Startup, Login, Logout scripts

By
James Reynolds
Startup, Login, Logout scripts

- **Startup script?**
  - Runs at startup time

- **Login script?**
  - Runs right after user authenticates
  - Runs before Finder loads

- **Logout script?**
  - Runs right after Finder quits
  - Runs before login panel is displayed
Why learn any of this?

• Debug existing startup/login/logout scripts
  – Many existing applications install bad startup/login/logout scripts (TB2, KeyServer).

• Create your own system customizations
  – Execute a script or tool with root permissions
  – Custom administrative, accounting, or security tools/scripts
Why learn any of this?

• Startup customizations
  – Check machine status
  – Run maintenance like RsyncX or Radmind
  – Check permissions
  – Startup daemons
Why learn any of this?

• Login customizations
  – Delete home folder & replace with template
  – Debug login problems (when using Directory Services)
  – Check login username
    • Retrieve network home folder
    • Keep a log
    • Do something different for each user
      – Give elevated access (but not admin) to “super-users”
      – Notify the campus police for bad guys
Why learn any of this?

- Logout customizations
  - Perform home folder backup
  - Sync home folder to a network home folder server
  - Clean up after the user
Security

• Startup items, login, & logout scripts run as root!
• Prevent your script from being “hijacked”, use secure script writing practices
  – Scripts CANNOT be writable by “group” or “other”
  – Specify paths to commands
    • “/bin/reboot” NOT “reboot”
  – Every file interaction is a possible security breach
    • Check everything the startup/login/logout script executes
Startup Items

- Examples:
  - Apple/System
    - Network
    - Disks
    - Apache
    - ssh
    - loginwindow
  - 3rd party
    - PostgreSQL
    - Timbuktu
What is a startup item

• Parent folder
  – Name of the startup item

• Script
  – Must have same name as parent folder
  – The real guts

• Resources
  – Extra files

• Description
  – StartupParameters.plist
Parent Folder

- Contains everything you need
- Should have self explanatory name
- Where are they?
  - Custom startup items
    - /Library/StartupItems/
      - This folder does not exists after a vanilla install
  - System startup items
    - /System/Library/StartupItems/
Script

- Must have user executable set
- Should respond to “start”, “stop”, “restart”, like:
  
  ```bash
  /sbin/SystemStarter restart /System/Library/StartupItems/Apache
  ```
- Can be any executable, best to use:
  - `/bin/sh`
    - Has a script skeleton
    - Has a common file with useful functions
#!/bin/sh

##
# Your script name
##

. /etc/rc.common

StartService ()
{
}

StopService ()
{
}

RestartService ()
{
}

RunService "$1"
Script skeleton

• StartService ()
  – Startup code should go here

• StopService ()
  – Used for daemons, like apache
  – Shutdown code should go here
  – Shutdown or Restart does not call this function
    • It should be called, especially for databases
    • Apple may eventually fix it

• RestartService ()
  – Used for daemons, like apache
/etc/rc.common

• Tasks
  – Export Paths
    • /bin
    • /sbin
    • /usr/bin
    • /usr/sbin
    • /usr/libexec
    • /System/Library/CoreServices
  – Imports /etc/hostconfig
  – Enables CoreDumps
/etc/rc.common

• Functions:
  – ConsoleMessage()

    ConsoleMessage "Restarting Apache web server"

  – CheckForNetwork()

    CheckForNetwork
    if [ "${NETWORKUP}" = "-NO-" ]; then exit; fi

  – GetPID()

    • Returns the PID
/etc/rc.common

- Functions:
  - RunService($1)
    - Calls StartService if $1 is “start”
    - Calls StopService if $1 is “stop”
    - Calls RestartService if $1 is “restart”
  - purgedir()
    - Delete contents of directory, but leave directory structure alone
Example startup item

. /etc/rc.common
StartService()
{
  if [ "${WEBSERVER:=NO-}" = "YES-" ]; then
    ConsoleMessage "Starting Apache web server"
    apachectl start
  fi
}

StopService()
{
  ConsoleMessage "Stopping Apache web server"
  apachectl stop
}

RestartService()
{
  if [ "${WEBSERVER:=NO-}" = "YES-" ]; then
    ConsoleMessage "Restarting Apache web server"
    apachectl restart
  else
    StopService
  fi
}

RunService "$1"
Example startup item

• Lots of examples in /System/Library/StartupItems
Resources

• Put extra files in here
  – Localization
  – Images
  – Other
    • scripts
    • daemons
    • whatever you want

• Not required
StartupParameters.plist

• Description
  – This is displayed in the startup window

• Provides
  – This is what other scripts may "Require" or "Use"
  – Should be unique

• If 2 have the same provides, the first startup item encountered will be executed
StartupParameters.plist

• Requires
  – Startup items that MUST finish before this one can run

• Uses
  – Startup items that should finish before this one can run
  – Startup item should be able to start without the availability of these Startup items
StartupParameters.plist

- OrderPreference
  - Advisory value if 2 startup items have identical "Requires" and "Uses"
    - "First"
    - "Early"
    - "None" (default)
    - "Late"
    - "Last"
  - This value might be ignored
Example StartupParameters.plist

```
{
    Description = "Apache web server";
    Provides    = ("Web Server");
    Requires    = ("DirectoryServices");
    Uses        = ("Disks", "NFS", "Network Time");
    OrderPreference = "None";
}
```
When do startup items execute?

