Managing System Information, Processes, and Performance in Oracle® Solaris 11.3

Part No: E54798
November 2016

Managing System Information, Processes, and Performance in Oracle Solaris 11.3

Part No: E54798

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Référence: E54798

Accessibilité de la documentation

Pour plus d'informations sur l'engagement d'Oracle pour l'accessibilité à la documentation, visitez le site Web Oracle Accessibility Program, à l'adresse `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Accès aux services de support Oracle

Les clients Oracle qui ont souscrit un contrat de support ont accès au support électronique via My Oracle Support. Pour plus d'informations, visitez le site `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` ou le site `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` si vous êtes malentendant.

Contents

Using This Documentation

- **Overview** – Describes tasks for managing system information, processes, and monitoring performance
- **Audience** – System administrators using the Oracle Solaris 11 release
- **Required knowledge** – Experience administering UNIX systems

Product Documentation Library

Documentation and resources for this product and related products are available at `http://www.oracle.com/pls/topic/lookup?ctx=E53394`.

Feedback

Provide feedback about this documentation at `http://www.oracle.com/goto/docfeedback`.

CHAPTER 1

Managing System Information

This chapter describes the tasks that are required to display and change basic system information.

For information about resource management that enables you to allocate, monitor, and control system resources in a flexible way, see Chapter 1, "Introduction to Resource Management" in *Administering Resource Management in Oracle Solaris 11.3*.

This is a list of the information that is in this chapter:

- "Displaying System Information" on page 11
- "Changing System Information" on page 23

Displaying System Information

This section describes commands that enable you to display general system information.

Commands That Are Used to Display System Information

TABLE 1 Commands for Displaying System Information

Command	System Information Displayed	Man Page
`date`	Date and time	`date`(1)
`hostid`	Host ID number	`hostid`(1)
`isainfo`	The number of bits supported by *native* applications on the running system, which can be passed as a token to scripts	`isainfo`(1)
`isalist`	Processor type	`isalist`(1)
`prtconf`	System configuration information, installed memory, device properties, and product name	`prtconf`(1M)
`prtdiag`	System configuration and diagnostic information, including any failed field replacement units (FRUs)	`prtdiag`(1M)

Command	System Information Displayed	Man Page
psrinfo	Processor information	psrinfo(1M)
uname	Operating system name, release, version, node name, hardware name, and processor type	uname(1)

Displaying a System's Release Information

Display the contents of the /etc/release file to identify your release version.

```
$ cat /etc/release
```

Displaying the Date and Time

To display the current date and time according to your system clock, use the date command.

The following example shows sample output from the date command.

```
$ date
Fri Jun  1 16:07:44 MDT 2012
$
```

Displaying a System's Host ID Number

To display the host ID number in a numeric (hexadecimal) format, use the hostid command.

The following example shows sample output from the hostid command.

```
$ hostid
80a5d34c
```

Displaying a System's Architecture Type

Use the isainfo command to display the architecture type and names of the native instruction sets for applications that are supported by the current operating system.

The following sample output is from an x86 based system:

```
$ isainfo
amd64 i386
```

The following sample output is from a SPARC based system:

```
$ isainfo
sparcv9 sparc
```

The `isainfo -v` command displays 32-bit and 64-bit application support. For example, the following sample output is from a SPARC based system:

```
$ isainfo -v
64-bit sparcv9 applications
        asi_blk_init
32-bit sparc applications
        asi_blk_init v8plus div32 mul32
#
```

The following example shows the output of the `isainfo -v` command from an x86 based system:

```
$ isainfo -v
64-bit amd64 applications
        sse4.1 ssse3 ahf cx16 sse3 sse2 sse fxsr mmx cmov amd_sysc cx8 tsc fpu
32-bit i386 applications
        sse4.1 ssse3 ahf cx16 sse3 sse2 sse fxsr mmx cmov sep cx8 tsc fpu
```

See the `isainfo(1)` man page.

For more information, see the `isainfo`(1) man page.

Displaying a System's Processor Type

Use the `isalist` command to display information about a system's processor type.

The following sample output is from an x86 based system:

```
$ isalist
pentium_pro+mmx pentium_pro pentium+mmx pentium i486 i386 i86
```

The following sample output is from a SPARC based system:

```
$ isalist
sparcv9 sparcv8plus sparcv8 sparcv8-fsmuld sparcv7 sparc sparcv9+vis sparcv9+vis2 \
sparcv8plus+vis sparcv8plus+vis2
```

See the `isalist(1)` man page.

Displaying a System's Product Name

To display the product name for your system, use the `prtconf` command with the `-b` option:

```
$ prtconf -b
```

For more information, see the `prtconf`(1M) man page.

The following example shows sample output from the `prtconf -b` command on a SPARC based system.

```
$ prtconf -b
name:  ORCL,SPARC-T4-2
banner-name:  SPARC T4-2
compatible: 'sun4v'
$
```

The following example shows sample output from the `prtconf -vb` command on a SPARC based system. The added `-v` option specifies verbose output.

```
$ prtconf -vb
name:  ORCL,SPARC-T3-4
banner-name:  SPARC T3-4
compatible: 'sun4v'
idprom: 01840014.4fa02d28.00000000.a02d28de.00000000.00000000.00000000.00000000
openprom model:  SUNW,4.33.0.b
openprom version: 'OBP 4.33.0.b 2011/05/16 16:26'
```

Displaying a System's Installed Memory

To display the amount of memory that is installed on your system, use the `prtconf` command with the `grep Memory` command. The following example shows sample output where the `grep Memory` command selects output from the `prtconf` command to display memory information only.

```
$ prtconf | grep Memory
Memory size: 523776 Megabytes
```

Displaying Default and Customized Property Values for a Device

You can display both the default and customized property values for devices, use the `prtconf` command with the `-u` option.

```
$ prtconf -u
```

The output of the prtconf -u command displays the default and customized properties for all of the drivers that are on the system.

For more information about this option, see the prtconf(1M) man page.

EXAMPLE 1 SPARC: Displaying Default and Custom Device Properties

This example shows the default and custom properties for the bge.conf file. Note that vendor-provided configuration files are located in the /kernel and /platform directories, while the corresponding modified driver configuration files are located in the /etc/driver/drv directory.

```
$ prtconf -u
System Configuration:  Oracle Corporation  sun4v
Memory size: 523776 Megabytes
System Peripherals (Software Nodes):

ORCL,SPARC-T3-4
    scsi_vhci, instance #0
        disk, instance #4
        disk, instance #5
        disk, instance #6
        disk, instance #8
        disk, instance #9
        disk, instance #10
        disk, instance #11
        disk, instance #12
    packages (driver not attached)
        SUNW,builtin-drivers (driver not attached)
        deblocker (driver not attached)
        disk-label (driver not attached)
        terminal-emulator (driver not attached)
        dropins (driver not attached)
        SUNW,asr (driver not attached)
        kbd-translator (driver not attached)
        obp-tftp (driver not attached)
        zfs-file-system (driver not attached)
        hsfs-file-system (driver not attached)
    chosen (driver not attached)
    openprom (driver not attached)
        client-services (driver not attached)
    options, instance #0
    aliases (driver not attached)
    memory (driver not attached)
    virtual-memory (driver not attached)
    iscsi-hba (driver not attached)
        disk, instance #0 (driver not attached)
    virtual-devices, instance #0
        flashprom (driver not attached)
        tpm, instance #0 (driver not attached)
        n2cp, instance #0
```

```
        ncp, instance #0
        random-number-generator, instance #0
        console, instance #0
        channel-devices, instance #0
            virtual-channel, instance #0
            virtual-channel, instance #1
            virtual-channel-client, instance #2
            virtual-channel-client, instance #3
            virtual-domain-service, instance #0
    cpu (driver not attached)
    cpu (driver not attached)
    cpu (driver not attached)
    cpu (driver not attached)
    cpu (driver not attached)
    cpu (driver not attached)
    cpu (driver not attached)
```

EXAMPLE 2 x86: Displaying Default and Custom Device Properties

This example shows the default and custom properties for the `bge.conf` file. Note that vendor-provided configuration files are located in the `/kernel` and `/platform` directories, while the corresponding modified driver configuration files are located in the `/etc/driver/drv` directory.

```
$ prtconf -u
System Configuration:  Oracle Corporation  i86pc
Memory size: 8192 Megabytes
System Peripherals (Software Nodes):

i86pc
    scsi_vhci, instance #0
    pci, instance #0
        pci10de,5e (driver not attached)
        isa, instance #0
            asy, instance #0
            motherboard (driver not attached)
            pit_beep, instance #0
        pci10de,cb84 (driver not attached)
        pci108e,cb84, instance #0
            device, instance #0
                keyboard, instance #0
                mouse, instance #1
        pci108e,cb84, instance #0
        pci-ide, instance #0
            ide, instance #0
                sd, instance #0
            ide (driver not attached)
        pci10de,5c, instance #0
            display, instance #0
        pci10de,cb84, instance #0
        pci10de,5d (driver not attached)
```

```
        pci10de,5d (driver not attached)
        pci10de,5d (driver not attached)
        pci10de,5d (driver not attached)
        pci1022,1100, instance #0
        pci1022,1101, instance #1
        pci1022,1102, instance #2
        pci1022,1103 (driver not attached)
        pci1022,1100, instance #3
        pci1022,1101, instance #4
        pci1022,1102, instance #5
        pci1022,1103 (driver not attached)
    pci, instance #1
        pci10de,5e (driver not attached)
        pci10de,cb84 (driver not attached)
        pci10de,cb84, instance #1
        pci10de,5d (driver not attached)
        pci10de,5d (driver not attached)
        pci10de,5d (driver not attached)
        pci10de,5d (driver not attached)
        pci1022,7458, instance #1
        pci1022,7459 (driver not attached)
        pci1022,7458, instance #2
            pci8086,1011, instance #0
            pci8086,1011, instance #1
            pci1000,3060, instance #0
                sd, instance #1
                sd, instance #2
        pci1022,7459 (driver not attached)
    ioapics (driver not attached)
        ioapic, instance #0 (driver not attached)
        ioapic, instance #1 (driver not attached)
    fw, instance #0
        cpu (driver not attached)
        cpu (driver not attached)
        cpu (driver not attached)
        cpu (driver not attached)
        sb, instance #1
    used-resources (driver not attached)
    iscsi, instance #0
    fcoe, instance #0
    pseudo, instance #0
    options, instance #0
    xsvc, instance #0
    vga_arbiter, instance #0
```

EXAMPLE 3 x86: Displaying System Configuration Information

The following example shows how to use the `prtconf` command with the `-v` option on an x86 based system to identify which disk, tape, and DVD devices are connected to a system. The output of this command displays "driver not attached" messages next to the device instances for which no device exists.

```
$ prtconf -v | more
System Configuration:  Oracle Corporation  i86pc
Memory size: 8192 Megabytes
System Peripherals (Software Nodes):

i86pc
    System properties:
        name='#size-cells' type=int items=1
            value=00000002
        name='#address-cells' type=int items=1
            value=00000003
        name='relative-addressing' type=int items=1
            value=00000001
        name='MMU_PAGEOFFSET' type=int items=1
            value=00000fff
        name='MMU_PAGESIZE' type=int items=1
            value=00001000
        name='PAGESIZE' type=int items=1
            value=00001000
        name='acpi-status' type=int items=1
            value=00000013
        name='biosdev-0x81' type=byte items=588
            value=01.38.74.0e.08.1e.db.e4.fe.00.d0.ed.fe.f8.6b.04.08.d3.db.e4.fe
.
.
.
```

For more information, see the `driver`(4), `driver.conf`(4), and `prtconf`(1M) man pages.

For instructions on how to create administratively provided configuration files, see Chapter 1, "Managing Devices in Oracle Solaris" in *Managing Devices in Oracle Solaris 11.3*.

Displaying System Diagnostic Information

Use the `prtdiag` command to display configuration and diagnostic information for a system.

```
$ prtdiag [-v] [-l]
```

`-v`	Verbose mode.
`-l`	Log output. If failures or errors exist in the system, output this information to `syslogd`(1M) only.

EXAMPLE 4 SPARC: Displaying System Diagnostic Information

The following example shows the output for the `prtdiag -v` command on a SPARC based system. For the sake of brevity, the example has been truncated.

```
$ prtdiag -v | more
System Configuration:  Oracle Corporation  sun4v Sun Fire T200
Memory size: 16256 Megabytes

================================ Virtual CPUs ================================


CPU ID Frequency Implementation         Status
------ --------- ---------------------- -------
0      1200 MHz  SUNW,UltraSPARC-T1     on-line
1      1200 MHz  SUNW,UltraSPARC-T1     on-line
2      1200 MHz  SUNW,UltraSPARC-T1     on-line
3      1200 MHz  SUNW,UltraSPARC-T1     on-line
4      1200 MHz  SUNW,UltraSPARC-T1     on-line
5      1200 MHz  SUNW,UltraSPARC-T1     on-line
6      1200 MHz  SUNW,UltraSPARC-T1     on-line
.
.
.
======================= Physical Memory Configuration ========================
Segment Table:
--------------------------------------------------------------
Base           Segment  Interleave  Bank     Contains
Address        Size     Factor      Size     Modules
--------------------------------------------------------------
0x0            16 GB    4           2 GB     MB/CMP0/CH0/R0/D0
                                             MB/CMP0/CH0/R0/D1
                                    2 GB     MB/CMP0/CH0/R1/D0
                                             MB/CMP0/CH0/R1/D1
                                    2 GB     MB/CMP0/CH1/R0/D0
                                             MB/CMP0/CH1/R0/D1
                                    2 GB     MB/CMP0/CH1/R1/D0
.
.
System PROM revisions:
----------------------
OBP 4.30.4.d 2011/07/06 14:29

IO ASIC revisions:
------------------
Location             Path                                  Device
                  Revision
-------------------- ------------------
IOBD/IO-BRIDGE                                    /pci@780      SUNW,sun4v-pci     0
.
.
```

.

EXAMPLE 5 x86: Displaying System Diagnostic Information

The following example shows the output for the `prtdiag -l` command on an x86 based system.

```
$ prtdiag -l
System Configuration: ... Sun Fire X4100 M2
BIOS Configuration: American Megatrends Inc. 0ABJX104 04/09/2009
BMC Configuration: IPMI 1.5 (KCS: Keyboard Controller Style)

==== Processor Sockets ====================================

Version                          Location Tag
-------------------------------- --------------------------
Dual-Core AMD Opteron(tm) Processor 2220 CPU 1
Dual-Core AMD Opteron(tm) Processor 2220 CPU 2

==== Memory Device Sockets ================================

Type        Status Set Device Locator      Bank Locator
----------- ------ --- ------------------- ---------------
unknown     empty  0   DIMM0                NODE0
unknown     empty  0   DIMM1                NODE0
DDR2        in use 0   DIMM2                NODE0
DDR2        in use 0   DIMM3                NODE0
unknown     empty  0   DIMM0                NODE1
unknown     empty  0   DIMM1                NODE1
DDR2        in use 0   DIMM2                NODE1
DDR2        in use 0   DIMM3                NODE1

==== On-Board Devices =====================================
 LSI serial-SCSI #1
 Gigabit Ethernet #1
 ATI Rage XL VGA

==== Upgradeable Slots ====================================

ID  Status    Type             Description
--- --------- ---------------- ----------------------------
1   available PCI Express      PCIExp SLOT0
2   available PCI Express      PCIExp SLOT1
3   available PCI-X            PCIX SLOT2
4   available PCI Express      PCIExp SLOT3
5   available PCI Express      PCIExp SLOT4
$
```

Identifying Information About Chip Multithreading Features

The `psrinfo` command has been modified to provide information about physical processors in addition to information about virtual processors. This enhanced functionality has been added to identify chip multithreading (CMT) features. The `-p` option reports the total number of physical processors that are in a system. The `-t` option displays a tree of the system's processors and their associated socket, core, and cpu ids.

Using the `psrinfo -pv` command lists all the physical processors that are in the system as well as the virtual processors that are associated with each physical processor. The default output of the `psrinfo` command continues to display the virtual processor information for a system.

For more information, see the `psrinfo`(1M) man page.

Displaying a System's Physical Processor Type

Use the `psrinfo -p` command to display the total number of physical processors on a system.

```
$ psrinfo -p
1
```

Add the `-v` option to also display information about the virtual processor that is associated with each physical processor. For example:

```
$ psrinfo -pv
The physical processor has 8 cores and 32 virtual processors (0-31)
  The core has 4 virtual processors (0-3)
  The core has 4 virtual processors (4-7)
  The core has 4 virtual processors (8-11)
  The core has 4 virtual processors (12-15)
  The core has 4 virtual processors (16-19)
  The core has 4 virtual processors (20-23)
  The core has 4 virtual processors (24-27)
  The core has 4 virtual processors (28-31)
    UltraSPARC-T1 (chipid 0, clock 1000 MHz)
```

The following example shows sample output for the `psrinfo -pv` command on an x86 based system.

```
$ psrinfo -pv
The physical processor has 2 virtual processors (0 1)
  x86 (AuthenticAMD 40F13 family 15 model 65 step 3 clock 2793 MHz)
        Dual-Core AMD Opteron(tm) Processor 2220        [ Socket: F(1207) ]
The physical processor has 2 virtual processors (2 3)
```

```
  x86 (AuthenticAMD 40F13 family 15 model 65 step 3 clock 2793 MHz)
        Dual-Core AMD Opteron(tm) Processor 2220        [ Socket: F(1207) ]
```

Displaying a System's Virtual Processor Type

Use the `psrinfo -v` command to display information about a system's virtual processor type.

```
$ psrinfo -v
```

On an x86 based system, use the `isalist` command to display the virtual processor type. For example:

```
$ isalist
amd64 pentium_pro+mmx pentium_pro pentium+mmx pentium i486 i386 i86
```

EXAMPLE 6 SPARC: Displaying a System's Virtual Processor Type

This example shows how to display information about a SPARC based system's virtual processor type.

```
$ psrinfo -v
Status of virtual processor 28 as of: 09/13/2010 14:07:47
  on-line since 04/08/2010 21:27:56.
  The sparcv9 processor operates at 1400 MHz,
        and has a sparcv9 floating point processor.
Status of virtual processor 29 as of: 09/13/2010 14:07:47
  on-line since 04/08/2010 21:27:56.
  The sparcv9 processor operates at 1400 MHz,
        and has a sparcv9 floating point processor.
```

EXAMPLE 7 SPARC: Displaying the Virtual Processor That Is Associated With Each Physical Processor on a System

The following example shows the output of the `psrinfo` command, when run with the `-pv` options on an Oracle SPARC T4-4 server. The output displays both the chip (physical processor) and the core information about the thread location. This information can be helpful in determining which physical CPU a thread is on, and how it is mapped at the core level.

```
$ psrinfo -pv
The physical processor has 8 cores and 64 virtual processors (0-63)
  The core has 8 virtual processors (0-7)
  The core has 8 virtual processors (8-15)
  The core has 8 virtual processors (16-23)
  The core has 8 virtual processors (24-31)
  The core has 8 virtual processors (32-39)
  The core has 8 virtual processors (40-47)
  The core has 8 virtual processors (48-55)
```

```
  The core has 8 virtual processors (56-63)
    SPARC-T4 (chipid 0, clock 2998 MHz)
The physical processor has 8 cores and 64 virtual processors (64-127)
  The core has 8 virtual processors (64-71)
  The core has 8 virtual processors (72-79)
  The core has 8 virtual processors (80-87)
  The core has 8 virtual processors (88-95)
  The core has 8 virtual processors (96-103)
  The core has 8 virtual processors (104-111)
  The core has 8 virtual processors (112-119)
  The core has 8 virtual processors (120-127)
    SPARC-T4 (chipid 1, clock 2998 MHz)
```

Changing System Information

This section describes commands that enable you to change general system information.

