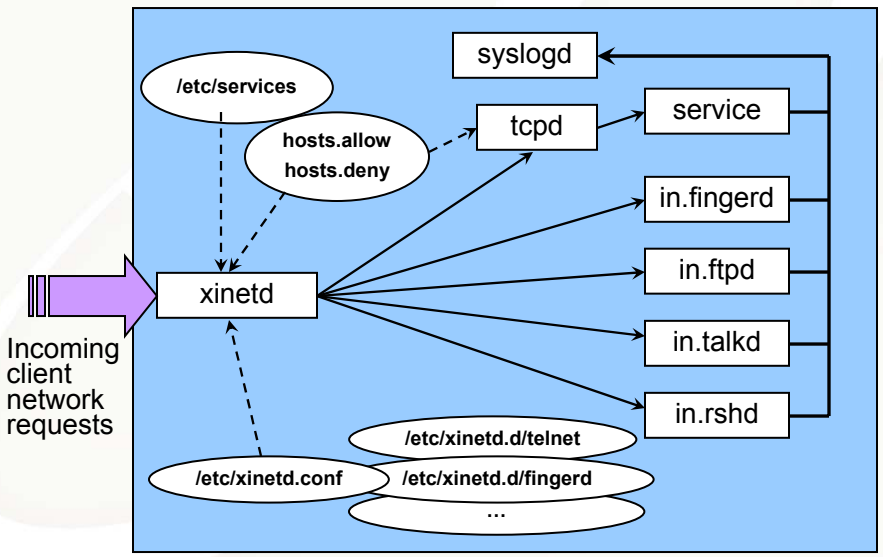
- **A network socket or port is a gaping hole in the wall of your system castle**
- **The best way to prevent access is not to open the hole – disable the service**
- **If you need to enable a service, then you need to be able to filter and control who can connect to it**
- **There are a couple of security mechanisms you need to know about:**
  - **TCP Wrappers *(/etc/hosts.allow* and */etc/hosts.deny*)**
  - **Xinetd also with *(/etc/hosts.allow* and */etc/hosts.deny*)**
- **TCP Wrappers are implemented with *tcpd* (external) or may be part of the service (like with *xinetd*)**
- **This way existing services are protected and you can add your own services by inserting *tcpd* between them and the outside world**

Xinetd Configuration File, /etc/xinetd.conf

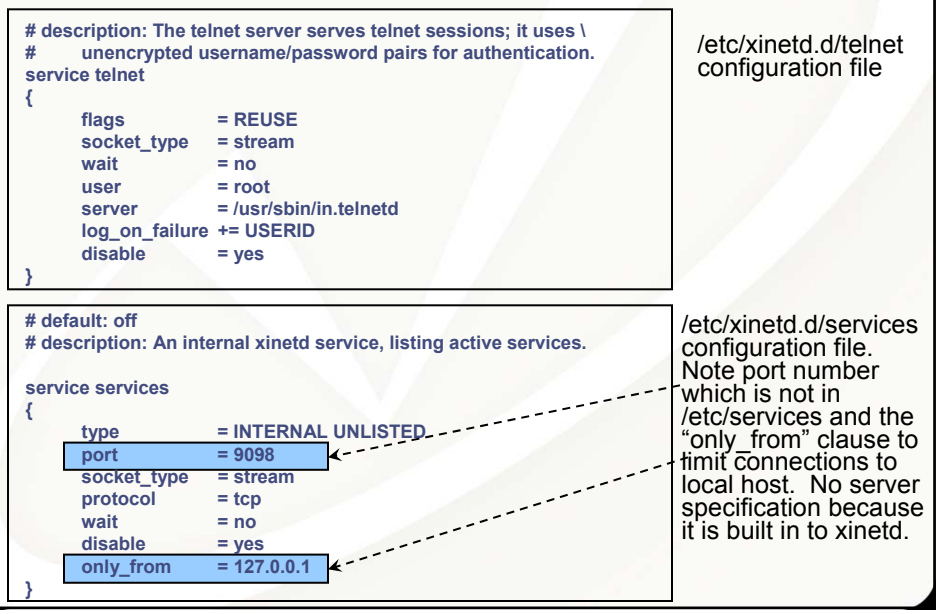
```
#
# Simple configuration file for xinetd
#
# Some defaults, and include /etc/xinetd.d/

defaults
{
        instances               = 60
        log_type                = SYSLOG authpriv
        log_on_success                  = HOST PID
        log_on_failure                  = HOST
        cps                             = 25 30
}

includedir /etc/xinetd.d
```

The files in the /etc/xinetd.d directory take the place of the contents of the old /etc/inetd.confile. This makes adding and deleting services easier.

A Picture Will Help …



Example Configuration Files from /etc/xinetd.d



```
# description: The telnet server serves telnet sessions; it uses \
#       unencrypted username/password pairs for authentication.
service telnet
{
        flags           = REUSE
        socket_type     = stream
        wait            = no
        user            = root
        server          = /usr/sbin/in.telnetd
        log_on_failure  += USERID
        disable         = yes
}
```

/etc/xinetd.d/telnet configuration file

```
# default: off
# description: An internal xinetd service, listing active services.

service services
{
        type            = INTERNAL UNLISTED
        port            = 9098
        socket_type     = stream
        protocol        = tcp
        wait            = no
        disable         = yes
        only_from       = 127.0.0.1
}
```

/etc/xinetd.d/services configuration file. Note port number which is not in /etc/services and the "only_from" clause to limit connections to local host. No server specification because it is built in to xinetd.

The /etc/hosts.allow and /etc/hosts.deny Files

- **The syntax for these files is documented in the man pages for *hosts_options*, *hosts_access*, *hosts.allow,* and *hosts.deny***
- **Basic syntax is *"daemon list: client list: options[: options]"***
- **Simple rules are:**
  - **Access granted if (daemon, client) match in *hosts.allow***
  - **Access denied if (daemon, client) match in *hosts.deny***
  - **Otherwise, access granted**
- **Simplest *hosts.deny*: "ALL: ALL", which denies everything to start**
- **Simple *hosts.allow:* "ALL: LOCAL", which matches any hostname that does not have a "." in it**
- **Between the two files, the specification can get fairly complicated, we don't have time to dive in any deeper here**

Secure Shell (SSH) Overview

- **SSH allows secure connections to other machines, solving the "my-password-is-in-clear-text-on-the-network" problem**
- **The *"sshd"* program is configured by default when you install Redhat Linux**
- **The installation or boot process will generate pairs of keys for your host one "public" and one "private"**
- **The public key may be shared with other entities and can be used to verify that data was encrypted with the private key**
- **The protocol is too involved to go into at this point, but you will notice that there are "DSA" and "RSA" keys in the */etc/ssh* directory**
- **These *ssh_host*key** keys will be used to identify the system to users that log in, and they will be added to the user's *~/.ssh/known_hosts* file. If you re-install the system, the keys will be regenerated and cause the user to be prompted. Save the keys and restore them if this is a cause for concern.**

Example SSH DSA (public key) File contents

- **Just so you're not surprised, here is what a key can look like:**


```
ssh-dss
AAAAB3NzaC1kc3MAAACBANXdDbDjSMuYevUebXkD66CeeeJlEx5tyjub9UxRgwsr2jpRe
PAw/jwZsg9BuJMfkeI6wdC1snFZdbUPH6lWsjH1Ox5ok97/zA6O/Gq0wlsmnoS0lM6nJ9g5H
Dpgju78LVyfaf5LgF3Ju4SHSGaBao16hYGvQHcQ2BsNHfBclWL5AAAAFQDLsx+tS3QlvHz
5w2XefyHcCK7CxwAAAIEAw7ft2F7Vrv/XAsykLmBrB9i9w5eDlmNnT9jNiLdvGgF+56r4iNk7
9K8Hi5MZX5LDWU1oNR1HE2V4hlPJMaO0wQPH563+opONGTssBVR0/F2wwKjaSN7sQni
BhgQm3DhwiwbCVw3ass7rB01rTnbUTQLFnyx08z4PSKnAAuggjPYAAACATCipshTA1RE
LJqUUy/KXKAdHXNoFffzA2hvAxLdhDjLJrPTBwFHdD0JiPYf878e+HlvfH7pyZSfjly5tnISI8w
Rt98coCUTdWkH1r9StW0C3sBwoPJCdZ8jJneDUXEf+cMLIEYO3iprijzNc3EP5eRdcDXSx8
SqumkAPI53o1Fw=
```


- **Don't ever share the private keys! (yours or the system's)**
- **SSH provides other commands:**
  - **ssh** Secure shell
  - **scp** Secure copy
  - **sftp** Secure FTP
- **SSH will also create a DISPLAY variable on the other side of the link that will redirect (secure) X-windows traffic to your client**

Generating Keys for SSH

- **Redhat Linux 9.0 uses SSH protocol version 2 and RSA keys by default**
- **You can generate an RSA private/public key pair by executing "*ssh-keygen –t RSA*"**
  - **Accept the default key file *~/.ssh/id_rsa,* this is the private key**
  - **The public key is written to *~/.ssh/id_rsa.pub***
  - **You can protect the keys with a pass-phrase so that they cannot be used without entering it**
- **How can you use this?**
  - **On each host, log in as root and generate RSA keys**
  - **Use *scp* to copy and append the *id_rsa.pub* file contents from all systems to the *~/.ssh/authorized_keys* file on one of the systems**
  - **Use *scp* to copy the *authorized_keys* file to the /root/.ssh directory**
  - **You can now log into any system as root and securely *ssh* to any other system without typing a password**

Other SSH Tidbits

- **There are a whole host (no pun intended) of services that can use SSH for secure access and data transport**
- **You can use SSH to open encrypted tunnels between systems and do encrypted port-forwarding**
- **These uses (and all of the others) are beyond the scope of this seminar, and are left as an exercise for the student**
- **There is a parallel-distributed shell that will allow you to execute commands on multiple systems at once with SSH, it is called "*pdsh*" and is available from http://www.llnl.gov/linux/pdsh/pdsh.html**
- **Turn off *telnet*, *rsh*, *rlogin*, and others and use SSH instead!**

Scanning Your Network

- There are a number of tools on Linux (and other systems) that may be used to do bad things
- These same tools are useful to check your systems for vulnerabilities or for troubleshooting
- The two tools that we will discuss are *"nmap"* and *"ethereal"*
- *"nmap"* is a network/system/port scanning tool
- *"ethereal"* is a network sniffer
- You can and should use these tools because the bad guys will use them *against* you
- Note: If you are in a corporate network, it may really alarm your network security people to start doing port scans. *ALWAYS* clear the activity ahead of time if there is a question. People have been fired for doing the right thing at the wrong time.

