Chapter 3

System Startup and Shutdown

This chapter describes the procedures for starting and stopping your system, bringing your system to the various default levels of operation, and how to create your own custom levels of operation.

- Starting an IRIX system. See "Starting the System" on page 45.
- Shutting down an IRIX system. See "Shutting Down the System" on page 46.
- Definition of the operating system run levels; how they are controlled and how to change them. See "IRIX Operating Levels" on page 48.

Starting the System

To start up a system, follow these steps:

1. Make sure all cables (such as power, display monitor cable, and keyboard) are properly connected. See your owner's guide and hardware guide for complete information about cabling your particular workstation or server.
2. Turn on the power switches on the display monitor (or console terminal) and the computer.

   The computer runs power-on diagnostics and displays some copyright messages and some system startup information. These messages appear on the *console* screen or on the screen of a *diagnostics terminal* (an ASCII terminal connected to the first serial port) of a server. A copy of these messages is also written to the */var/adm/SYSLOG* file in case you miss them.

   If you are restarting the system after a power loss or some other unexpected shutdown, you may see an error message regarding your SCSI bus or other hardware problem. This may be a temporary condition based on the disks' need to spin up to using speed or the SCSI bus may simply need a moment to reset itself. Wait 30 seconds and attempt to boot the operating system again.

If the operating system determines that the file systems need checking, it checks them with the *fsck* program. *fsck* fixes any problems it finds before the operating system mounts the file systems. *fsck* will run if the system is not shut down properly, such as in the event of a power failure. For information about using *fsck*, see the *IRIX Admin: Disks and Filesystems* guide and the *fsck*(1M) reference page. Note that it is not necessarily a problem if *fsck* runs, it is merely a precaution.

The system now comes up in multiuser mode and you can log in. You should leave your system running at all times. The IRIX operating system works best when it is allowed to run continuously, so that scheduled operations and "housekeeping" processes can be performed on schedule.

Shutting Down the System

This section describes how to turn off a workstation or server from multiuser or single-user mode.

Shutting Down from Multiuser Mode

To shut down the system from multiuser mode:

1. Use the *who*(1) command to determine which users are logged in to the operating system, if any:

   who

   Notify any users that the system is shutting down. Issue the */etc/wall*(1M) command:

   wall

   Enter your message. For example, you might enter:

   ```
   There is a problem with the building's power system.
   I will be shutting down the system in 10 minutes.
   Please clean up and log off.
   Sorry for the inconvenience,
   norton
   ```

2. When you finish entering your message, type **<Ctrl-D>**. The message is sent to all users on the system. They see something like this:

   ```
   Broadcast Message from root Tue Oct 17 17:02:27...
   There is a problem with the building's power system.
   I will be shutting down the system in 10 minutes.
   ```

```
Please clean up and log off.
Sorry for the inconvenience,
norton
```

3. Issue the */etc/shutdown* command:

   `/etc/shutdown -y -i0 -g600`

   The above command specifies a 10 minute (600 second) grace period to allow users to clean up and log off. The other flags indicate that the system is to be completely shut down (**-i0**) and that the system can assume that all answers to any prompts regarding the shutdown are "yes." You see the following message:

   ```
   Shutdown started. Fri Aug 28 17:10:57...
   Broadcast Message from root (console) Fri Aug 28 17:10:59
   The system will be shut down in 600 seconds.
   Please log off now.
   ```

   After ten minutes, you see this message:

   ```
   INIT: New run level: 0
   The system is coming down. Please wait.
   ```

   The Command Monitor prompt or System Maintenance menu appears. Wait for a Command Monitor prompt or System Maintenance menu to appear before turning off power to the workstation or you may damage your hard disk.

4. You can now turn off the power.

For more information on shutting down the system, see the *halt*(1M) and *shutdown*(1M) reference pages. Remember that you should shut down the system only when something is wrong or if modifications to the software or hardware are necessary. IRIX is designed to run continuously, even when no users are logged in and the system is not in use.