Changing System Information Task Map

Task	Directions	For Instructions
Manually set a system's date and time.	Manually set your system's date and time by using the `date` *mmddHHMM [[cc]yy]* command-line syntax.	"How to Manually Set a System's Date and Time" on page 23
Set up a message-of-the-day.	Set up a message-of-the-day on your system by editing the `/etc/motd` file.	"How to Set Up a Message-Of-The-Day" on page 24
Change a system's identity.	Change a system's identity by using the `hostname` command.	"How to Change a System's Identity" on page 25

▼ How to Manually Set a System's Date and Time

1. **Become an administrator.**

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Provide the new date and time.**

   `$` **`date`** *mmddHHMM[[cc]yy]*

   mm Month, using two digits

dd	Day of the month, using two digits
HH	Hour, using two digits and a 24-hour clock
MM	Minutes, using two digits
cc	Century, using two digits
yy	Year, using two digits

See the `date(1)` man page for more information.

3. **Verify that you have reset your system's date correctly by using the `date` command with no options.**

Example 8 Manually Setting a System's Date and Time

The following example shows how to use the `date` command to manually set a system's date and time.

```
# date
Monday, September 13. 2010 02:00:16 PM MDT
# date 0921173404
Thu Sep 17:34:34 MST 2010
```

▼ How to Set Up a Message-Of-The-Day

You can edit the message-of-the-day file, `/etc/motd`, to include announcements or inquiries to all users of a system when they log in. Use this feature sparingly, and edit this file regularly to remove obsolete messages.

1. **Assume a role that has the `Administrator Message Edit` profile assigned to it.**

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Use the `pfedit` command to edit the `/etc/motd` file and add a message of your choice.**

   ```
   $ pfedit /etc/motd
   ```

   Edit the text to include the message that will be displayed during user login. Include spaces, tabs, and carriage returns.

3. **Verify the changes by displaying the contents of the `/etc/motd` file.**

```
$ cat /etc/motd
Welcome to the UNIX universe. Have a nice day.
```

How to Change a System's Identity

1. **Become an administrator.**

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Set the name of the host for the system.**

   ```
   # hostname name
   ```

   The `hostname` and `domainname` commands enable you to permanently set the host name and domain name. When you use these commands, the corresponding SMF properties and associated SMF service, are also automatically updated.

   For more information, see the `hostname(1)`, `domainname(1M)`, and `nodename(4)` man pages.

◆ ◆ ◆ CHAPTER 2

Managing System Processes

This chapter describes procedures for managing system processes.

This chapter covers the following topics:

System Processes That do not Require Administration

The Oracle Solaris 10 and Oracle Solaris 11 releases include system processes that perform a specific task but do not require any administration.

Process	Description
`fsflush`	System daemon that flushes pages to disk
`init`	Initial system process that starts and restarts other processes and SMF components
`intrd`	System process that monitors and balances system load due to interrupts
`kmem_task`	System process that monitors memory cache sizes
`pageout`	System process that controls memory paging to disk
`sched`	System process that is responsible for OS scheduling and process swapping
`vm_tasks`	System process with one thread per processor that balances and distributes virtual memory related workloads across CPUs for better performance.
`zpool-`*pool-name*	System process for each ZFS storage pool containing the I/O task threads for the associated pool

Managing System Processes

This section describes the various tasks for managing system processes.

Managing System Processes Task Map

Task	Description	For Instructions
List processes.	Use the `ps` command to list all the processes on a system.	"How to List Processes" on page 31
Display information about processes.	Use the `pgrep` command to obtain the process IDs for processes that you want to display more information about.	"How to Display Information About Processes" on page 33
Control processes.	Locate processes by using the `pgrep` command. Then, use the appropriate `pcommand` (`/proc`) to control the process. See Table 3, "Process Commands (`/proc`)," on page 30 for a description of the (`/proc`) commands.	"How to Control Processes" on page 34
Kill a process.	Locate a process, either by process name or process ID. You can use either the `pkill` or `kill` commands to terminate the process.	"How to Terminate a Process (`pkill`)" on page 34 "How to Terminate a Process (`kill`)" on page 35

Commands for Managing System Processes

The following table describes the commands for managing system processes.

TABLE 2 Commands for Managing Processes

Command	Description	Man Page
`ps`, `pgrep`, `prstat`, `pkill`	Check the status of active processes on a system, and also displays detailed information about the processes.	`ps(1)`, `pgrep(1)`, and `prstat(1M)`
`pkill`	Functions identically to `pgrep` but finds or signals processes by name or other attribute and terminates the process. Each matching process is signaled as if by the `kill` command instead of having its process ID printed.	`pgrep(1)`, and `pkill(1)` `kill(1)`
`pargs, preap`	Assists with process debugging.	`pargs(1)`, and `preap(1)`
`dispadmin`	Lists default process scheduling policies.	`dispadmin(1M)`
`priocntl`	Assigns processes to a priority class and manages process priorities.	`priocntl(1)`
`nice`	Changes the priority of a timesharing process.	`nice(1)`

Command	Description	Man Page
psrset	Binds specific process groups to a group of processors rather than to just a single processor.	psrset(1M)

Using the ps Command

The ps command enables you to check the status of active processes on a system, and also display technical information about the processes. This data is useful for administrative tasks, such as determining how to set process priorities.

Depending on which options you use, the ps command reports the following information:

- Current status of the process
- Process ID
- Parent process ID
- User ID
- Scheduling class
- Priority
- Address of the process
- Memory used
- CPU time used

The following list describes some fields that are reported by the ps command. The fields that are displayed depend on which option you choose. For a description of all available options, see the ps(1) man page.

UID — The effective user ID of the process's owner.

PID — The process ID.

PPID — The parent process ID.

C — The processor utilization for scheduling. This field is not displayed when the -c option is used.

CLS — The scheduling class to which the process belongs such as real-time, system, or timesharing. This field is included only with the -c option.

PRI — The kernel thread's scheduling priority. Higher numbers indicate a higher priority.

NI — The process's nice number, which contributes to its scheduling priority. Making a process “nicer” means lowering its priority.

ADDR	The address of the proc structure.
SZ	The virtual address size of the process.
WCHAN	The address of an event or lock for which the process is sleeping.
STIME	The starting time of the process in hours, minutes, and seconds.
TTY	The terminal from which the process, or its parent, was started. A question mark indicates that there is no controlling terminal.
TIME	The total amount of CPU time used by the process since it began.
CMD	The command that generated the process.

Using the /proc File System and Commands

You can display detailed information about the processes that are listed in the /proc directory by using process commands. The following table lists the /proc process commands. The /proc directory is also known as the process file system (PROCFS). Images of active processes are stored in the PROCFS by their process ID number.

TABLE 3 Process Commands (/proc)

Process Command	Description
pcred	Displays process credential information
pfiles	Reports fstat and fcntl information for open files in a process
pflags	Displays /proc tracing flags, pending signals and held signals, and other status information
pldd	Lists the dynamic libraries that are linked into a process
pmap	Displays the address space map of each process
psig	Lists the signal actions and handlers of each process
prun	Starts each process
pstack	Displays a hex+symbolic stack trace for each lightweight process in each process
pstop	Stops each process
ptime	Times a process by using microstate accounting
ptree	Displays the process trees that contain the process
pwait	Displays status information after a process terminates
pwdx	Displays the current working directory for a process

For more information, see the proc(1) man page.

The process tools are similar to some options of the `ps` command, except that the output that is provided by these commands is more detailed.

The process commands do the following:

- Display more information about processes, such as `fstat` and `fcntl`, working directories, and trees of parent and child processes
- Provide control over processes by allowing users to stop or resume them

Managing Processes by Using Process Commands (`/proc`)

You can display detailed technical information about processes or control active processes by using some of the process commands. Table 3, "Process Commands (/proc)," on page 30 lists some of the `/proc` commands.

If a process becomes trapped in an endless loop, or if the process takes too long to execute, you might want to stop (kill) the process. For more information about stopping processes using the `kill` or the `pkill` command, see Chapter 2, "Managing System Processes".

The `/proc` file system is a directory hierarchy that contains additional subdirectories for state information and control functions.

The `/proc` file system also provides an `xwatchpoint` facility that is used to remap read-and-write permissions on the individual pages of a process's address space. This facility has no restrictions and is MT-safe.

Debugging tools have been modified to use the `xwatchpoint` facility, which means that the entire `xwatchpoint` process is faster.

The following restrictions no longer apply when you set `xwatchpoints` by using the `dbx` debugging tool:

- Setting `xwatchpoints` on local variables on the stack due to SPARC based system register windows.
- Setting `xwatchpoints` on multithreaded processes.

For more information, see the `proc(4)` and `mdb(1)` man pages.

▼ How to List Processes

● **Use the `ps` command to list all the processes on a system.**

```
$ ps [-efc]
```

`ps`	Displays only the processes that are associated with your login session.

`-ef` Displays full information about all the processes that are being executed on the system.

`-c` Displays process scheduler information.

Example 9 Listing Processes

The following example shows output from the `ps` command when no options are used.

```
$ ps
   PID TTY      TIME COMD
  1664 pts/4    0:06 csh
  2081 pts/4    0:00 ps
```

The following example shows output from the `ps -ef` command. This output shows that the first process that is executed when the system boots is `sched` (the swapper) followed by the `init` process, `pageout`, and so on.

```
$ ps -ef
 UID   PID  PPID   C    STIME TTY         TIME CMD
    root     0     0   0 18:04:04 ?           0:15 sched
    root     5     0   0 18:04:03 ?           0:05 zpool-rpool
    root     1     0   0 18:04:05 ?           0:00 /sbin/init
    root     2     0   0 18:04:05 ?           0:00 pageout
    root     3     0   0 18:04:05 ?           2:52 fsflush
    root     6     0   0 18:04:05 ?           0:02 vmtasks
  daemon   739     1   0 19:03:58 ?           0:00 /usr/lib/nfs/nfs4cbd
    root     9     1   0 18:04:06 ?           0:14 /lib/svc/bin/svc.startd
    root    11     1   0 18:04:06 ?           0:45 /lib/svc/bin/svc.configd
  daemon   559     1   0 18:04:49 ?           0:00 /usr/sbin/rpcbind
  netcfg    47     1   0 18:04:19 ?           0:01 /lib/inet/netcfgd
   dladm    44     1   0 18:04:17 ?           0:00 /sbin/dlmgmtd
  netadm    51     1   0 18:04:22 ?           0:01 /lib/inet/ipmgmtd
    root   372   338   0 18:04:43 ?           0:00 /usr/lib/hal/hald-addon-cpufreq
    root    67     1   0 18:04:30 ?           0:02 /lib/inet/in.mpathd
    root   141     1   0 18:04:38 ?           0:00 /usr/lib/pfexecd
  netadm    89     1   0 18:04:31 ?           0:03 /lib/inet/nwamd
    root   602     1   0 18:04:50 ?           0:02 /usr/lib/inet/inetd start
    root   131     1   0 18:04:35 ?           0:01 /sbin/dhcpagent
  daemon   119     1   0 18:04:33 ?           0:00 /lib/crypto/kcfd
    root   333     1   0 18:04:41 ?           0:07 /usr/lib/hal/hald --daemon=yes
    root   370   338   0 18:04:43 ?           0:00 /usr/lib/hal/hald-addon-network-
discovery
    root   159     1   0 18:04:39 ?           0:00 /usr/lib/sysevent/syseventd
    root   236     1   0 18:04:40 ?           0:00 /usr/lib/ldoms/drd
    root   535     1   0 18:04:46 ?           0:09 /usr/sbin/nscd
    root   305     1   0 18:04:40 ?           0:00 /usr/lib/zones/zonestatd
    root   326     1   0 18:04:41 ?           0:03 /usr/lib/devfsadm/devfsadmd
    root   314     1   0 18:04:40 ?           0:00 /usr/lib/dbus-daemon --system
```

```
.
.
.
```

How to Display Information About Processes

1. **Obtain the process ID of the process that you want to display more information about.**

   `# pgrep` *process*

   The process ID is displayed in the first column of the output.

2. **Display the process information.**

   `# /usr/bin/`*pcommand PID*

pcommand	The process command that you want to run. Table 3, "Process Commands (`/proc`)," on page 30 lists and describes these commands.
PID	Identifies the process ID.

Example 10 Displaying Information About Processes

The following example shows how to use process commands to display more information about a `cron` process.

```
# pgrep cron        Obtains the process ID for the cron process
4780
# pwdx 4780        Displays the current working directory for the cron process
4780:    /var/spool/cron/atjobs
# ptree 4780        Displays the process tree that contains the cron process
4780  /usr/sbin/cron
# pfiles 4780        Displays fstat and fcntl information
4780:    /usr/sbin/cron
  Current rlimit: 256 file descriptors
   0: S_IFCHR mode:0666 dev:290,0 ino:6815752 uid:0 gid:3 rdev:13,2
      O_RDONLY|O_LARGEFILE
      /devices/pseudo/mm@0:null
   1: S_IFREG mode:0600 dev:32,128 ino:42054 uid:0 gid:0 size:9771
      O_WRONLY|O_APPEND|O_CREAT|O_LARGEFILE
      /var/cron/log
   2: S_IFREG mode:0600 dev:32,128 ino:42054 uid:0 gid:0 size:9771
      O_WRONLY|O_APPEND|O_CREAT|O_LARGEFILE
      /var/cron/log
   3: S_IFIFO mode:0600 dev:32,128 ino:42049 uid:0 gid:0 size:0
      O_RDWR|O_LARGEFILE
      /etc/cron.d/FIFO
```

```
4: S_IFIFO mode:0000 dev:293,0 ino:4630 uid:0 gid:0 size:0
   O_RDWR|O_NONBLOCK
5: S_IFIFO mode:0000 dev:293,0 ino:4630 uid:0 gid:0 size:0
   O_RDWR
```

▼ How to Control Processes

1. **Obtain the process ID of the process that you want to control.**

   `# pgrep` *process*

   The process ID displayed in the first column of the output.

2. **Use the appropriate process command to control the process.**

   `# /usr/bin/`*pcommand PID*

   pcommand — The process command that you want to run. Table 3, "Process Commands (`/proc`)," on page 30 lists and describes these commands.

   PID — Identifies the process ID.

3. **Verify the process status.**

   `# ps -ef | grep` *PID*

Terminating a Process (`pkill, kill`)

You might need to stop (kill) a process that is in an endless loop, or stop a large job before it is completed. You can kill any process that you own. The system administrator can kill any process in the system except for those processes with process IDs of `0`, `1`, `2`, `3`, and `4`. Killing these processes most likely will crash the system.

For more information, see the `pgrep(1)`, `pkill(1)`, and `kill(1)` man pages.

▼ How to Terminate a Process (`pkill`)

1. **To terminate the process of another user, assume the `root` role.**
2. **Obtain the process ID for the process that you want to terminate.**

   `$ pgrep` *process*

   For example:

   `$ pgrep netscape`

```
587
566
```

The process ID is displayed in the output.

Note - To obtain information about processes on a Sun Ray™system, use the following commands:

To list all user processes:

ps -fu *user*

To locate a specific process owned by a user:

ps -fu *user* **| grep** *process*

---

3. **Terminate the process.**

   $ **pkill** [*signal*] *PID*

signal	When no signal is included in the pkill command-line syntax, the default signal that is used is –15 (SIGKILL). Using the –9 signal (SIGTERM) with the pkill command ensures that the process terminates promptly. However, the –9 signal should not be used to kill certain processes such as a database process or an LDAP server process because data might be lost.
PID	The name of the process to stop.

Tip - When using the pkill command to terminate a process, first try using the command by itself without including a signal option. If the process does not terminate after a few minutes, use the pkill command with the -9 signal.

---

4. **Verify that the process has been terminated.**

   $ **pgrep** *process*

   The process you terminated should no longer be listed in the output of the pgrep command.

How to Terminate a Process (kill)

1. **To terminate the process of another user, assume the root role.**
2. **Obtain the process ID of the process that you want to terminate.**

   # **ps -fu** *user*

where *user* is the owner of the process.
The process ID is displayed in the first column of the output.

3. **Terminate the process.**

   `# kill [signal-number] PID`

signal	When no signal is included in the `kill` command-line syntax, the default signal that is used is –15 (`SIGKILL`). Using the –9 signal (`SIGTERM`) with the `kill` command ensures that the process terminates promptly. However, the –9 signal should not be used to kill certain processes such as a database process or an LDAP server process because data might be lost.
PID	Is the process ID of the process that you want to terminate.

   Tip - When using the `kill` command to stop a process, first try using the command by itself, without including a signal option. Wait a few minutes to see if the process terminates before using the `kill` command with the -9 signal.

   ---

4. **Verify that the process has been terminated.**

   `$ ps`

   The process you terminated should no longer be listed in the output of the `ps` command.

Debugging a Process (pargs, preap)

The `pargs` command and the `preap` command improve process debugging. The `pargs` command prints the arguments and environment variables that are associated with a live process or core file. The `preap` command removes defunct (zombie) processes. A zombie process has not yet had its exit status claimed by its parent. These processes are generally harmless but can consume system resources if they are numerous. You can use the `pargs` and `preap` commands to examine any process that you have the privileges to examine. When you become an administrator, you can examine any process.

