RHCSA
BOOT CAMP

A whole week of geeky fun!

ABOUT THE INSTRUCTOR

- Nathan Isburgh
  - instructor@edgecloud.com
- Unix user 15+ years, teaching it 10+ years
- RHCE, CISSP
- Forgetful, goofy, patient :)

ABOUT THE CLASS

- 40 hour boot-camp style prep course
- Monday-Thursday: Lecture and labs
- Friday: Practice exam
- Hours:
  - 8:30am - 5:00pm
  - Open lab time in the afternoons
  - Lunch: 11:45am - 1:00pm

ABOUT THE COLLEGE

- Rackspace Parking Sticker = good to go
- Breakroom downstairs - labeled “Laundry”
- Sodas - bottles in machine ($1.25) or cans in mini-fridge ($0.50)
- Cafeteria
- Do not speed!
- No smoking anywhere on campus, even in your car.

ABOUT THE STUDENTS

- Name?
- Time served, I mean employed, at Rackspace?
- Department?
- Unix skill level?
- First attempt at RHCSA or equivalent?
- What are you most worried/interested about with RHCSA?

EXPECTATIONS OF STUDENTS

- Strong foundation in basic Linux use and administration
- Ask Questions!
- Complete the labs
- Email if you’re going to be late/miss class
- Have fun
- Learn something
- Pass your exam!

ABOUT RHCSA EXAM

- The RHCSA exam is 100% hands-on. You will have a computer and a list of tasks to accomplish. When you are finished, an automatic grader will check your machine to be sure that everything is set up correctly.
- There are no questions, only tasks.
- You will have 2.5 hours and access to all RHEL 6 Server software.
- You will not have internet access.

AFTER RHCSA?

- After successfully passing RHCSA, the next certification is RHCE: Red Hat Certified Engineer.

- The RHCE class will also be a week long, boot-camp style class.

- The RHCE exam is also 100% practical, and it is 2.0 hours long. Most of the focus on this certification is configuring network services.

TO PASS EXAM:

- Details specific to RHEL v. 6
- Basic System Administration and Unix interaction
- Configuration and deployment of storage and filesystems.
- Implementation of networking and basic security / traffic filtering technologies
- **Locating Local Reference Materials (--help, man)**
- Red Hat might even install non-standard software to check that the candidate can locate documentation!

SCHEDULE

Monday	Booting, Packages, System Administration
Tuesday	File Systems, Users
Wednesday	Kernel Features, File Sharing, Web Services
Thursday	Network Security, Virtualization
Friday	Practice Exam

```
slideshow.end();
```

RHCSA
BOOT CAMP

The Boot Process

OVERVIEW

- The boot process gets a machine from the useless off state to the feature rich operating system we all know and love

- Requires cooperation between hardware and software to correctly hand off processing

- Akin to the life cycle of a human - birth, newborn, infant, toddler, teen, adult

BIRTH

- Power switch flipped on
- Electricity flows from wall, through power supply where it gets converted to the levels necessary for the computer, and on to the motherboard, drives, CPU and more
- Completely unaware of the world or even what's attached to the motherboard.

INFANT

- BIOS - Basic Input/Output System - CPU looks for instructions starting at a specific address, which happens to be where BIOS resides. BIOS initializes and starts the....

- POST - Power On Self Test - A simple set of tests that BIOS performs to verify basic functioning of attached hardware.

- Like an infant, extremely limited understanding of world

- Searches for valid MBR, loads the software found there and transfers control to the...

TODDLER

- Boot Loader - Special software installed to the MBR of the boot partition which selects and loads the kernel.
- Can be configured to immediately load the default OS, or can offer choice to user
- Slightly better understanding of world - can read linux filesystems, sometimes includes powerful debugging and configuration support.
- Main job: select and load kernel, transfer control to kernel

TEENAGER

- Dreaded teenager age: knows a lot about the world, but doesn't contribute a thing. Still pretty useless.
- Kernel loads and initializes. Device drivers are loaded and initialized. Basic hardware checks performed.
- The First Process is *created from nothing*: `init`

ADULT

- init loads the inittab, specifying what the default runlevel should be, then additional configuration files specify what software needs to be started. init starts running all of the specified startup scripts at this point.

- Services are started by init, including network configurations, X Windows, network services, databases, etc.

- At this point, the machine is finally becoming useful: otherwise, an adult

- Eventually, login processes are started and the boot process is complete!

MORE ON INIT

- RHEL 6 marks Red Hat's departure from the old style SystemV initialization framework. Time to [mostly] forget about inittab!

- RHEL 6 now uses Upstart to handle startup, shutdown and service management.

  - `http://upstart.ubuntu.com`

- The only configuration `/etc/inittab` provides anymore is what the default runlevel should be, as Upstart supports the notion of runlevels to ease transition from SysV style initialization to Upstart.

UPSTART

- The configuration files for Upstart are under:
  - `/etc/init`
- Files in this directory detail configuration for certain global events, like ctrl-alt-delete, as well as maintaining TTY gettys, handling runlevels and more.
- A runlevel defines what services are running...

RUNLEVELS

- Runlevels:
  - S: System startup (not *really* a runlevel, but listed here as reminder)
  - 0: OS stopped, machine halted (usually powers off as well)
  - 1: Single user mode - for maintenance
  - 2: Multiuser, no NFS shares
  - 3: Full multiuser, TUI
  - 4: Unused
  - 5: Full multiuser, GUI
  - 6: Reboot

RUNLEVELS

- `telinit`: Signal the init process to change the current runlevel

- Switching runlevels is fairly uncommon - generally only used if system maintenance needs to be performed

- Runlevels can be used to control what services a machine provides, and can sometimes be useful to quickly reconfigure a machine for a new task

UPSTART OVERVIEW

- So the basic flow of operation for Upstart is as follows:
  - At bootup, the kernel starts `/sbin/init`. After `/sbin/init` loads configuration files and is ready, the first event is emitted: **`startup`**
  - The **`startup`** event causes `/etc/init/rcS.conf` to fire, which in turn runs the familiar `/etc/rc.d/rc.sysinit`. After `rc.sysinit` finishes, `rcS.conf` uses `/etc/inittab` to determine the default runlevel, then runs `telinit` to that runlevel.

UPSTART OVERVIEW

- `telinit` emits the **`runlevel`** event, which fires off `/etc/init/rc.conf`

- `rc.conf` fires off the familiar `/etc/rc.d/rc` script with the appropriate runlevel to fire off all of the startup scripts in the appropriate `/etc/rcX.d` directory.

- **WHEW!**

- All of this, mainly so that the transition to Upstart is relatively painless for the system administrators more comfortable with SysV initialization.

