RHCE
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
  - 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();
```