For information about using the `preap` command, see the `preap(1)` man page. For information about the using the `pargs` command, see the `pargs(1)` man page. See also the `proc(1)` man page.

EXAMPLE 11 Debugging a Process (`pargs`)

The `pargs` command solves a long-standing problem of being unable to display with the `ps` command all the arguments that are passed to a process. The following example shows how to use the `pargs` command in combination with the `pgrep` command to display all the arguments that are passed to a process.

```
# pargs `pgrep ttymon`
579: /usr/lib/saf/ttymon -g -h -p system-name console login:
-T sun -d /dev/console -l
argv[0]: /usr/lib/saf/ttymon
argv[1]: -g
argv[2]: -h
argv[3]: -p
argv[4]: system-name console login:
argv[5]: -T
argv[6]: sun
argv[7]: -d
argv[8]: /dev/console
argv[9]: -l
argv[10]: console
argv[11]: -m
argv[12]: ldterm,ttcompat
548: /usr/lib/saf/ttymon
argv[0]: /usr/lib/saf/ttymon
```

The following example shows how to use the `pargs -e` command to display the environment variables that are associated with a process.

```
$ pargs -e 6763
6763: tcsh
envp[0]: DISPLAY=:0.0
```

Displaying and Managing Process Class Information

You can configure the process scheduling classes on your system and the user priority range for the timesharing class.

The possible process scheduling classes are as follows:

- Fair share (`FSS`)
- Fixed (`FX`)
- System (`SYS`)
- Interactive (`IA`)
- Real-time (`RT`)
- Timesharing (`TS`)
  - The user-supplied priority ranges from -60 to +60.
  - The priority of a process is inherited from the parent process. This priority is referred to as the *user-mode priority*.

- The system looks up the user-mode priority in the timesharing dispatch parameter table. Then, the system adds in any `nice` or `priocntl` (user-supplied) priority and ensures a 0–59 range to create a *global priority*.

Displaying Process Class Information

This section covers the following topics:

"Displaying Process Priority Information" on page 38	Use the `priocntl -l` command to display process scheduling classes and priority ranges.
"Displaying the Global Priority of a Process" on page 38	Use the `ps -ecl` command to display the global priority of a process.

Displaying Process Priority Information

Use the `priocntl -l` command to display process scheduling classes and priority ranges.

```
$ priocntl -l
```

The following example shows output from the `priocntl -l` command.

```
# priocntl -l
CONFIGURED CLASSES
==================

SYS (System Class)

TS (Time Sharing)
        Configured TS User Priority Range: -60 through 60

FX (Fixed priority)
        Configured FX User Priority Range: 0 through 60

IA (Interactive)
        Configured IA User Priority Range: -60 through 60
```

Displaying the Global Priority of a Process

Use the `ps` command to display the global priority of a process.

```
$ ps -ecl
```

The global priority is listed under the PRI column.

The following example shows `ps -ecl` command output. The values in the PRI column show the priority for each process.

```
$ ps -ecl
 F S    UID   PID  PPID  CLS PRI     ADDR     SZ    WCHAN TTY         TIME CMD
 1 T      0     0     0  SYS  96        ?      0          ?         0:11 sched
 1 S      0     5     0  SDC  99        ?      0        ? ?         0:01 zpool-rp
 0 S      0     1     0   TS  59        ?    688        ? ?         0:00 init
 1 S      0     2     0  SYS  98        ?      0        ? ?         0:00 pageout
 1 S      0     3     0  SYS  60        ?      0        ? ?         2:31 fsflush
 1 S      0     6     0  SDC  99        ?      0        ? ?         0:00 vmtasks
 0 S     16    56     1   TS  59        ?   1026        ? ?         0:01 ipmgmtd
 0 S      0     9     1   TS  59        ?   3480        ? ?         0:04 svc.star
 0 S      0    11     1   TS  59        ?   3480        ? ?         0:13 svc.conf
 0 S      0   162     1   TS  59        ?    533        ? ?         0:00 pfexecd
 0 S      0  1738  1730   TS  59        ?    817        ? pts/ 1       0:00 bash
 0 S      1   852     1   TS  59        ?    851        ? ?         0:17 rpcbind
 0 S     17    43     1   TS  59        ?   1096        ? ?         0:01 netcfgd
 0 S     15    47     1   TS  59        ?    765        ? ?         0:00 dlmgmtd
 0 S      0    68     1   TS  59        ?    694        ? ?         0:01 in.mpath
 0 S      1  1220     1   FX  60        ?    682        ? ?         0:00 nfs4cbd
 0 S     16    89     1   TS  59        ?   1673        ? ?         0:02 nwamd
 0 S      0   146     1   TS  59        ?    629        ? ?         0:01 dhcpagen
 0 S      1   129     1   TS  59        ?   1843        ? ?         0:00 kcfd
 0 S      1  1215     1   FX  60        ?    738        ? ?         0:00 lockd
 0 S      0   829   828   TS  59        ?    968        ? ?         0:00 hald-run
 0 S      0   361     1   TS  59        ?   1081        ? ?         0:01 devfsadm
 0 S      0   879     1   TS  59        ?   1166        ? ?         0:01 inetd
 0 O 119764  1773   880   TS  59        ?    557          cons ole     0:00 ps
 0 S      0   844   829   TS  59        ?    996        ? ?         0:00 hald-add
 0 S      0   895   866   TS  59        ?    590        ? ?         0:00 ttymon
 0 S      0   840     1   TS  59        ?    495        ? ?         0:00 cron
 0 S      0   874     1   TS  59        ?    425        ? ?         0:00 utmpd
 0 S      0  1724   956   TS  59        ?   2215        ? ?         0:00 sshd
 0 S 119764   880     9   TS  59        ?    565        ? cons ole     0:00 csh
 0 S      0   210     1   TS  59        ?   1622        ? ?         0:00 sysevent
 0 S      0   279     1   TS  59        ?    472        ? ?         0:00 iscsid
 0 S      1  1221     1   TS  59        ?   1349        ? ?         0:00 nfsmapid
 1 S      0   374     0  SDC  99        ?      0        ? ?         0:00 zpool-us
 0 S      0  1207     1   TS  59        ?   1063        ? ?         0:00 rmvolmgr
 0 S      0   828     1   TS  59        ?   1776        ? ?         0:03 hald
 0 S      0   853   829   TS  59        ?    896        ? ?         0:02 hald-add
 0 S      0   373     1   TS  59        ?    985        ? ?         0:00 picld
 0 S      0   299     1   TS  59        ?    836        ? ?         0:00 dbus-dae
 0 S  12524  1730  1725   TS  59        ?    452        ? pts/ 1       0:00 csh
 0 S      0   370     1   TS  59        ?    574        ? ?         0:00 powerd
 0 S      0   264     1   FX  60        ?    637        ? ?         0:00 zonestat
 0 S      0   866     9   TS  59        ?    555        ? ?         0:00 sac
 0 S      0   851   829   TS  59        ?    998        ? ?         0:00 hald-add
```

```
0 S  12524  1725  1724   TS  59          ?   2732          ? ?          0:00 sshd
0 S      1  1211     1   TS  59          ?    783          ? ?          0:00 statd
0 S      0  1046     1   TS  59          ?   1770          ? ?          0:13 intrd
0 S      0   889     1   TS  59          ?   1063          ? ?          0:00 syslogd
0 S      0  1209     1   TS  59          ?    792          ? ?          0:00 in.ndpd
0 S      0  1188  1186   TS  59          ?    951          ? ?          0:15 automoun
0 S      0  1172   829   TS  59          ?    725          ? ?          0:00 hald-add
0 S      0  1186     1   TS  59          ?    692          ? ?          0:00 automoun
0 S    101  1739  1738   TS  59          ?    817          ? pts/ 1        0:00 bash
0 S      0  1199     1   TS  59          ?   1495          ? ?          0:02 sendmail
0 S      0   956     1   TS  59          ?   1729          ? ?          0:00 sshd
0 S     25  1192     1   TS  59          ?   1528          ? ?          0:00 sendmail
0 S      0   934     1   TS  59          ?   6897          ? ?          0:14 fmd
0 S      0  1131     1   TS  59          ?   1691          ? ?          0:07 nscd
0 S      1  1181     1   TS  59          ?    699          ? ?          0:00 ypbind
```

Managing Process Class Information Task Map

Use the following procedures to manage your process classes.

Task	Description	For Instructions
Designate a process priority.	Start a process with a designated priority by using the `priocntl -e -c command`.	"How to Designate a Process Priority (`priocntl`)" on page 41
Change scheduling parameters of a timesharing process.	Use the `priocntl -s -m` command to change scheduling parameters in a timesharing process.	"How to Change Scheduling Parameters of a Timesharing Process (`priocntl`)" on page 41
Change the class of a process.	Use the `priocntl -s -c` command to change the class of a process.	"How to Change the Class of a Process (`priocntl`)" on page 42
Change the priority of a process.	Use the `/usr/bin/nice` command with the appropriate options to lower or raise the priority of a process.	"Changing the Priority of a Process (`nice`)" on page 44

Changing the Scheduling Priority of Processes (`priocntl`)

The scheduling priority of a process is the priority assigned by the process scheduler, according to scheduling policies. The `dispadmin` command lists the default scheduling policies. For more information, see the `dispadmin`(1M) man page.

You can use the `priocntl` command to assign processes to a priority class and to manage process priorities as shown in the following procedure.

How to Designate a Process Priority (`priocntl`)

1. **Assume the `root` role.**

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Start a process with a designated priority.**

   `# priocntl -e -c` *class* **`-m`** *user-limit* **`-p`** *PRI command-name*

`-e`	Executes the command.
`-c` *class*	Specifies the class within which to run the process. The valid classes are `TS` (timesharing), `RT` (real time), IA (interactive), FSS (fair share), and FX (fixed priority).
`-m` *user-limit*	When you use the `-p` option with this option, the maximum amount you can raise or lower your priority is also specified.
`-p` *PRI*	Enables you specify the relative priority in the RT class for a real-time thread. For a timesharing process, the `-p` option enables you to specify the user-supplied priority, which ranges from -60 to +60.
command-name	Specifies the name of the command that will be executed.

3. **Verify the process status.**

   `# ps -ecl | grep` *command-name*

Example 12 Designating a Process Priority (`priocntl`)

The following example shows how to start the `find` command with the highest possible user-supplied priority.

```
# priocntl -e -c TS -m 60 -p 60 find . -name core -print
# ps -ecl | grep find
```

How to Change Scheduling Parameters of a Timesharing Process (`priocntl`)

1. **Assume the `root` role.**

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Change the scheduling parameters of a running timesharing process.**

```
# priocntl -s -m user-limit [-p user-priority] -i ID type  ID list
```

`-s`	Lets you set the upper limit on the user priority range and change the current priority.
`-m` *user-limit*	When you use the `-p` option, specifies the maximum amount you can raise or lower the priority.
`-p` *user-priority*	Allows you to designate a priority.
`-i` *ID type ID list*	Uses a combination of *ID type* and *ID list* to identify the process or processes. *ID type* specifies the type of ID, such as the process ID or the user ID. *ID list* identifies a list of process IDs or user IDs.

3. **Verify the process status.**

```
# ps -ecl | grep ID list
```

Example 13 Changing Scheduling Parameters of a Timesharing Process (priocntl)

The following example shows how to execute a command with a 500-millisecond time slice, a priority of 20 in the RT class, and a global priority of 120.

```
# priocntl -e -c RT -m 500 -p 20 myprog
# ps -ecl | grep myprog
```

▼ How to Change the Class of a Process (priocntl)

1. **(Optional) Assume the `root` role.**

Note - You must assume the root role or be working in a real-time shell to change a process from, or to, a real-time process. If, in the root role, you change a user process to the real-time class, the user cannot subsequently change the real-time scheduling parameters by using the `priocntl -s` command.

See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Change the class of a process.**

```
# priocntl -s -c class -i ID type  ID list
```

`-s`	Lets you set the upper limit on the user priority range and change the current priority.
`-c` *class*	Specifies the class, `TS` for time-sharing or `RT for real-time`, to which you are changing the process.
`-i` *ID type ID list*	Uses a combination of *ID type* and *ID list* to identify the process or processes. *ID type* specifies the type of ID, such as the process ID or user ID. *ID list* identifies a list of process IDs or user IDs.

3. **Verify the process status.**

   ```
   # ps -ecl | grep ID list
   ```

Example 14 Changing the Class of a Process (`priocntl`)

The following example shows how to change all the processes that belong to user `15249` to real-time processes.

```
# priocntl -s -c RT -i uid 15249
# ps -ecl | grep 15249
```

Changing the Priority of a Timesharing Process (`nice`)

The `nice` command is supported only for backward compatibility to previous releases. The `priocntl` command provides more flexibility in managing processes.

The priority of a process is determined by the policies of its scheduling class and by its *nice number*. Each timesharing process has a global priority. The global priority is calculated by adding the user-supplied priority, which can be influenced by the `nice` or `priocntl` commands, and the system-calculated priority.

The execution priority number of a process is assigned by the operating system. The priority number is determined by several factors, including the process's scheduling class, how much CPU time it has used and in the case of a timesharing process, its `nice` number.

Each timesharing process starts with a default `nice` number, which it inherits from its parent process. The `nice` number is shown in the `NI` column of the `ps` report.

A user can lower the priority of a process by increasing its user-supplied priority. However, only an administrator can lower a `nice` number to increase the priority of a process. This restriction prevents users from increasing the priorities of their own processes, thereby monopolizing a greater share of the CPU.

The nice numbers range from 0 to +39, with 0 representing the highest priority. The default nice value for each timesharing process is 20. Two versions of the command are available: the standard version, /usr/bin/nice, and the C shell built-in command.

Changing the Priority of a Process (nice)

As a user, you can lower the priority of a process. Become an administrator to raise or lower the priority of a process.

- As a user, you can lower the priority of a command by increasing the nice number.

  The following nice command executes *command-name* with a lower priority by raising the nice number by 5 units.

  ```
  $ /usr/bin/nice -5 command-name
  ```

  In this command, the minus sign designates that what follows is an option. This command could also be specified as follows:

  ```
  $ /usr/bin/nice -n 5 command-name
  ```

  The following nice command lowers the priority of *command-name* by raising the nice number by the default increment of 10 units, but not beyond the maximum value of 39.

  ```
  $ /usr/bin/nice command-name
  ```
- As an administrator, you can raise or lower the priority of a command by changing the nice number.

  The following nice command raises the priority of *command-name* by lowering the nice number by 10 units. It is not lowered below the minimum value of 0.

  ```
  # /usr/bin/nice --10 command-name
  ```

  In this command, the first minus sign designates that what follows is an option. The second minus sign indicates a negative number.

  The following nice command lowers the priority of *command-name* by raising the nice number by 5 units. It does not exceed the maximum value of 39.

  ```
  # /usr/bin/nice -5 command-name
  ```

For more information, see the nice(1) man page.

Troubleshooting Problems With System Processes

Some common system process problems you might encounter are as follows:

- Look for several identical jobs that are owned by the same user. This problem might occur because of a running script that starts a lot of background jobs without waiting for any of the jobs to finish.
- Look for a process that has accumulated a large amount of CPU time. You can identify this problem by checking the `TIME` field in the `ps` output. This value could indicate that the process is in an endless loop.
- Look for a process that is running with a priority that is too high. Use the `ps -c` command to check the `CLS` field, which displays the scheduling class of each process. A process executing as a real-time (`RT`) process can monopolize the CPU. Or, look for a timesharing (`TS`) process with a high `nice` number. An administrator might have increased the priority of a process. The system administrator can lower the priority by using the `nice` command.
- Look for a runaway process that progressively uses increasing amounts of CPU time. You can identify this problem by looking at the time when the process started (`STIME`) and by watching the cumulation of CPU time (`TIME`) for a while.

♦ ♦ ♦ CHAPTER 3

Monitoring System Performance

Achieving good performance from a computer or network is an important part of system administration. This chapter describes some factors that contribute to managing the performance of the computer systems in your care. In addition, this chapter describes procedures for monitoring system performance by using the `vmstat`, `iostat`, `df`, and `sar` commands.

This chapter covers the following topics:

Where to Find Information About Monitoring System Performance

System Performance Task	For More Information
Manage processes	Chapter 2, "Managing System Processes"
Monitor system performance	Chapter 3, "Monitoring System Performance"
Change tunable parameters	*Oracle Solaris 11.3 Tunable Parameters Reference Manual*
Manage system performance tasks	Chapter 2, "About Projects and Tasks" in *Administering Resource Management in Oracle Solaris 11.3*
Manage processes with FX and FS schedulers	Chapter 8, "About Fair Share Scheduler" in *Administering Resource Management in Oracle Solaris 11.3*

Managing Performance Using Oracle Enterprise Manager Ops Center

If you need to monitor, analyze, and improve performance for physical and virtual operating systems, servers, and storage devices in a large deployment, rather than just monitoring performance within individual systems, you can use the comprehensive system management solutions available in the Oracle Enterprise Manager Ops Center.

The monitoring feature in the Enterprise Manager Ops Center provides extensive information about the monitored operating systems and zones in a large deployment. You can use the information to evaluate performance, identify issues, and perform tuning. Analytics are available for the Oracle Solaris operating system, for Linux, and for OS virtualization technologies including Oracle Solaris Zones, Oracle VM Server for SPARC, and Oracle VM Server for x86 guests.

For information, see `http://www.oracle.com/pls/topic/lookup?ctx=oc122`.

About System Resources That Affect System Performance

The performance of a computer system depends upon how the system uses and allocates its resources. Monitor your system's performance regularly so that you know how it behaves under normal conditions. You should have a good idea of what to expect, and be able to recognize a problem when it occurs.

System resources that affect performance are:

Central processing unit (CPU)	The CPU processes instructions by fetching instructions from memory and executing them.
Input/output (I/O) devices	I/O devices transfer information into and out of the computer. Such a device could be a terminal and keyboard, a disk drive, or a printer.
Memory	Physical (or main) memory is the amount of random access memory (RAM) on the system.

Chapter 3, "Monitoring System Performance" describes the tools that display statistics about the system's activity and performance.