Capabilities of "nmap"

```
Nmap V. 3.00 Usage: nmap [Scan Type(s)] [Options] <host or net list>
Some Common Scan Types ('*' options require root privileges)
* -sS TCP SYN stealth port scan (default if privileged (root))
  -sT TCP connect() port scan (default for unprivileged users)
* -sU UDP port scan
  -sP ping scan (Find any reachable machines)
* -sF,-sX,-sN Stealth FIN, Xmas, or Null scan (experts only)
  -sR/-I RPC/Identd scan (use with other scan types)
Some Common Options (none are required, most can be combined):
* -O Use TCP/IP fingerprinting to guess remote operating system
  -p <range> ports to scan.  Example range: '1-1024,1080,6666,31337'
  -F Only scans ports listed in nmap-services
  -v Verbose. Its use is recommended.  Use twice for greater effect.
  -P0 Don't ping hosts (needed to scan www.microsoft.com and others)
* -Ddecoy_host1,decoy2[,...] Hide scan using many decoys
  -T <Paranoid|Sneaky|Polite|Normal|Aggressive|Insane> General timing policy
  -n/-R Never do DNS resolution/Always resolve [default: sometimes resolve]
  -oN/-oX/-oG <logfile> Output normal/XML/grepable scan logs to <logfile>
  -iL <inputfile> Get targets from file; Use '-' for stdin
* -S <your_IP>/-e <devicename> Specify source address or network interface
  --interactive Go into interactive mode (then press h for help)
Example: nmap -v -sS -O www.my.com 192.168.0.0/16 '192.88-90.*.*'
```

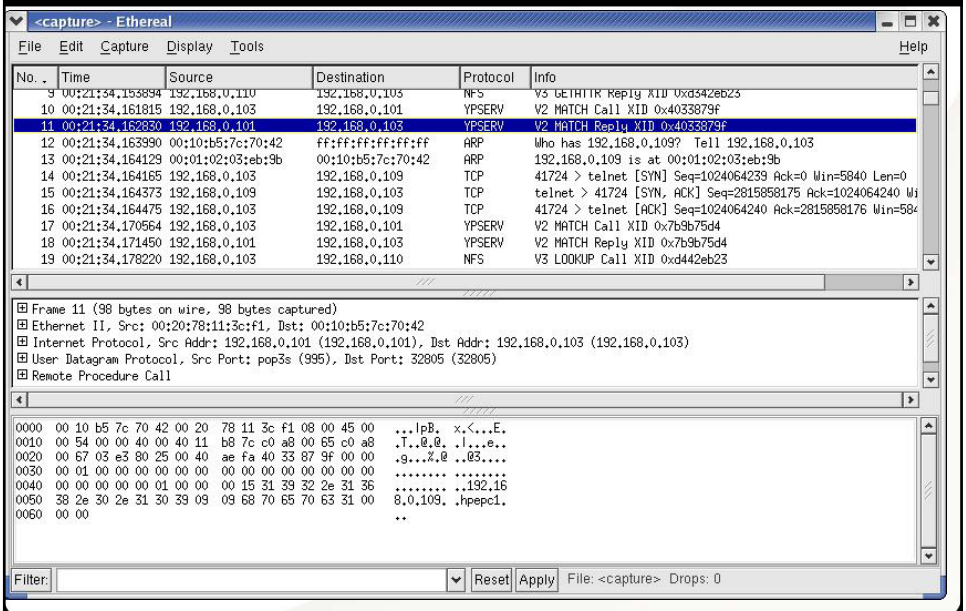
Output from An “nmap” Linux Host Scan

```
# nmap -v -sS -O  192.168.0.1/16

Host nec1 (192.168.0.101) appears to be up ... good.
Initiating SYN Stealth Scan against nec1 (192.168.0.101)
The SYN Stealth Scan took 1 second to scan 1601 ports.
For OSScan assuming that port 22 is open and port 1 is closed and neither are firewalled
Interesting ports on nec1 (192.168.0.101):
(The 1590 ports scanned but not shown below are in state: closed)
Port       State       Service
22/tcp     open        ssh
111/tcp    open        sunrpc
139/tcp    open        netbios-ssn
817/tcp    open        unknown
901/tcp    open        samba-swat
998/tcp    open        busboy
1019/tcp   open        unknown
1024/tcp   open        kdm
1026/tcp   open        LSA-or-nterm
3052/tcp   open        PowerChute
6000/tcp   open        X11
Remote operating system guess: Linux Kernel 2.4.0 - 2.5.20
Uptime 14.325 days (since Thu Jun 19 16:10:32 2003)
TCP Sequence Prediction: Class=random positive increments
                         Difficulty=5743772 (Good luck!)
IPID Sequence Generation: All zeros
```

Output from An “nmap” Windows XP Scan

```
Host hpvpw1 (192.168.0.102) appears to be up ... good.
Initiating SYN Stealth Scan against hpvpw1 (192.168.0.102)
Adding open port 445/tcp
Adding open port 139/tcp
Adding open port 3389/tcp
Adding open port 135/tcp
Adding open port 5000/tcp
Adding open port 3052/tcp
Adding open port 1025/tcp
The SYN Stealth Scan took 1 second to scan 1601 ports.
For OSScan assuming that port 135 is open and port 1 is closed and neither are firewalled
Interesting ports on hpvpw1 (192.168.0.102):
(The 1594 ports scanned but not shown below are in state: closed)
Port       State       Service
135/tcp    open        loc-srv
139/tcp    open        netbios-ssn
445/tcp    open        microsoft-ds
1025/tcp   open        NFS-or-IIS
3052/tcp   open        PowerChute
3389/tcp   open        ms-term-serv
5000/tcp   open        UPnP
Remote operating system guess: Windows 2000/XP/ME
TCP Sequence Prediction: Class=random positive increments
                         Difficulty=15737 (Worthy challenge)
IPID Sequence Generation: Incremental
```

Ethereal Network Analyzer



Linux *"netfilter"* Facility

- **Linux has an extensive ability to manipulate network packets**
- **The most recent facility is called *"netfilter"* and you can see it start up at boot time if it is enabled**
- **There is a little confusion because the facility is called *"netfilter"*, but the interface routine is *"iptables"***
- **The two terms are sometimes used interchangably**
- **One of the very useful things that *iptables*, er, *netfilter* can do is implement a firewall**
- **There are so many tools that generate the proper rules that it is impossible to count**
- **A good book on the subject is "*Linux Firewalls*" by Robert L. Ziegler, ISBN # 0-7357-1099-6**
- **Another good book is "*Linux Routers*" by Tony Mancill, ISBN # 0-13-009026-3**
- **We only have enough time to skim the surface of this topic**

Checking the State of IpTables

```
# service iptables status
      --or--
# iptables –L

Table: filter
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

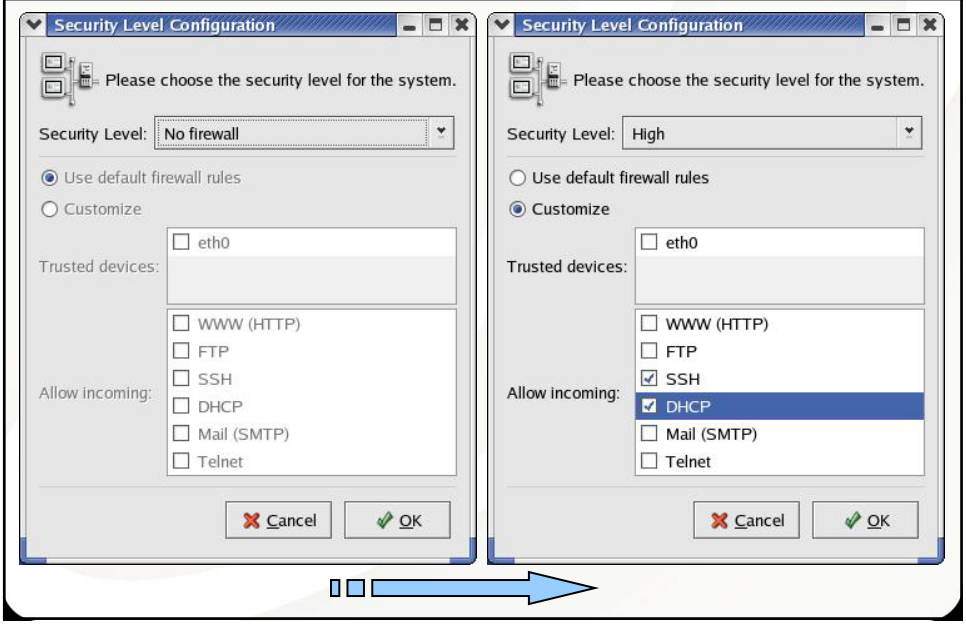
Adding a Simple Rule to a Chain

```
# iptables --table filter -A INPUT -j ACCEPT     \
>                       --in-interface eth0      \
>                       --source 192.168.0.101   \
>                       --proto tcp --dport 22

# iptables -L

Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  nec1                 anywhere          tcp dpt:ssh

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

With Enough Rules …

- You can implement a tremendous firewall …
- You can lock the world *out* of your system …
- You can also “tag” incoming traffic and shape, filter, redirect, mangle, or otherwise alter it as it passes through your system
- This allows you to implement very flexible, inexpensive routers with Linux if you desire
- The *iptables* functionality is implemented as dynamically loadable kernel modules with an extensible framework
- The facility implements “*stateful packet inspection*”, that is, it will remember which network connections you initiated and let them through the firewall while blocking outside attempts to make *new* connections
- It can examine the contents of packets to determine which to accept, reject, drop, or forward
- It is best to hunt down a firewall generator for *iptables* unless you want to learn all of the details