INIT SCRIPTS

- What is actually running in a given runlevel is defined by the init scripts for that level.
- That standard location for the init scripts is:
  - `/etc/rcX.d`
  - Where the `X` corresponds to the runlevel
- For example, `/etc/rc5.d` contains all of the init scripts that, combined, provide runlevel 5 service

RC DIRECTORIES

- The files in the rc directories start with either an S or a K:
  - S means to start the service, ie run the command with "start" as an argument
  - K means to kill the service, ie run the command with "stop" as an argument
- After the S or K, there is a two digit number which is used for ordering the execution of the scripts

ENTERING A RUNLEVEL

- So when the init process “enters” a runlevel, the steps are:
  - Run all of the Kill scripts, in order, with “stop” as an argument
  - Run all of the Start scripts, in order, with “start” as an argument

INIT SCRIPTS

- If you look closely, you will see that `/etc/rcX.d` actually holds a collection of symbolic links
- The actual script files are stored in `/etc/init.d`
- The main reason for this is so that there is only one copy of each init script, reducing the chance that a script change won't be reflected in all runlevels.
- You can run the scripts directly, or use the `service` command to start/stop various components of the OS.

MANAGING INIT SCRIPTS

- You can manage the links to the init scripts manually, or you can use the `chkconfig` command to get the job done:

- **`chkconfig --list`**
  - List all processes and display their default status in each run-level.
- **`chkconfig [--level levels] name <on|off|reset>`**
  - This command will modify the chkconfig configuration for a particular service, setting it on/off for the given runlevels.

GRUB

- Grand Unified Boot Loader
- Recall that GRUB is responsible for the initial kernel load at boot time.
- Using GRUB, an administrator can control what kernel is loaded, what options are passed to the kernel, as well as initial ramdisk configurations.

GRUB CONFIGURATION

- GRUB's configuration file is `/boot/grub/grub.conf`, which is configured as follows:

```
default=0

timeout=10

splashimage=(hd0,0)/grub/splash.xpm.gz

title RedHat Enterprise Linux

    root (hd0,0)

    kernel /vmlinuz ro root=LABEL=/

    initrd /initrd
```

GRUB SHELL

- Command mode – Pressing “c” while the boot menu is displayed will provide the user with the GRUB shell, where a limited set of commands can be used to explore the filesystem before booting. A full list of the commands available can be found by pressing Tab while in command mode.

- Editing mode – Pressing “e” while the boot menu is displayed will provide the user with the opportunity to edit a line in GRUB’s configuration file.

- Append mode – Pressing “a” while the boot menu is displayed will allow the user to append to the kernel line for the default kernel in GRUB’s configuration file

- Esc – can be pressed at any time to return you to the previous menu

BOOTING TO A GIVEN RUNLEVEL

- Using GRUB, add a number to the end of the kernel command line to override the default runlevel.

- Also, adding the letter “**`s`**” or the word “**`single`**” to the end of the command line is very important: this boots into single user mode, which by default, will not require a password to obtain a root shell.

- Very important!

LAB

1. Reboot your machine into the single user runlevel and verify root access without a password.

2. Review a few of the `init.d` scripts

3. Review the configuration files in `/etc/init`

```
slideshow.end();
```

RHCSA
BOOT CAMP

Package Management

RPM

- Redhat Package Manager
- RPM's provide full software packaging features: pre-install scripts, post-install scripts, dependencies, meta information, and an installed software database to name a few.
- The RPM system maintains a database of all installed software on a machine - this is useful for tracking and updating reasons, as well as dependency verification and software management.

RPM

- `rpm`: The Redhat Package Manager tool. Provides interface to RPM system, performing queries, installs, upgrades, uninstalls and general database maintenance operations.
  - `-i` option: install the given package
  - `-q` option: query the database
  - `-e` option: erase the given package from the system

RPM QUERIES

- Below are just a few examples of the types of queries you can run against the RPM database.
  - **`rpm -qa`** Queries for the names of all installed rpms.
  - **`rpm -qi`** Queries the rpm database for package information.
  - **`rpm -qf`** Determines which rpm a file is associated with.
  - **`rpm -ql`** Queries the rpm database to determine which files are associated with a particular rpm.
- With any of these commands, you can add the `-p` option to run the command against a package before it is installed.

RPM INSTALLATION VERIFICATION

- In addition to storing information about where a package is installed, `rpm` also stores permissions, file sizes, md5sums, and ownership information. This information can be easily referenced to see if anything has been changed.
  - **`rpm -Va`** Verifies all installed packages.
  - **`rpm –Vi <package>`** Verifies given package.
- Rackspace Best Practice Example
  - `rpm –Va | grep ^..5`

RPM VERIFY OUTPUT

- **S** File Size differs
- **M** Mode differs (includes permissions and file type)
- **5** MD5 sum differs
- **D** Device major/minor number mismatch
- **L** readLink(2) path mismatch
- **U** User ownership differs
- **G** Group ownership differs
- **T** mTime differs
- **C** SELinux Context differs

EXTRACT RPM CONTENTS

- Use this technique to make a clean working copy of the files and directories that would be installed with a package.
  - `cd /temp/dir`
  - `rpm2cpio /path/to/package | cpio -i –d –m`
- This would allow you to:
  - Replace one corrupted file without un-installing and then re-installing a package
  - Compare original configuration files versus modified files in the running system to quickly locate changed lines, for example with the ‘diff’ utility

YUM

- yum: Yellowdog Updater Modified
  - Supports package installation over the network through repositories.
  - RPM backend
  - Simple interface

REPOSITORIES

- Repositories of packages must be listed in files in the `/etc/yum.repos.d` directory with names ending in `.repo` and having a format like:
  - `[label-for-repo]`
  - `name = descriptive text`
  - `baseurl = protocol://path/to/directory/of/packages`
- Access to the Red Hat Network, including any Satellite Servers, is implemented through a plugin to the yum tool itself and not as a repository definition in the above format.

LAB

1. Connect to http://server1.example.com and read the information there.

2. Download the OpenOffice archive from `server1` and choose an appropriate location to extract all its RPMs

3. Install the `createrepo` package and use it to to turn your collection of OpenOffice packages into a yum repository

4. Add that repository to your local yum configuration

5. Using yum, install the “openoffice.org3-writer” package, and/or any others from your new repository

```
slideshow.end();
```

RHCSA
BOOT CAMP

System Administration

INSTALLATION

- Installing RHEL 6 is a straightforward process when performed interactively. I expect every single person in here can install RHEL 6 from media.

- Unattended install using a Kickstart file is another matter entirely, though.

KICKSTART FILES

- Fortunately, Kickstart files are *extremely simple* to understand and create.
- A Kickstart file is a flat text file which answers all of the installation questions automatically. Therefore, logically, it contains details on:
  - Partitioning and filesystems
  - Software packages
  - Users, Groups, Passwords
  - Features, networking and more

KICKSTART FILES

- There are three primary means of creating a Kickstart file:
  - From scratch
  - From an existing Kickstart file (perhaps from a recent install?)
  - Using `system-config-kickstart`

LAB

1. Examine `/root/anaconda-ks.cfg`

2. Install and run `system-config-kickstart` and create a simple kickstart file to install a basic desktop RHEL 6 machine.

NETWORK CONFIGURATION

- There are two main approaches to configuring a machine for network access:
  - Static configuration
  - Dynamic configuration
- Static configuration uses set parameters for the configuration, which is known by the machine and the network and never changes. Generally used with servers.
- Dynamic configuration configures network machines on the fly, where a service on the network provides all configuration parameters to a machine when it joins the network. Generally used with workstations.