About Processes and System Performance

Some terms that are related to processes are:

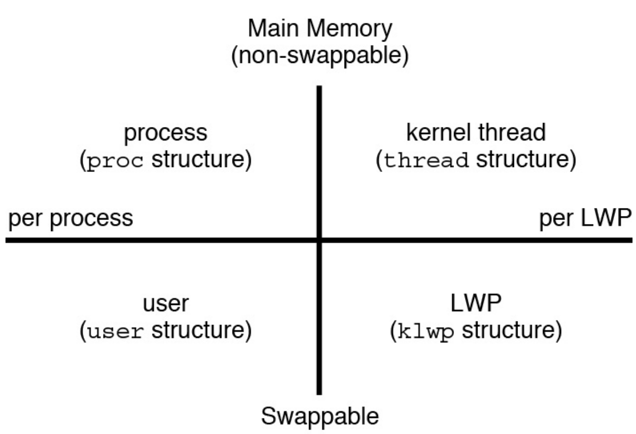
Process	Any system activity or job. Each time you boot a system, execute a command, or start an application, the system activates one or more processes.
Lightweight process (LWP)	A virtual CPU or execution resource. LWPs are scheduled by the kernel to use available CPU resources based on their scheduling class and priority. An LWP contains information that is swappable and a kernel thread that contains information that has to be in memory all the time.
Application thread	A series of instructions with a separate stack that can execute independently in a user's address space. Application threads can be multiplexed on top of LWPs.

A process can consist of multiple LWPs and multiple application threads. The kernel schedules a kernel-thread structure, which is the scheduling entity in the Oracle Solaris environment. Various process structures are as follows:

`proc`	Contains information that pertains to the whole process and must be in main memory all the time
`kthread`	Contains information that pertains to one LWP and must always be in main memory
`user`	Contains the "per process" information that is swappable
`klwp`	Contains the "per LWP process" information that is swappable

The following figure illustrates the relationships among these process structures.

FIGURE 1 Relationships Among Process Structures



Most process resources are accessible to all the threads in the process. Almost all process virtual memory is shared. A change in shared data by one thread is available to the other threads in the process.

About Monitoring System Performance

While your computer is running, counters in the operating system are incremented to track various system activities.

System activities that are tracked are as follows:

- Central processing unit (CPU) utilization
- Buffer usage
- Disk and tape input/output (I/O) activity
- Terminal device activity
- System call activity
- Context switching
- File access
- Queue activity
- Kernel tables
- Interprocess communication
- Paging
- Free memory and swap space
- Kernel memory allocation (KMA)

Monitoring Tools

The Oracle Solaris software provides several tools to help you track how your system is performing.

TABLE 4 Performance Monitoring Tools

Command	Description	For More Information
`cpustat` and `cputrack` commands	Monitors performance of a system or a process using CPU performance counters.	`cpustat`(1M) and `cputrack`(1)
`netstat` and `nfsstat` commands	Displays information about network performance.	`netstat`(1M) and `nfsstat`(1M)
`ps` and `prstat` commands	Displays information about active processes.	Chapter 2, "Managing System Processes"
`sar` and `sadc` commands	Collects and reports on system activity data.	Chapter 3, "Monitoring System Performance"

Command	Description	For More Information
`swap` command	Displays information about available swap space on your system.	Chapter 3, "Configuring Additional Swap Space" in *Managing File Systems in Oracle Solaris 11.3*
`vmstat` and `iostat` commands	Summarizes system activity data, such as virtual memory statistics, disk usage, and CPU activity.	Chapter 3, "Monitoring System Performance"
`kstat` and `mpstat` commands	Examines the available kernel statistics, or `kstats`, on the system and reports those statistics which match the criteria specified on the command line. The `mpstat` command reports processor statistics in tabular form.	`kstat`(1M) and `mpstat`(1M) man pages.

Displaying System Performance Information

This section describes the tasks for monitoring and displaying system performance information.

Displaying Virtual Memory Statistics

You can use the `vmstat` command to report virtual memory statistics and information about system events such as CPU load, paging, number of context switches, device interrupts, and system calls. The `vmstat` command can also display statistics on swapping, cache flushing, and interrupts.

TABLE 5 Output From the `vmstat` Command

Category	Field Name	Description
`procs`		
	`r`	The number of kernel threads in the dispatch queue
	`b`	The number of blocked kernel threads that are waiting for resources
	`w`	The number of swapped-out LWPs that are waiting for processing resources to finish
`memory`		Reports on usage of real memory and virtual memory
	`swap`	Available swap space
	`free`	Size of the free list
`page`		Reports on page faults and paging activity, in units per second
	`re`	Pages reclaimed
	`mf`	Minor faults and major faults
	`pi`	Kbytes paged in
	`po`	Kbytes paged out

Category	Field Name	Description
	fr	Kbytes freed
	de	Anticipated memory that is needed by recently swapped-in processes
	sr	Pages scanned by the page daemon not currently in use. If sr does not equal zero, the page daemon has been running.
disk		Reports the number of disk operations per second, showing data on up to four disks
faults		Reports the trap/interrupt rates per second
	in	Interrupts per second
	sy	System calls per second
	cs	CPU context switch rate
cpu		Reports on the use of CPU time
	us	User time
	sy	System time
	id	Idle time

For a more detailed description of this command, see the vmstat(1M) man page.

Displaying Virtual memory Statistics (vmstat)

To display virtual memory statistics, use the vmstat command with a time interval in seconds.

```
$ vmstat n
```

where *n* is the interval in seconds between reports.

The following example shows the vmstat display of statistics that were gathered at five-second intervals:

```
$ vmstat 5
kthr      memory            page            disk          faults      cpu
 r b w   swap  free  re  mf pi po fr de sr dd f0 s1 --   in   sy   cs us sy id
 0 0 0 863160 365680  0   3  1  0  0  0  0  0  0  0  0  406  378  209  1  0 99
 0 0 0 765640 208568  0  36  0  0  0  0  0  0  0  0  0  479 4445 1378  3  3 94
 0 0 0 765640 208568  0   0  0  0  0  0  0  0  0  0  0  423  214  235  0  0 100
 0 0 0 765712 208640  0   0  0  0  0  0  0  3  0  0  0  412  158  181  0  0 100
 0 0 0 765832 208760  0   0  0  0  0  0  0  0  0  0  0  402  157  179  0  0 100
 0 0 0 765832 208760  0   0  0  0  0  0  0  0  0  0  0  403  153  182  0  0 100
 0 0 0 765832 208760  0   0  0  0  0  0  0  0  0  0  0  402  168  177  0  0 100
 0 0 0 765832 208760  0   0  0  0  0  0  0  0  0  0  0  402  153  178  0  0 100
 0 0 0 765832 208760  0  18  0  0  0  0  0  0  0  0  0  407  165  186  0  0 100
```

Displaying System Event Information (`vmstat -s`)

Run the `vmstat -s` command to show how many system events have taken place since the last time the system was booted.

```
$ vmstat -s
        0 swap ins
        0 swap outs
        0 pages swapped in
        0 pages swapped out
   522586 total address trans. faults taken
    17006 page ins
       25 page outs
    23361 pages paged in
       28 pages paged out
    45594 total reclaims
    45592 reclaims from free list
        0 micro (hat) faults
   522586 minor (as) faults
    16189 major faults
    98241 copy-on-write faults
   137280 zero fill page faults
    45052 pages examined by the clock daemon
        0 revolutions of the clock hand
       26 pages freed by the clock daemon
     2857 forks
       78 vforks
     1647 execs
 34673885 cpu context switches
 65943468 device interrupts
   711250 traps
 63957605 system calls
  3523925 total name lookups (cache hits 99%)
    92590 user   cpu
    65952 system cpu
 16085832 idle   cpu
     7450 wait   cpu
```

Displaying Swapping Statistics (`vmstat -S`)

Run `vmstat -S` to show swapping statistics.

```
$ vmstat -S
kthr      memory            page            disk          faults      cpu
 r b w   swap  free  si  so pi po fr de sr dd f0 s1 --   in   sy   cs us sy id
 0 0 0 862608 364792  0   0  1  0  0  0  0  0  0  0  0  406  394  213  1  0 99
```

The swapping statistics fields are described in the following list. For a description of the other fields, see Table 5, "Output From the `vmstat` Command," on page 51.

`si`	Average number of LWPs that are swapped in per second
`so`	Number of whole processes that are swapped out

Note - The `vmstat` command truncates the output of `si` and `so` fields. Use the `sar` command to display a more accurate accounting of swap statistics.

Displaying Interrupts Per Device (`vmstat -i`)

Run the `vmstat -i` command to show the number of interrupts per device.

The following example shows output from the `vmstat -i` command.

```
$ vmstat -i
interrupt         total     rate
--------------------------------
clock          52163269      100
esp0            2600077        4
zsc0              25341        0
zsc1              48917        0
cgsixc0             459        0
lec0             400882        0
fdc0                 14        0
bppc0                 0        0
audiocs0              0        0
--------------------------------
Total          55238959      105
```

Displaying Disk Utilization Information

Use the `iostat` command to report statistics about disk input and output, and to produce measures of throughput, utilization, queue lengths, transaction rates, and service time. For a detailed description of this command, refer to the `iostat(1M)` man page.

Displaying Disk Utilization Information (`iostat`)

You can display disk utilization information by using the `iostat` command with a time interval in seconds.

```
$ iostat 5
     tty          fd0           sd3          nfs1         nfs31          cpu
tin tout kps tps serv  kps tps serv  kps tps serv  kps tps serv  us sy wt id
  0    1   0   0  410    3   0   29    0   0    9    3   0   47   4  2  0 94
```

The first line of output shows the statistics since the last time the system was booted. Each subsequent line shows the interval statistics. The default is to show statistics for the terminal (`tty`), disks (`fd` and `sd`), and CPU (`cpu`).

The following example shows disk statistics that were gathered every five seconds.

```
$ iostat 5
tty        sd0           sd6           nfs1         nfs49          cpu
tin tout kps tps serv  kps tps serv  kps tps serv  kps tps serv  us sy wt id
 0    0   1   0   49    0   0    0    0   0    0    0   0   15   0  0  0 100
 0   47   0   0    0    0   0    0    0   0    0    0   0    0   0  0  0 100
 0   16   0   0    0    0   0    0    0   0    0    0   0    0   0  0  0 100
 0   16   0   0    0    0   0    0    0   0    0    0   0    0   0  0  0 100
 0   16  44   6  132    0   0    0    0   0    0    0   0    0   0  0  1 99
 0   16   0   0    0    0   0    0    0   0    0    0   0    0   0  0  0 100
 0   16   0   0    0    0   0    0    0   0    0    0   0    0   0  0  0 100
 0   16   0   0    0    0   0    0    0   0    0    0   0    0   0  0  0 100
 0   16   0   0    0    0   0    0    0   0    0    0   0    0   0  0  0 100
 0   16   0   0    0    0   0    0    0   0    0    0   0    0   0  0  0 100
 0   16   3   1   23    0   0    0    0   0    0    0   0    0   0  0  1 99
 0   16   0   0    0    0   0    0    0   0    0    0   0    0   0  0  0 100
 0   16   0   0    0    0   0    0    0   0    0    0   0    0   0  0  0 100
 0   16   0   0    0    0   0    0    0   0    0    0   0    0   0  0  0 100
```

The following table describes the fields in the output of the `iostat` *n* command.

Device Type	Field Name	Description
Terminal		
	`tin`	Number of characters in the terminal input queue
	`tout`	Number of characters in the terminal output queue
Disk		
	`bps`	Blocks per second
	`tps`	Transactions per second
	`serv`	Average service time, in milliseconds
CPU		
	`us`	In user mode
	`sy`	In system mode
	`wt`	Waiting for I/O
	`id`	Idle

Displaying Extended Disk Statistics (`iostat -xtc`)

Run the `iostat −xt` command to display extended disk statistics.

```
$ iostat &minus;xt
device     r/s    w/s   kr/s   kw/s wait actv wsvc_t asvc_t  %w  %b  tin tout
blkdev0    0.0    0.0    0.0    0.0  0.0  0.0    0.0    0.0   0   0    0    1
sd0        0.1   19.3    1.4   92.4  0.0  0.0    0.2    1.6   0   1
sd1        0.0    0.0    0.0    0.0  0.0  0.0    0.0    0.0   0   0
nfs9       0.0    0.0    0.0    0.0  0.0  0.0    0.0    1.0   0   0
nfs10      0.0    0.0    0.0    0.0  0.0  0.0    0.0    7.6   0   0
nfs11      0.0    0.0    0.0    0.0  0.0  0.0    0.0   15.6   0   0
nfs12      0.3    0.0    1.9    0.0  0.0  0.0    0.0   30.5   0   1
```

The `iostat −xt` command displays a line of output for each disk. The output fields are as follows:

`r/s` — Reads per second

`w/s` — Writes per second

`kr/s` — Kbytes read per second

`kw/s` — Kbytes written per second

`wait` — Average number of transactions that are waiting for service (queue length)

`actv` — Average number of transactions that are actively being serviced

`svc_t` — Average service time, in milliseconds

`%w` — Percentage of time that the queue is not empty

`%b` — Percentage of time that the disk is busy

Displaying Disk Space Statistics (`df`)

Use the `df` command to show the amount of free disk space on each mounted disk. The *usable* disk space that is reported by `df` reflects only 90 percent of full capacity because the reporting statistics allows for 10 percent above the total available space. This *head room* normally stays empty for better performance.

The percentage of disk space actually reported by the `df` command is used space divided by usable space.

If the file system exceeds 90 percent capacity, you could transfer files to a disk that is not as full by using the `cp` command. Alternately, you could transfer files to a tape by using the `tar` or `cpio` commands. Or, you could remove the files.

For a detailed description of this command, see the `df`(1M) man page.

Displaying Disk Space Information (`df -k`)

Use the `df -k` command to display disk space information in Kbytes.

```
$ df -k
Filesystem            kbytes    used   avail capacity  Mounted on
/dev/dsk/c0t3d0s0     192807   40231  133296    24%    /
```

EXAMPLE 15 Displaying File System Information

The following example shows the output from the `df -k` command for a SPARC system.

```
$ df -k
Filesystem           1024-blocks        Used   Available Capacity  Mounted on
rpool/ROOT/solaris-161   191987712     6004395   140577816     5%     /
/devices                     0           0           0     0%    /devices
/dev                         0           0           0     0%    /dev
ctfs                         0           0           0     0%    /system/contract
proc                         0           0           0     0%    /proc
mnttab                       0           0           0     0%    /etc/mnttab
swap                   4184236         496     4183740     1%    /system/volatile
objfs                        0           0           0     0%    /system/object
sharefs                      0           0           0     0%    /etc/dfs/sharetab
/usr/lib/libc/libc_hwcap1.so.1   146582211     6004395   140577816     5%    /lib/
libc.so.1
fd                           0           0           0     0%    /dev/fd
swap                   4183784          60     4183724     1%    /tmp
rpool/export         191987712          35   140577816     1%    /export
rpool/export/home    191987712          32   140577816     1%    /export/home
rpool/export/home/123   191987712     13108813   140577816     9%    /export/home/123
rpool/export/repo    191987712    11187204   140577816     8%    /export/repo
rpool/export/repo2010_11   191987712           31   140577816     1%    /export/
repo2010_11
rpool                191987712     5238974   140577816     4%    /rpool
/export/home/123     153686630    13108813   140577816     9%    /home/123
```

The output fields of the `df -k` command are as follows:

`1024-blocks` Total size of usable space in the file system

`Used`	Amount of space used
`Available`	Amount of space available for use
`Capacity`	Amount of space used, as a percentage of the total capacity
`Mounted on`	Mount point

EXAMPLE 16 Displaying File System Information by Using the `df` Command Without Any Options

When the `df` command is used without operands or options, it reports on all mounted file systems, as shown in the following example.

```
$ df
/                  (rpool/ROOT/solaris):100715496 blocks 100715496 files
/devices           (/devices          ):        0 blocks         0 files
/dev               (/dev              ):        0 blocks         0 files
/system/contract   (ctfs              ):        0 blocks 2147483601 files
/proc              (proc              ):        0 blocks     29946 files
/etc/mnttab        (mnttab            ):        0 blocks         0 files
/system/volatile   (swap              ):42257568 blocks  2276112 files
/system/object     (objfs             ):        0 blocks 2147483441 files
/etc/dfs/sharetab  (sharefs           ):        0 blocks 2147483646 files
/dev/fd            (fd                ):        0 blocks         0 files
/tmp               (swap              ):42257568 blocks  2276112 files
/export            (rpool/export      ):100715496 blocks 100715496 files
/export/home       (rpool/export/home ):100715496 blocks 100715496 files
/export/home/admin (rpool/export/home/admin):100715496 blocks 100715496 files
/rpool             (rpool             ):100715496 blocks 100715496 files
/export/repo2010_11(rpool/export/repo2010_11):281155639 blocks 281155639 files
/rpool             (rpool             ):281155639 blocks 281155639 files
```

Monitoring System Activities

This section describes activities for monitoring system activities.

Monitoring System Activities (`sar`)

Use the `sar` command to perform the following tasks:

- Organize and view data about system activity.
- Access system activity data on a special request basis.
- Generate automatic reports to measure and monitor system performance as well as special request reports to pinpoint specific performance problems. For information about how to

set up the `sar` command to run on your system as well as a description of these tools, see "Collecting System Activity Data Automatically (sar)" on page 75.

For a detailed description of this command, see the sar(1) man page.

Checking File Access (`sar -a`)

Display file access operation statistics with the `sar -a` command.

```
$ sar -a

SunOS t2k-brm-24 5.10 Generic_144500-10 sun4v    ...

00:00:00  iget/s namei/s dirbk/s
01:00:00       0       3       0
02:00:00       0       3       0
03:00:00       0       3       0
04:00:00       0       3       0
05:00:00       0       3       0
06:00:00       0       3       0
07:00:00       0       3       0
08:00:00       0       3       0
08:20:01       0       3       0
08:40:00       0       3       0
09:00:00       0       3       0
09:20:01       0      10       0
09:40:01       0       1       0
10:00:02       0       5       0

Average        0       4       0
```

The operating system routines that are reported by the `sar -a` command are as follows:

`iget/s` The number of requests made for inodes that were not in the directory name look-up cache (DNLC).

`namei/s` The number of file system path searches per second. If `namei` does not find a directory name in the DNLC, it calls `iget` to get the inode for either a file or directory. Hence, most `igets` are the result of DNLC misses.

`dirbk/s` The number of directory block reads issued per second.

The larger the reported values for these operating system routines, the more time the kernel is spending to access user files. The amount of time reflects how heavily programs and applications are using the file systems. The `-a` option is helpful for viewing how disk-dependent an application is.

Checking Buffer Activity (`sar -b`)

Display buffer activity statistics with the `sar -b` command.

The buffer is used to cache metadata. Metadata includes inodes, cylinder group blocks, and indirect blocks.

```
$ sar -b
00:00:00 bread/s lread/s %rcache bwrit/s lwrit/s %wcache pread/s pwrit/s
01:00:00       0       0     100       0       0      55       0       0
```

The following table describes the buffer activities that are displayed by the `-b` option.