Redhat Firewall Configurator “*redhat-config-securitylevel*”



Firewall Configurator “gnome-lokkit”

The Generated Rules for “iptables”

 # iptables –L

```
Chain INPUT (policy ACCEPT)
target                  prot opt source          destination
RH-Lokkit-0-50-INPUT    all   --  anywhere        anywhere

Chain FORWARD (policy ACCEPT)
target                  prot opt source          destination
RH-Lokkit-0-50-INPUT    all   --  anywhere        anywhere

Chain OUTPUT (policy ACCEPT)
target                  prot opt source          destination
```

The Generated Rules for "iptables" (continued 1)

```
Chain RH-Lokkit-0-50-INPUT (2 references)
target     prot opt source          destination

ACCEPT     tcp  --  anywhere        anywhere          tcp  dpt:ssh
    flags:SYN,RST,ACK/SYN

ACCEPT     udp  --  anywhere        anywhere          udp spts:bootps:bootpc
    dpts:bootps:bootpc

ACCEPT     udp  --  anywhere        anywhere          udp spts:bootps:bootpc
    dpts:bootps:bootpc

ACCEPT     all  --  anywhere        anywhere

ACCEPT     udp  --  192.168.0.1     anywhere          udp spt:domain

REJECT     tcp  --  anywhere        anywhere          tcp
    flags:SYN,RST,ACK/SYN reject-with icmp-port-unreachable

REJECT     udp  --  anywhere        anywhere          udp reject-with icmp-port-
    unreachable
```

Firewall Tidbits

- Useful *"iptables"* commands:
  - *"service iptables status"* lists current rules
  - *"service iptables stop"* stops firewall
  - *"service iptables save"* saves current rules
  - *"service iptables start"* starts (restores rules)
  - *"service iptables panic"* stops all network activity!
- The rules from both *"gnome-lokkit"* and *"redhat-config-securitylevel"* appear to be the same
- Another firewall generator is *"gShield"* which may be plugged into your DHCP client activity to generate dynamic firewall rules. See http://muse.linuxmafia.org/gshield.html for gShield
- There are so many other firewall rule generators for *"iptables"* that it would take pages to list them

From the gShield Developer's Page

gShield

gShield is an iptables firewall for use with the 2.4.x series of the Linux kernel. It is easily configured through a single, well commented configuration file. If your needs are more minimal, see levy, a iptables ruleset generator.

Features include: support for multiple NATs, configurable public service access, access control lists, routable protection, DMZ support, port-forwarding, MAC-specific filtering, configurable outgoing filtering, blacklists, support for transparent proxy, QoS marking of common transports and more.

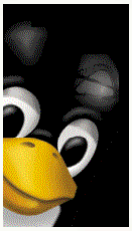
gShield in no way taunts Happy Fun Ball, and is released under the GNU General Public License (GPLv2).

http://muse.linuxmafia.org/gshield.html

Lab #3: Linux System Configuration

See Lab #3 Handout for details

Linux File Systems, RAID, and Quotas



- Working with EXT2 and EXT3
- Other File Systems
- Software RAID
- Overview of Quotas

Working with EXT2 and EXT3

- **The default Redhat file system, EXT2 was upgraded at Redhat release 7.2 to EXT3**
- **The EXT2 file system is still available**
- **EXT3 Provides:**
  - Journaling to reduce e2fsck (fsck) times in the event of a failure
  - Easy transition from EXT2 to EXT3 (and back)
  - Control over the amount and type of data being journaled at mount time
- **EXT2/EXT3 tools:**
  - resize2fs
  - e2fsck
  - tune2fs
  - mke2fs

Creating and Mounting an EXT3 File System

- "*mke2fs –b 4096 –j –J size=400 –L MYLABEL –O sparse_super –T largefile4 –v –m 1 /dev/sda1*"

  makes an EXT3 file system with a journal sized at 400 MB, a block size of 4KB, one inode for every 4 MB of file data, reserving 1% of the disk for the root user, and limited duplicate superblocks on the device partition /dev/sda1

- "*mount –t ext3 –o data=ordered /dev/sda1 /mnt/test"*

  mounts the previously created file system with all data written to the file system before the metadata is committed to the journal

- **The settings for the EXT3 data=<mode> are either *journal*, *ordered (default),* or *writeback*. "*journal*" writes all data to the journal before committing it to the file system, "*ordered*" writes data to the file system before writing meta-data to journal, and "*writeback*" does not preserve data and meta-data ordering**

Using "tune2fs"

- ***"tune2fs –l /dev/sda1"* will list the contents of the superblock**

- ***"tune2fs –L MYLABLE"* will set the file system label value, which can be used by mount, fsck, and /etc/fstab instead of the block special device by specifying *"LABEL=MYLABLE"* in its place**

- **Note that Linux has the habit of shifting device names when new devices are added to the system, the ability to use "LABEL=<value>" for the device special file in the mount request removes the dependency on the device name (i.e. /dev/sda1)**

Working With the “chattr” Command and EXT*

- **Usage: chattr [–RV] --+=ASDacdijsTtu –v version files …**
- **Attributes that may be set on an EXT* file system:**
  - **A** Don’t update atime on access
  - **S** Synchronous updates
  - **D** Synchronous updates of directory (2.5.19 and later)
  - **a** Append only
  - **c** Compress
  - **d** No dump
  - **i** Immutable (cannot be deleted, written to, or linked)
  - **j** Data journaling
  - **s** Secure deletion
  - **T** Top of directory hierarchy (2.5.19 and later)
  - **t** No tail merging
  - **u** Undeletable

File Systems On Redhat (See “man 5 fs”)

- **Commonly used file systems:**
  - **ext2/ext3** default Redhat fs
  - **jfs** journaled fs from IBM
  - **umsdos** DOS fs, plus UID/GID, permissions, long filenames
  - **msdos** DOS fs, 8.3 file names
  - **vfat** later version of FAT, long names
  - **reiserfs** popular journaled fs for Linux
  - **ISO9660** CD-ROM/DVD fs includes Sierra and Rockridge
  - **xfs** journaled fs from SGI, must be added to Redhat
  - **smb** Microsoft server message block, CIFS and Samba
  - **nfs** network file system from Sun Microsystems
- **Less commonly used**
  - **cramfs** read-only compressed fs
  - **minix** first file system to run under Linux
  - **xiafs** extension of minix
  - **ext** first extended fs, extension of minix
  - **ncpfs** uses NCP protocol for Novell Netware
  - **sysv** Xenix fs, SystemV/386 fs, Coherent fs

Linux Software RAID (Multi-Disk or MD)

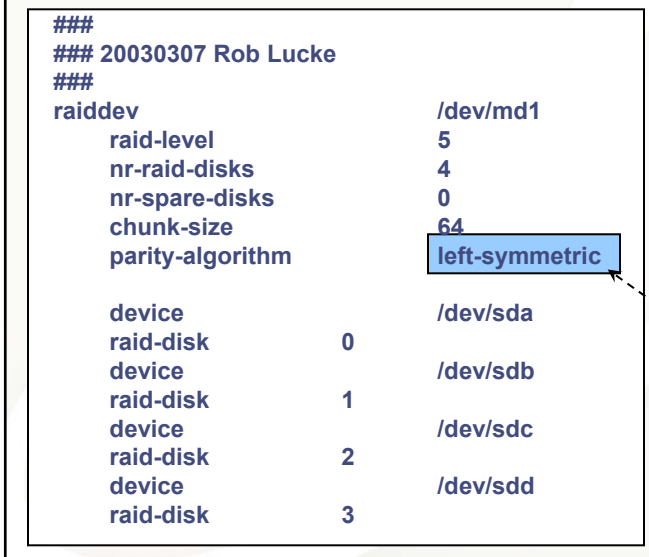
- **The software RAID feature of Linux is handy and easy to use**
- **Remember that all parity calculations and I/O are being performed by the local CPU (i.e. they are not hidden inside dedicated RAID box)**
- **Linux Software RAID modes:**
  - Linear — Concatenates disks — (0% space overhead)
  - 0 — Striped data — (0% space overhead)
  - 1 — Mirrored data — (100% space overhead)
  - 4 — One disk for parity (infrequently used) — (33% space overhead)
  - 5 — Parity on multiple disks — (~20% space overhead)

Creating a Software RAID Device

- **Create /etc/raidtab entry**
- ***"mkraid /dev/md<nn>"***
- ***"raidstart /dev/md<nn>"***
- **Create file system**
- **Mount device**

- **See *http://en.tldp.org/HOWTO/Software-RAID-HOWTO-6.html* for information on how to recover from RAID failures**
- **Commands**
  - ***mkraid*** — *create MD device from raidtab*
  - ***raidstart*** — *start an MD device*
  - ***raidstop*** — *stop an MD device*
  - ***raidhotadd*** — *recovery tool, add disk to array*
  - ***raidhotremove*** — *recovery tool, remove disk from array*
  - ***"cat /proc/mdstat"*** — *get live MD device status*
  - ***lsraid –R –a /dev/md0*** — *recreate raidtab from live array*

Software RAID Table from Rob's Server



```
###
### 20030307 Rob Lucke
###
raiddev                         /dev/md1
      raid-level                5
      nr-raid-disks             4
      nr-spare-disks            0
      chunk-size                64
      parity-algorithm          left-symmetric

      device                    /dev/sda
      raid-disk           0
      device                    /dev/sdb
      raid-disk           1
      device                    /dev/sdc
      raid-disk           2
      device                    /dev/sdd
      raid-disk           3
```

You can see the speed calculation for this algorithm in the *"dmesg"* output from the system boot, see next slide

RAID Startup and Checksumming Speed Calculation

```
[…]

raid5: measuring checksumming speed
   8regs     :   731.136 MB/sec
   32regs    :   372.736 MB/sec
   pII_mmx   :   888.832 MB/sec
   p5_mmx    :   931.840 MB/sec
raid5: using function: p5_mmx (931.840 MB/sec)
md: raid5 personality registered as nr 4
Journalled Block Device driver loaded
md: Autodetecting RAID arrays.
md: autorun ...
md: ... autorun DONE.

[…]
```

Functioning Software RAID Status from /proc/mdstat

 # cat /proc/mdstat

```
Personalities : [raid5]
read_ahead 1024 sectors
md1 : active raid5 sdd[3] sdc[2] sdb[1] sda[0]
      215061888 blocks level 5, 64k chunk, algorithm 2 [4/4] [UUUU]

unused devices: <none>
```

The "U" indicates "up-to-date" or some such status. If the array is updating parity, the "U" will change to "_".

