The journal of Apple technology.

Volume Number: 19 (2003)
Issue Number: 8
Column Tag: Programming

Section 7

The Process Tree

by Rich Morin

Which processes do what

The initial design of Unix (by Ken Thompson, Dennis Ritchie, et al) was based on the idea that large jobs could be done by lots of little programs, working together. Part of the reason for this was that the available hardware had only minuscule amounts of RAM. Another part was the designers' dislike of large, monolithic programs.

Mac OS X inherits this modular approach. Most of its programs aren't "little", especially by the standards of early Unix, but there are certainly lots of them. On my freshly-booted desktop machine, I found 62:

```
% ps -ax | wc -l
      62
```

The command-line options (-ax) tell ps to show all processes, even if they belong to another user or have no controlling terminal. Using another of ps's many options, we can take a detailed look, concentrating on the commands and their parentage. Stretch a Terminal window really wide, then try:

```
% ps -axo pid,ppid,tt,user,command
  PID  PPID  TT  USER COMMAND
    1     0  ??  root /sbin/init
    2     1  ??  root /sbin/mach_init
   51     1  ??  root kextd
71     1  ??  root update
...
  539   536 std  root login -pf rdm
  540   539 std  rdm  -tcsh (tcsh)
1506   540 std  root ps -axo pid ppid tt user command
 674   536  p2  root login -pf rdm
 675   674  p2  rdm  -tcsh (tcsh)
```

The PID (Process ID) and PPID (Parent Process ID) columns allow us to guess at the "genealogy" of each process. If a process has PPID 1, for instance, its "parent" has PID 1. Note, however, that a PPID of 1 may also mean that the original parent process has terminated.

The TT ("terminal"; originally from teletype) column tells us the identity of the controlling terminal ("??" for none). Because most OSX applications have no controlling terminal, this field is insufficient to distinguish apps from background processes.

Fortunately, the USER (username) and COMMAND columns help to clear this up. Apps generally run with the username of the logged-in user; in addition, most apps have long path names which include directories such as "Applications". Also, the names of BSD-derived daemons often end in "d" (e.g., configd, cupsd).

If you're seriously interested in looking at processes (e.g., to figure out which one is bogging down your system :-), you should also investigate pstat(1), top(1), vm_stat(1). Apple's Process Viewer utility is also handy, but note that it may not report information in a manner that is consistent with the command-line tools. For example, it shows a process named "Window Manager", which ps(1) and top(1) do not list...

Background processes

When OSX starts up, it initiates a number of background processes (commonly referred to as "daemons", after the "attendant spirits" found in Greek mythology). Because these daemons have low Process IDs, they appear at the start of the listing.

Note: Although OSX uses a 32-bit value (pid_t) for process IDs, it "wraps" the IDs at ~32K, so a ps(1) listing taken from a long-running (and/or busy) system may show other processes with low IDs. Interestingly, FreeBSD does not seem to limit IDs in this manner.

Most daemons are started up by init(8), as part of the system start-up process. This accounts for the long list of processes whose PPID is 1. Scattered in this list, however, you may see a few processes with PPIDs of 2, indicating that they are children of mach_init(8). Finally, some applications (e.g., iChat) and daemons (e.g., loginwindow, WindowServer) start up their own daemons.

Here's an annotated list of some background processes that show up on my machine. Lacking any other organizing rationale, I'll use alphabetical order. If nothing else, that will make the list easy to navigate (:-):

AppleFileServer - personal file sharing server; supports AppleTalk Filing Protocol (AFP) over Internet Protocol (IP).

ATSServer - Apple Type Solution server; enables system-wide font management.

autodiskmount(8) - checks and mounts disk-based file systems.

automount - automatically mounts and unmounts network (NFS and AFP) file systems. See amd(8) for information on the traditional (NFS-only) variant of this daemon.

configd - maintains dynamic configuration information about the computer and its environment (e.g. network).

coreservicesd - core services daemon.

crashreporterd - logs information about program crashes.

cron(8) - executes scheduled commands. See also crontab(1,5).

cupsd(8) - Common Unix Printing System daemon. Run "apropos cups" to see an extensive list of man pages.