Field Name	Description
`bread/s`	Average number of reads per second that are submitted to the buffer cache from the disk
`lread/s`	Average number of logical reads per second from the buffer cache
`%rcache`	Fraction of logical reads that are found in the buffer cache (100 % minus the ratio of `bread/s` to `lread/s`)
`bwrit/s`	Average number of physical blocks (512 bytes) that are written from the buffer cache to disk per second
`lwrit/s`	Average number of logical writes to the buffer cache per second
`%wcache`	Fraction of logical writes that are found in the buffer cache (100 % minus the ratio of `bwrit/s` to `lwrit/s`)
`pread/s`	Average number of physical reads per second that use character device interfaces
`pwrit/s`	Average number of physical write requests per second that use character device interfaces

The most important entries are the cache hit ratios `%rcache` and `%wcache`. These entries measure the effectiveness of system buffering. If `%rcache` falls below 90 percent or if `%wcache` falls below 65 percent, you might be able to improve performance by increasing the buffer space.

EXAMPLE 17 Checking Buffer Activity (`sar -b`)

The following example of `sar -b` command output shows that the `%rcache` and `%wcache` buffers are not causing any slowdowns. All the data is within acceptable limits.

```
$ sar -b

SunOS t2k-brm-24 5.10 Generic_144500-10 sun4v    ...

00:00:04 bread/s lread/s %rcache bwrit/s lwrit/s %wcache pread/s pwrit/s
01:00:00       0       0     100       0       0      94       0       0
02:00:01       0       0     100       0       0      94       0       0
```

```
03:00:00       0       0     100       0       0      92       0       0
04:00:00       0       1     100       0       1      94       0       0
05:00:00       0       0     100       0       0      93       0       0
06:00:00       0       0     100       0       0      93       0       0
07:00:00       0       0     100       0       0      93       0       0
08:00:00       0       0     100       0       0      93       0       0
08:20:00       0       1     100       0       1      94       0       0
08:40:01       0       1     100       0       1      93       0       0
09:00:00       0       1     100       0       1      93       0       0
09:20:00       0       1     100       0       1      93       0       0
09:40:00       0       2     100       0       1      89       0       0
10:00:00       0       9     100       0       5      92       0       0
10:20:00       0       0     100       0       0      68       0       0
10:40:00       0       1      98       0       1      70       0       0
11:00:00       0       1     100       0       1      75       0       0

Average        0       1     100       0       1      91       0       0
```

Checking System Call Statistics (`sar -c`)

Display system call statistics by using the `sar -c` command.

```
$ sar -c
00:00:00 scall/s sread/s swrit/s  fork/s  exec/s rchar/s wchar/s
01:00:00      38       2       2    0.00    0.00     149     120
```

The following list describes the system call categories that are reported by the `-c` option. Typically, reads and writes account for about half of the total system calls. However, the percentage varies greatly with the activities that are being performed by the system.

`scall/s` The number of all types of system calls per second, which is generally about 30 per second on a system with four to six users.

`sread/s` The number of `read` system calls per second.

`swrit/s` The number of `write` system calls per second.

`fork/s` The number of `fork` system calls per second, which is about 0.5 per second on a system with four to six users. This number increases if shell scripts are running.

`exec/s` The number of `exec` system calls per second. If `exec/s` divided by `fork/s` is greater than 3, look for inefficient `PATH` variables.

`rchar/s` The number of characters (bytes) transferred by `read` system calls per second.

`wchar/s` — The number of characters (bytes) transferred by `write` system calls per second.

EXAMPLE 18 Checking System Call Statistics (`sar -c`)

The following example shows output from the `sar -c` command.

```
$ sar -c

SunOS balmy 5.10 Generic_144500-10 sun4v ...
00:00:04 scall/s sread/s swrit/s  fork/s  exec/s rchar/s wchar/s
01:00:00      89      14       9    0.01    0.00    2906    2394
02:00:01      89      14       9    0.01    0.00    2905    2393
03:00:00      89      14       9    0.01    0.00    2908    2393
04:00:00      90      14       9    0.01    0.00    2912    2393
05:00:00      89      14       9    0.01    0.00    2905    2393
06:00:00      89      14       9    0.01    0.00    2905    2393
07:00:00      89      14       9    0.01    0.00    2905    2393
08:00:00      89      14       9    0.01    0.00    2906    2393
08:20:00      90      14       9    0.01    0.01    2914    2395
08:40:01      90      14       9    0.01    0.00    2914    2396
09:00:00      90      14       9    0.01    0.01    2915    2396
09:20:00      90      14       9    0.01    0.01    2915    2396
09:40:00     880     207     156    0.08    0.08   26671    9290
10:00:00    2020     530     322    0.14    0.13   57675   36393
10:20:00     853     129      75    0.02    0.01   10500    8594
10:40:00    2061     524     450    0.08    0.08  579217  567072
11:00:00    1658     404     350    0.07    0.06 1152916 1144203

Average      302      66      49    0.02    0.01   57842   55544
```

Checking Disk Activity (`sar -d`)

Display disk activity statistics with the `sar -d` command.

```
$ sar -d

00:00:00   device        %busy   avque   r+w/s  blks/s  avwait  avserv
```

The following list describes the disk device activities that are reported by the `-d` option.

`device` — Name of the disk device that is being monitored.

`%busy` — Portion of time the device was busy servicing a transfer request.

`avque` — Average number of requests during the time the device was busy servicing a transfer request.

`r+w/s` Number of read-and-write transfers to the device, per second.

`blks/s` Number of 512-byte blocks that are transferred to the device, per second.

`avwait` Average time, in milliseconds, that transfer requests wait in the queue. This time is measured only when the queue is occupied.

`avserv` Average time, in milliseconds, for a transfer request to be completed by the device. For disks, this value includes seek times, rotational latency times, and data transfer times.

EXAMPLE 19 Checking Disk Activity

This abbreviated example illustrates the output from the `sar -d` command.

```
$ sar -d

SunOS balmy 5.10 Generic_144500-10 sun4v    ...

12:36:32   device        %busy   avque   r+w/s  blks/s  avwait  avserv

12:40:01   dad1             15     0.7      26     399    18.1    10.0
           dad1,a           15     0.7      26     398    18.1    10.0
           dad1,b            0     0.0       0       1     1.0     3.0
           dad1,c            0     0.0       0       0     0.0     0.0
           dad1,h            0     0.0       0       0     0.0     6.0
           fd0               0     0.0       0       0     0.0     0.0
           nfs1              0     0.0       0       0     0.0     0.0
           nfs2              1     0.0       1      12     0.0    13.2
           nfs3              0     0.0       0       2     0.0     1.9
           nfs4              0     0.0       0       0     0.0     7.0
           nfs5              0     0.0       0       0     0.0    57.1
           nfs6              1     0.0       6     125     4.3     3.2
           nfs7              0     0.0       0       0     0.0     6.0
           sd1               0     0.0       0       0     0.0     5.4
           ohci0,bu          0     0.0       0       0     0.0     0.0
           ohci0,ct          0     0.0       0       0     0.0     0.0
           ohci0,in          0     0.0       7       0     0.0     0.0
           ohci0,is          0     0.0       0       0     0.0     0.0
           ohci0,to          0     0.0       7       0     0.0     0.0
```

Note that queue lengths and wait times are measured when something is in the queue. If `%busy` is small, large queues and service times probably represent the periodic efforts by the system to ensure that altered blocks are promptly written to the disk.

Checking Page-Out and Memory (`sar -g`)

Use the `sar -g` command to display page-out and memory freeing activities in averages.

```
$ sar -g
00:00:00  pgout/s ppgout/s pgfree/s pgscan/s %ufs_ipf
01:00:00     0.00     0.00     0.00     0.00     0.00
```

The output displayed by the `sar -g` command is a good indicator of whether more memory might be needed. Use the `ps -elf` command to show the number of cycles that are used by the `page` daemon. A high number of cycles, combined with high values for the `pgfree/s` and `pgscan/s` fields, indicates a memory shortage.

The `sar -g` command also shows whether inodes are being recycled too quickly and causing a loss of reusable pages.

The following list describes the output from the `-g` option.

`pgout/s` — The number of page-out requests per second.

`ppgout/s` — The actual number of pages that are paged-out per second. A single page-out request might involve paging-out multiple pages.

`pgfree/s` — The number of pages per second that are placed on the free list.

`pgscan/s` — The number of pages per second that are scanned by the `page` daemon. If this value is high, the `page` daemon is spending a lot of time checking for free memory. This situation implies that more memory might be needed.

`%ufs_ipf` — The percentage of `ufs` inodes taken off the free list by `iget` that had reusable pages associated with them. These pages are flushed and cannot be reclaimed by processes. Thus, this field represents the percentage of `igets` with page flushes. A high value indicates that the free list of inodes is page-bound, and that the number of `ufs` inodes might need to be increased.

EXAMPLE 20 Checking Page-Out and Memory (`sar -g`)

The following example shows output from the `sar -g` command.

```
$ sar -g

SunOS balmy 5.10 Generic_144500-10 sun4v    ...

00:00:00  pgout/s ppgout/s pgfree/s pgscan/s %ufs_ipf
01:00:00     0.00     0.00     0.00     0.00     0.00
02:00:00     0.01     0.01     0.01     0.00     0.00
03:00:00     0.00     0.00     0.00     0.00     0.00
04:00:00     0.00     0.00     0.00     0.00     0.00
05:00:00     0.00     0.00     0.00     0.00     0.00
06:00:00     0.00     0.00     0.00     0.00     0.00
```

```
07:00:00     0.00     0.00     0.00     0.00     0.00
08:00:00     0.00     0.00     0.00     0.00     0.00
08:20:01     0.00     0.00     0.00     0.00     0.00
08:40:00     0.00     0.00     0.00     0.00     0.00
09:00:00     0.00     0.00     0.00     0.00     0.00
09:20:01     0.05     0.52     1.62    10.16     0.00
09:40:01     0.03     0.44     1.47     4.77     0.00
10:00:02     0.13     2.00     4.38    12.28     0.00
10:20:03     0.37     4.68    12.26    33.80     0.00

Average      0.02     0.25     0.64     1.97     0.00
```

Checking Kernel Memory Allocation

The Kernel Memory Allocation (KMA) allows a kernel subsystem to allocate and free memory as needed.

Rather than statically allocating the maximum amount of memory that might be needed under peak load, the KMA divides requests for memory into three categories:

- Small (less than 256 bytes)
- Large (512 bytes to 4 Kbytes)
- Oversized (greater than 4 Kbytes)

The KMA keeps two pools of memory to satisfy small requests and large requests. The oversized requests are satisfied by allocating memory from the system page allocator.

The `sar -k` command is useful if you are checking a system that is being used to write drivers or STREAMS that use KMA resources. . Any driver or module that uses KMA resources but does not specifically return the resources before it exits can create a memory leak. A memory leak causes the amount of memory that is allocated by KMA to increase over time. Thus, if the `alloc` fields of the `sar -k` command increase steadily over time, there might be a memory leak. Another indication of a memory leak is failed requests. If this problem occurs, a memory leak has probably caused KMA to be unable to reserve and allocate memory.

If it appears that a memory leak has occurred, you should check any drivers or STREAMS that might have requested memory from KMA and not returned it.

Checking Kernel Memory Allocation (`sar -k`)

Use the `sar -k` command to report on activities of the Kernel Memory Allocator (KMA).

```
$ sar -k
00:00:00 sml_mem   alloc  fail  lg_mem   alloc   fail  ovsz_alloc  fail
01:00:00 2523136 1866512     0 18939904 14762364    0      360448     0
02:00:02 2523136 1861724     0 18939904 14778748    0      360448     0
```

The following list describes the output from the `-k` option.

`sml_mem`	The amount of memory in bytes that the KMA has available in the small memory request pool. In this pool, a small request is less than 256 bytes.
`alloc`	The amount of memory in bytes that the KMA has allocated from its small memory request pool to small memory requests.
`fail`	The number of requests for small amounts of memory that failed.
`lg_mem`	The amount of memory in bytes that the KMA has available in the large memory request pool. In this pool, a large request is from 512 bytes to 4 Kbytes.
`alloc`	The amount of memory in bytes that the KMA has allocated from its large memory request pool to large memory requests.
`fail`	The number of failed requests for large amounts of memory.
`ovsz_alloc`	The amount of memory that is allocated for oversized requests, which are requests that are greater than 4 Kbytes. These requests are satisfied by the page allocator. Thus, there is no pool.
`fail`	The number of failed requests for oversized amounts of memory.

EXAMPLE 21 Checking Kernel Memory Allocation (`sar -k`)

The following example shows an abbreviated example of `sar -k` output.

```
$ sar -k

SunOS balmy 5.10 Generic_144500-10 sun4v    ...
00:00:04 sml_mem   alloc  fail  lg_mem   alloc  fail  ovsz_alloc  fail
01:00:00 6119744 4852865     0 60243968 54334808   156     9666560     0
02:00:01 6119744 4853057     0 60243968 54336088   156     9666560     0
03:00:00 6119744 4853297     0 60243968 54335760   156     9666560     0
04:00:00 6119744 4857673     0 60252160 54375280   156     9666560     0
05:00:00 6119744 4858097     0 60252160 54376240   156     9666560     0
06:00:00 6119744 4858289     0 60252160 54375608   156     9666560     0
07:00:00 6119744 4858793     0 60252160 54442424   156     9666560     0
08:00:00 6119744 4858985     0 60252160 54474552   156     9666560     0
08:20:00 6119744 4858169     0 60252160 54377400   156     9666560     0
08:40:01 6119744 4857345     0 60252160 54376880   156     9666560     0
09:00:00 6119744 4859433     0 60252160 54539752   156     9666560     0
09:20:00 6119744 4858633     0 60252160 54410920   156     9666560     0
09:40:00 6127936 5262064     0 60530688 55619816   156     9666560     0
10:00:00 6545728 5823137     0 62996480 58391136   156     9666560     0
10:20:00 6545728 5758997     0 62996480 57907400   156     9666560     0
```

```
10:40:00 6734144 6035759     0 64389120 59743064   156    10493952     0
11:00:00 6996288 6394872     0 65437696 60935936   156    10493952     0

Average  6258044 5150556     0 61138340 55609004   156     9763900     0
```

Checking Interprocess Communication (`sar -m`)

Use the `sar -m` command to report interprocess communication activities.

```
$ sar -m
00:00:00   msg/s  sema/s
01:00:00    0.00    0.00
```

These figures are usually zero (0.00), unless you are running applications that use messages or semaphores.

The output from the `-m` option is as follows:

`msg/s`	The number of message operations (sends and receives) per second
`sema/s`	The number of semaphore operations per second

The following abbreviated example shows output from the `sar -m` command.

```
$ sar -m

SunOS balmy 5.10 Generic_144500-10 sun4v    ...

00:00:00   msg/s  sema/s
01:00:00    0.00    0.00
02:00:02    0.00    0.00
03:00:00    0.00    0.00
04:00:00    0.00    0.00
05:00:01    0.00    0.00
06:00:00    0.00    0.00

Average     0.00    0.00
```

Checking Page-In Activity (`sar -p`)

Use the `sar -p` command to report page-in activity, which includes protection and translation faults.

```
$ sar -p
00:00:00  atch/s  pgin/s ppgin/s  pflt/s  vflt/s slock/s
01:00:00    0.07    0.00    0.00    0.21    0.39    0.00
```

The following list describes the reported statistics from the `-p` option.

`atch/s` — The number of page faults per second that are satisfied by reclaiming a page currently in memory (attaches per second). Instances include reclaiming an invalid page from the free list and sharing a page of text that is currently being used by another process. An example is two or more processes that are accessing the same program text.

`pgin/s` — The number of times per second that file systems receive page-in requests.

`ppgin/s` — The number of pages paged in per second. A single page-in request, such as a soft-lock request (see `slock/s`) or a large block size, might involve paging-in multiple pages.

`pflt/s` — The number of page faults from protection errors. Instances of protection faults indicate illegal access to a page and copy-on-writes. Generally, this number consists primarily of copy-on-writes.

`vflt/s` — The number of address translation page faults per second. These faults are known as validity faults, faults occur when a valid process table entry does not exist for a given virtual address.

`slock/s` — The number of faults per second caused by software lock requests that require physical I/O. An example of the occurrence of a soft-lock request is the transfer of data from a disk to memory. The system locks the page that is to receive the data so that the page cannot be claimed and used by another process.

EXAMPLE 22 Checking Page-In Activity (`sar -p`)

The following example shows output from the `sar -p` command.

```
$ sar -p

SunOS balmy 5.10 Generic_144500-10 sun4v    ...

00:00:04  atch/s  pgin/s ppgin/s  pflt/s  vflt/s slock/s
01:00:00    0.09    0.00    0.00    0.78    2.02    0.00
02:00:01    0.08    0.00    0.00    0.78    2.02    0.00
03:00:00    0.09    0.00    0.00    0.81    2.07    0.00
04:00:00    0.11    0.01    0.01    0.86    2.18    0.00
05:00:00    0.08    0.00    0.00    0.78    2.02    0.00
06:00:00    0.09    0.00    0.00    0.78    2.02    0.00
07:00:00    0.08    0.00    0.00    0.78    2.02    0.00
08:00:00    0.09    0.00    0.00    0.78    2.02    0.00
08:20:00    0.11    0.00    0.00    0.87    2.24    0.00
08:40:01    0.13    0.00    0.00    0.90    2.29    0.00
09:00:00    0.11    0.00    0.00    0.88    2.24    0.00
09:20:00    0.10    0.00    0.00    0.88    2.24    0.00
```

```
09:40:00    2.91    1.80    2.38    4.61   17.62    0.00
10:00:00    2.74    2.03    3.08    8.17   21.76    0.00
10:20:00    0.16    0.04    0.04    1.92    2.96    0.00
10:40:00    2.10    2.50    3.42    6.62   16.51    0.00
11:00:00    3.36    0.87    1.35    3.92   15.12    0.00

Average     0.42    0.22    0.31    1.45    4.00    0.00
```

Checking Queue Activity (`sar -q`)

Use the `sar -q` command to report the following information:

- The average queue length while the queue is occupied.
- The percentage of time that the queue is occupied.

```
$ sar -q
00:00:00 runq-sz %runocc swpq-sz %swpocc
```

The output from the `-q` option is as follows.