Some Configuration Notes – RAID and EXT

- **Note that if you are using RAID 5, there is a special option to the "*mke2fs*" command to set the file system stride to match the array chunk size:**

  "–R stride=<fs_blocks_per_chunk>"

- **Example: an ext3 file system created with 4KB (the maximum) block size and a Linux software RAID 5 device with a 128 KB chunk size. The option for "*mke2fs*" would be: *"-R stride=32"* because 32x4=128**

- **The "RAID HowTo" and several other sources report that setting this properly is important but don't quantify the effect**

- **The author has not verified the performance impact of the different settings on file system performance (yet)**

Overview of Linux Disk Quotas

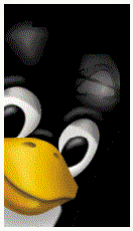
- Linux disk quotas are calculated on a per file-system basis
- Quotas are enabled for local file systems at file-system mount time
- Quotas may be enabled per user, per group, or both
- The mount options are *"usrquota"* and *"grpquota"*
- Once the file-systems are mounted with the quota options, current usage statistics are generated with the *"quotacheck"* command
- The system manager establishes individual settings for groups and users with the *"edquota"* command, which uses the text editor defined by the EDITOR environment variable to modify the settings
- As part of on-going maintenance, *"quotacheck"* should be run periodically to update the quota information
- Use the *"repquota"* command to report on the current quota status

Linux Quota Commands

- Linux quota commands:
  - quotacheck — check and update quota statistics
  - edquota — edit quota settings
  - repquota — report quota usages per filesystem
  - quota — report quota information per user or group
  - quotaon — enable file-system quotas
  - quotaoff — disable file-system quotas (link to quotaon)
  - quotastats — report quota system performance statistics
- Linux quota files:
  - aquota.user — per file-system user quota information
  - aquota.group — per file-system group quota information

Linux DHCP, NFS, NIS, and Samba

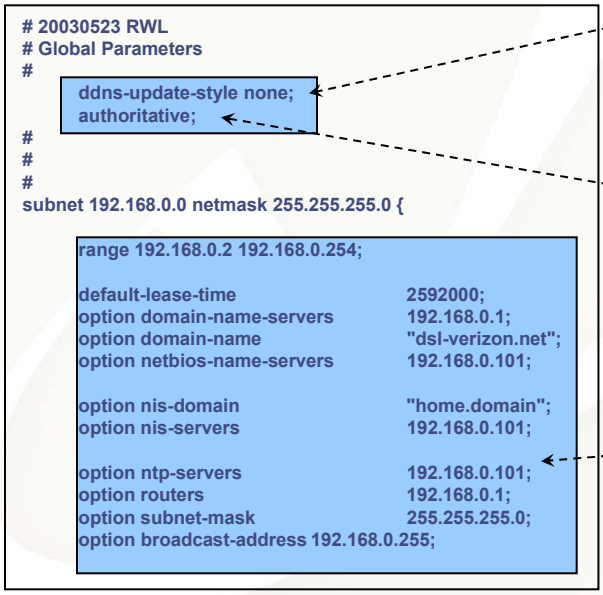
- Configuring DHCP
- Configuring NFS
- Configuring NIS
- Samba and Swat



Configuring Linux DHCP Services

- The DHCP server can be a big help to you as a system manager it can “fill in” lots of client configuration information for you
- You can specify default parameters to all client systems serviced by the DHCP server
- The server configuration file is */etc/dhcpd.conf*
- The DHCP client (dhcpcd) will fill in all of the information in configuration files on the system at boot time
- To see the client information for DHCP, look at the */var/lib/dhcp/dhclient-eth0.leases* file
- See man pages for *dhcp.conf, dhcp.leases, dhcp-options,* and *dhclient.conf*

Example DHCP Configuration File

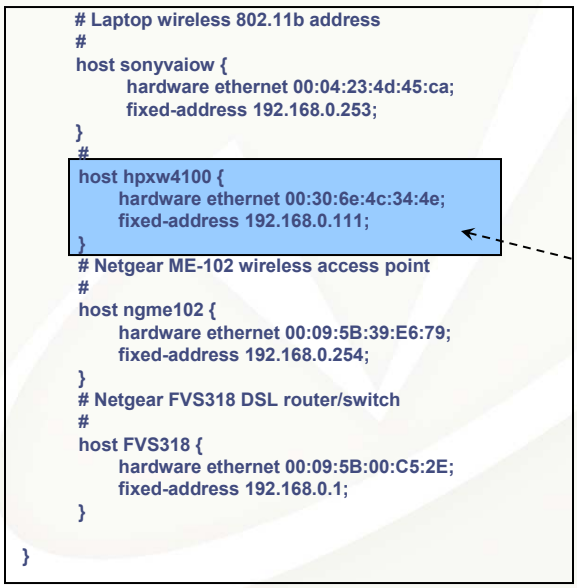


DHCP can update dynamic DNS if you have that service. It will register the host and IP information into the DNS server.

If you have another DHCP server in your network, make sure to set “non-authoritative” or you will NAK the requests of other systems and your IT guys will not like you any more.

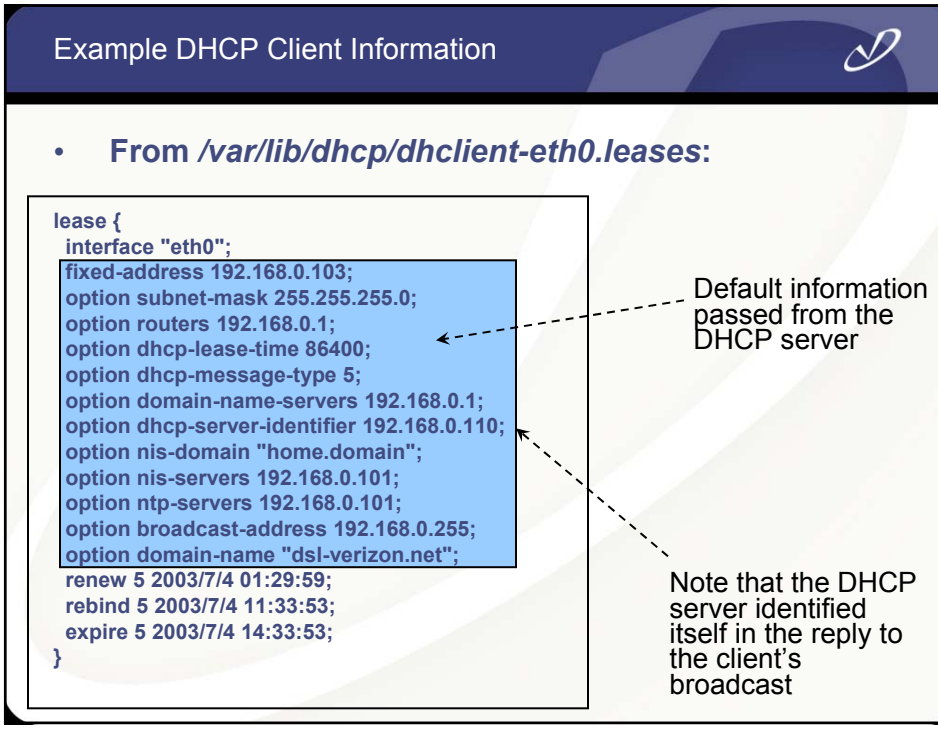
Default options sent to all DHCP clients for the subnet.

Example DHCP Configuration File (continued 1)



Entry specifying hostname, MAC address, and IP address for one system

Example DHCP Client Information



- **From */var/lib/dhcp/dhclient-eth0.leases*:**

```
lease {
  interface "eth0";
  fixed-address 192.168.0.103;
  option subnet-mask 255.255.255.0;
  option routers 192.168.0.1;
  option dhcp-lease-time 86400;
  option dhcp-message-type 5;
  option domain-name-servers 192.168.0.1;
  option dhcp-server-identifier 192.168.0.110;
  option nis-domain "home.domain";
  option nis-servers 192.168.0.101;
  option ntp-servers 192.168.0.101;
  option broadcast-address 192.168.0.255;
  option domain-name "dsl-verizon.net";
  renew 5 2003/7/4 01:29:59;
  rebind 5 2003/7/4 11:33:53;
  expire 5 2003/7/4 14:33:53;
}
```


Setting Up the DHCP Server, "dhcpd"

- **Make sure you installed the DHCP server rpm file …**
- **Create the /etc/dhcpd.conf file**
- **Execute *"chkconfig dhcpd on"* to create startup links**
- **Execute *"service dhcpd start"* to start the server**
- **Check the */var/log/messages* file for DHCP server logging messages**
- **Edit the */etc/dhcpd.conf* file as necessary**
- **Execute *"service dhcpd restart"* to stop and restart the DHCP server**
- **Note that */etc/sysconfig/dhcpd* contains a variable definition, DHCPDARGS, that is sourced by the startup – this is useful for setting options:**

  DHCPDARGS="eth0"

- **Note: There is a DHCP protcol relay agent *"dhcrelay"* that can forward DHCP requests from a subnet to a server with no direct connection, see *"man dhcrelay"* for information**

Tips on Configuring NFS Client and Server

- The NFS client software on Linux seems to work okay with NFS PV2 and PV3 (without large 32KB blocks?)
- The NFS server software on Linux seems to have problems, maybe related to the buffer cache and does not yet support all of the PV3 stuff
- Some of the statistics we are used to from *"nfsstat"* are not available – darn it, we have to fly blind
- Server support of NFS over TCP/IP and large blocks (32KB) may not be supported without kernel patches (experimental patches, that is)
- Careful! The default values for *"rsize"* and *"wsize"* is 1024 bytes (1 KB) – you have to set the values in the mount/automount options fields
- If you have a firewall, you must ensure that the *portmapper* service is accessible by your NFS clients or the network is filled with silence
- This all changes on a weekly basis, it seems …