Turning Off from Single-user Mode

If the system is in single-user mode, follow these steps:

1. Use the *shutdown* command to turn off the system and guarantee file system integrity. As *root*, enter the command:

   `shutdown -y -i0 -g0`

   where:

   -y assumes yes answers to all questions, **-i0** goes to state 0 (System Maintenance Menu), and **-g0** allows a grace period of 0 seconds.

You see a display similar to this one:

```
Shutdown started. Fri Aug 28 17:11:50 EDT 1987
INIT: New run level: 0
The system is coming down. Please wait.
```

The system stops all services and the Command Monitor prompt or System Maintenance Menu appears.

Wait for the Command Monitor prompt or System Maintenance menu to appear or for a message that the system can be powered off before turning off power to the computer. Doing so prematurely may damage your hard disk.

2. Turn off power to the computer.

IRIX Operating Levels

The IRIX system can run in either single-user or multiuser mode. In single-user mode, only a few processes are active on the system, no graphics are available, and only a single login is allowed. In multiuser mode, there can be multiple login sessions, many files open at once, and many active processes, including numerous background daemons.

The *init* program controls whether the system is in the multiuser or single-user state. Each possible state that the system can be in is assigned a label, either a number or a letter. The shutdown state is state *0*. Single-user mode is state *s*.

Multiuser state labeling is more complex, because there can be great variations in multiuser states. For example, in one multiuser state, there can be unlimited logins, but in another state there can be a restricted number of logins. The different states can all be assigned different numbers.

The state of the system is controlled by the file */etc/inittab*. This file lists the possible states, and the label associated with each.

When you bring the system to standard multiuser mode, *init* state 2, the following happen:

- The file systems are mounted.
- The *cron* daemon is started for scheduled tasks.
- Network services are started, if turned on.
- The serial-line networking functions of *uucp* are available for use.

- The spooling and scheduling functions of the *lp* package (if added to the system) are available for use.
- Users can log in.

Not all activities can or should be performed in the multiuser state. Some tasks, such as installing software that requires the miniroot and checking file systems must be done with the system in single-user mode.

There are many synonyms for the system state. These include:

- init state
- run state
- run level
- run mode
- system state

Likewise, each system state may be referred to in a number of ways; for example:

- single user
- single-user mode
- run level 1

Table 3-1 shows the various possible states of the operating system as it is shipped. You can, of course, create your own custom states.

Table 3-1 System States

Run Level	Description
0	Power-down state.
1, s, or S	Single-user mode is used to install/remove software utilities, run file system backups/restores, and check file systems. This state unmounts everything except root, and kills all user processes except those that relate to the console.

Table 3-1 (continued) System States

Run Level	Description
2	Multiuser mode is the normal operating mode for the system. The default is that the *root* (*/*) and user (*/usr*) file systems are mounted in this mode. When the system is powered up, it is put in multiuser mode.
6	Reboot mode is used to bring down the system and then bring it back up again. This mode is useful when you are changing certain system configuration parameters.

How init Controls the System State

The *init* process is the first general process created by the system at startup. It reads the file */etc/inittab*, which defines exactly which processes exist for each run level.

In the multiuser state (run level 2), *init* scans the file for entries that have a tag (also 2) for the run level and executes everything after the third colon on the line containing the tag. For complete information, see the *inittab*(4) reference page.

The system */etc/inittab* looks something like this:

```
is:2:initdefault:
fs::sysinit:/etc/bcheckrc </dev/console >/dev/console 2>&1
mt::sysinit:/etc/brc </dev/console >/dev/console 2>&1
s0:06s:wait:/etc/rc0 >/dev/console 2>&1 </dev/console
s1:1:wait:/etc/shutdown -y -iS -g0 >/dev/console 2>&1 </dev/console
s2:23:wait:/etc/rc2 >/dev/console 2>&1 </dev/console
s3:3:wait:/etc/rc3 >/dev/console 2>&1 </dev/console
or:06:wait:/etc/umount -ak -b /proc,/debug > /dev/console 2>&1
of:0:wait:/etc/uadmin 2 0 >/dev/console 2>&1 </dev/console
RB:6:wait:echo "The system is being restarted." >/dev/console
rb:6:wait:/etc/uadmin 2 1 >/dev/console 2>&1 </dev/console
#
```