`runq-sz`	The number of kernel threads in memory that are waiting for a CPU to run. Typically, this value should be less than 2. Consistently higher values mean that the system might be CPU-bound.
`%runocc`	The percentage of time that the dispatch queues are occupied.
`swpq-sz`	The average number of swapped out processes.
`%swpocc`	The percentage of time in which the processes are swapped out.

EXAMPLE 23 Checking Queue Activity

The following example shows output from the `sar -q` command. If the `%runocc` value is high (greater than 90 percent) and the `runq-sz` value is greater than 2, the CPU is heavily loaded and response is degraded. In this case, additional CPU capacity might be required to obtain acceptable system response.

```
# sar -q
SunOS balmy 5.10 Generic_144500-10 sun4v    ...

00:00:00 runq-sz %runocc swpq-sz %swpocc
01:00:00     1.0       7     0.0       0
02:00:00     1.0       7     0.0       0
03:00:00     1.0       7     0.0       0
04:00:00     1.0       7     0.0       0
05:00:00     1.0       6     0.0       0
06:00:00     1.0       7     0.0       0
```

```
Average      1.0       7     0.0       0
```

Checking Unused Memory (`sar -r`)

Use the `sar -r` command to report the number of memory pages and swap-file disk blocks that are currently unused.

```
$ sar -r
00:00:00 freemem freeswap
01:00:00    2135   401922
```

The output from the `-r` option is as follows:

`freemem` — The average number of memory pages that are available to user processes over the intervals sampled by the command. Page size is system-dependent.

`freeswap` — The number of 512-byte disk blocks that are available for page swapping.

EXAMPLE 24 Checking Unused Memory (`sar -r`)

The following example shows output from the `sar -r` command.

```
$ sar -r

SunOS balmy 5.10 Generic_144500-10 sun4v    ...

00:00:04 freemem freeswap
01:00:00   44717  1715062
02:00:01   44733  1715496
03:00:00   44715  1714746
04:00:00   44751  1715403
05:00:00   44784  1714743
06:00:00   44794  1715186
07:00:00   44793  1715159
08:00:00   44786  1714914
08:20:00   44805  1715576
08:40:01   44797  1715347
09:00:00   44761  1713948
09:20:00   44802  1715478
09:40:00   41770  1682239
10:00:00   35401  1610833
10:20:00   34295  1599141
10:40:00   33943  1598425
11:00:00   30500  1561959

Average    43312  1699242
```

Checking CPU Utilization (`sar -u`)

Use the `sar -u` command to display CPU utilization statistics.

```
$ sar -u
00:00:00    %usr    %sys    %wio   %idle
01:00:00       0       0       0     100
```

The `sar` command without any options is equivalent to the `sar -u` command. At any given moment, the processor is either busy or idle. When busy, the processor is in either user mode or system mode. When idle, the processor is either waiting for I/O completion or has no work to do.

The output from the `-u` option is as follows:

`%usr` The percentage of time that the processor is in user mode

`%sys` The percentage of time that the processor is in system mode

`%wio` The percentage of time that the processor is idle and waiting for I/O completion

`%idle` The percentage of time that the processor is idle and not waiting for I/O

A high `%wio` value generally means that a disk slowdown has occurred.

EXAMPLE 25 Checking CPU Utilization (`sar -u`)

The following example shows output from the `sar -u` command.

```
$ sar -u

00:00:04    %usr    %sys    %wio   %idle
01:00:00       0       0       0     100
02:00:01       0       0       0     100
03:00:00       0       0       0     100
04:00:00       0       0       0     100
05:00:00       0       0       0     100
06:00:00       0       0       0     100
07:00:00       0       0       0     100
08:00:00       0       0       0     100
08:20:00       0       0       0      99
08:40:01       0       0       0      99
09:00:00       0       0       0      99
09:20:00       0       0       0      99
09:40:00       4       1       0      95
10:00:00       4       2       0      94
10:20:00       1       1       0      98
```

```
10:40:00      18       3       0      79
11:00:00      25       3       0      72

Average        2       0       0      98
```

Checking System Table Status (`sar -v`)

Use the `sar -v` command to report the status of the process table, inode table, file table, and shared memory record table.

```
$ sar -v
00:00:00  proc-sz    ov  inod-sz    ov  file-sz    ov   lock-sz
01:00:00   43/922     0 2984/4236    0  322/322     0    0/0
```

Output from the `-v` option is described in the following list.

`proc-sz` The number of process entries (`proc` structures) that are currently being used, or allocated, in the kernel.

`inod-sz` The total number of inodes in memory compared to the maximum number of inodes that are allocated in the kernel. This number is not a strict high watermark. The number can overflow.

`file-sz` The size of the open system file table. `sz` is given as `0`, because space is allocated dynamically for the file table.

`ov` The overflows that occur between sampling points for each table.

`lock-sz` The number of shared memory record table entries that are currently being used, or allocated, in the kernel. `sz` is given as `0` because space is allocated dynamically for the shared memory record table.

EXAMPLE 26 Checking System Table Status (`sar -v`)

The following abbreviated example shows output from the `sar -v` command. This example shows that all tables are large enough to have no overflows. These tables are all dynamically allocated based on the amount of physical memory.

```
$ sar -v

00:00:04  proc-sz    ov  inod-sz    ov  file-sz    ov   lock-sz
01:00:00   69/8010    0 3476/34703    0    0/0       0    0/0
02:00:01   69/8010    0 3476/34703    0    0/0       0    0/0
03:00:00   69/8010    0 3476/34703    0    0/0       0    0/0
04:00:00   69/8010    0 3494/34703    0    0/0       0    0/0
05:00:00   69/8010    0 3494/34703    0    0/0       0    0/0
```

```
06:00:00   69/8010    0 3494/34703    0    0/0       0    0/0
07:00:00   69/8010    0 3494/34703    0    0/0       0    0/0
08:00:00   69/8010    0 3494/34703    0    0/0       0    0/0
08:20:00   69/8010    0 3494/34703    0    0/0       0    0/0
08:40:01   69/8010    0 3494/34703    0    0/0       0    0/0
09:00:00   69/8010    0 3494/34703    0    0/0       0    0/0
09:20:00   69/8010    0 3494/34703    0    0/0       0    0/0
09:40:00   74/8010    0 3494/34703    0    0/0       0    0/0
10:00:00   75/8010    0 4918/34703    0    0/0       0    0/0
10:20:00   72/8010    0 4918/34703    0    0/0       0    0/0
10:40:00   71/8010    0 5018/34703    0    0/0       0    0/0
11:00:00   77/8010    0 5018/34703    0    0/0       0    0/0
```

Checking Swapping Activity (`sar -w`)

Use the `sar -w` command to report swapping and switching activity.

```
$ sar -w
00:00:00 swpin/s bswin/s swpot/s bswot/s pswch/s
01:00:00    0.00     0.0    0.00     0.0      22
```

The following list describes target values and observations related to the `sar -w` command output.

`swpin/s`
: The number of LWP transfers into memory per second.

`bswin/s`
: The number of blocks transferred for swap-ins per second. `/* (float) PGTOBLK(xx->cvmi.pgswapin) / sec_diff */`.

`swpot/s`
: The average number of processes that are swapped out of memory per second. If the number is greater than 1, you might need to increase memory.

`bswot/s`
: The number of blocks that are transferred for swap-outs per second.

`pswch/s`
: The number of kernel thread switches per second.

Note - All process swap-ins include process initialization.

EXAMPLE 27 Checking Swap Activity (`sar -w`)

The following example shows output from the `sar -w` command.

```
$ sar -w

00:00:04 swpin/s bswin/s swpot/s bswot/s pswch/s
```

```
01:00:00    0.00     0.0    0.00     0.0     132
02:00:01    0.00     0.0    0.00     0.0     133
03:00:00    0.00     0.0    0.00     0.0     133
04:00:00    0.00     0.0    0.00     0.0     134
05:00:00    0.00     0.0    0.00     0.0     133
06:00:00    0.00     0.0    0.00     0.0     133
07:00:00    0.00     0.0    0.00     0.0     132
08:00:00    0.00     0.0    0.00     0.0     131
08:20:00    0.00     0.0    0.00     0.0     133
08:40:01    0.00     0.0    0.00     0.0     132
09:00:00    0.00     0.0    0.00     0.0     132
09:20:00    0.00     0.0    0.00     0.0     132
09:40:00    0.00     0.0    0.00     0.0     335
10:00:00    0.00     0.0    0.00     0.0     601
10:20:00    0.00     0.0    0.00     0.0     353
10:40:00    0.00     0.0    0.00     0.0     747
11:00:00    0.00     0.0    0.00     0.0     804

Average     0.00     0.0    0.00     0.0     198
```

Checking Terminal Activity (`sar -y`)

Use the `sar -y` command to monitor terminal device activities.

```
$ sar -y
00:00:00 rawch/s canch/s outch/s rcvin/s xmtin/s mdmin/s
01:00:00       0       0       0       0       0       0
```

If you have a lot of terminal I/O, you can use this report to determine whether any bad lines exist. The activities recorded are defined in the following list.

`rawch/s` — Input characters (raw queue) per second

`canch/s` — Input characters that are processed by canon (canonical queue) per second

`outch/s` — Output characters (output queue) per second

`rcvin/s` — Receiver hardware interrupts per second

`xmtin/s` — Transmitter hardware interrupts per second

`mdmin/s` — Modem interrupts per second

The number of modem interrupts per second (`mdmin/s`) should be close to zero. The receive and transmit interrupts per second (`xmtin/s` and `rcvin/s`) should be less than or equal to the number of incoming or outgoing characters, respectively. If not, check for bad lines.

EXAMPLE 28 Checking Terminal Activity (`sar -y`)

The following example shows output from the `sar -y` command.

```
$ sar -y

00:00:04 rawch/s canch/s outch/s rcvin/s xmtin/s mdmin/s
01:00:00       0       0       0       0       0       0
02:00:01       0       0       0       0       0       0
03:00:00       0       0       0       0       0       0
04:00:00       0       0       0       0       0       0
05:00:00       0       0       0       0       0       0
06:00:00       0       0       0       0       0       0
07:00:00       0       0       0       0       0       0
08:00:00       0       0       0       0       0       0
08:20:00       0       0       0       0       0       0
08:40:01       0       0       0       0       0       0
09:00:00       0       0       0       0       0       0
09:20:00       0       0       0       0       0       0
09:40:00       0       0       1       0       0       0
10:00:00       0       0      37       0       0       0
10:20:00       0       0       0       0       0       0
10:40:00       0       0       3       0       0       0
11:00:00       0       0       3       0       0       0

Average        0       0       1       0       0       0
```

Checking Overall System Performance (`sar -A`)

Use the `sar -A` command to display statistics from all options to provide a view of overall system performance.

This command provides a more global perspective. If data from more than a single time segment is shown, the report includes averages.

Collecting System Activity Data Automatically (`sar`)

Three commands are involved in the automatic collection of system activity data: `sadc`, `sa1`, and `sa2`.

The `sadc` data collection utility periodically collects data on system activity and saves the data in a file in binary format, one file for each 24-hour period. You can set up the `sadc` command to run periodically (usually once each hour), and whenever the system boots to multi-user mode.

The data files are placed in the `/var/adm/sa` directory. Each file is named `sa`*dd*, where *dd* is the current date. The format of the command is as follows:

```
/usr/lib/sa/sadc [t n] [ofile]
```

The command samples *n* times with an interval of *t* seconds, which should be greater than five seconds between samples. This command then writes to the binary *ofile* file, or to standard output.

Running the `sadc` Command When Booting

The `sadc` command should be run at system boot time to record the statistics from when the counters are reset to zero. To make sure that the `sadc` command is run at boot time, the `svcadm enable system/sar:default` command writes a record to the daily data file.

The command entry has the following format:

```
/usr/bin/su sys -c "/usr/lib/sa/sadc /var/adm/sa/sa`date +%d`"
```

Running the `sadc` Command Periodically With the `sa1` Script

To generate periodic records, you need to run the `sadc` command regularly. The simplest way to do so is to uncomment the following lines in the `/var/spool/cron/crontabs/sys` file:

```
# 0 * * * 0-6 /usr/lib/sa/sa1
# 20,40 8-17 * * 1-5 /usr/lib/sa/sa1
# 5 18 * * 1-5 /usr/lib/sa/sa2 -s 8:00 -e 18:01 -i 1200 -A
```

The default `sys crontab` entries do the following:

- The first two `crontab` entries cause a record to be written to the `/var/adm/sa/sa`*dd* file every 20 minutes from 8 am to 5 pm, Monday through Friday, and every hour on the hour otherwise.
- The third entry writes a record to the `/var/adm/sa/sar`*dd* file hourly, Monday through Friday, and includes all `sar` options.

You can change these defaults to meet your needs.

Producing Reports With the `sa2` Shell Script

Another shell script, `sa2`, produces reports rather than binary data files. The `sa2` command invokes the `sar` command and writes the ASCII output to a report file.

Setting Up Automatic Data Collection (`sar`)

The `sar` command can be used either to gather system activity data itself or to report what has been collected in the daily activity files that are created by the `sadc` command.

The `sar` command has the following formats:

`sar [-aAbcdgkmpqruvwy] [-o` *file*`]` *t [n]*

`sar [-aAbcdgkmpqruvwy] [-s` *time*`] [-e` *time*`] [-i` *sec*`] [-f` *file*`]`

The first format samples cumulative activity counters in the operating system every *t* seconds, *n* times. The *t* should be five seconds or greater. Otherwise, the command itself might affect the sample. You must specify a time interval in which to take the samples. Otherwise, the command operates according to the second format. The default value of *n* is 1.

The following example, using the second format, takes two samples separated by 10 seconds. If the `-o` option were specified, samples are saved in binary format.

```
$ sar -u 10 2
```

The `sar` command with the second format, with no sampling interval or number of samples specified, extracts data from a previously recorded file. This file is either the file specified by the `-f` option or, by default, the standard daily activity file, `/var/adm/sa/sa`*dd*, for the most recent day.

The `-s` and `-e` options define the starting time and the ending time for the report. Starting and ending times are of the form *hh*[*:mm*[*:ss*]], where *hh*, *mm*, and *ss* represent hours, minutes, and seconds.

The `-i` option specifies, in seconds, the intervals between record selection. If the `-i` option is not included, all intervals that are found in the daily activity file are reported.

The `sar` options and their actions are as follows:

Note - Using no option is equivalent to calling the `sar` command with the `-u` option.

`-a`	Checks file access operations
`-b`	Checks buffer activity
`-c`	Checks system calls
`-d`	Checks activity for each block device
`-g`	Checks page-out and memory freeing
`-k`	Checks kernel memory allocation

`-m`	Checks interprocess communication
`-nv`	Checks system table status
`-p`	Checks swap and dispatch activity
`-q`	Checks queue activity
`-r`	Checks unused memory
`-u`	Checks CPU utilization
`-w`	Checks swapping and switching volume
`-y`	Checks terminal activity
`-A`	Reports overall system performance, which is the same as entering all options

▼ How to Set Up Automatic Data Collection

1. **Assume the `root` role.**

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Run the `svcadm enable system/sar:default` command.**

   This version of the `sadc` command writes a special record that marks the time when the counters are reset to zero (boot time).

3. **Edit the `/var/spool/cron/crontabs/sys crontab` file.**

   Note - Do not edit a `crontab` file directly. Instead, use the `crontab -e` command to make changes to an existing `crontab` file.

   ```
   # crontab -e sys
   ```

4. **Uncomment the following lines:**

   ```
   0 * * * 0-6 /usr/lib/sa/sa1
   20,40 8-17 * * 1-5 /usr/lib/sa/sa1
   5 18 * * 1-5 /usr/lib/sa/sa2 -s 8:00 -e 18:01 -i 1200 -A
   ```

   For more information, see the `crontab(1)` man page.

CHAPTER 4

Scheduling System Tasks

This chapter describes how to schedule routine system tasks or single (one-time) tasks by using the Oracle Solaris Service Management Facility (SMF), the `crontab` command, and `at` command.

It covers the following topics:

Overview of Scheduled System Tasks

You can set up many system tasks to execute automatically. Some of these tasks should occur at regular intervals, while others for only a specific duration. Then, there are tasks that need to run only once, perhaps during off peak hours such as evenings and weekends.

This section contains overview information about scheduling system tasks using the Oracle Solaris Service Management Facility (SMF), or the `crontab` command or `at` command. Each has a primary function for which it should be used. In general, SMF provides a a simpler scheduling mechanism for resource monitoring, though it does have `cron`-like scheduling capabilities. With SMF, you can set up both periodic services and scheduled services. A periodic service in SMF runs routine tasks within a designated time frame or "interval". An SMF scheduled service always runs according to an assigned start time. To execute repetitive or routine jobs (at a specific start time), use the `crontab` command. To schedule a single job or to execute a task once (at a specific time), use the `at` command.

The following table summarizes SMF, the `crontab` command, and the `at` command, as well as the files or propertiess for these functions.

TABLE 6 Tools for Automatically Executing Tasks

Tool	What It Schedules	Access Control Files or Properties
Periodic SMF service	Multiple system tasks at regular intervals (relative to the time of previous invocation)	`solaris.smf.modify` `solaris.smf.manage`
Scheduled SMF service	Multiple system tasks at a specific time	`scheduled/schedule` sets the frequency to run. Other `scheduled` properties specify the time to run.
`crontab` command	Multiple system tasks at regular intervals (but at specific times)	`/var/spool/cron/crontabs`
`at` command	A single system task at a specific time	`/var/spool/cron/atjobs`

Scheduling a Periodic or Scheduled Task With SMF

You can manage running applications or services with a Solaris service management facility known as SMF. Services are represented in the SMF framework by service and instance objects and their configuration settings. The configuration of the local Oracle Solaris instance is called the `localhost scope`, and is the only supported scope as of now. For additional information, check the `smf(5)` man page.

SMF services that help systems run routine maintenance tasks at regular intervals are called *periodic* or *scheduled* services. A scheduled service is a type of periodic service that occurs at a specific time. Use a scheduled service for tasks that run occasionally or on a specific schedule (such as off-peak hours). For more information on scheduled services, see Chapter 4, "Creating a Service to Run on a Specific Schedule" in *Developing System Services in Oracle Solaris 11.3*.

A periodic service, on the other hand, begins the start method at a time relative to the last run. It is used for maintenance tasks that occur more frequently or regularly. In SMF, a periodic service is managed by the delegated restarter `svc:/system/svc/periodic-restarter`. This restarter runs the start method only for the instances that it manages. That is, the scheduled task will begin only at specified intervals for the duration of time that such instances are online. For more information on periodic services, see Chapter 3, "Creating a Service to Run Periodically" in *Developing System Services in Oracle Solaris 11.3*.