DYNAMIC CONFIGURATION

- Dynamic configuration is the easiest to use.
- The machine just needs to set up it's interfaces with the DHCP protocol.
- DHCP: Dynamic Host Configuration Protocol.
- A lease is obtained from the DHCP server, providing all network configuration details for the client. The lease expires after some amount of time and is renewed by the client to maintain network access.

STATIC CONFIGURATION

- Static configuration requires four configuration parameters in order to allow full network functionality:
  - IP Address
  - Netmask
  - Default Gateway or Router
  - DNS Server(s)

DNS?

- Domain Name Service: This is the glue between network names and IP addresses.

- Remember: Humans like names, computers like numbers. DNS is a service like so many others, mapping names to numbers and numbers to names. Mostly a convenience.

- Also provides for email functionality, geographic load balancing and limited service failover capabilities.

STATIC CONFIGURATION

- The first two components of static configuration are IP address and netmask.
- These provide LAN-level access
- To view current address:
  - `ip addr list`

GATEWAYS

- The third configuration parameter is the default gateway.
- Provides access to *inter-networking*, or moving from just the local LAN to other LAN's
- To see the current routing entries:
  - `ip route show`

DNS SERVERS

- Final piece of configuration information.
- List of one or more IP addresses which provide the DNS service, allowing name to IP address mapping
- To view current nameservers, see:
  - `/etc/resolv.conf`
- Also consider `/etc/nsswitch.conf`

STATIC CONFIGURATION

- Once all four pieces of information are configured on the system, full network service will be available.
- To test local connectivity, try pinging the gateway
- To test inter-networking connectivity, try pinging `8.8.8.8` or some other external IP address.
- To test name resolution, try pinging `google.com` or another public DNS name.

CHANGING NETWORKING

- To change the IP address, hostname, netmask and gateway, you have to edit two configuration files:
  - `/etc/sysconfig/network-scripts/ifcfg-eth0`
  - `/etc/sysconfig/network`

/ETC/SYSCONFIG/NETWORK

```
NETWORKING={yes|no}

HOSTNAME=<fqdn>

NISDOMAIN=<nis domain name>
```

IFCFG-* FILES

- To configure a device to use dhcp, the ifcfg file should contain the following:

```
DEVICE=eth0

BOOTPROTO=dhcp

ONBOOT=yes
```

IFCFG-* FILES

- To configure a device with static settings, the ifcfg file should contain the following:

```
DEVICE=eth0

BOOTPROTO=none

IPADDR=<ip>

NETMASK=<netmask> ( or PREFIX=<net bits> )

ONBOOT=yes

GATEWAY=<gateway ip>

DNS1=<dns server ip>

DNS2=<dns server ip>

DOMAIN=<dns search domain>
```

NETWORK MANAGER

- In RHEL 6, Network interfaces are now handled via Network Manager. Some notable commands/tools:
  - `nmcli` - simple CLI to Network Manager
  - `nm-connection-editor` - excellent GUI tool for managing all network connections.
- On the test, you should decide if you are going to use Network Manager or not, and if so, only use NM and don't edit the config files by hand. Otherwise, disable NM and edit the files by hand.

LAB

1. Determine your current network settings (which were assigned by DHCP) and change your machine to a static network configuration using these settings.

2. When you are satisfied with your configuration, restart the network service to put your changes into effect.

3. Test your connectivity to `server1` to make sure you are still online.

4. Refer back to DHCP settings if necessary to correct any mistakes in your static configuration.

5. Once complete, switch everything back to DHCP.

CRON

- `crond` is the cron daemon. Cron provides for the ability to execute commands on a regular basis.

- Generally used to run hourly, daily and weekly type system maintenance scripts.

- Also useful to run reports, cleanup jobs and much, much more.

SYSTEM CRONS

- `/etc/crontab` and `/etc/cron.d/*` define the system cron jobs.
- `/etc/anacrontab` defines system cron jobs that are run even if the machine was not running when the job normally executes.
- Many distributions use the `run-parts` script to execute all scripts found in `/etc/cron.hourly`, `/etc/cron.daily`, etc on the appropriate schedule.
  - `/etc/anacrontab` defines the times for each schedule: daily, weekly, monthly
  - Due to limitations in `anacrontab`, the hourly scripts are configured to run via `/etc/cron.d/0hourly`

USING CRON

- Cron is controlled through crontab files.
- There are system-wide crons as discussed previously.
- Every user has their own crontab, accessible through the `crontab` command

CRONTAB

- `crontab`: View, edit or remove crontabs
  - The `-l` option prints the crontab. The `-e` option opens the crontab for editing. The `-r` option removes the crontab.
- Root can work with the crontab for any user by specifying the username on the command line:
  - `crontab -e -u bob`

CRONTAB SYNTAX

- There are two main components to a crontab entry:
  - The timespec specifies when the command should be run
  - The command is what gets executed every time the timespec is matched

CRONTAB TIMESPECS

- The timespec is broken down into 5 fields, separated by spaces:
  - `minute hour day-of-month month day-of-week`
- Each field can contain a number, a range of numbers, a comma-separated list of numbers, an asterisk or a number slash division rate
- Mostly self-explanatory - some examples will help...

TIMESPEC EXAMPLES

- `0 23 * * *` *11pm every day*
- `30 * * * 1-5` *30 minutes after every hour, M-F*
- `0 7 1 * *` *7am, first of every month*
- `* * * * *` *Every single minute*
- `0,10,20,30,40,50 * * * *` *Every 10 minutes*
- `*/5 8-17 * * 1-5` *Every 5 minutes, 8am-5pm, M-F*

EXAMPLE CRONTAB

```
01 4 * * * /usr/local/bin/restart-webserver
00 8 1 * * /usr/bin/mail-report boss@mycompany.com
*/5 * * * * /monitor/bin/check-site -e admin@mycompany.com -o /var/log/check.log
```

- There are various additional options and features available to the cron system. Check the man pages for reference:
  - `cron`, `crontab` (sections 1 and 5)

LAB

1. Create a cronjob for the user root that checks the amount of available space on the system every Friday at 12:34pm.

2. Create a cronjob as a regular user that lists the contents of `/tmp` at 3:54am on Sunday, January 2.

LOGS

- One of the easiest places to find the cause of a problem is in the log files.

- Log files store informational messages from software. The types of messages include debug information, status information, warnings, errors and more.

- Some applications manage their own log files. Others use the system-wide logging package...

SYSLOG

- `rsyslog` - The system logger. A framework consisting of a library, a daemon, a configuration file and logs.
- Any application can use the library and log messages through `rsyslog` with simple function calls.
- Log messages consist of 3 parts:
  - Facility
  - Level
  - Message

SYSLOG

- The facility describes what part of the operating system generated the message, and is selected by the software:
  - `auth, authpriv, cron, daemon, ftp, kern, lpr, mail, news, security, syslog, user, uucp, local0-local7`
- The level represents the importance of the message, and is also chosen by the software:
  - `emergency, alert, critical, error, warning, notice, info, debug`

/ETC/RSYSLOG.CONF

- `/etc/rsyslog.conf` defines where all of the log messages should go. Destinations include files, screens of logged in users, console, other syslog servers. Additional configuration is available as well.