BootROM

/System/Library/CoreServices/BootX
/sbin/mach_init
/sbin/init
/private/etc/rc.boot
/private/etc/rc
/System/Library/CoreServices/SystemStarter
/System/Library/StartupItems/*
/Library/StartupItems/*
SystemStarter

- SystemStarter executes valid startup items in
  - /System/Library/StartupItems
  - /Library/StartupItems
- Startup order
  - StartupParameters.plist
    - Script with no “Requires” or “Uses” executed first
      - “System Tuning”
    - Typical order: "System Tuning", "ConfigServer", "Network", "Disks" and then startup items that require network and disks
More info StartupItems

• Creating startup items

• About SystemStarter
  – man SystemStarter

• The boot process

• Learning BASH scripting
Login/Logout Scripts (Hooks)

- Special feature provided by Apple
- By default, it is not used
- Some 3rd party applications use it
  - KeyServer
    - If you use KeyServer and you create your own LoginHook, you will need to add KeyServer’s LoginHook to your LoginHook since you can only have one at a time
What is a Login/Logout Hook?

• Any executable file, script or tool

• Must have user executable set

• Must exit with 0
How to enable Hooks?

• Edit /etc/ttys
  – Change:
    console "/System/Library/CoreServices/loginwindow.app/Contents/
    MacOS/loginwindow" vt100 on secure window="/System/Library/
    CoreServices/WindowServer onoption="/usr/libexec/getty std.9600"
    #tahoe's only
  – To:
    console "/System/Library/CoreServices/loginwindow.app/Contents/
    MacOS/loginwindow -LoginHook /path/to/login_executable -LogoutHook
    /path/to/logout_executable" vt100 on secure window="/System/Library/
    CoreServices/WindowServer onoption="/usr/libexec/getty std.9600"
    #tahoe's only
  – Do not add line breaks!!!
    • Mac will not login, need to boot to single usermode to fix.
Fixing a broke /etc/ttys file

• Reboot Mac
• Boot into single user mode
  – Hold Command-s until black display with text appears
• Type “fsck -y”
  – If there are errors, repeat until there are no errors
• Type “mount -uw /”
• Either restore a backup of ttys or edit the file with “pico /etc/ttys” (or “vi” / “emacs”)
• Type “reboot”
Hook parameters?

• This will not work:

  console "/System/Library/CoreServices/loginwindow.app/Contents/MacOS/
  loginwindow -LoginHook /path/to/login_executable parameter -LogoutHook /
  path/to/logout_executable" vt100 on secure window=/System/Library/
  CoreServices/WindowServer onoption="/usr/libexec/getty std.9600"
  #tahoe's only

• loginwindow already passes in a parameter:
  BASH: echo $1
  Perl: print $ARGV[0];
  Will print the username logging in

• You can not grab the password! Sorry!
When LoginHook?

/sbin/init
/private/etc/rc.boot
/private/etc/rc
/System/Library/CoreServices/SystemStarter
/usr/libexec/getty /private/etc/ttys /System/Library/CoreServices/loginwindow.app/Contents/MacOS/loginwindow

• User logs in
• LoginHook executed
• Login panel goes away, Finder loads, etc.,
When LogoutHook?

/sbin/init
/private/etc/rc.boot
/private/etc/rc
   /System/Library/CoreServices/SystemStarter
/usr/libexec/getty /private/etc/ttys /System/Library/CoreServices/loginwindow.app/Contents/MacOS/loginwindow

• User logs out
• Finder and all other applications quit
• LogoutHook executed
• Login panel loads (background graphic also changes)
More info Login/Logout Hooks

• Login and Logout Procedures
    SystemOverview/BootingLogin/chapter_4_section_14.html
  – http://www.macosxlabs.org/documentation/
    hard_disk_maintenance/configuration/config.html
    http://www.bombich.com/mactips/loginhooks.html

• Old man pages for loginwindow
  – http://www.vorlesungen.uni-osnabrueck.de/informatik/shellscrip/
    Html/Man/_Man_NeXT_html/html8/loginwindow.8.html

• Nifty tool (LoginWindow Manager)