Advantages of using SMF to schedule tasks include the following:

- Automatically restarts any failed services in dependency order (whereas `cron` typically does not restart itself)
- Services are well integrated with the operating system and can easily be controlled with dependencies

- Debugs and reports on service problems, detailing why or how a scheduled service has failed
- Ensures that the task runs only when the (required) `IPS` package software is running
- Requires no additional steps for removing the scheduled task (uninstalls automatically with the `IPS` package)
- Delegates tasks to non-root users, with the ability to modify properties and manage services (which isn't possible in `cron`)
- Manages multiple users in different time zones and settings

---

Note - Users without `root` access are unable to create their own scheduled services in SMF. Therefore, `cron` or `at` are the only options for underprivileged users.

For step-by-step instructions on scheduling SMF tasks, see "Scheduling Repetitive System Tasks Using SMF" on page 83.

Scheduling a Routine System Task With `crontab`

`cron` runs a process that executes commands at specified dates and times. Unlike SMF, it only examines `crontab` or `at` command files during its own process initialization phase or when the two commands are run. It does not check for new or changed files at regularly scheduled intervals.

You can specify regularly scheduled commands to `cron` according to instructions found in the `crontab` files. Users can submit their own system tasks in `cron` using the `crontab` command; `root` access is not required as it is for SMF. The `crontab` command also allows you to schedule tasks that need to be executed more than once, whereas the `at` command only allows for a one-time run. Note that `cron` never exits, so it should only be executed once.

For step-by-step instructions on scheduling `crontab` jobs, see "How to Create or Edit a `crontab` File" on page 87.

Scheduling a Single System Task With `at`

The `at` command enables you to schedule a one-time or infrequent task for execution at a prescribed time. The job can consist of a single command or script.

Similar to `crontab`, the `at` command allows you to schedule the automatic execution of a system task. However, unlike `crontab` files, `at` files execute their tasks just once. They are then removed from their directory. Therefore, the at command is most useful for running simple commands or scripts that direct output into separate files for later examination.

Submitting an at job involves typing a command and following the at command syntax to specify options to schedule the time your job will be executed. For more information about submitting at jobs, see "Submitting an at Job File" on page 95.

The at command stores the command or script you ran, along with a copy of your current environment variable, in the /var/spool/cron/atjobs directory. The file name for an at job consists of a long number that specifies its location in the at queue followed by the .a extension, for example, 793962000.a.

The cron daemon checks for at jobs at startup and listens for new jobs that are submitted. After the cron daemon executes an at job, the at job's file is removed from the atjobs directory. For more information, see the at(1) man page.

For step-by-step instructions on scheduling at jobs, see "How to Create an at Job" on page 95.

Examples of Repetitive System Tasks

You can schedule routine system administration tasks to execute daily, weekly, or monthly. Depending on the task requirements or assigned access control rights, you can use one of the aforementioned scheduling tools in Solaris.

Daily system administration tasks might include the following:

- Removing files more than a few days old from temporary directories
- Executing accounting summary commands
- Taking snapshots of the system by using the df and ps commands
- Performing daily security monitoring
- Running system backups

Weekly system administration tasks might include the following:

- Rebuilding the catman database for use by the man -k command
- Running the fsck -n command to list any disk problems

Monthly system administration tasks might include the following:

- Listing files not used during a specific month
- Producing monthly accounting reports

Additionally, you can schedule other routine system tasks, such as sending reminders and removing backup files.

Scheduling Repetitive System Tasks Using SMF

The following section includes information about how to create, modify, and display SMF service instances. It also covers information about SMF access controls:

- "How SMF Handles Scheduling" on page 83
- "Scheduling Method and Time Values for SMF" on page 83
- "How SMF Handles Scheduling" on page 83

How SMF Handles Scheduling

Each SMF scheduled service is managed by a restarter. The master restarter `svc.startd` manages states for the entire set of service instances and their dependencies. The master restarter acts on behalf of its services and on delegated restarters that can provide specific execution environments for certain application classes. For example, `inetd` is a delegated restarter that provides its service instances with an initial environment. Each service instance delegated to `inetd` is in the online state. While the daemon of a particular instance might not be running, the instance is available to run. As dependencies are satisfied when instances move to the online state, `svc.startd` invokes start methods of other instances which may overlap. See the `smf_restarter`(5) man page to view the configuration settings for all SMF restarters.

Each service or service instance must define a set of methods that start, stop, and, refresh the service. See the `smf_method`(5) man page for a more complete description of the method conventions for `svc.startd` and similar restarters. Administrative methods, such as for the capture of legacy configuration information into the repository, are discussed in the `svccfg`(1M) man page.

Scheduling Method and Time Values for SMF

A scheduled service instance in SMF requires a specific time which is delegated by the `scheduled_method` element. The `scheduled_method` element specifies both method and scheduling information for scheduled services.

TABLE 7 Acceptable Numerical Values for SMF Scheduling

Time Field	Numerical Values
Minute	0-59
Hour	0-23
Day of month	1-31
Month	1-12
Day of week	0-6 (0 = Sunday)

For more information on constraints and how to specify the scheduled_method element, see "Specifying the scheduled_method Element" in *Developing System Services in Oracle Solaris 11.3*.

Examples of SMF Manifests

EXAMPLE 29 Creating an SMF Scheduled Service

The following example shows how to create an SMF scheduled service instance to run automatically at 1:00 a.m. every Sunday morning.

```
<?xml version='1.0'?>
<!DOCTYPE service_bundle
  SYSTEM '/usr/share/lib/xml/dtd/service_bundle.dtd.1'>
<service_bundle type='manifest' name='site/sample-periodic-svc'>
    <service type='service' version='1' name='site/sample-periodic-svc'>

        <instance name='default' enabled='false'>

            <scheduled_method
                schedule='month'
                day='0'
                hour='1'
                minute='0'
                exec='/usr/bin/scheduled_service_method'
                timeout_seconds='0'>
                    <method_context>
                        <method_credential user='root' group='root' />
                    </method_context>
            </scheduled_method>

        </instance>
    </service>
</service_bundle>
```

For more step-by-step information on how to create a periodic service, see Chapter 3, "Creating a Service to Run Periodically" in *Developing System Services in Oracle Solaris 11.3*.

Scheduling a Repetitive System Task Using crontab

The following section describes how to schedule routine system tasks by using the crontab command. Information on how to create, edit, display, and remove crontab files are in the following sections:

Inside a crontab File

The cron daemon schedules system tasks according to commands found within each crontab file. A crontab file consists of commands, one command per line, that will be executed at regular intervals. The beginning of each line contains date and time information that tells the cron daemon when to execute the command.

For example, a crontab file named root is supplied during Oracle Solaris software installation. The file's contents include these command lines:

```
10 3 * * * /usr/sbin/logadm     (1)
15 3 * * 0 /usr/lib/fs/nfs/nfsfind     (2)
1 2 * * * [ -x /usr/sbin/rtc ] && /usr/sbin/rtc -c > /dev/null 2>&1     (3)
30 3 * * * [ -x /usr/lib/gss/gsscred_clean ] && /usr/lib/gss/gsscred_clean     (4)
```

The output for each of these command lines is as follows:

- The first line runs the logadm command at 3:10 am every day.
- The second line executes the nfsfind script every Sunday at 3:15 am.
- The third line runs a script that checks for daylight savings time (and make corrections, if necessary) at 2:10 am daily.

  If there is no RTC time zone or /etc/rtc_config file, this entry does nothing.

  ---

  x86 only - The /usr/sbin/rtc script can be run only on an x86 based system.

- The fourth line checks for (and removes) duplicate entries in the Generic Security Service table, /etc/gss/gsscred_db, at 3:30 am daily.

For more information about the syntax of lines within a crontab file, see "Syntax of crontab File Entries" on page 86.

The crontab files are stored in the /var/spool/cron/crontabs directory. Several crontab files besides root are provided during Oracle Solaris software installation.

adm	Accounting

`root`	General system functions and file system cleanup
`sys`	Performance data collection
`uucp`	General `uucp` cleanup

Besides the default `crontab` files, you can create `crontab` files to schedule your own system tasks. Custom `crontab` files are named after the user accounts in which they are created, such as `bob`, `mary`, `smith`, or `jones`.

To access `crontab` files that belong to `root` or other users, superuser privileges are required.

How the `cron` Daemon Handles Scheduling

The `cron` daemon manages the automatic scheduling of `crontab` commands. The role of the `cron` daemon is to check the `/var/spool/cron/crontab` directory for the presence of `crontab` files.

The `cron` daemon performs the following tasks at startup:

- Checks for new `crontab` files.
- Reads the execution times that are listed within the files.
- Submits the commands for execution at the proper times.
- Listens for notifications from the `crontab` commands regarding updated `crontab` files.

In much the same way, the `cron` daemon controls the scheduling of `at` files. These files are stored in the `/var/spool/cron/atjobs` directory. The `cron` daemon also listens for notifications from the `crontab` commands regarding submitted `at` jobs.

Syntax of `crontab` File Entries

A `crontab` file consists of commands, one command per line, that execute automatically at the time specified by the first five fields of each command line, which are separated by spaces.

TABLE 8 Acceptable Values for `crontab` Time Fields

Time Field	Values
Minute	0-59
Hour	0-23
Day of month	1-31
Month	1-12
Day of week	0-6 (0 = Sunday)

Follow these guidelines for using special characters in crontab time fields:

- Use a space to separate each field.
- Use a comma to separate multiple values.
- Use a hyphen to designate a range of values.
- Use an asterisk as a wildcard to include all possible values.
- Use a comment mark (#) at the beginning of a line to indicate a comment or a blank line.

For example, the following crontab command entry displays a reminder in the user's console window at 4 pm on the first and fifteenth days of every month.

```
0 16 1,15 * * echo Timesheets Due > /dev/console
```

Each command within a crontab file must consist of one line, even if that line is very long. The crontab file does not recognize extra carriage returns. For more detailed information about crontab entries and command options, refer to the crontab(1) man page.

Creating and Editing crontab Files

The simplest way to create a crontab file is to use the crontab -e command. This command invokes the text editor that has been defined for your system environment in the EDITOR environment variable. If this variable has not been set, the crontab command uses the default editor, ed.

The following example shows how to determine whether an editor has been defined, and sets up vi as the default editor.

```
$ which $EDITOR
$
$ EDITOR=vi
$ export EDITOR
```

When you create a crontab file, it is automatically placed in the /var/spool/cron/crontabs directory and is given your user name. You can create or edit a crontab file for another user, or root if you have root privileges.

▼ How to Create or Edit a crontab File

Before You Begin If you are creating or editing a crontab file that belongs to another user, you must assume the root role. See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

You do not need to assume the root role to edit your own crontab file.

1. **Create a new crontab file, or edit an existing file.**

```
# crontab -e [username]
```

where *username* specifies the name of the user's account for which you want to create or edit a `crontab` file. You can create your own `crontab` file without superuser privileges, but you must have superuser privileges to creating or edit a `crontab` file for `root` or another user.

Caution - If you accidentally type the `crontab` command with no option, press the interrupt character for your editor that enables you to quit without saving changes. If you instead save changes and exit the file, the existing `crontab` file will be overwritten with an empty file.

2. **Add command lines to the `crontab` file.**

   Follow the syntax described in "Syntax of `crontab` File Entries" on page 86. The `crontab` file will be placed in the `/var/spool/cron/crontabs` directory.

3. **Verify your `crontab` file changes.**

```
# crontab -l [username]
```

Example 30 Creating a `crontab` File

The following example shows how to create a `crontab` file for another user.

```
# crontab -e mary
```

The following command entry added to a new `crontab` file automatically removes any log files from Mary's home directory at 1:00 am every Sunday morning. Because the command entry does not redirect output, redirect characters are added to the command line after `*.log`. Doing so ensures that the command executes properly.

```
# This command helps clean up user accounts.
1 0 * * 0 rm /home/mary/*.log > /dev/null 2>&1
```

Displaying and Verifying `crontab` Files

You can use the `crontab -l` command to display and verify contents of a `crontab` file.

Verifying That a `crontab` File Exists

To verify that a `crontab` file exists for a user, use the `ls -l` command in the `/var/spool/cron/crontabs` directory. For example, the following sample output shows that `crontab` files exist for various users on the system.

```
$ ls -l /var/spool/cron/crontabs
```

```
drwxr-xr-x   2 root     sys          12 Nov 26 16:55 ./
drwxr-xr-x   4 root     sys           4 Apr 28  2012 ../
-rw-------   1 root     sys         190 Jun 28  2011 adm
-rw-------   1 root     staff         0 Nov 13  2012 mary
-rw-------   1 root     un          437 Oct  8  2012 johndoe
-r--------   1 root     root        453 Apr 28  2012 lp
-rw-------   1 root     sparccad     63 Jul 17 10:39 mary2
-rw-------   1 root     sparccad    387 Oct 14 15:15 johndoe2
-rw-------   1 root     other      2467 Nov 26 16:55 root
-rw-------   1 root     sys         308 Jun 28  2011 sys
-rw-------   1 root     siete       163 Nov 20 10:40 mary3
-r--------   1 root     sys         404 Jan 24  2013 uucp
```

Displaying a `crontab` File

The `crontab -l` command displays the contents of a `crontab` file much the same way that the `cat` command displays the contents of other types of files. You do not have to change the directory to `/var/spool/cron/crontabs` directory (where `crontab` files are located) to use this command.

By default, the `crontab -l` command displays your own `crontab` file. To display `crontab` files that belong to other users, you must assume the root role.

`You can use the crontab command as follows:`

`#` **`crontab -l`** `[`*username*`]`

where *username* specifies the name of the user's account for which you want to display a `crontab` file. Displaying another user's `crontab` file requires `root` privileges.

Caution - If you accidentally type the `crontab` command with no option, press the interrupt character for your editor to quit without saving changes. If you instead save changes and exit the file, the existing `crontab` file will be overwritten with an empty file.

EXAMPLE 31 Displaying a `crontab` File

This example shows how to use the `crontab -l` command to display the contents of the your default `crontab` file.

```
$ crontab -l
13 13 * * * chmod g+w /home1/documents/*.book > /dev/null 2>&1
```

EXAMPLE 32 Displaying the Default `root` `crontab` file.

This example shows how to display the default `root` `crontab` file.

```
$ su
Password:

# crontab -l
#ident  "@(#)root       1.19    98/07/06 SMI"   /* SVr4.0 1.1.3.1       */
#
# The root crontab should be used to perform accounting data collection.
#
#
10 3 * * * /usr/sbin/logadm
15 3 * * 0 /usr/lib/fs/nfs/nfsfind
30 3 * * * [ -x /usr/lib/gss/gsscred_clean ] && /usr/lib/gss/gsscred_clean
#10 3 * * * /usr/lib/krb5/kprop_script ___slave_kdcs___
```

EXAMPLE 33 Displaying the `crontab` File of Another User

This example shows how to display the `crontab` file that belongs to another user.

```
$ su
Password:
# crontab -l jones
13 13 * * * cp /home/jones/work_files /usr/backup/. > /dev/null 2>&1
```

Removing `crontab` Files

By default, `crontab` file protections are set up so that you cannot inadvertently delete a `crontab` file by using the `rm` command. Instead, use the `crontab -r` command to remove `crontab` files.

By default, the `crontab -r` command removes your own `crontab` file.

You do not have to change the directory to `/var/spool/cron/crontabs` (where `crontab` files are located) to use this command.

▼ How to Remove a `crontab` File

Before You Begin Assume the `root` role to remove a `crontab` file that belongs to `root` or another user. Roles contain authorizations and privileged commands. See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

You do not need to assume the `root` role to remove your own `crontab` file.

1. **Remove the `crontab` file.**

   ```
   # crontab -r [username]
   ```

where *username* specifies the name of the user's account for which you want to remove a `crontab` file. To remove `crontab` files for another user, assume the root role.

Caution - If you accidentally type the `crontab` command with no option, press the interrupt character for your editor to quit without saving changes. If you instead save changes and exit the file, the existing `crontab` file will be overwritten with an empty file.

2. **Verify that the `crontab` file has been removed.**

```
# ls /var/spool/cron/crontabs
```

Example 34 Removing a `crontab` File

The following example shows how user `smith` uses the `crontab -r` command to remove his own `crontab` file.

```
$ ls /var/spool/cron/crontabs
adm     jones     root    smith    sys     uucp
$ crontab -r
$ ls /var/spool/cron/crontabs
adm     jones root    sys    uucp
```

Controlling Access to the `crontab` Command

You can control access to the `crontab` command by using two files in the `/etc/cron.d` directory: `cron.deny` and `cron.allow`. These files permit only specified users to perform `crontab` command tasks such as creating, editing, displaying, or removing their own `crontab` files.

The `cron.deny` and `cron.allow` files consist of a list of user names, one user name per line.

These access control files work together as follows:

- If `cron.allow` exists, only the users who are listed in this file can create, edit, display, or remove `crontab` files.
- If `cron.allow` does not exist, all users can submit `crontab` files except for users who are listed in `cron.deny`.
- If neither `cron.allow` nor `cron.deny` exists, you must assume the root role to run the `crontab` command.
- In order to edit or create the `cron.deny` and `cron.allow` files, you must assume the root role.

The `cron.deny` file, which is created during Oracle Solaris software installation, contains the following user names:

```
$ cat /etc/cron.d/cron.deny
daemon
bin
smtp
nuucp
listen
nobody
noaccess
```

None of the user names in the default `cron.deny` file can access the `crontab` command. You can edit this file to add other users that will be denied access to the `crontab` command.

Because no default `cron.allow` file is supplied, all users except users who are listed in the default `cron.deny` file can access the `crontab` command. If you create a `cron.allow` file, only these users can access the `crontab` command.

How to Deny `crontab` Command Access

1. **Assume the `root` role.**

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Edit the `/etc/cron.d/cron.deny` file and add user names, one user per line, who will be denied access to the `crontab` commands.**

   ```
   daemon
   bin
   smtp
   nuucp
   listen
   nobody
   noaccess
   username1
   username2
   username3
   .
   .
   .
   ```

3. **Verify that the `/etc/cron.d/cron.deny` file contains the new entries.**

   ```
   # cat /etc/cron.d/cron.deny
   daemon
   bin
   nuucp
   listen
   nobody
   ```

```
noaccess
```

▼ How to Limit `crontab` Command Access to Specified Users

1. **Assume the `root` role.**

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Create the `/etc/cron.d/cron.allow` file.**

3. **Add the `root` role to the `cron.allow` file.**

   If you do not add `root` to the file, root access to `crontab` commands will be denied.

