Booting sequence of UNIX systems

Remco Hobo

September 20, 2004

Preface

In this document I will discuss the different Bootup procedures of Gentoo Linux, BSD, and Mac OS X 10.3. I will look at the similarities between these and their main differences.

Gentoo linux

- When an I386 computer starts, it will first start with initializing devices like the video card. Next it will scan for hard drives, cd-rom players etc. After this (The post-process), the computer will look in a pre-defined order for bootable devices. In this case it will start from the hard drive.
- The bootloader will be loaded. In this case Yaboot. The old kernel, which was loaded from the BIOS, is replaced with the Yaboot kernel.
- When Yaboot selects a partition to boot from, the kernel on this partition will be unpacked and executed, replacing the Yaboot kernel.
- Yaboot will load the kernel image into memory after which it tells the CPU to run the kernel. When the kernel is loaded and run, it initializes all kernel-specific structures and tasks and starts the init process. This process then makes sure that all filesystems (defined in /etc/fstab) are mounted and ready to be used. Then it executes several scripts located in /etc/init.d, which will start the services you need in order to have a successfully booted system. Following steps are executed:
  - The filesystems are mounted from /etc/inittab. This is done with the command: si::sysinit:/sbin/rc sysinit

  - All scripts that have symlinks to /etc/runlevels/boot are executed. This is done with the command: rc::bootwait:/sbin/rc boot
  - Init will how check which runlevel should be executed, it reads this from /etc/inittab. This file contains a line like: id:3:initdefault indicating a runlevel of 3.
  - Init will now check what it should do with runlevel 3. Runlevel 3 is defined as:. l3:3:wait:/sbin/rc default
  - Init will now start the services with the default argument.
- Finally, when all scripts are executed, init activates the terminals (in most cases just the virtual consoles which are hidden beneath Alt-F1, Alt-F2, etc.) attaching a special process called agetty to it. This process will then make sure you are able to log on through these terminals by running login.
  - The terminals are activiated with lines like: c1:12345:respawn:/sbin/agetty 38400 tty1 linux

As an attachment, the inittab script can be found

FreeBSD

- When an I386 computer starts, it will first start with initializing devices like the video card. Next it will scan for hard drives, cd-rom players etc. After this (The post-process), the computer will look in a pre-defined order for bootable devices. In this case it will start from the hard drive.
- The boot0 stage will be initialized, showing the different devices to boot from
- The boot2 stage is initialized
- Load stage, showing something similair:
  - BTX loader 1.0 BTX version is 1.01
  - BIOS drive A: is disk0
  - BIOS drive C: is disk1
  - BIOS 639kB/64512kB available memory

  - FreeBSD/i386 bootstrap loader, Revision 0.8
  - Console internal video/keyboard
  - (jkh@bento.freebsd.org, Mon Nov 20 11:41:23 GMT 2000)
  - /kernel text=0x1234 data=0x2345 syms=[0x4+0x3456]
  - Hit [Enter] to boot immediately, or any other key for command prompt
  - Booting [kernel] in 9 seconds...

- The kernel is loaded:
  - Copyright (c) 1992-2002 The FreeBSD Project.
  - Copyright (c) 1979, 1980, 1983, 1986, 1988, 1989, 1991, 1992, 1993, 1994
  - The Regents of the University of California. All rights reserved.
  - FreeBSD 4.6-RC #0: Sat May 4 22:49:02 GMT 2002
  - devnull@kukas:/usr/obj/usr/src/sys/DEVNULL
  - Timecounter "i8254" frequency 1193182 Hz

After this, things will be executed in user level (super user)

- sys/kern/init_main.c will be run,
- /sbin/init will take over, and /etc/default/rc.conf script will be run.
- After this, the inlog screen appears.

Mac OS X 10.3

- A Mac computer works quite differently then the two mentioned above. A Mac uses open firmware; this means it will make a tree of all devices that are available on the Mac. It will also check for bootable devices attached to the computer. This means it will check the network for a bootable image, the cd-rom player, hard drives, bootable firewire devices, bootable USB devices etc.
- The device tree looks something like this:
  - 0 ¿ dev / ls
  - ff880d90: /cpus

  - ff881068: /PowerPC,750@0
  - ff881488: /l2-cache
  - ff882148: /chosen
  - ff882388: /memory@0
  - ff882650: /openprom
  - ff882828: /client-services
  - The Open Firmware uses a FCode dialect (FCode is an ANS Forth compliant dialect that supports compilation of FCode source to bytecode)

- Control passes to /System/Library/CoreServices/BootX, the boot loader. BootX loads the kernel and also draws the OS badges, if any.

- BootX tries to load a previously cached list of device drivers (created/updated by /usr/sbin/kextcache). Such a cache is of the type mkext and contains the info dictionaries and binary files for multiple kernel extensions.

- The init routine of the kernel is executed. The root device of the booting system is determined. At this point, Open Firmware is not accessible any more.

- Various Mach/BSD data structures are initialized by the kernel.

- The I/O Kit is initialized.

- The kernel starts /sbin/mach_init, the Mach service naming (bootstrap) daemon. mach_init maintains mappings between service names and the Mach ports that provide access to those services.

From this point, startup becomes user level (of course, it still runs as superuser)

- mach_init starts /sbin/init, the traditional BSD init process. init determines the runlevel, and runs /etc/rc.boot, which sets up the machine enough to run single-user.

- rc.boot figures out the type of boot (Multi-User, Safe, CD-ROM, Network etc.). In case of a network boot (the sysctl variable kern.netboot will be set to 1 in which case), it runs /etc/rc.netboot with a start argument.