Configuring Linux NFS Client Software

- **Useful NFS client commands:**
  - *"service autofs start"*
  - *"service autofs status"*
  - *"service autofs reload"*
  - *"service autofs stop"*
  - *"chkconfig autofs on"*
- **There is an */etc/auto.misc* file loaded by default that has some interesting Linux twists on autofs mounts**
- **The "normal" *auto.master* behavior exists, either a local file or from NIS**

Configuring Linux NFS Server

- **Useful commands:**
  - *"chkconfig nfs on"*
  - *"chkconfig nfslock on"*
  - *"chkconfig portmap on"*
  - *"service nfs start"*
  - *"service nfslock start"*
  - *"service portmap start"*
- **Note that the *"portmap"* service must be visible through any firewall for NFS server to work**
- **Also, the daemons that are associated with Sun RPC all start with the *"rpc."* prefix, for example "rpc.mountd"**
- **Redhat's *"portmap"* is compiled to use TCP wrappers, so *hosts.allow* and *hosts.deny* work in addition to the controls in the */etc/exports* file**
- **As an aside, example after example shows the use of the *"soft"* option for mounts: *Do not use this unless you *want* data corruption*!**

Sun RPC Daemons from "pmap_dump" Command

#pmap_dump

100000	2	tcp	111	portmapper
100000	2	udp	111	portmapper
100024	1	udp	32768	status
100024	1	tcp	32768	status
100007	2	udp	702	ypbind
100007	1	udp	702	ypbind
100007	2	tcp	705	ypbind
100007	1	tcp	705	ypbind
391002	2	tcp	32769	sgi_fam
100011	1	udp	859	rquotad
100011	2	udp	859	rquotad
100011	1	tcp	862	rquotad
100011	2	tcp	862	rquotad

100003	2	udp	2049	nfs
100003	3	udp	2049	nfs
100021	1	udp	32770	nlockmgr
100021	3	udp	32770	nlockmgr
100021	4	udp	32770	nlockmgr
100005	1	udp	32771	mountd
100005	1	tcp	32770	mountd
100005	2	udp	32771	mountd
100005	2	tcp	32770	mountd
100005	3	udp	32771	mountd
100005	3	tcp	32770	mountd

Example “auto.misc” File

```
# $Id: auto.misc,v 1.2 1997/10/06 21:52:04 hpa Exp $
# This is an automounter map and it has the following format
# key [ -mount-options-separated-by-comma ] location
# Details may be found in the autofs(5) manpage

cd        -fstype=iso9660,ro,nosuid,nodev        :/dev/cdrom

# the following entries are samples to pique your imagination
#linux          -ro,soft,intr              ftp.example.org:/pub/linux
#boot           -fstype=ext2               :/dev/hda1
#floppy         -fstype=auto               :/dev/fd0
#floppy         -fstype=ext2               :/dev/fd0
#e2floppy       -fstype=ext2               :/dev/fd0
#jaz            -fstype=ext2               :/dev/sdc1
#removable      -fstype=ext2               :/dev/hdd
```

My “autofs” Map Files

```
# ypcat -k auto.master

/data   auto.data    -vers=3,rsize=32768,wsize=32768
/home auto.home  -vers=3,rsize=32768,wsize=32768

# ypcat -k auto.data

music      nec2:/bigdata/SambaShare/Music
software nec2:/bigdata/SambaShare/Software
pictures  nec2:/bigdata/SambaShare/Pictures
bigdata   nec2:/bigdata

# ypcat -k auto.home

teri  nec2:/bigdata/LocalHomes/Teri
rob  nec2:/bigdata/LocalHomes/Rob
```

NFS Server Commands from /etc/init.d/nfs

```
# Check for and source configuration file otherwise set defaults
# TUNE_QUEUE: controls whether to up the size of input queues
[ -f /etc/sysconfig/nfs ] && . /etc/sysconfig/nfs

[ -z "$MOUNTD_NFS_V2" ] && MOUNTD_NFS_V2=auto
[ -z "$MOUNTD_NFS_V3" ] && MOUNTD_NFS_V3=auto

# Number of servers to be started by default
[ -z "$RPCNFSDCOUNT" ] && RPCNFSDCOUNT=32

# Remote quota server
[ -z "$RQUOTAD" ] && RQUOTAD=`type -path rpc.rquotad`

# Get the initial values for the input sock queues
# at the time of running the script.
if [ "$TUNE_QUEUE" = "yes" ]; then
   RMEM_DEFAULT=`/sbin/sysctl -n net.core.rmem_default`
   RMEM_MAX=`/sbin/sysctl -n net.core.rmem_max`
   # 256kb recommended minimum size based on SPECsfs NFS benchmarks
   [ -z "$NFS_QS" ] && NFS_QS=262144
fi
```

NFS Server Commands from /etc/init.d/nfs

```
start)
      # Start daemons.
      # Apply input queue increase for nfs server
      if [ "$TUNE_QUEUE" = "yes" ]; then
         /sbin/sysctl -w net.core.rmem_default=$NFSD_QS >/dev/null 2>&1
         /sbin/sysctl -w net.core.rmem_max=$NFSD_QS >/dev/null 2>&1
      fi
      action $"Starting NFS services: " /usr/sbin/exportfs -r
      if [ -n "$RQUOTAD" -a "$RQUOTAD" != "no" ]; then
         echo -n "Starting NFS quotas: "
         daemon rpc.rquotad
         echo
      fi
      echo -n $"Starting NFS daemon: "
      daemon rpc.nfsd $RPCNFSDCOUNT
      echo
```

Two important “tunes” for an NFS server are done here.
1) increase the network memory allocation pool for incoming packets and
2) start the “nfsd” threads to answer incoming requests

NFS Server – Create /etc/sysconfig/nfs

- You should create the */etc/sysconfig/nfs* file for your NFS server and add the following lines:

```
TUNE_QUEUE=yes
RPCNFSDCOUNT=<number of threads to start>
```

- This is much safer than modifying the */etc/init.d/nfs* script, which is what you used to have to do with earlier revisions of Linux
- How many threads *should* you start? Well, in the words of a consultant, "That depends …" on:
  - The number of client requests
  - The network link to the server (is it 100baseT or GbE?)
  - Whether you are seeing NFS socket overflows (port 2049)

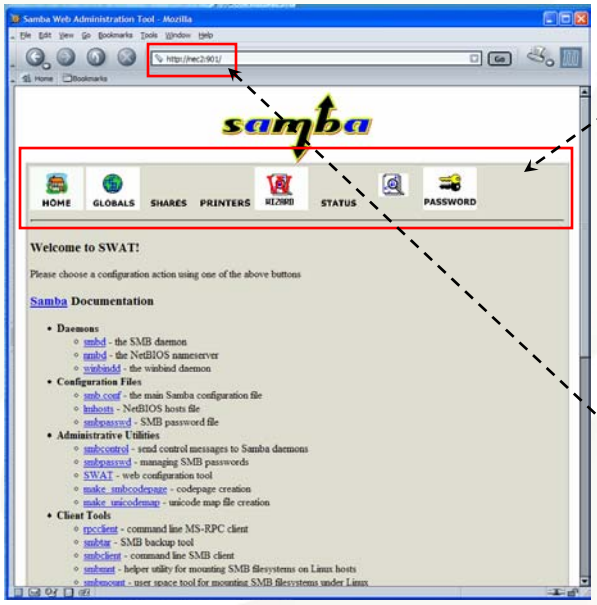
Configuring NIS

- NIS works much as you would expect
- For clients, there is a */etc/yp.conf* file that determines how to find the server
- Setting NISDOMAIN="<domain>" in the */etc/sysconfig/network* script will properly execute the "*domainname*" command at boot time
- *"chkconfig ypbind on" "service ypbind start"* will fire up the client side
- For the client, you must make sure that your password encoding (remember MD5?) matches whatever the server is providing – non-Linux servers will not provide what Linux is expecting
- If you have a Linux NIS server, you can merge the shadow file and the password file before creating the *passwd* map – there is an option in */var/yp/Makefile* for that, but it defeats the security
- Your Linux NIS server must *provide* password encoding that any client will expect – if the system type doesn't handle MD5, you must use the lowest common denominator

An Overview of Samba

- **Samba provides server message block (SMB) and common internet file system (CIFS) protocol to Microsoft ® Windows ® clients**
- **Samba works *very* well and also provides domain controller functionality that will map NIS and/or passwd information to the proper authentication**
- **Samba may be used to export data from your NFS server so that Microsoft Windows clients can see the same data as their Linux bretheren**
- **Samba can be quite complex to configure if you start using all of the features, but is not too hard for simple situations**
- **Note that there is an O'Reilly book on Samba that is invaluable if you will be setting up large or complex environments**
- **We will show some initial ways to get Samba up and running so you can experiment …**

Introduction to Samba Web Administration Tool (SWAT)

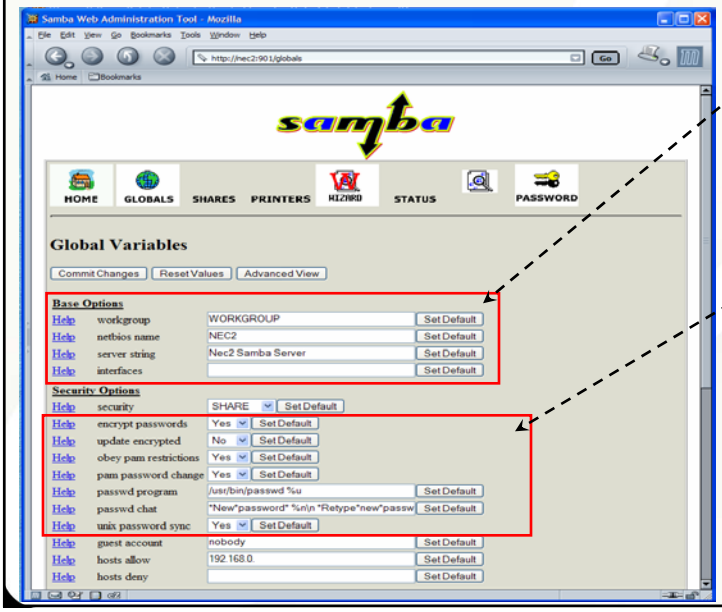


The SWAT command toolbar gives you access to configuration information in the Samba configuration file in an easy-to use graphical interface

There are, of course, security issues with opening this service, but the *xinetd* configuration file for Swat lets you control where the tool is run from with the *only_from* option.