4. **Add the user names, one user name per line, who will be allowed to use the `crontab` command.**

   `root`
   username1
   username2
   username3
   .
   .
   .

Example 35 Limiting `crontab` Command Access to Specified Users

The following example shows a `cron.deny` file that prevents user names `jones`, `temp`, and `visitor` from accessing the `crontab` command.

```
$ cat /etc/cron.d/cron.deny
daemon
bin
smtp
nuucp
listen
nobody
noaccess
jones
temp
visitor
```

The following example shows a `cron.allow` file. The users `root`, `jones`, and `smith` are the only users who can access the `crontab` command.

```
$ cat /etc/cron.d/cron.allow
root
jones
```

```
smith
```

Verifying Limited crontab Command Access

To verify whether a specific user can access the `crontab` command, use the `crontab -l` command while you are logged into the user account.

```
$ crontab -l
```

If the user can access the `crontab` command and already has created a `crontab` file, the file is displayed. Otherwise, if the user can access the `crontab` command but no `crontab` file exists, a message similar to the following message is displayed:

```
crontab: can't open your crontab file
```

Either this user either is listed in the `cron.allow` file (if the file exists) or the user is not listed in the `cron.deny` file.

If the user cannot access the `crontab` command, the following message is displayed regardless of whether a previous `crontab` file exists:

```
crontab: you are not authorized to use cron. Sorry.
```

This message means that either the user is not listed in the `cron.allow` file (if the file exists) or the user is listed in the `cron.deny` file.

Scheduling A Single System Task by Using the at Command

The following section describes how to schedule routine system tasks by using the `at` command. Examples for scheduling system tasks with `at` are in the following sections:

- "How to Create an at Job" on page 95
- "Displaying the at Queue" on page 96
- "Verifying an at Job" on page 96
- "Displaying at Jobs" on page 97
- "How to Remove at Jobs" on page 97
- "Denying Access to the at Command" on page 98

By default, users can create, display, and remove their own `at` job files. To access `at` files that belong to `root` or other users, you must assume the root role.

Submitting an at Job File

When you submit an at job, it is assigned a job identification number along with the .a extension. This designation becomes the job's file name as well as its queue number.

Submitting an at job file involves these steps:

1. Invoking the at utility and specifying a command execution time.
2. Typing a command or script to execute later.

Note - If output from this command or script is important, be sure to direct the output to a file for later examination.

For example, the following at job removes core files from the user account smith near midnight on the last day of July.

```
$ at 11:45pm July 31
at> rm /home/smith/*core*
at> Press Control-d
commands will be executed using /bin/csh
job 933486300.a at Tue Jul 31 23:45:00 2004
```

Creating an at Job

The following task describes how to create and at job.

▼ How to Create an at Job

1. **Start the at utility, specifying the time you want your job executed.**

```
$ at [-m] time [date]
```

-m	Specifies to send you an email after the job is completed.
time	Specifies the hour that you want to schedule the job. Add am or pm if you do not specify the hours according to the 24-hour clock. Acceptable keywords are midnight, noon, and now. Minutes are optional.
date	Specifies the first three or more letters of a month, a day of the week, or the keywords today or tomorrow.

2. **At the `at` prompt, type the commands or scripts that you want to execute, one per line.**

   You may type more than one command by pressing Return at the end of each line.

3. **Press Control-D to exit the `at` utility and save the `at` job.**

   Your `at` job is assigned a queue number, which is also the job's file name. This number is displayed when you exit the `at` utility.

Example 36 Creating an `at` Job

The following example shows the `at` job that user `jones` created to remove her backup files at 7:30 p.m. She used the `-m` option so that she would receive an email message after her job completed.

```
$ at -m 1930
at> rm /home/jones/*.backup
at> Press Control-D
job 897355800.a at Thu Jul  12 19:30:00 2004
```

She received a email message which confirmed the execution of her `at` job.

```
Your “at” job “rm /home/jones/*.backup”
completed.
```

The following example shows how `jones` scheduled a large `at` job for 4:00 a.m. Saturday morning. The job output was directed to a file named `big.file`.

```
$ at 4 am Saturday
at> sort -r /usr/dict/words > /export/home/jones/big.file
```

Displaying the `at` Queue

To check your jobs that are waiting in the `at` queue, use the `atq` command.

```
$ atq
```

This command displays status information about the `at` jobs that you have created.

Verifying an `at` Job

To verify that you have created an `at` job, use the `atq` command. In the following example, the `atq` command confirms that `at` jobs that belong to `jones` have been submitted to the queue.

```
$ atq
Rank   Execution Date     Owner     Job         Queue   Job Name
```

```
1st   Jul 12, 2004 19:30   jones  897355800.a     a     stdin
2nd   Jul 14, 2004 23:45   jones  897543900.a     a     stdin
3rd   Jul 17, 2004 04:00   jones  897732000.a     a     stdin
```

Displaying `at` Jobs

To display information about the execution times of your `at` jobs, use the `at -l` command.

```
$ at -l [job-id]
```

where `-l` *job-id* is the optional identification number of a specific job whose status you want to display. Without an ID, the command displays the status of all jobs submitted by a user.

EXAMPLE 37 Displaying `at` Jobs

The following example shows sample output from the `at -l` command, which provides information about the status of all jobs submitted by a user.

```
$ at -l
897543900.a Sat Jul 14 23:45:00 2004
897355800.a Thu Jul 12 19:30:00 2004
897732000.a Tue Jul 17 04:00:00 2004
```

The following example shows sample output that is displayed when a single job is specified with the `at -l` command.

```
$ at -l 897732000.a
897732000.a  Tue Jul 17 04:00:00 2004
```

▼ How to Remove `at` Jobs

Before You Begin Assume the `root` role to remove an `at` job that belongs to `root` or another user. See “Using Your Assigned Administrative Rights” in *Securing Users and Processes in Oracle Solaris 11.3*.

You do not need to assume the `root` role to remove your own `at` job.

1. **Remove the `at` job from the queue before the job is executed.**

   ```
   # at -r [job-id]
   ```

   where the `-r` *job-id* option specifies the identification number of the job you want to remove.

2. **Verify that the `at` job is removed by using the `at -l` (or the `atq`) command.**
   The `at -l` command displays the jobs remaining in the `at` queue. The job whose identification number you specified should not appear.

```
$ at -l [job-id]
```

Example 38 Removing at Jobs

In the following example, a user wants to remove an at job that was scheduled to execute at 4 a.m. on July 17th. First, the user displays the at queue to locate the job identification number. Next, the user removes this job from the at queue. Finally, the user verifies that this job has been removed from the queue.

```
$ at -l
897543900.a Sat Jul 14 23:45:00 2003
897355800.a Thu Jul 12 19:30:00 2003
897732000.a Tue Jul 17 04:00:00 2003
$ at -r 897732000.a
$ at -l 897732000.a
at: 858142000.a: No such file or directory
```

Controlling Access to the at Command

You can set up a file to control access to the at command, permitting only specified users to create, remove, or display queue information about their at jobs. The file that controls access to the at command, /etc/cron.d/at.deny, consists of a list of user names, one user name per line. The users who are listed in this file cannot access at commands.

The at.deny file, which is created during Oracle Solaris software installation, contains the following user names:

```
daemon
bin
smtp
nuucp
listen
nobody
noaccess
```

With superuser privileges, you can edit the at.deny file to add other user names whose at command access you want to restrict.

Denying Access to the at Command

As root, edit the /etc/cron.d/at.deny file to add the names of users, one user name per line, that you want to prevent from using the at commands.

```
daemon
bin
smtp
nuucp
listen
nobody
noaccess
username1
username2
username3
.
.
.
```

EXAMPLE 39 Denying at Access

The following example shows an at.deny file that has been edited so that the users smith and jones cannot access the at command.

```
$ cat at.deny
daemon
bin
smtp
nuucp
listen
nobody
noaccess
jones
smith
```

Verifying That the at Command Access is Denied

To verify that a username was added correctly to the /etc/cron.d/at.deny file, use the at -l command while logged in as the user. For example, if the logged-in user smith cannot access the at command, the following message is displayed:

```
# su smith
Password:
# at -l
at: you are not authorized to use at.  Sorry.
```

Likewise, if the user tries to submit an at job, the following message is displayed:

```
# at 2:30pm
at: you are not authorized to use at.  Sorry.
```

This message confirms that the user is listed in the at.deny file.

If at command access is allowed, then the at -l command returns nothing.

♦ ♦ ♦ CHAPTER 5

Managing the System Console, Terminal Devices, and Power Services

This chapter describes how to manage the system console and locally connected terminal devices through the `ttymon` program and system power services.

This chapter covers the following topics:

Managing System Console and Locally Connected Terminal Devices

The system console is a terminal that has special attributes and is used for certain purposes. For example, kernel messages that are meant for an administrator are sent to the console and not other terminals.

A terminal is a means of interacting with Oracle Solaris. Your system's bitmapped graphics display is not the same as an alphanumeric terminal. An alphanumeric terminal connects to a serial port and displays only text. You do not have to perform any special steps to administer the graphics display.

A terminal could also be associated with the physical monitor and keyboard layout of a computer. What sets the graphical terminal apart is that it must be associated with the graphics card and monitor of a computer. So, instead of transmitting characters out of a serial port, the characters are drawn onto the memory of the graphics card that is in the computer.

SMF Services That Manage the System Console and Locally Connected Terminal Devices

The system console and locally connected terminal devices are represented as instances of the SMF service, `svc:/system/console`. This service defines most of the behavior, with each instance having specific overrides to the settings that are inherited from the service. The `ttymon` program is used to offer login services for these terminals. Each terminal uses a separate instance of the `ttymon` program. Command-line arguments that are passed by the service to the `ttymon` program govern its behavior.

The service instances that are supplied with the system are as follows:

- `svc:/system/console-login:default`

  The default instance always represents that the `ttymon` program offer a login to the system hardware console.

- `svc:/system/console-login:{vt2, vt3, vt4, vt5, vt6}`

  Additional service instances are provided for the system's virtual consoles. If virtual consoles are not available, these services are automatically disabled. For more information, see the `vtdaemon(1M)` man page.

- `svc:/system/console-login:{terma, termb}`

  The `svc:/system/console-login:terma` and `svc:/system/console-login:termb` services are provided as a convenience. These services can assist you in setting up login services for additional `/dev/term/a` and `/dev/term/b` ports. These services are disabled by default.

You can define additional service instances as part of the `svc:system/console-login` service. For example, if you have a `/dev/term/f` port that you need to support, you could instantiate `svc:/system/console-login:termf` and configure it appropriately.

▼ How to Set Up Login Services on Auxiliary Terminals

For terminals that are connected to `/dev/term/a` or `/dev/term/b` serial ports on a system, predefined services are provided.

1. **Assume the `root` role.**

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Enable the service instance.**

   For example, to enable login services for `/dev/term/a`:

   ```
   # svcadm enable svc:/system/console-login:terma
   ```

3. **Check that the service is online.**

   ```
   # svcs svc:/system/console-login:terma
   ```

   The output should show that the service is online. If the service is in maintenance mode, consult the service's log file for further details.

▼ How to Set the Baud Rate Speed on the Console

Support for console speeds on x86 based systems are dependent on the specific platform.

The following are supported console speeds for SPARC based systems:

- `9600 bps`
- `19200 bps`
- `38400 bps`

1. **Become an administrator.**

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Use the `eeprom` command to set a baud rate speed that is appropriate for your system type.**

   ```
   # eeprom ttya-mode=baud-rate,8,n,1,-
   ```

   For example, to change the baud rate on an x86 based system's console to `38400`, type:

   ```
   # eeprom ttya-mode=38400,8,n,1,-
   ```

3. **Change the console line in the `/etc/ttydefs` file as follows:**

   ```
   console baud-rate hupcl opost onlcr:baud-rate::console
   ```

4. **Make the following additional changes for your system type.**

   Note that these changes are platform-dependent.

   - **On SPARC based systems:** Change the baud rate speed in the version of the `options.conf` file that is in the `/etc/driver/drv` directory. For example:

     To change the baud rate to `9600`:

     ```
     # 9600            :bd:
     ttymodes="2502:1805:bd:8a3b:3:1c:7f:15:4:0:0:0:11:13:1a:19:12:f:17:16";
     ```

     To change the baud rate speed to `19200`.

     ```
     # 19200           :be:
     ttymodes="2502:1805:be:8a3b:3:1c:7f:15:4:0:0:0:11:13:1a:19:12:f:17:16";
     ```

     To change the baud rate speed to `38400`:

     ```
     # 38400           :bf:
     ttymodes="2502:1805:bf:8a3b:3:1c:7f:15:4:0:0:0:11:13:1a:19:12:f:17:16";
     ```

   - **On x86 based systems:** Change the console speed if the BIOS serial redirection is enabled.

Managing System Power Services

In the Oracle Solaris 11 operating system, power management configuration has moved into an SMF configuration repository. The new `poweradm` command is used to manage system power management properties directly rather than using a combination of power-related command, daemon, and configuration file. These changes are part of a wider set of changes to modernize the power management framework in the Oracle Solaris 11 operating system.

The following power management features are no longer available:

- `/etc/power.conf`
- `pmconfig` and `powerd`
- Device power management

The following properties describe power management components:

- `administrative-authority` – Defines the source of administrative control for Oracle Solaris power management. This property can be set to `none`, `platform` (default value), or `smf`.

  When set to `platform`, the values of `time-to-full-capacity` and `time-to-minimum-responsiveness` are taken from the platform's power management commands.

  When set to `smf`, the values of `time-to-full-capacity` and `time-to-minimum-responsiveness` are taken from SMF.

  If you attempt to set `time-to-full-capacity` or `time-to-minimum-responsiveness` from either a platform command or an SMF service property when in the opposite venue, the value is ignored.

  When `administrative-authority` is set to `none`, power management within the Oracle Solaris instance is turned off.
- `time-to-full-capacity` – Defines the maximum time (in microseconds) the system is allowed to reach its full capacity, from any lower-capacity or less-responsive state, while the system is in active state. The maximum time includes the time while it has been using any or all of the PM features falling within this boundary.

  By default, this value is taken from the platform, `i86pc` for example, because the default setting for `administrative-authority` is set to platform.

  Alternatively, if `administrative-authority` is set to `smf`, this value is taken from the definition provided by the SMF power service. At installation time, this value is undefined. If you choose to modify this property, a value appropriate to the needs of the system's workload or applications should be considered.
- `time-to-minimum-responsiveness` – Defines how long the system is allowed to return to its active state in milliseconds. This parameter provides the minimum capacity required to meet the `time-to-full-capacity` constraint. Because the default setting for `administrative-authority` is set to platform by default, this parameter value is taken from the platform, `i86pc` for example..

Alternatively, if `administrative-authority` is set to `smf`, this value is taken from the definition provided by the SMF power service . At installation time, this value is undefined. If you choose to modify this property, use a value appropriate to the needs of the system's workload or applications.

Moderate values, seconds for example, allow hardware components or subsystems on the platform to be placed in slower-response inactive states. Larger values, 30 seconds to minutes, for example, allow for whole system suspension, using techniques such as suspend-to-RAM.

- `suspend-enable` – By default, no system running Oracle Solaris is permitted to attempt a suspend operation. Setting this property to true permits a suspend operation to be attempted. The value of the `administrative-authority` has no effect upon this property.
- `platform-disabled` – When `platform-disabled` is set to `true`, the platform has disabled power management. When set to `false`, the default value, power management is controlled by the value of the above properties.

To display a brief summary of power management status, issue the following command:

```
$ /usr/sbin/poweradm show
Power management is enabled with the hardware platform as the authority:
time-to-full-capacity set to 250 microseconds
time-to-minimum-responsiveness set to 0 milliseconds
```

To display power management properties, issue the following command:

```
$ /usr/sbin/poweradm list
active_config/time-to-full-capacity          current=250, platform=250
active_config/time-to-minimum-responsiveness current=0, platform=0
active_control/administrative-authority      current=platform, smf=platform
suspend/suspend-enable                       current=false
platform-disabled                            current=false
```

In this output, the `active_control/administrative-authority` indicates the source of the configuration with two settings:

- `platform` – Configuration for power management comes from the platform. This is the default value.
- `smf` – Allows the other power management properties to be set using the `poweradm` command.

The `platform-disabled` property in the output indicates that the platform power management is enabled:

```
platform-disabled                            current=false
```

For more information, see the `poweradm`(1M) man page.

EXAMPLE 40 Enabling and Disabling Power Management

If you previously enabled S3-support in the `/etc/power.conf` file to suspend and resume your system, similar `poweradm` syntax is:

```
# poweradm set suspend-enable=true
```

The `suspend-enable` property is set to `false` by default.

Use the following syntax to disable power management:

```
# poweradm set administrative-authority=none
```

Disabling the following SMF power management service does not disable power management:

```
online         Sep_02   svc:/system/power:default
```

Use the following syntax to disable suspend and resume:

```
# poweradm set suspend-enable=false
```

EXAMPLE 41 Setting and Displaying Power Management Parameters

The following example shows how to set `time-to-full-capacity` to 300 microseconds and `time-to-minimum-responsiveness` to 500 milliseconds. Lastly, the Oracle Solaris instance is informed of the new values.

```
# poweradm set time-to-full-capacity=300
# poweradm set time-to-minimum-responsiveness=500
# poweradm set administrative-authority=smf
```

The following command shows the current `time-to-full-capacity` value.

```
# poweradm get time-to-full-capacity
300
```

The following command retrieves the `time-to-full-capacity` value set by the platform.

```
# poweradm get -a platform time-to-full-capacity
```

Note that this value will be the same as the current value only if `administrative-authority` is set to platform. For more information, see the above `administrative-authority` property description.

How to Recover from Power Service in Maintenance Mode

If `administrative-authority` is set to `smf` before both `time-to-full-capacity` and `time-to-minimum-responsiveness` have been set, the service will go into maintenance mode. See the task below to recover from this scenario.

1. **Become an administrator.**

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Set `administrative-authority` to `none`.**

   ```
   # poweradm set administrative-authority=none
   ```

3. **Set both `time-to-full-capacity` and `time-to-minimum-responsiveness` to their desired values.**

   ```
   # poweradm set time-to-full-capacity=value
   # poweradm set time-to-minimum-responsiveness=value
   ```

4. **Clear the service.**

   ```
   # svcadm clear power
   ```

5. **Set `administrative-authority` to `smf`.**

   ```
   # poweradm set administrative-authority=smf
   ```

Index

R

S