- rc.boot figures out if a file system consistency check is required. Single-user and CD-ROM boots do not run fsck. SafeBoot always runs fsck. rc.boot handles the return status of fsck as well.
- If rc.boot exits successfully, /etc/rc, the multi-user startup script is then run. If booting from a CD-ROM, the script switches over to /etc/rc.cdrom (installation).
- /etc/rc mounts local file systems (HFS+, HFS, UFS, /dev/fd, /.vol), ensures that the directory /private/var/tmp exists, and runs /etc/rc.installer_cleanup, if one exists (left by an installer before reboot).
- /etc/rc.cleanup is run. It "cleans" a number of Unix and Mac specific directories/files.
- BootCache is started.
- Various sysctl variables are set (such as for maximum number of vnodes, System V IPC, etc.). If /etc/sysctl.conf exists (plus /etc/sysctl-macosxserver.conf on Mac OS X Server), it is read and sysctl variables contained therein are set.
- syslogd is started.
- The Mach symbol file is created.
- /etc/rc starts kextd, the daemon process that loads kernel extension on demand from kernel or client processes.
- /usr/libexec/register_mach_bootstrap_servers is run to load various Mach bootstrap based services contained in /etc/mach_init.d
- portmap and netinfo are started.
- If /System/Library/Extensions.mkext is older than /System/Library/Extensions, /etc/rc deletes the existing mkext and creates a new one. It also creates one if one doesn't exist.
- /etc/rc starts /usr/sbin/update, the daemon that flushes internal file system caches to disk frequently.
- /etc/rc starts the virtual memory system. /private/var/vm is set up as the swap directory. /sbin/dynamic_pager is started with the appropriate arguments (swap filename path template, size of swap files

created, high and low water alert triggers specifying when to create additional swap files or delete existing ones).

- /etc/rc starts /usr/libexec/fix_prebinding to fix incorrectly prebound binaries.
- /etc/rc executes /etc/rc.cleanup to clean up and reset files and devices.
- /etc/rc finally launches /sbin/SystemStarter to handle startup items from locations such as /System/Library/StartupItems and /Library/StartupItems. A StartupItem is a program, usually a shell script, whose name matches the folder name. The folder contains a property list file containing key-value pairs such as Description, Provides, Requires, OrderPreference, start/stop messages etc. You can run SystemStarter -n -D as root to have the program print debugging and dependency information (without actually running anything).
- The CoreGraphics startup item starts the Apple Type Services daemon (ATSServer) as well as the Window Server (WindowServer).

Conclusion

Since Mac OS X 10.3 will only run on an Mac, it's booting procedures where quite different then the other two. Also, a Mac normally boots up to a graphical interface, which is much more elaborate to initialize then a simple login prompt.

- Gentoo uses symlinks from /etc/rc?.d to /etc/init.d/ to start all services
- FreeBSD uses /etc/default/rc.conf to start all services
- Mac uses the /etc/rc script to start all services

All of the three above use a unix kernel, hand-crafted to it's own needs. The basis is the same, it is only configurated for specific needs.

References

[1] http://www.kernelthread.com/mac/osx/arch_boot.html

[2] http://www.freebsd.org/doc/en_US.ISO8859-1/books/arch-handbook/boot.html

Attachment: The Gentoo inittab script

Most of this information comes from /etc/inittab, which looks somewhat like:

```
    #
    # /etc/inittab: This file describes how the INIT process should set up
    # the system in a certain run-level.
    #
    # Author: Miquel van Smoorenburg, ¡miquels@cistron.nl¿
    # Modified by: Patrick J. Volkerding, ¡volkerdi@ftp.cdrom.com¿
    # Modified by: Daniel Robbins, ¡drobbins@gentoo.org¿
    # Modified by: Martin Schlemmer, ¡azarah@gentoo.org¿
    #
    # Header: /home/cvsroot/gentoo-src/rc-scripts/etc/inittab,v 1.6 azarah
Exp
    #
    # Default runlevel.
    #i d:3:initdefault:
    # System initialization, mount local filesystems, etc.
    # si::sysinit:/sbin/rc sysinit
    # Further system initialization, brings up the boot runlevel.
    # rc::bootwait:/sbin/rc boot
    # l0:0:wait:/sbin/rc shutdown
    # l1:S1:wait:/sbin/rc single
    # l2:2:wait:/sbin/rc nonetwork
    # l3:3:wait:/sbin/rc default
    # l4:4:wait:/sbin/rc default
    # l5:5:wait:/sbin/rc default
    # l6:6:wait:/sbin/rc reboot
    #z6:6:respawn:/sbin/sulogin
    # TERMINALS
    # c1:12345:respawn:/sbin/agetty 38400 tty1 linux
    # c2:12345:respawn:/sbin/agetty 38400 tty2 linux
    # c3:12345:respawn:/sbin/agetty 38400 tty3 linux
    # c4:12345:respawn:/sbin/agetty 38400 tty4 linux
    # c5:12345:respawn:/sbin/agetty 38400 tty5 linux
    # c6:12345:respawn:/sbin/agetty 38400 tty6 linux
    # What to do at the "Three Finger Salute".
    # ca:12345:ctrlaltdel:/sbin/shutdown -r now
    # Used by /etc/init.d/xdm to control DM startup.
    # Read the comments in /etc/init.d/xdm for more
    # info. Do NOT remove, as this will start nothing
    # extra at boot if /etc/init.d/xdm is not added
    # to the "default" runlevel.
    # x:a:once:/etc/X11/startDM.sh
    # End of /etc/inittab
```