Swat is run from port 901 by entering the browser URL *http://<server>:901*

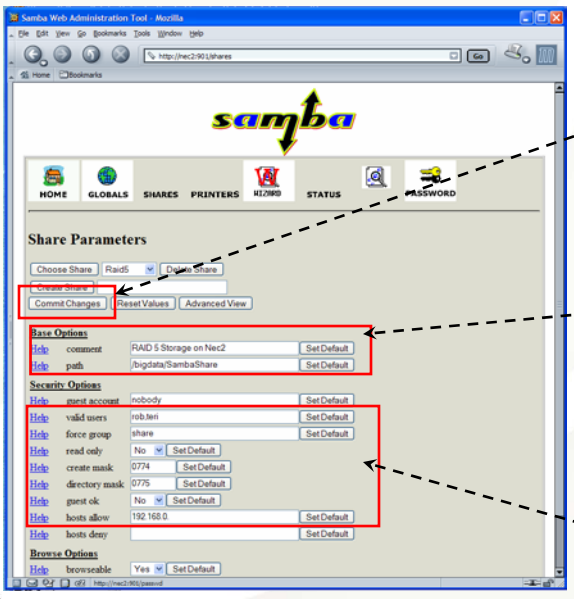
The Swat Globals Form



Settings that determine how the share appears to Windows clients

Password options that determine interoperability with NIS or the */etc/passwd* file

A Swat Share Form



Always make sure to “Commit Changes” after making changes to the form information – otherwise you will lose your work … bad form!

Settings for the base path to the share’s storage and the share comment seen by Windows clients

File access, valid user list, default file and directory create masks, default ownership, and the hosts that are allowed to connect

Swat Creates the /etc/samba/smb.conf File

```
# Samba config file created using SWAT
# Global parameters

[global]
        netbios name = NEC2
        server string = Nec2 Samba Server
        security = SHARE
        encrypt passwords = Yes
        obey pam restrictions = Yes
        pam password change = Yes
        passwd program = /usr/bin/passwd %u
        passwd chat = *New*password* %n\n *Retype*new*password* %n\n
        *passwd:*all*authentication*tokens*updated*successfully*
        unix password sync = Yes
        log file = /var/log/samba/%m.log
        max log size = 0
        socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192
        local master = No
        dns proxy = No
        wins server = 192.168.0.101
        hosts allow = 192.168.0.
        printing = lprng
```

Swat Creates the /etc/samba/smb.conf File (continued 1)

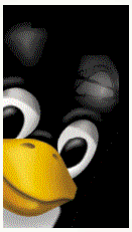
```
[printers]
        comment = All Printers
        path = /var/spool/samba
        printable = Yes
        browseable = No

[Raid5]
        comment = RAID 5 Storage on Nec2
        path = /bigdata/SambaShare
        valid users = rob,teri
        force group = share
        read only = No
        create mask = 0774
        directory mask = 0775

[Music]
        comment = Nec2 RAID5 Music Storage
        path = /bigdata/SambaShare/Music
        valid users = rob teri
        force user = rob
        force group = share
        read only = No
```

Samba Tidbits

- The Samba configuration information is underneath the */etc/samba* directory
- Files in this directory include:
  - *smbpasswd* — Contains UID and password for users
  - *smbusers* — Maps Windows users to Unix names
  - *lmhosts* — Windows hosts for NetBIOS
  - *smb.conf* — Samba configuration file
- The */var/log/samba* directory contains log files for *smbd, nmbd,* and clients that have connected (or attempted to connect)
- For each client connection, a new *smbd* process is created
- The *nmbd* daemon handles the netBIOS name service requests for Samba clients

Building the Linux Kernel and Modules

- Basic Steps
- Kernel Building Tips
- Running the Configuration Tool



Basic Kernel-Building Steps

- Note: Redhat Linux is compiled for maximum compatibility, for an i386 processor! Later distributions *may* have CPU optimizations available (i.e. kernel-2.4.20-9.i686.rpm)
- To build the kernel:

a) cd /usr/src/linux2.4 (a link to the current version)

b) Build the graphical configurator:
   - make xconfig –or– (X-windows configurator tool)
   - make menuconfig (VGA mode configurator tool)

c) Save the current configuration to a file!
   1. The current configuration is kept in /usr/src/linux2.4/.config
   2. The default Redhat configurations are kept in /usr/src/linux/configs

d) Make changes using the configurator

e) Save new configuration to a file!

f) Build the kernel (Be prepared for a wait …)
   - make deps (build dependencies)
   - make bzImage (a compressed kernel)
   - make modules

!! See Warnings on Next Slide !!

Basic Kernel-Building Steps WARNINGS!

- You are operating on the source tree for your current kernel
- At each and every step you need to make sure you can back out *any* changes you have made
- The top-level *Makefile* has been modified by Redhat to save you from yourself: *"custom"* has been added to the end of the kernel version to prevent overwriting existing module directories or kernels

```
VERSION = 2
PATCHLEVEL = 4
SUBLEVEL = 20
EXTRAVERSION = -18.9custom
KERNELRELEASE=$(VERSION).$(PATCHLEVEL).$(SUBLEVEL)$(EXTRAVERSION)
```

Kernel Building Basics (continued 1)

a) Back up the current module information!
b) Backup the current kernel and system files!
c) Always make a boot disk! (/sbin/mkbootdisk)
d) make modules_install (install new modules)
e) Move kernel and other system files from /usr/src/linux/arch/i386/boot, this will be called "bzImage"
   - Can use /sbin/installkernel or /sbin/new-kernel-package
f) Run LILO to update map information (offset into partition for kernel file) in boot record (not necessary with GRUB, just modify the /boot/grub/grub.conf file)
g) Reboot and test

- Naming conventions are important
  a) Add a "test" label to the boot configuration file?
  b) With links and "generic" boot configuration labels, you can implement a relatively fail-safe way of installing new kernels

Kernel Building Tips

- **Tip**: The /usr/src/linux2.4/Makefile contains lines like the following:

  ```
  VERSION = 2
  PATCHLEVEL = 4
  SUBLEVEL = 18
  EXTRAVERSION = -3
  ```

- To keep from screwing up your current installation when building a new kernel and associated modules, change the EXTRAVERSION variable to something like

  "EXTRAVERSION=-3custom"

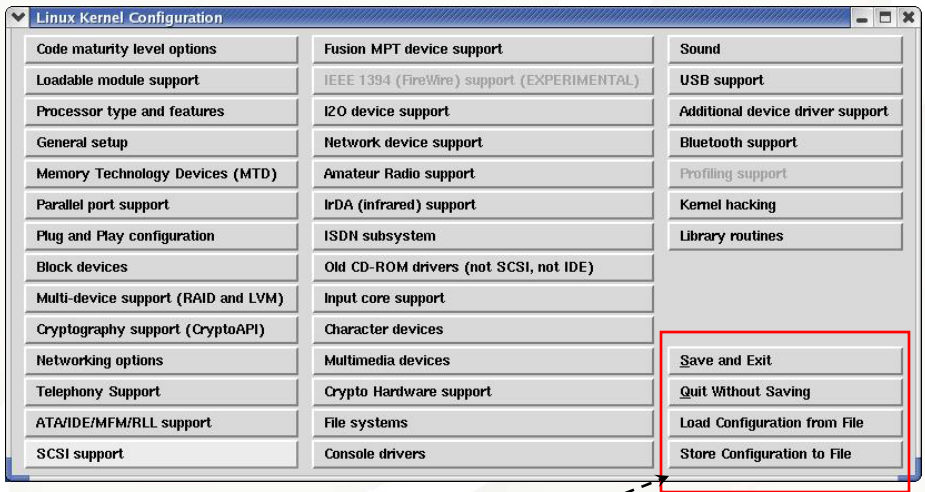
- This will allow you to keep your modules, initrd, and kernel separate from your working version (/usr/lib/modules-2.4.18-3custom, /boot/vmlinux-2.4.18-3custom, /boot/initrd-2.4.18-3custom, etc.)

- **Tip**: If you have more than one CPU, you can type "make –j <N> bzimage modules" to run N simultaneous compile jobs

Running the Kernel Configuration Application

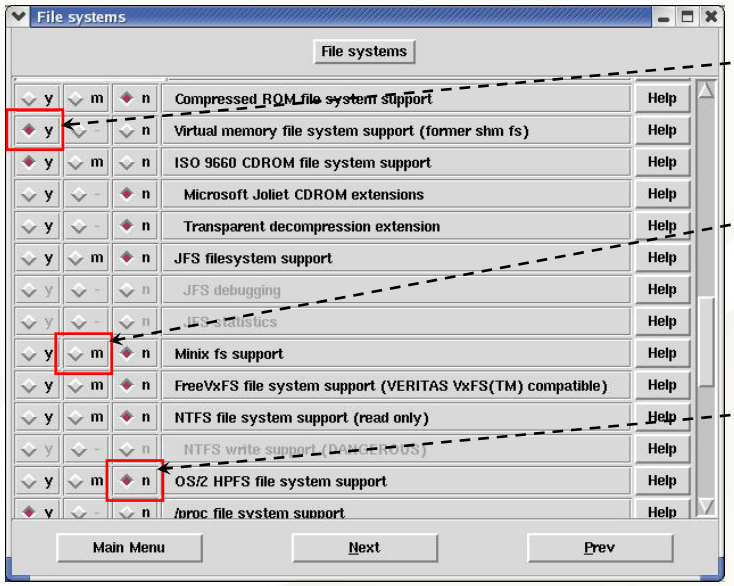
- **"cd /usr/src/linux2.4"; make xconfig (or menuconfig)**
- **Always save a new copy of the current configuration to work on so you will not screw-up the *original* configuration information**
- **Remember to load/save the configuration changes if you want to keep them across rebuilds**
- **Redhat keeps kernel configuration files in the */usr/src/linux-2.4/configs* directory**
- **Always save a new copy of the current configuration to work on so you will not screw-up the *original* configuration information**

Main Kernel Configuration Menu (xconfig)



Remember what you are supposed to do here?