- Basic rule format:

  - `facility.level  destination`

- Examples:

  - `*.err  /dev/console`

  - `mail.*  /var/log/maillog`

  - `*.info;mail.none;authpriv.none  /var/log/messages`

/VAR/LOG

- `maillog`: messages from the email subsystem
- `secure`: authentication and security messages
- `cron`: cron messages
- `boot.log`: boot messages
- `messages`: catch-all
- `dmesg` : hardware and kernel events generated before syslogd

LOGS

- As mentioned earlier, not all software uses the syslog framework to handle it's logging. Quite a bit of software manages it's own logs.

- This can make it difficult to track down all of the log locations on an unfamiliar system. The best way to handle this is to start from the init scripts...

LOCATING APPLICATION LOGS

- To track down the log file location for an application, you need to find it's configuration file so you can see where the logs are being written.

- Of course, finding the configuration file might be just as difficult, so it's best to start at the source.

- `init` starts all of the system services, and so there is an init script somewhere that is starting up the application in question.

- The init script almost always references the configuration file

LOCATING APPLICATION LOGS

- Now that the configuration file location is known, it only takes a few moments to scan through it and find out where logs are being written.

- As for the format of the log file, that's completely dependent on the application. Some will be similar to syslog, others, like Apache or Qmail, will be completely foreign looking.

- Fortunately, a little common sense and judicious application of Google Ointment will get the information you seek.

MAINTAINING LOGS

- `/etc/logrotate.conf`
  - This is the main configuration file for logrotate.
- `/etc/logrotate.d/`
  - EVERYTHING in this directory will be parsed as if it is a logrotate configuration file. Usually, applications such as Apache and Sendmail will have configuration files in this directory to control how their logs will be rotated.
- `logrotate -vf /etc/logrotate.conf`
  - Can be run as root at any time to force log rotation and check for errors.

TROUBLESHOOTING

- There will be some basic troubleshooting objectives on the exam, mostly to test basic knowledge of how permissions should work, SELinux and locating error messages in log files.

- Mentioned here are a few useful tools to remember

TOP

- `top`: Self-updating tool displays combination summary at top, followed by ordered list of processes. Fully customizable.
  - The summary includes uptime information, memory breakdowns, CPU utilization and process state summaries
  - The process display can be customized and sorted to suit need

```
top - 16:39:32 up 682 days, 10:41,  2 users,  load average: 0.01, 0.00, 0.00
Tasks: 118 total,   1 running, 116 sleeping,   1 stopped,   0 zombie
Cpu(s):  0.1%us,  0.0%sy,  0.0%ni, 99.8%id,  0.0%wa,  0.0%hi,  0.0%si,  0.1%st
Mem:    262316k total,   258024k used,     4292k free,     7380k buffers
Swap:   524280k total,    74564k used,   449716k free,    67808k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
    1 root      15   0 10316  648  592 S    0  0.2   0:06.24 init
    2 root      RT   0     0    0    0 S    0  0.0   0:04.88 migration/0
    3 root      34  19     0    0    0 S    0  0.0   0:00.19 ksoftirqd/0
```

DF

- `df`: lists filesystem utilization
  - Breaks down size and use information for each mounted filesystem
  - -h is useful option to display in “human-friendly” format

```
[root@dev1 ~]# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/sda1             9.4G  7.2G  1.8G  81% /
none                  129M     0  129M   0% /dev/shm
[root@dev1 ~]#
```

LDD, LDCONFIG

- `ldd`: List library dependencies
- `ldconfig`: Update library location database
- `/etc/ld.so.conf` and `/etc/ld.so.conf.d/*.conf` for list of pathnames to search for libraries, creates database for dynamic linker

```
[root@dev1 ~]# ldd /bin/bash
    libtermcap.so.2 => /lib64/libtermcap.so.2 (0x00002ac044572000)
    libdl.so.2 => /lib64/libdl.so.2 (0x00002ac044775000)
    libc.so.6 => /lib64/libc.so.6 (0x00002ac044979000)
    /lib64/ld-linux-x86-64.so.2 (0x00002ac044357000)
[root@dev1 ~]# cat /etc/ld.so.conf.d/mysql-x86_64.conf
/usr/lib64/mysql
[root@dev1 ~]# ldconfig
[root@dev1 ~]#
```

NICE LEVEL

- The nice level represents one influence on the calculations the kernel uses when assigning priorities.
- Originally designed and named to allow users to be “nice” to other users of the system by assigning a higher nice value to an intensive process, which in turn lowers it’s priority.
- Ranges from -20 to 19. Default nice level is 0.
- Only root can assign negative nice values.
- See `nice` and `renice` commands

LAB

1. Take a few minutes to browse through the various logs in `/var/log`. Familiarize yourself with the kinds of information available.

2. Browse the man page for `rsyslog.conf`

3. Find where the audit service keeps its log and add a corresponding new entry to your logrotate configuration. Force a rotation to see everything work.

4. Remove the audit logrotate configuration and restart the auditd service.

5. Locate the PIDs of the highest memory and highest CPU utilization processes. Play with their nice levels.

```
slideshow.end();
```

RHCSA
BOOT CAMP

Filesystem Administration

PARTITIONING

- What is partitioning?
  - Splitting up a hard drive into organizable chunks
- Why?
  - Isolates filesystem corruption
  - Simplifies/speeds backups
  - Allows optimizing filesystems to tasks

FDISK

- `fdisk`: partitioning tool
  - Works on one disk at a time, allows for viewing and manipulating partition table.
    - Online help (hit ‘m’) makes tool easy to use
- At boot, the kernel loads a copy of the partition table into memory. Most partition editing commands only update the partition table on the drive, and not in memory. As such, the command `partprobe` was used to trick the kernel and force a reload. **`partprobe` does not work in RHEL 6!**

MKFS

- `mkfs`: format a device to create a new filesystem
  - "Paints the parking stripes" for the filesystem structure
  - For Linux extended filesystems, this means creating the superblock, block groups, superblock copies, bitmaps and inode tables and creates basic structure on disk
  - Through `-t` option, mkfs can create different types of filesystems

EXT2

- Benefits
  - Default file system for pre - 7.x versions of Red Hat
  - Heavily tested / Rock solid stability
- Drawbacks
  - Does not have a journal
  - File system check (fsck) required to mount a “dirty” file system
  - System offline and unavailable while fsck is running

EXT3

- Benefits
  - Default file system of the old 7.x Red Hat to RHEL 5.x releases
  - Based on proven stability of Ext2
  - Has journal for increased reliability
- Drawbacks
  - Inodes allocated when file system is created (other file systems create them dynamically as they are needed)
  - Not as efficient as other file systems when dealing with lots of small files

EXT4

- Benefits
  - Default file system of RHEL 6.x releases and newer
  - Built from a series of extensions to ext3
    - Many improvements over ext3, including larger scales, timestamps, performance and more
- Drawbacks
  - Inodes allocated when file system is created (other file systems create them dynamically as they are needed)
  - Delayed allocation can potentially lead to data loss (patches in place)

JOURNALING

- Journaling - How does it help?
- Deleting a file in Linux requires two steps:
  1. The file's directory entry must be removed.
  2. The file's inode must be marked as free in the free space map.
- If step 1 happens before a crash, an inode will be orphaned and the file will be lost.
- If step 2 happens first before a crash, the inode will be marked free and will possibly be overwritten.
- Journaling keeps a journal of the changes that are planned for the file system ahead of time. The journal can then replay the changes in the journal at any time to keep the file system clean.

