NAME

init, telinit – process control initialization

SYNOPSIS

/sbin/init [0123456abcQqSs]

/etc/telinit [0123456abcQqSs]

AVAILABILITY

SUNWcsu

DESCRIPTION

init is a general process spawner. Its primary role is to create processes from information stored in the file **/etc/inittab**.

Run Level Defined

At any given time, the system is in one of eight possible run levels. A run level is a software configuration under which only a selected group of processes exists. Processes spawned by **init** for each of these run levels are defined in **/etc/inittab**. **init** can be in one of eight run levels, **0**–**6** and **S** or **s** (**S** and **s** are identical). The run level changes when a privileged user runs **/sbin/init**. This sends appropriate signals to the original **init** spawned by the operating system at boot time, saying which run level to invoke.

init and System Booting

When the system is booted, **init** is invoked and the following occurs. First, it reads **/etc/default/init** to set environment variables. This is typically where **TZ** (time zone) and locale-related environments such as **LANG** or **LC_CTYPE** get set.

init then looks in **/etc/inittab** for the **initdefault** entry (see **inittab**(4)). If the **initdefault** entry:

exists
: **init** usually uses the run level specified in that entry as the initial run level to enter.

does not exist
: **/etc/inittab**, **init** asks the user to enter a run level from the system console.

- **S** or **s** — **init** goes to the single-user state. In this state, the system console device (**/dev/console**) is opened for reading and writing and the command **/sbin/su**, (see **su**(1M)), is invoked. Use either **init** or **telinit** to change the run level of the system. Note that if the shell is terminated (using an end-of-file), **init** only re-initializes to the single-user state if **/etc/inittab** does not exist.
- **0-6** — **init** enters the corresponding run level. Run levels **0**, **5**, and **6** are reserved states for shutting the system down. Run levels **2**, **3**, and **4** are available as multi-user operating states.

If this is the first time since power up that **init** has entered a run level other than single-user state, **init** first scans **/etc/inittab** for **boot** and **bootwait** entries (see **inittab**(4)). These entries are performed before any other processing of **/etc/inittab** takes place, providing that the run level entered matches that of the entry. In this way any special initialization of the operating system, such as mounting file systems, can take place before users are allowed onto the system. **init** then scans **/etc/inittab** and executes all other entries that are to be processed for that run level.

To spawn each process in **/etc/inittab**, **init** reads each entry and for each entry that should be respawned, it forks a child process. After it has spawned all of the processes specified by **/etc/inittab**, **init** waits for one of its descendant processes to die, a **powerfail** signal, or a signal from another **init** or **telinit** process to change the system's run level. When one of these conditions occurs, **init** re-examines **/etc/inittab**.

inittab Additions

New entries can be added to **/etc/inittab** at any time; however, **init** still waits for one of the above three conditions to occur before re-examining **/etc/inittab**. To get around this, **init Q** or **init q** command wakes **init** to re-examine **/etc/inittab** immediately.

When **init** comes up at boot time and whenever the system changes from the single-user state to another run state, **init** sets the **ioctl**(2) states of the console to those modes saved in the file **/etc/ioctl.syscon**. **init** writes this file whenever the single-user state is entered.

Run Level Changes

When a run level change request is made, **init** sends the warning signal (**SIGTERM**) to all processes that are undefined in the target run level. **init** waits five seconds before forcibly terminating these processes by sending a kill signal (**SIGKILL**).

When **init** receives a signal telling it that a process it spawned has died, it records the fact and the reason it died in **/var/adm/utmp** and **/var/adm/wtmp** if it exists (see **who**(1)). A history of the processes spawned is kept in **/var/adm/wtmp.**

If **init** receives a **powerfail** signal (**SIGPWR**) it scans **/etc/inittab** for special entries of the type **powerfail** and **powerwait**. These entries are invoked (if the run levels permit) before any further processing takes place. In this way **init** can perform various cleanup and recording functions during the powerdown of the operating system.