Kernel Configuration File System Sub-Menu (xconfig)

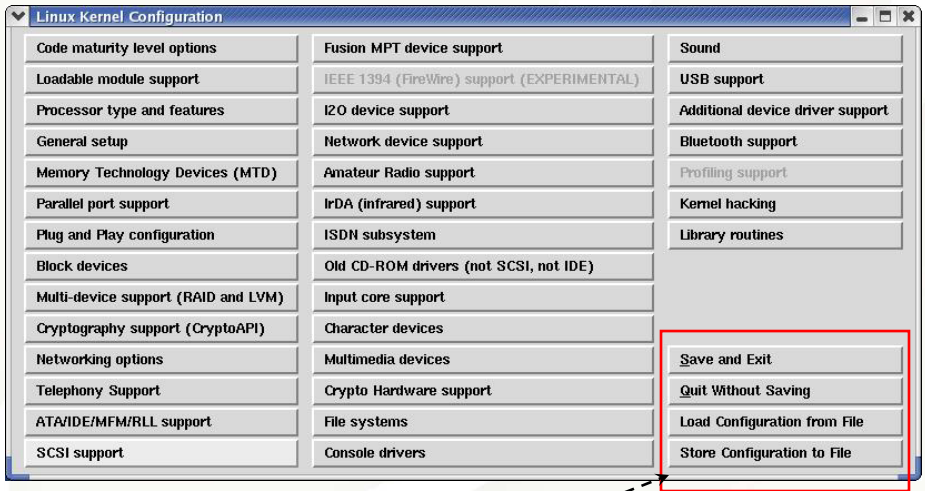


Functionality statically built into the kernel

Functionality built into a dynamically loadable module

Functionality not used at all

Main Kernel Configuration Menu (xconfig)



Remember what you are supposed to do here?

Kernel Installation

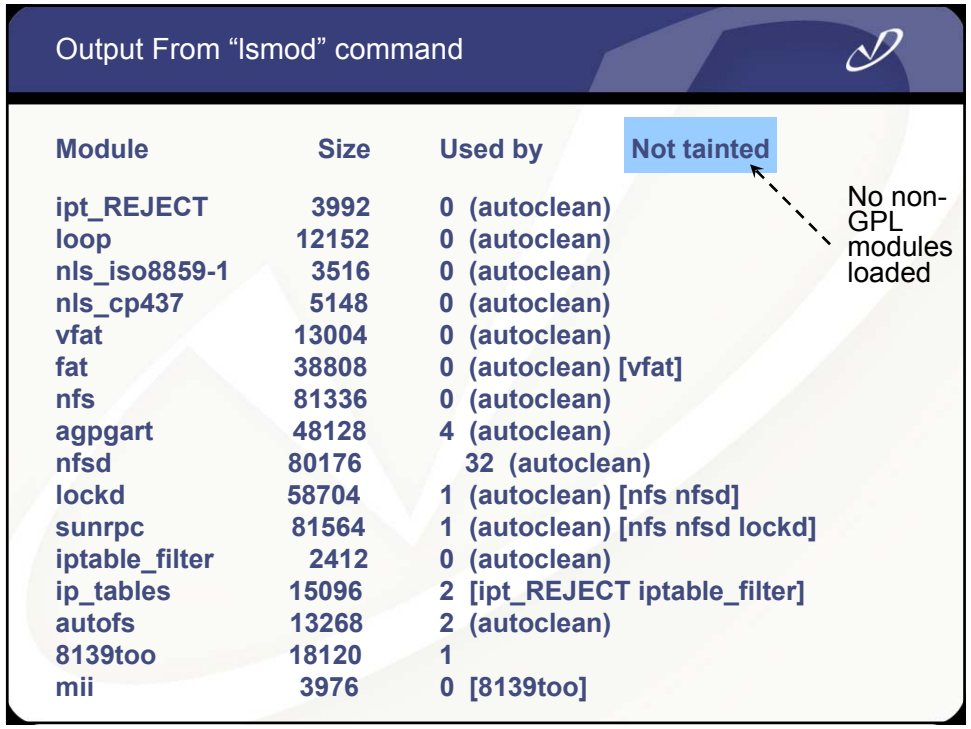
- **Once you have configured and built your kernel and modules, you will need to:**
  - **Create an initrd image**
  - **Install the kernel and the initrd**
  - **Create the module dependencies**
  - **Install the kernel information in the boot loader config**
- **The *<u>/sbin/new-kernel-pkg</u>* script will do this for you, or at least give you an idea of what you need to do manually**

```
# new-kernel-pkg --help

Usage: new-kernel-pkg [-v] [--mkinitrd] [--rminitrd]
    [--initrdfile=<initrd-image>] [--depmod] [--rmmoddep]
    <--install | --remove> <kernel-version>
    (ex: new-kernel-pkg --mkinitrd --depmod --install 2.4.7-2)
```

Linux Dynamic Kernel Modules

- **When configuring a kernel, you have the choice to build things in (monolithic) or to dynamically load modules (slower) but much easier ...**
- **The "*lsmod*" command lists all loaded modules and their current state**
- **The startup operation (kudzu) will detect hardware and load the proper module (usually) for any *supported* hardware**
- **At boot time, the startup process runs "*depmod -a*", which builds a dependency list for all modules**
- **The */etc/modules.conf* file contains commands for the module commands "*insmod*" and "*modprobe*"**
- **Unused modules are unloaded until needed again, and show up as status "autoclean"**
- **Modules are located under *"/lib/modules/<kernel_rev>"***
- **Modules are generally *not* compatible across kernel revisions!**
- **The kernel is smart enough to locate the proper modules to match its version, provided they are in the standard location (*/lib/modules*) Example: */lib/modules/2.4.20-9smp***

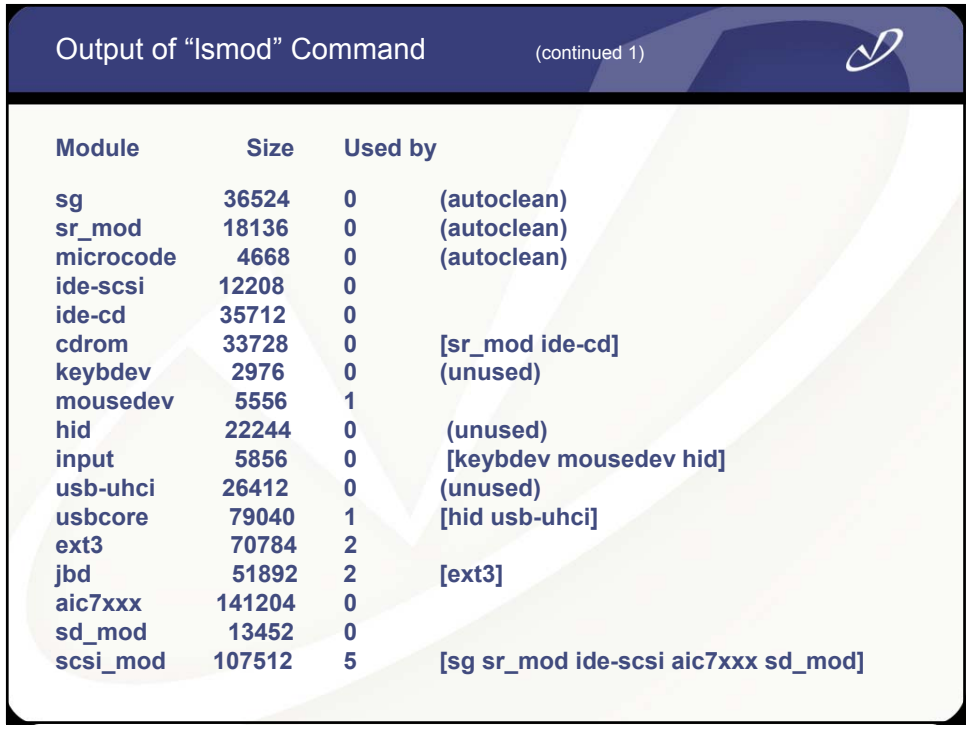
Output From “lsmod” command



Module	Size	Used by	Not tainted
ipt_REJECT	3992	0 (autoclean)	
loop	12152	0 (autoclean)	
nls_iso8859-1	3516	0 (autoclean)	
nls_cp437	5148	0 (autoclean)	
vfat	13004	0 (autoclean)	
fat	38808	0 (autoclean) [vfat]	
nfs	81336	0 (autoclean)	
agpgart	48128	4 (autoclean)	
nfsd	80176	32 (autoclean)	
lockd	58704	1 (autoclean) [nfs nfsd]	
sunrpc	81564	1 (autoclean) [nfs nfsd lockd]	
iptable_filter	2412	0 (autoclean)	
ip_tables	15096	2 [ipt_REJECT iptable_filter]	
autofs	13268	2 (autoclean)	
8139too	18120	1	
mii	3976	0 [8139too]	

No non-GPL modules loaded

Output of “lsmod” Command (continued 1)



Module	Size	Used by	
sg	36524	0	(autoclean)
sr_mod	18136	0	(autoclean)
microcode	4668	0	(autoclean)
ide-scsi	12208	0	
ide-cd	35712	0	
cdrom	33728	0	[sr_mod ide-cd]
keybdev	2976	0	(unused)
mousedev	5556	1	
hid	22244	0	(unused)
input	5856	0	[keybdev mousedev hid]
usb-uhci	26412	0	(unused)
usbcore	79040	1	[hid usb-uhci]
ext3	70784	2	
jbd	51892	2	[ext3]
aic7xxx	141204	0	
sd_mod	13452	0	
scsi_mod	107512	5	[sg sr_mod ide-scsi aic7xxx sd_mod]

Example /etc/modules.conf File

```
alias           eth0                8139too
alias           scsi_hostadapter    aic7xxx
alias           sound-slot-0        i810_audio
post-install    sound-slot-0        /bin/aumix-minimal -f /etc/.aumixrc -
    L  \                                    >/dev/null 2>&1 || :
pre-remove      sound-slot-0        /bin/aumix-minimal -f /etc/.aumixrc -
    S  \                                    >/dev/null 2>&1 || :
alias           usb-controller      usb-uhci
```

Alias commands map "internal" names to actual physical module names (minus the ".o" extension)

Modules may have commands executed after installation and before removal. This example restores sound mixer settings after the module is loaded.