FILESYSTEM INTEGRITY CHECKS

- `fsck`: Filesystem Check
  - Generally only run when a filesystem needs it:
    - Mount count
    - Last check
    - Dirty
- Checks all of the filesystem structures for accuracy and completeness

FILE SYSTEM TOOLS

- `e2label`: View/set filesystem label
- `tune2fs`: View/set filesystem attributes
- `mount/umount`: You better know these already. :)

FSTAB

- `/etc/fstab` is parsed during boot by `rc.sysinit` to determine what file systems should be mounted and how. After boot, this file is referenced by the `mount` command.

- The file is space delimited and organized as follows:

```
device  mount_point  fs_type  options  dump  fsck
```

LAB

1. Using `fdisk`, create a new 100MB partition.

2. Create a new filesystem on this partition using ext4, a blocksize of 1k, and a reserve space of 2%. Confirm settings with `tune2fs`. Mount the new filesystem as `/u01` and set it to mount at boot.

3. Un-mount the `/u01` filesystem and force an integrity check. Re-mount the `/u01` filesystem. Use `e2label` to set the filesystem label on `/u01` to '`/u01`'.

AUTOMOUNT

- The `autofs` service can be configured to monitor certain directories and automatically mount a file system when a call is made to files in that directory.

- When `autofs` starts, it parses the configuration file `/etc/auto.master` to determine which directories it should be monitoring. Each directory can then have its own configuration file determining how each file system should be mounted, or the default file `/etc/auto.misc` can be used.

AUTO.MASTER

- Basic format for `auto.master`:
- Path Config file
- `/misc /etc/auto.misc`
- This tells `automountd` to “watch” the `/misc` pathname for activity, and if activity is observed, consult `/etc/auto.misc` for instructions.

AUTOMOUNT PATH CONFIG FILES

- Basic syntax:
- path options mount device
- `nfs -fstype=nfs,ro nfsserver:/share/nfs`
- This tells `automountd` to dynamically mount the nfs share "`/share/nfs`" on `nfsserver` when access is attempted on the "`nfs`" pathname under a watched pathname (`/misc` for example)

LAB

1. Configure your server to automatically mount `/share` as an NFS share from `server1` to `/server1/share` when a process changes directories there.

EXTENDED ATTRIBUTES

- The Linux Extended filesystem supports attributes that affect how data can be manipulated.
- The `chattr` command can change these file system attributes.
- The `lsattr` command will list the file system attributes.
- Extended attributes can only be set by the root user, unless the `user_xattr` mount option is in effect.

COMMON EXTENDED FILE ATTRIBUTES

- `i` Immutable. The file can not be changed. By anyone. Period.
- `a` Append-only. File can only be opened for appending.
- Most of the others are experimental and/or esoteric. Surprising? ;)

ACL's

- The Linux Extended Filesystem supports access control lists, which allow for more flexible permissions than standard file system permissions.
- ACL's can be listed with the `getfacl` command.
- They can be modified with the `setfacl` command.
- To use ACLs, a file system must have the `acl` mount option.
- Use `dumpe2fs -h <block device node>` to see default mount options.

ACL EXAMPLES

- `setfacl -m u:bob:w memo.txt`
- `setfacl -x g:ru report.txt`
- `setfacl -m g:ru:r another-report.txt`

QUOTAS

- Quotas are used to limit how many filesystem resources are available to a user.
- Inodes and space are controllable.
- Hard and soft limits are available, with grace periods.
- Enabling quotes is an involved process...

ENABLING QUOTAS

- `usrquota` and `grpquota` options must be enabled on the filesystem mount
- Run `quotacheck -mavug`
- Two files will be created at the root of the filesystem: `aquota.user` and `aquota.group`
- Turn on quotas by running `quotaon` with the mount point as argument.
- Now you can use `edquota` to set up the quotas
- See man pages: `quota, repquota, edquota, quotaon, quotacheck`

LAB

1. Create a quota for the user `student` with:
   - a block soft limit of `100M` and a hard limit of `150M`
   - a soft inode limit of `30` and a hard inode limit of `100`
2. Create a quota for the group `gdm` so that its members collectively have:
   - a block soft limit of `200M` and a hard limit of `300M`
   - a soft inode limit of `50` and a hard inode limit of `200`

DISK ENCRYPTION

- Disk encryption is supported under Linux via the Device Mapper functionality introduced in the 2.6 kernel.
- The Device Mapper allows arbitrary device path mapping.
- Disk encryption is most commonly implemented with the dm-crypt Device Mapper module, supporting transparent device encryption.
- dm-crypt supports a simple, internal encryption specification, as well as the more common LUKS, Linux Unified Key Setup.

LUKS

- LUKS is an open standard disk encryption specification.
- LUKS is a preferred standard due to it's broad compatibility and secure implementation.
- Using `cryptsetup`, a LUKS encrypted device can be created, accessed and modified.

CRYPTSETUP

- To create a new LUKS encrypted device:
  - `cryptsetup luksFormat <device>`
- Then, to establish access to the device:
  - `cryptsetup luksOpen <device> <mapname>`
- This command will verify the password and setup a new dm-crypt device mapper mapping of:
  - `<device> -> dm-crypt(LUKS) -> <mapname>`
- Creating `/dev/mapper/mapname`

CRYPTSETUP

- After the `/dev/mapper/mapname` is in place, all operations operate on the mapper device:
  - `mkfs -t ext4 /dev/mapper/mapname`
  - `mount /dev/mapper/mapname /crypt`
- To remove access to an encrypted device, unmount the filesystem if it's mounted, then:
  - `cryptsetup luksClose mapname`

LUKS PERSISTENCE

- To make a LUKS encrypted device available at boot time, use the `/etc/crypttab` file:
  - `<mapname> <device> [keyfile] [options]`
- To create an insecure keyfile:
  - `echo -n 'your pass phrase' > /etc/keyfile`
- To create a secure keyfile:
  - `dd if=/dev/urandom of=/etc/keyfile bs=1k count=4`
  - `cryptsetup luksAddKey <device> /etc/keyfile`

LAB

1. Create a new 100M partition, then set up a LUKS encrypted ext4 filesystem on the partition which will be persistent across reboots.

2. Reboot your machine to verify the LUKS filesystems prompt for the passphrase and become accessible automatically after bootup.