This display has been edited for brevity; the actual */etc/inittab* file is lengthy. If */etc/inittab* is removed by mistake and is missing during shutdown, *init* enters the single-user state (*init s*). While entering single-user state, */usr* remains mounted, and processes not spawned by *init* continue to run. Immediately replace */etc/inittab* before changing states again. The format of each line in *inittab* is:

id:*level*:*action*:*process*

where:

- *id* is one or two characters that uniquely identify an entry.
- *level* is zero or more numbers and letters (0 through 6, s, a, b, and c) that determine in what *level*(s) *action* is to take place. If *level* is null, the *action* is valid in all levels.
- *action* can be one of the following:
  - **sysinit**

    Run *process* before *init* sends anything to the system console (Console Login).
  - **bootwait**

    Start *process* the first time *init* goes from single-user to multiuser state after the system is started. (If *initdefault* is set to 2, the process runs right after the startup.) *init* starts the process, waits for its termination and, when it dies, does not restart the process.
  - **wait**

    When going to *level*, start *process* and wait until it has finished.
  - **initdefault**

    When *init* starts, it enters *level*; the *process* field for this *action* has no meaning.
  - **once**

    Run *process* once. If it finishes, don't start it again.
  - **powerfail**

    Run *process* whenever a direct powerdown of the computer is requested.
  - **respawn**

    If *process* does not exist, start it, wait for it to finish, and then start another.

  - **ondemand**

    Synonymous with *respawn*, but used only with *level* a, b, or c.

  - **off**

    When in *level*, kill *process* or ignore it.

- *process* is any executable program, including shell procedures.
- # can be used to add a comment to the end of a line. *init* ignores all lines beginning with the # character.

When changing levels, *init* kills all processes not specified for that level.

Entering the Multiuser State from System Shutdown

When your system is up and running, it is usually in multiuser mode. It is only in this mode that the full power of IRIX is available to your users.

Powering Up the System

When you power up your system, it enters multiuser mode by default. (You can change the default by modifying the *initdefault* line in your *inittab* file.) In effect, going to the multiuser state follows these stages (see Table 3-1, " System States," on page 49):

1. The operating system loads and the early system initializations are started by *init*.
2. The run level change is prepared by the */etc/rc2* procedure.
3. The system is made public through the spawning of *getty* processes along the enabled terminal lines.

Early Initialization

Just after the operating system is first loaded into physical memory through the specialized boot programs that are resident in the PROM hardware, the *init* process is created. It immediately scans */etc/inittab* for entries of the type *sysinit*:

```
fs::sysinit:/etc/bcheckrc </dev/console >/dev/console 2>&1
mt::sysinit:/etc/brc </dev/console >/dev/console 2>&1
```

These entries are executed in sequence and perform the necessary early initializations of the system. Note that each entry indicates a standard input/output relationship with

/dev/console. This establishes communication with the system console before the system is brought to the multiuser state.

Preparing the Run Level Change

Now the system is placed in a particular run level. First, *init* scans the table to find an entry that specifies an *action* of the type *initdefault*. If it finds one, it uses the run level of that entry as the tag to select the next entries to be executed. In our sample */etc/inittab*, the *initdefault* entry is run level 2 (the multiuser state):

```
is:2:initdefault:
s2:23:wait:/etc/rc2 >/dev/console 2>&1 </dev/console
co:23:respawn:/etc/gl/conslog
t1:23:respawn:/etc/getty -s console ttyd1 co_9600 #altconsole
t2:23:off:/etc/getty ttyd2 co_9600 # port 2
t3:23:off:/etc/getty ttyd3 co_9600 # port 3
t4:23:off:/etc/getty ttyd4 co_9600 # port 4
```

The other entries shown above specify the actions necessary to prepare the system to change to the multiuser run level. First, */etc/rc2* is executed. It executes all files in */etc/rc2.d* that begin with the letter S, accomplishing (among other things) the following:

- Sets up and mounts the file systems
- Starts the *cron* daemon
- Makes *uucp* available for use
- Makes the line printer (*lp*) system available for use, if installed
- Starts accounting, if installed and configured on
- Starts networking, if installed and configured on
- Starts *sar*, if installed and configured on
- Starts the mail daemon
- Starts the system monitor

init then starts a *getty* for the console and starts *getty* on the lines connected to the ports indicated.