Contents of /lib/modules/<version> Directory

Module dependency information file

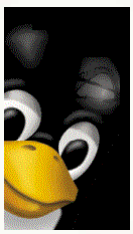
```
# ls -al /lib/modules/2.4.20-18.9

total 360
drwxr-xr-x    3 root     root         4096 Jun 19 14:57 .
drwxr-xr-x    6 root     root         4096 Jun 19 14:56 ..
lrwxrwxrwx    1 root     root           34 Jun 19 14:56 build ->
                                          ../../../usr/src/linux-2.4.20-18.9
drwxr-xr-x    8 root     root         4096 Jun 19 14:56 kernel
-rw-r--r--    1 root     root       104170 Jun 19 14:57 modules.dep
-rw-r--r--    1 root     root           31 Jun 19 14:57 modules.generic_string
-rw-r--r--    1 root     root          147 Jun 19 14:57 modules.ieee1394map
-rw-r--r--    1 root     root         8330 Jun 19 14:57 modules.isapnpmap
-rw-r--r--    1 root     root           29 Jun 19 14:57 modules.parportmap
-rw-r--r--    1 root     root        65563 Jun 19 14:57 modules.pcimap
-rw-r--r--    1 root     root           24 Jun 19 14:57 modules.pnpbiosmap
-rw-r--r--    1 root     root       135925 Jun 19 14:57 modules.usbmap
```

Manipulating Dynamically Loadable Modules

- **Dynamic module commands:**
  - **"*ksyms*"** — **List exported module symbols**
  - **"*insmod*"** — **install module (low level)**
  - **"*modprobe*"** — **Install module and dependencies (high level)**
  - **"*rmmod*"** — **remove module**
  - **"*depmod*"** — **create module dependencies**
  - **"*lsmod*"** — **list installed modules**

Miscellaneous Linux Tidbits

- **HP-UX to Linux Commands**
- **MANPATH**
- **Shared Library Loading**
- **Using *"strace"***



Some Linux to HP-UX Command Mappings

HP-UX	Linux
swapinfo	*swapon -s*
chown root:root /root_home	*chown root.root /dev/fd0*
rm	*rm -f* (defaults to "safe")
ll	*alias ll='ls -al'*
bdf	*df*
lanscan	*ifconfig*

Note: Many Linux commands support both the "-v" option style and the GNU "--verbose" option style. Check the man page for the command for details

Tidbits Concerning "man" on Linux

- The *"man"* command on Linux works differently than you might be used to:
  - The MANPATH variable is usually empty
  - The *"man"* command uses */etc/man.conf* to determine a base level of paths for man pages
  - The "man" command will also search according to your PATH variable value, looking "in the neighborhood of the executable" for man pages – this works well for self-contained packages
  - If you set MANPATH, it will *override* all other behavior, which will make things stop working – lots of software that is not written explicitly for Linux will set MANPATH and break things
  - If your *"man"* command behavior is strange, check for something setting *"MANPATH=${MANPATH}:NewPath"* during installation
  - Like, execute *"man man"*, man, if you want gory details

Shared Library Loading

- **Occasionally, we all have to deal with shared library loading issues, particularly if we are installing software**
- **Useful shared library information:**
  - ***ldd*** — **list executable's shared library dependencies**
  - ***ld.so*** — **shared object loader, see *ld.so(8)***
  - ***/etc/ld.so.conf*** — **configuration file for shared libraries in the */usr/lib* and */lib* directories**
  - ***ldconfig*** — **program to configure shared library cache information, see *ldconfig(8)***
  - **LD_LIBRARY_PATH** — **one of many environment variables that affect the libraries loaded by an application**

Troubleshooting Application Behavior on Linux

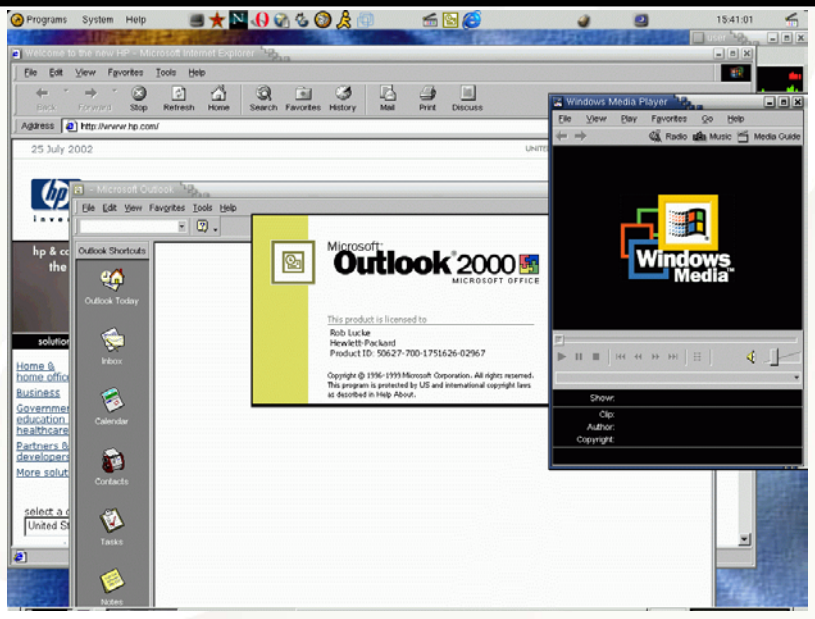
- **A useful tool for system administrators is the *"strace"* utility – it can tell you *everything* that an application is doing while it is running**
- **You can use *"strace"* to see:**
  - **application environment startup**
  - **shared library loading**
  - **kernel calls**
  - **library calls**
  - **… and a whole lot more …**
- **An example: After installing a to-remain-unnamed load-balancing facility, applications on the system slowed *way* down. Using *"strace"* on the *"ls"* command we were able to see that the shared library searches were failing multiple times before finally loading the correct system libraries. We traced this to the application adding its shared library paths on the front of the LD_LIBRARY_PATH variable, instead of at the end. We found this operation in a startup file being sourced by every user's shell at invocation, and fixed it. We would never have found this without *"strace"*.**

Example of “strace” Output

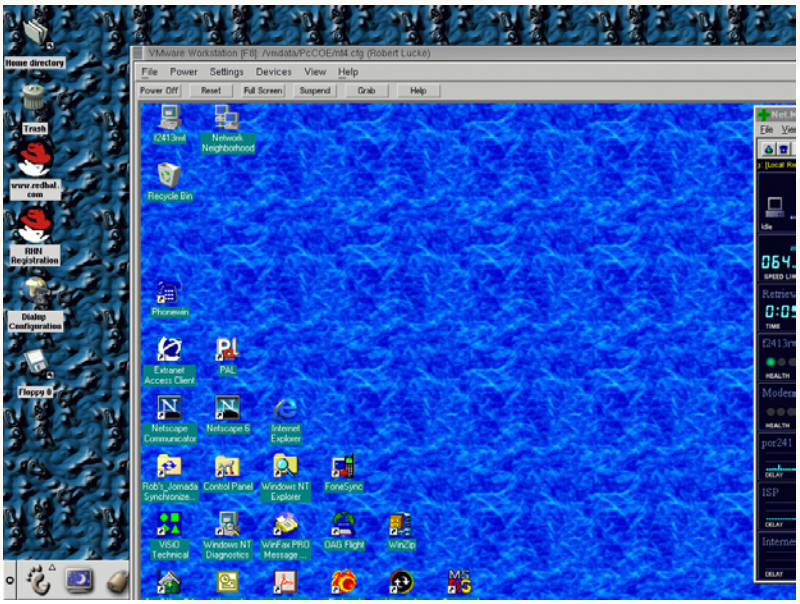
```
execve("/bin/ls", ["ls", "."], [/* 38 vars */]) = 0
uname({sys="Linux", node="hppav1", ...}) = 0
brk(0)                                  = 0x80586c8
old_mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
        0) = 0x40016000
open("/etc/ld.so.preload", O_RDONLY)    = -1 ENOENT (No such file or directory)
open("/etc/ld.so.cache", O_RDONLY)      = 3
fstat64(3, {st_mode=S_IFREG|0644, st_size=116342, ...}) = 0
old_mmap(NULL, 116342, PROT_READ, MAP_PRIVATE, 3, 0) = 0x40017000
close(3)                                = 0
open("/lib/libtermcap.so.2", O_RDONLY)  = 3
read(3, "\177ELF\1\1\1\0\0\0\0\0\0\0\0\0\3\0\3\0\1\0\0\0\340\r\0"..., 512) = 512
fstat64(3, {st_mode=S_IFREG|0755, st_size=11784, ...}) = 0
old_mmap(NULL, 14856, PROT_READ|PROT_EXEC, MAP_PRIVATE, 3, 0) = 0x40034000
old_mmap(0x40037000, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED, 3,
        0x2000) = 0x40037000
close(3)                                = 0
open("/lib/tls/libc.so.6", O_RDONLY)    = 3
```

You get the idea, this goes on for pages and pages on a complicated application

Running Windows Applications in CrossOver Office



Running Windows in a VMware Virtual Machine



Lab #4: Kernel, Modules, Software, and Miscellaneous

See Lab #4 Handout for details