3. Browse through the man pages on `cryptsetup` and `crypttab`.

SELINUX

- Every process or object has an SELinux context:
  - `identity:role:domain/type`
- The SELinux policy controls:
  - What identities can use which roles
  - What roles can enter which domains
  - What domains can access which types

SELINUX

- Adding the `-Z` option to several commands will show how they are running in regards to SELinux:
  - `ps -Z` lists the process contexts
  - `ls -Z` lists the file contexts
- To change the context of a file, you can use the `chcon` command:
  - `chcon -R --reference=/var/www/html <file>`
- SELinux will log all policy violations to `/var/log/audit/audit.log` as AVC (access vector cache) denials.

LABELING

- The SELinux policy includes a specification for default contexts on all common pathnames in a standard Linux filesystem, known as the default filesystem labels.

- Relabeling involves using the defaults from the policy and applying the contexts to files. The tool for relabeling is:

  - `restorecon [-R] <path> [path...]`

- `restorecon` can work on individual pathnames as well as recursively apply contexts to a pathname.

CONTROLLING SELINUX

- The tool `system-config-selinux` can be used to configure SELinux.
- The file `/etc/sysconfig/selinux` can be edited.
- The command `getenforce` will show the current SELinux mode, and `setenforce` will allow you to change the mode.
- To change the SELinux mode during boot, you can pass the `enforcing=0` option to the kernel in GRUB.
- See also the members of the "`policycoreutils`" and "`setroubleshoot`" packages.

ADDITIONAL SELINUX TOOLS

- `restorecon` Will restore default filesystem contexts from policy.
- `getsebool` View SELinux boolean(s)
- `setsebool` Set SELinux boolean
- `seinfo` View SELinux information - types, domains, roles, etc.

LAB

1. With SELinux enforcing, configure a website to be served from `/srv`

2. Don't focus on advanced Apache settings, accomplish this in the simplest way possible: just change the global `DocumentRoot`.

3. Populate a simple `index.html` file. Plain text is acceptable.

4. The `setroubleshoot` tool is useful here. Don't be confused by any typos in its output.

```
slideshow.end();
```

RHCSA
BOOT CAMP

Users and Groups

USERS AND GROUPS

- Users and Groups define access to the operating system through the file permission scheme.
- Root is the super user, and the only user with special permissions
- Every user is a member of at least one group, which is called their primary group. The main purpose of this primary relationship is to define group owner of created files.
- Users can have a secondary group membership in as many groups as needed. These secondary relationships exist to broaden a user's access to the files on the system.

CONFIG FILES

- User information is stored in two files:
  - `/etc/passwd`
  - `/etc/shadow`
- Group information is stored in one file:
  - `/etc/group`

/ETC/PASSWD

- List of user records, one per line, with columns separated by colons. Format:
- `login:x:userid:groupid:gecos:homedir:shell`
- Examples:
  - `root:x:0:0:root:/root:/bin/bash`
  - `mysql:x:27:27:MySQL Server:/var/lib/mysql:/bin/bash`

/ETC/SHADOW

- Similar colon-separated-column list of records:
- `login:password:`*password aging fields*
- Aging fields track dates for password resets, locks, etc
- Examples:
  - root:pB8msP1fCbCqc:13904:0:99999:7:::
  - nisburgh:vRoPw6a/jQsp.:14466:0:99999:7:::

/ETC/GROUP

- Same colon-separated-column list of records format
- `groupname:grouppassword:groupid:secondarymembers`
- Group passwords allow temporary management to a group, are rarely used and not set up by default
- Examples:
  - `daemon:x:2:root,bin,daemon`
  - `apache:x:48:jack,nisburgh`

MANAGEMENT

- While it is possible to edit the three files directly, it's easier and safer to use the management commands to create, modify and delete users and groups:
  - `useradd, usermod, userdel`
  - `groupadd, groupmod, groupdel`

USERADD

- `useradd`: Add a new user to the system

- Accepts various arguments to control the settings on the user account. Most common is the -g option to specify the primary group of the user, and the -G option to list secondary group memberships. Examples:

  - `useradd lisa`

  - `useradd -g clowns -G trouble,simpson bart`

USERMOD, USERDEL

- `usermod`: Modify a user's settings. Example:
  - `usermod -G detention bart`
- `userdel`: Remove a user from the system. Main option to consider is `-r`, which tells `userdel` to remove the user's home and spool directories. Example:
  - `userdel moe`

GROUP COMMANDS

- `groupadd`: Adds a new group to the system. Example:
  - `groupadd bullies`
- `groupmod`: Mainly used to rename a group. Example:
  - `groupmod -n mktg mkg`
- `groupdel`: Remove a group. Example:
  - `groupdel microsoft`

PASSWORDS

- `passwd`: Change login password.
- Root can change the password for any user on the system
- Root can also setup password aging, allowing for timed password resets and account disabling (or use `chage`)
- `passwd` is also the preferred way to lock a user account:
  - `passwd -l mary`

PASSWORD AGING

- To set the maximum lifetime for a user's password:
  - `passwd -x days login`
- When a user's password has expired, you can set the number of days it can remain expired before disabling the account completely:
  - `passwd -i days login`

IMPORTANT USER ENVIRONMENT FILES

- `/etc/skel` default template for a newly-added user's home directory
- `/etc/profile` sets environmental variables used by all users
- `/etc/profile.d` contains scripts specific to certain rpms
- `/etc/bashrc` contains global aliases and system settings
- `~/.bashrc` contains user aliases and functions
- `~/.bash_profile` contains user environment settings and can be set to automatically start programs at login

LAB

1. Create a new group '`dev`'. Create a new user '`alice`' as a member of the '`dev`' group, with a description of "`Alice from Dev`" and a default shell of '`/bin/csh`'. Use the `passwd` command to set a password for `alice`, then log in as `alice` and verify her access.

2. Set a maximum password lifetime of 4 weeks for the `alice` account. Look at the `passwd`, `shadow` and `group` files.

3. Configure the users `guido`, `linus`, and `richard`. Set all their passwords to "`linux`".

4. Make these users part of the `ru` group.

5. Configure the directory `/home/linux` so that each user from the `ru` group can read, create, and modify files.

6. Configure the directory `/home/linux/work` so that each user can create and read files, but only the file's owner can delete.