DirectoryService - directory server for Apple's Open Directory architecture.

dynamic_pager - assists the kernel with managing swap files for virtual memory.

httpd(8) - Apache hypertext transfer protocol server; may be replicated, for performance. Run "apropos http" to see an extensive list of man pages.

inetd(8) - internet "super-server"; listens for connections on certain internet sockets. When a connection is found on one of its sockets, it decides what service the socket corresponds to, and invokes a program to service the request.

kextd(8) - handles requests from the kernel to load kernel extensions (kexts).

loginwindow - handles miscellaneous process- and session-monitoring functions (e.g., login, logout, restart, shutdown, and restarting of the Dock and Finder).

lookupd(8) - caches directory service information (e.g., user accounts, groups, printers, e-mail aliases and distribution lists, computer names, Internet addresses).

mDNSResponder - multicast-DNS responder; advertises network services (such as AFP file sharing) provided by this computer (part of Rendezvous).

netinfod(8) - serves NetInfo information to the network. Run "apropos netinfo" to see an extensive list of man pages.

nfsiod(8) - typically, several copies of this daemon will be running on any NFS client machine, servicing asynchronous I/O requests. This daemon improves NFS performance, but it is not required for correct operation. Run "apropos nfs" to see an extensive list of man pages (but ignore the bogus hits for config(5) and confstr(3) :-).

ntpd(8:FreeBSD) - Network Time Protocol (NTP) daemon; sets and maintains the system time of day in synchronism with Internet standard time servers.

pbs - pasteboard server (similar to the Clipboard in Mac OS 9); enables the exchange of data between applications. It is also the data-transfer mechanism used in dragging operations. (started by loginwindow).

SecurityServer - oversees system authorization, authentication, and keychain access.

slpd - Service Location Protocol (SLP) responder; advertises network services (such as AFP file sharing) provided by this computer.

sshd(8) - with ssh(1), replaces rlogin(1) and rsh(1), providing secure encrypted communications between two untrusted hosts over an insecure network. Run "apropos ssh" to see an extensive list of man pages.

syslogd(8) - reads and logs messages to the system console, log files, other machines, and/or users as specified by syslog.conf(5), its configuration file. See also syslog(3).

SystemUIServer - displays items on the right-hand end of the menu bar; loads menu and dock extras as plug-ins (started by WindowServer).

update(8) - helps protect the integrity of disk volumes by flushing volatile cached file system data to disk at thirty second intervals.

WindowServer - responsible for rudimentary screen displays, window compositing and management, event routing, and cursor management. It coordinates low-level windowing behavior and enforces a fundamental uniformity in what appears on the screen.

Interactive Processes

Most apps are children of the WindowServer daemon. Their names should be quite familiar: Dock, Finder, iChat, Preview, Terminal. As noted above, some apps start up their own daemons. For example, iChat starts up iChatAgent (probably to handle communications).

The lineage for command-line programs is a bit more complex. Assuming that you're using Terminal, it should look something like this:

```
PID PPID  TT  USER COMMAND
173    1  ??  rdm  .../WindowServer ...
536  173  ??  rdm  .../Terminal ...
539  536  p1  root login -pf rdm
540  539  p1  rdm  -tcsh (tcsh)
873  540  p1  root top
```

As these lines show, init(8) started up WindowServer, which started up the Terminal. Then, to generate an interactive shell window, Terminal started up login, which started up tcsh(1). Finally, I ran top(1) from the command line.

Careful Reader will notice that the username for login and top is root, rather than rdm. Because these processes need to do some things which a normal user is not allowed to do, they have been created as setuid(2) executables:

```
% ls -l /usr/bin/{login,top}
-r-sr-xr-x 1 root wheel ... /usr/bin/login
-r-sr-xr-x 1 root wheel ... /usr/bin/top
```

Further Reading

In a traditional BSD system, the first place to look for information on commands and daemons would be the manual. Unfortunately, many OSX commands and daemons do not have accompanying manual pages. Even when man pages are present, they may not be up-to-date, let alone customized to reflect changes which Apple has made, other documentation it has developed, etc.

A comprehensive attack on this problem has been suggested by parties inside and outside of Apple, but (AFAIK) no significant effort has been made as yet. If you would like better manual pages, let ADC (etc.) know!

In the meanwhile, here are some places where I found useful information for this section:

http://www.westwind.com/reference/OS-X/background-processes.html

http://developer.apple.com/techpubs/macosx/Essentials/SystemOverview/ BootingLogin/chapter_4_section_5.html

---

Rich Morin has been using computers since 1970, Unix since 1983, and Mac-based Unix since 1986 (when he helped Apple create A/UX 1.0). When he isn't writing this column, Rich runs Prime Time Freeware (www.ptf.com), a publisher of books and CD-ROMs for the Free and Open Source software community. Feel free to write to Rich at rdm@ptf.com.