At this point, the full multiuser environment is established, and your system is available for users to log in.

Changing Run Levels

To change run levels, the system administrator enters a command that directs *init* to execute entries in */etc/inittab* for a new run level, such as *multi*, *single*, or *reboot*. Then key procedures, such as *shutdown*, */etc/rc0*, and */etc/rc2,* are run to initialize the new state.

The new state is reached. If it is state 1 (single-user mode), the system administrator can continue.

Run-level Directories

Run levels 0, 2, and 3 each have a directory of files that are executed in transitions to and from that level. These directories are *rc0.d*, *rc2.d*, and *rc3.d*, respectively. All files in these directories are linked to files in */etc/init.d*. The run-level file names look like this:

S00name

or

K00name

The filenames can be split into three parts:

S or K	The first letter defines whether the process should be started (S) or killed (K) upon entering the new run level.
00	The next two characters are a number from 00 to 99. They indicate the order in which the files will be started (S00, S01, S02, etc.) or stopped (K00, K01, K02, etc).
name	The rest of the filename is the */etc/init.d* filename to which this file is linked.

For example, the *init.d* file *cron* is linked to the *rc2.d* file and *rc0.d* file *S75cron*. When you enter *init 2*, this file is executed with the **start** option: **sh S75cron start**. When you enter *init 0*, this file is executed with the **stop** option: **sh K70cron stop**. This particular shell script executes */sbin/cron* when run with the **start** option and kills the *cron* process when run with the **stop** option.

Because these files are shell scripts, you can read them to see what they do. You can modify these files, though it is preferable to add your own since the delivered scripts may change in future releases. To create your own scripts, follow these rules:

- Place the file in */etc/init.d*.
- Symbolically link the file to files in appropriate run state directories, using the naming convention described above.
- Have the file accept the **start** and/or **stop** options.

Note that it may prove easier to simply copy an existing script from the directory and make appropriate changes.

Going to Single-user Mode From Multiuser Mode

Sometimes you must perform administrative functions, such as backing up the *root* file system, in single-user mode.

Use the *shutdown* command. This procedure executes all the files in */etc/rc0.d* by calling the */etc/rc0* procedure. The *shutdown* command does the following things, among others:

- Closes all open files and stops all user processes
- Stops all daemons and services
- Writes all system buffers out to the disk
- Unmounts all file systems except root

The entries for single-user processing in the sample */etc/inittab* are:

```
s1:1:wait:/etc/shutdown -y -iS -g0 >/dev/console 2>&1 </dev/console
```

There are three recommended ways to start the shutdown to single-user mode:

1. You can enter the *shutdown -i1* command (recommended). The option **-g** specifies a grace period between the first warning message and the final message.
2. You can enter the *single* command, which runs a shell script that switches to single-user mode and turns the *getty* processes off.
3. You can enter the *init 1* command, which forces the *init* process to scan the table. The first entry it finds is the **s1** entry, and *init* starts the shutdown processing according to that entry.

Now the system is in the single-user environment, and you can perform the appropriate administrative tasks.

/etc/inittab and Power Off

The following entries in */etc/inittab* power off the system:

```
s0:06s:wait:/etc/rc0 >/dev/console 2>&1 </dev/console
of:0:wait:/etc/uadmin 2 0 >/dev/console 2>&1 </dev/console
```

Always attempt to shut the system down gracefully. You can either enter the *powerdown* command, the *init 0* command, or directly invoke the */etc/shutdown -i0* command.

In either case, the */etc/shutdown* and */etc/rc0* procedures are called to clean up and stop all user processes, daemons, and other services and to unmount the file systems. Finally, the */sbin/uadmin* command is called, which indicates that the last step (physically removing power from the system) is under firmware control.