7. Use ACL's to allow `alice`, not in `ru`, full r/w access to the `work` folder.

NIS

- NIS Servers can be configured to centrally manage system and account information. These servers can share the contents of `/etc/passwd`, `/etc/shadow`, `/etc/group`, and several other files among any number of clients.
- To configure a client, you must install the `ypbind` and `portmap` RPMs, and then you can run `system-config-authentication`.
- This command will make the proper entries in:
  - `/etc/sysconfig/network`
  - `/etc/yp.conf`
  - `/etc/nsswitch.conf`
  - `/etc/pam.d/system-auth`

LAB

1. Configure your server to authenticate against `server1.example.com` using NIS.

2. You should then be able to log in to your server as `station#` (where `#` is your station number) with the password: `station#`

LDAP

- LDAP Servers can also be configured to centrally manage system and account information. LDAP is much more secure and flexible than a default NIS configuration, and as such is becoming much more popular.
- To configure a client, you must install the `nss-pam-ldapd` and `openldap` RPMs, and then you can run **`system-config-authentication`**.
- This command will make the proper entries in:
  - `/etc/ldap.conf`
  - `/etc/openldap/ldap.conf`
  - `/etc/nsswitch.conf`
  - `/etc/pam.d/system-auth`

LAB

1. Disable NIS authentication and verify you can no longer authenticate as `station#`.

2. Configure your server to authenticate against `server1.example.com` using LDAP.

3. You should then be able to log in to your server as `station#` (where `#` is your station number) with the password: `station#`

```
slideshow.end();
```

RHCSA
BOOT CAMP

Kernel Features

IMPORTANT KERNEL DIRECTORIES

- `/boot` — contains the `vmlinuz` and `initrd` required to boot the system
- `/proc` — virtual file system for seeing “into” the kernel

/PROC/*

- The `/proc` folder contains copious amounts of information useful for troubleshooting. Some examples:
  - `/proc/meminfo` Memory utilization breakdown
  - `/proc/devices` Mapping major numbers to drivers
  - `/proc/dma` dma channel assignments
  - `/proc/ioports` io port assignments
  - See the manpage for `proc` for more information and descriptions

/PROC/*

- Also in the `/proc` folder is detailed information on every process on the system.
  - Details on process status, environment, commandline, and more can be obtained
- Read the `proc` manpage - tons of information available through `/proc`

SYSCTL

- `sysctl`: Get/set kernel parameters
  - `sysctl -w kernel.pid_max=65535`
  - `sysctl -a`
  - `sysctl -w vm.swappiness=100`
- Also, you can view/edit runtime values under `/proc/sys`
- To make changes permanent, edit `/etc/sysctl.conf`

LAB

1. Configure your server to have an open file limit of `524288` files.
2. Configure your server to refuse any ping requests.
3. Configure your server to forward ipv4 packets.
4. Make all of these changes persistent across reboots.

LVM

- The Logical Volume Manager
  - Abstracts the physical hardware into logical drive spaces which can be dynamically grown/shrunk and span disparate physical devices
  - Simplifies hard drive management as it abstracts away the details of the underlying storage devices.
  - Adds a small amount of overhead to the VFS layer, slightly reducing performance.

LVM TERMINOLOGY

- **Physical Volume (pv)** — A physical volume is simply the partition/RAID device for the LVM space.
- **Physical Extent (pe)** — A physical extent in a chunk of disk space. Can be any size, but default to 4M.
- **Volume Group (vg)** — A volume group is a collection of physical volumes.
- **Logical Volume (lv)** — A logical volume is a grouping of physical extents from your physical volumes. This logical volume is where you can format a file system.

LVM BASIC IDEA

- To create a space suitable for `mkfs`, three steps must occur:
  - `pvcreate`: Create a physical volume
  - `vgcreate`: Create a volume group on PV
  - `lvcreate`: Create a logical volume on VG
- See also `pvdisplay`, `vgdisplay`, `lvdisplay`

PVCREATE

- Easiest of the LVM tools:

- `pvcreate /dev/sda4`

VGCREATE

- In basic form, you need to provide a name:
- `vgcreate VolGroup00 /dev/sda4`
- Note that `/dev/sda4` is actually a physical volume created with `pvcreate` - not just a device
- To set physical extent size, use the `-s` option.

LVCREATE

- `lvcreate -n myvol -L 10G VolGroup00`

- Creates a new logical volume called `myvol`, 10 gigs in size pulled from the `VolGroup00` Volume Group.

RESIZING LV'S

- `vgextend <volume group name> <new PV path>`
  - Add a new physical volume to a volume group
- `lvextend {-l <+extents>| -L <+size>} <lv>`
  - Grow a logical volume
  - NOTE: Use the + to give the amount of additional space added, otherwise specify the total desired size to end up with.

RESIZING LV'S

- `resize2fs <logical volume>`
  - Once the lv has been extended, you will need to extend the file system
  - You can grow the file system while it is mounted, but before shrinking it must first be unmounted.
- `lvresize -r {-l <+extents>| -L <+size>} <lv>`
  - Resizes logical volume **and** filesystem at same time!
  - Works like a champ in RHEL 6

LAB

1. Add logical volume management on top of a new partition. Use a physical extent size of 16MB.

2. Use half the available space for a logical volume formatted with ext4 and mounted persistently across reboots.

3. Take a snapshot of this logical volume and check the file system for errors.

4. Assuming none are found, reset the counter for days and mounts until a check is forced on the original file system.

5. Copy some data onto the LV, then expand it and the filesystem by 50MB. `fsck`, then re-mount the filesystem and verify it's contents. Also try reducing by 50MB.

SWAP SPACE

- Swap space allows the kernel to better manage limited system memory by copying segments of memory onto disk
  - Performance gains
  - “Expanded” memory space

`mkswap`	Create a new swap space for use by the kernel
`swapon/swapoff`	Enable/disable a swap area
`/proc/swaps`	Lists current swap areas

LAB

1. Add 500MB of swap space to your system using a device.

2. Add 500MB of swap space to your system using a swap file.

```
slideshow.end();
```

RHCSA
BOOT CAMP

File Sharing Services

NFS

- The Network File Service, or NFS, is used to share data with other servers.
- The command `rpcinfo` can be run to confirm that these services are running on a remote server:
  - `rpcinfo -p server1`
- To see the shared filesystems, use `showmount`:
  - `showmount -e server1`

ACCESSING NFS SHARES

- To mount an NFS share:
  - `mount server1:/share /server1/share`
- NFS mounts can be made persistent across reboots by adding the following to `/etc/fstab`:
  - `server1:/share  /server1/share  nfs  defaults 0 0`

LAB

1. Mount the `/share` NFS share from `server1`, and add it to your `fstab` for persistence across reboots.

VSFTPD

- VSFTPd is the default ftp server
- The primary configuration file is `/etc/vsftpd/vsftpd.conf`
- Provides two levels of user access:
  - **Anonymous:** by default these users are chrooted to `/var/ftp` for security
  - **User:** these users authenticate with a username/password and can download any file they can read and can upload into any directory in which they have write access
- Individual users can be denied by placing their names in:
  - `/etc/vsftpd/ftpusers`

LAB

1. Configure VSFTPd to only allow the user `richard` to ftp to your server.

2. Browse through the man page on `vsftpd.conf`.

3. Make sure vsftpd is started at boot time.

```
slideshow.end();
```

RHCSA
BOOT CAMP

Web Services

APACHE CONFIGURATION

- The main apache configuration file is `httpd.conf` and is found in `/etc/httpd/conf/`. This configuration file stores the core configuration of the web server.

- In Apache 2, the `/etc/httpd/conf.d` directory stores configurations that are specific to a particular Apache module. All files in this directory ending in `.conf` will be parsed as a configuration file.