/etc/defaults/init File

Default values can be set for the following flags in **/etc/default/init**. For example: **TZ**=US/Pacific

TZ	Either specifies the timezone information (see **ctime**(3C)) or the name of a timezone information file **/usr/share/lib/zoneinfo**.
LC_CTYPE	Character characterization information.
LC_MESSAGES	Message translation.
LC_MONETARY	Monetary formatting information.
LC_NUMERIC	Numeric formatting information.
LC_TIME	Time formatting information.
LC_ALL	If set, all other **LC_*** environmental variables take-on this value.
LANG	If **LC_ALL** is not set, and any particular **LC_*** is also not set, the value of **LANG** is used for that particular environmental variable.

telinit

telinit, which is linked to **/sbin/init**, is used to direct the actions of **init**. It takes a one-character argument and signals **init** to take the appropriate action.

OPTIONS

0 Go into firmware.

1 Put the system in system administrator mode. All file systems are mounted. Only a small set of essential kernel processes are left running. This mode is for administrative tasks such as installing optional utility packages. All files are accessible and no users are logged in on the system.

2 Put the system in multi-user mode. All multi-user environment terminal processes and daemons are spawned. This state is commonly referred to as the multi-user state.

3 Extend multi-user mode by making local resources available over the network.

4 Is available to be defined as an alternative multi-user environment configuration. It is not necessary for system operation and is usually not used.

5 Shut the machine down so that it is safe to remove the power. Have the machine remove power, if possible.

6 Stop the operating system and reboot to the state defined by the **initdefault** entry in **/etc/inittab**.

a, **b**, **c** process only those **/etc/inittab** entries having the **a**, **b**, or **c** run level set. These are pseudo-states, which may be defined to run certain commands, but which do not cause the current run level to change.

Q, **q** Re-examine **/etc/inittab**.

S, **s** Enter single-user mode. When this occurs, the terminal that executed this command becomes the system console. This is the only run level that doesn't require the existence of a properly formatted **/etc/inittab** file. If this file does not exist, then by default, the only legal run level that **init** can enter is the single-user mode. When the system comes up to **S** or **s**, file systems for users' files are not mounted and only essential kernel processes are running. When the system comes down to **S** or **s**, all mounted file systems remain mounted, and all processes started by **init** that should only be running in multi-user mode are killed. In addition, any process that has a **utmp** entry will be killed. This last condition insures that all port monitors started by the SAC are killed and all services started by these port monitors, including **ttymon** login services, are killed. Other processes not started directly by **init** will remain running. For example, **cron** remains running.

FILES

/etc/inittab	controls process dispatching by init
/var/adm/utmp	accounting information
/var/adm/wtmp	history of all logins since file was last created
/etc/ioctl.syscon	
/dev/console	system console device
/etc/default/init	environment variables.

SEE ALSO

login(1), **sh**(1), **stty**(1), **who**(1), **shutdown**(1M), **su**(1M), **ttymon**(1M), **ioctl**(2), **kill**(2), **ctime**(3C), **inittab**(4), **utmp**(4), **utmpx**(4), **termio**(7I)

DIAGNOSTICS

If **init** finds that it is respawning an entry from **/etc/inittab** more than ten times in two minutes, assumes that there is an error in the command string in the entry, and generates an error message on the system console. It will then refuse to respawn this entry until either five minutes has elapsed or it receives a signal from a user-spawned **init** or **telinit**. This prevents **init** from eating up system resources when someone makes a typographical error in the **inittab** file, or a program is removed that is referenced in **/etc/inittab**.

NOTES

init and **telinit** can be run only by a privileged user.

The **S** or **s** state must not be used indiscriminately in **/etc/inittab**. When modifying this file, it is best to avoid adding this state to any line other than **initdefault**.

If a default state is not specified in the **initdefault** entry in **/etc/inittab**, state **6** is entered. Consequently, the system will loop by going to firmware and rebooting continuously.

If the **utmp** file cannot be created when booting the system, the system will boot to state "**s**" regardless of the state specified in the **initdefault** entry in **/etc/inittab**. This can occur if the **/var** file system is not accessible.