APACHE CONFIGURATION

- You can find this example Apache VirtualHost definition at the bottom of `httpd.conf`:

```
<VirtualHost _____>
    ServerName name
    ServerAlias alias
    DocumentRoot path
    CustomLog /path/to/access_log combined
    ErrorLog /path/to/error_log
</VirtualHost>
```

- The `NameVirtualHost` directive **must be used** to specify an IP that can host multiple websites.

LAB

1. Configure two websites on your server. “X” represents your station #.

2. `wwwX.example.com` should be served from `/var/www/html` and should also respond to requests for the short hostname `wwwX`.

3. `vhostX.example.com` should be served from `/home/linus/html` and should also respond to requests for the short hostname `vhostX`.

4. Both should be listening on your primary ip address, but `wwwX.example.com` should be the default site.

SECURING APACHE

- Apache support access control through `allow` and `deny` directives:
    - `allow from <host|network|ALL>`
    - `deny from <host|network|ALL>`

- These can be applied in the given order:
    - `order allow,deny` — Allows explicitly allowed clients and **denies everyone else**. Anyone matching both the allow and deny are denied.
    - `order deny,allow` — Denies explicitly denied clients and **allows everyone else**. Anyone matching both the allow and deny are allowed.

SECURING APACHE

- These access control directive are applied through a per-Directory or per-File basis.

- The `allow`, `deny` and `order` directives are placed inside one of the following tags:

    - `<Directory>`

    - `<File>`

LAB

1. Reconfigure your two websites such that:

   - `wwwX.example.com` is accessible to everyone except for the person sitting to your left.

   - `vhostX.example.com` is only accessible to the person sitting to your right.

```
slideshow.end();
```

RHCSA
BOOT CAMP

Network Security

TCP WRAPPERS

- TCP Wrappers was originally written to provide host based access control for services which did not already include it.
- It was one of the first “firewalls” of a sort. :)
- Before you can set up tcp_wrappers on a service, you have to check if the service supports it, which involves:
  - Checking for libwrap in the binary
  - Checking that tcp wrappers is enabled in the config

CHECKING TCP WRAPPER SUPPORT

- Determine which binary the application runs as. Check `init` script or:

```
# which sshd

/usr/sbin/sshd
```

- Check for `libwrap` support in the binary.

- If you see `libwrap` support in the output, then you can configure access to the service with tcp_wrappers.

```
# ldd /usr/sbin/sshd | fgrep wrap

libwrap.so.0 => /usr/lib/libwrap.so.0 (0x009c5000)
```

TCP WRAPPER OPERATION

- If an application is compiled with support for `tcp_wrappers`, that application will check connection attempts against the `tcp_wrappers` configuration files:
  - `/etc/hosts.allow`
  - `/etc/hosts.deny`

TCP WRAPPER OPERATION

- These files are parsed in the following order:
    - The file `/etc/hosts.allow` is consulted. If the configuration of this file permits the requested connection, the connection is immediately allowed.
    - The file `/etc/hosts.deny` is consulted. If the configuration of this file does not permit the requested connection, the connection is immediately refused.
    - If the connection is not specifically accepted or rejected in either file, the connection is permitted.

TCP WRAPPER CONFIGURATION

- The basic syntax for these files is:
    - `<daemon>: <client>`
- For example, to disable `ssh` connections from `192.168.2.200`, add this line to `/etc/hosts.deny`:
    - `sshd: 192.168.2.200`

IPTABLES

- IPTables works at the kernel level, just above the network drivers, to provide several useful features.
  - Extremely powerful and flexible Layer 2 filtering engine.
  - NAT support
  - Port forwarding
  - And many more

IPTABLES RULE MATCHING

- The IPTables configuration is parsed from top to bottom.
- IPTables will respond based on the first match that it finds.
- If there is no specific match, the chain policy will apply.

IPTABLES TOOLS

- **`iptables`**: View/modify current firewall rules
- **`iptables-save`**: Script to save current firewall rules for use with iptables-restore
- **`iptables-restore`**: Restores iptables-save format firewall rules - useful to set up firewalls at boot
- Consider `iptables` init script for save/restore. Config file:

  `/etc/sysconfig/iptables`

IPTABLES RULES

- When creating a new rule, considerations include:
  - What chain should the rule apply to?
  - What is the traffic pattern to look for?
  - What should happen with the traffic?

IPTABLES CHAINS

- **`INPUT`**
  - This chain is responsible for filtering traffic destined for the local system.
- **`OUTPUT`**
  - This chain is responsible for handling outbound traffic.
- **`FORWARD`**
  - This chain is responsible for controlling traffic routed between different interfaces.

IPTABLES RULES

- Below are a few examples of possible IPTables match criteria:
  - incoming interface `-i`
  - protocol `-p`
  - source ip address `-s`
  - destination ip address `-d`
  - destination port `--dport`

IPTABLES RULES

- Finally, some examples of what to do with traffic when matched:
  - **`DROP`** Do not deliver, do not respond
  - **`REJECT`** Do not deliver, send reject notice
  - **`ACCEPT`** Deliver
  - **`LOG`** Just log the packet

IPTABLES RULES

- So to summarize the syntax:
    - `iptables`
    - What chain should the rule apply to?
        - `-A INPUT`
    - What is the traffic pattern to look for?
        - `-s 192.168.2.100`
    - What should happen with the traffic?
        - `-j REJECT`

LAB

1. Using `iptables`, configure your web server to NOT accept connections from the `192.168.1.0/24` network, EXCEPT for the ip address of whomever is sitting next to you. Work together to test the firewall settings, and remember, **web** server. :)

2. Browse through the man page for iptables.

3. Use `iptables` to allow ssh from the classroom network only.

```
slideshow.end();
```

RHCSA
BOOT CAMP

Virtualization

VIRTUALIZATION

- RHEL 6 virtualization is accomplished via:
  - KVM - Kernel-based Virtualization Machine
  - QEMU - Processor emulator
- RHEL 6 only supports virtualization via KVM/QEMU, and only on 64bit systems supporting virtualization extensions
  - Intel: Intel VT (flag: `vmx`)
  - AMD: AMD-V (flag: `svm`)

PACKAGES

- There are four package groups available to install the necessary and ancillary software to support virtualization.
  - `Virtualization`
  - `Virtualization Client`
  - `Virtualization Platform`
  - `Virtualization Tools`

LIBVIRT

- `libvirt` is the management framework used in RHEL 6 virtualization.
- The `libvirtd` daemon will always be running in the background to handle virtualization needs and management requests such as starting, stopping, installing, etc.
- Interface to `libvirt` is provided by:
  - `virsh` - command line client
  - `virt-manager` - GUI client

SCHEDULE FOR TOMORROW

- Exam starts at 9:00am **SHARP**
- Exam concludes at 11:30am.
- Lunch from 11:30am to 1:00pm
- Review exam on projector from 1:00pm until finished
- Final Q/A session
- Survey Monkey!

DEMONSTRATION

- A demonstration of basic virtualization tasks...

LAB

1. Create a VM on your machine using the RHEL 6 sources available on server1.

2. Make sure the guest starts on host reboot.

```
slideshow.end();
```