Managing System Information, Processes, and Performance in Oracle® Solaris 11.4

Part No: E60997
August 2021

ORACLE®

Managing System Information, Processes, and Performance in Oracle Solaris 11.4

Part No: E60997

Référence: E60997

Accessibilité de la documentation

Pour plus d'informations sur l'engagement d'Oracle pour l'accessibilité de 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 11

1 Managing System Information 13
Displaying System Information 13
Commands That Display System Information 13
Chip Multithreading Features 23
Changing System Information 24
▼ How to Manually Set the Date and Time of a System 24
▼ How to Set Up a Message-Of-The-Day 25
▼ How to Change the Identity of a System 25

2 Managing System Processes 27
System Processes That Do Not Require Administration 27
Tasks and Commands for Managing System Processes 28
Commands for Managing System Processes 29
Using the `ps` Command 29
Using the `/proc` File System and Commands 30
Managing Processes by Using Process Commands (`/proc`) 31
▼ How to List Processes 32
▼ How to Display Information About Processes 33
▼ How to Control Processes 34
Terminating a Process 35
▼ How to Terminate a Process With the `pkill` Command 35
▼ How to Terminate a Process With the `kill` Command 36
Debugging a Process 37
Displaying and Managing Process Class Information 38
Displaying Process Priority Information 38
Displaying the Global Priority of a Process 39

Changing the Scheduling Priority of Processes 40
▼ How to Designate a Process Priority 40
▼ How to Change Scheduling Parameters of a Timesharing Process 41
▼ How to Change the Class of a Process 42
Changing the Priority of a Timesharing Process 43
Changing the Priority of a Process 43
Troubleshooting Problems With System Processes 44

3 Monitoring System Performance 45
System Resources That Affect System Performance 45
Managing Performance Using Oracle Enterprise Manager Ops Center 46
About Processes and System Activities 46
System Activities That Are Monitored 47
Displaying System Performance Information 49
Displaying Virtual Memory Statistics 49
Displaying System Event Information 51
Displaying Swapping Statistics 52
Displaying Interrupts Per Device 52
Displaying Disk Utilization 53
Displaying Disk Space Statistics 55
Displaying Data Analytics Accelerator Statistics 57
Displaying DAX Information 59
Monitoring System Activities 60
Monitoring System Activities With the `sar` Command 60
Collecting System Activity Data Automatically 76

4 Scheduling System Tasks 81
Overview of Scheduled System Tasks 81
Scheduling a Periodic or Scheduled Task With SMF 82
Scheduling a Routine System Task With the `crontab` Command 83
Scheduling a Single System Task With `at` 83
Examples of Repetitive System Tasks 84
Scheduling Repetitive System Tasks With SMF 85
How SMF Handles Scheduling 85
Scheduling Method and Time Values for SMF 85
Sample SMF Manifest 86

Scheduling a Repetitive System Task With `crontab` 87
About the `crontab` File 87
How the `cron` Daemon Handles Scheduling 88
Syntax of `crontab` File Entries 89
Creating and Editing `crontab` Files 90
Listing `crontab` Files and Entries 91
Removing `crontab` Files 93
Controlling Access to the `crontab` Command 94
Scheduling a Single System Task With the `at` Command 97
Submitting an `at` Job 97
Creating an `at` Job 98
Displaying the `at` Queue 99
Verifying an `at` Job 99
Displaying `at` Jobs 100
▼ How to Remove `at` Jobs 100
Controlling Access to the `at` Command 101

5 Managing the System Console, Terminal Devices, and Power Services 105
SMF Services That Manage the System Console and Terminal Devices 105
▼ How to Set Up Login Services on Auxiliary Terminals 106
▼ How to Set the Baud Rate Speed on the Console 106
Managing System Power Services 108
▼ How to Recover from Power Service in Maintenance Mode 111

Index 113

Examples

EXAMPLE 1 SPARC: Displaying Default and Custom Device Properties 16
EXAMPLE 2 x86: Displaying Default and Custom Device Properties 18
EXAMPLE 3 x86: Displaying System Configuration Information 19
EXAMPLE 4 SPARC: Displaying System Diagnostic Information 20
EXAMPLE 5 x86: Displaying System Diagnostic Information 22
EXAMPLE 6 Manually Setting the Date and Time of a System 24
EXAMPLE 7 Listing Processes 32
EXAMPLE 8 Displaying Information About Processes 34
EXAMPLE 9 Debugging a Process With `pargs` and `pgrep` 37
EXAMPLE 10 Designating a Process Priority 41
EXAMPLE 11 Changing the Scheduling Parameters of a Timesharing Process 41
EXAMPLE 12 Changing the Class of a Process 42
EXAMPLE 13 Displaying VM Statistics at 5-Second Intervals 50
EXAMPLE 14 Displaying File System Information 55
EXAMPLE 15 Displaying File System Information by Using the `df` Command With No Options 56
EXAMPLE 16 Using the `daxstat` Command to Display Per-DAX Statistics 58
EXAMPLE 17 Using the `daxstat` Command to Display Per-CPU Statistics 58
EXAMPLE 18 Using the `daxstat` Command to Display Per-Queue Statistics 59
EXAMPLE 19 Displaying DAX Information 60
EXAMPLE 20 Reporting File Access 61
EXAMPLE 21 Reporting Buffer Activity 62
EXAMPLE 22 Reporting System Call Statistics 63
EXAMPLE 23 Reporting Disk Activity 64
EXAMPLE 24 Reporting Page-Out and Memory 66
EXAMPLE 25 Reporting Kernel Memory Allocation 68
EXAMPLE 26 Reporting Page-In Activity 70
EXAMPLE 27 Reporting Queue Activity 71

EXAMPLE 28 Reporting Unused Memory 71
EXAMPLE 29 Reporting CPU Utilization 72
EXAMPLE 30 Reporting System Table Status 74
EXAMPLE 31 Reporting Swap Activity 74
EXAMPLE 32 Reporting Terminal Activity 75
EXAMPLE 33 Creating a `crontab` File 91
EXAMPLE 34 Displaying a `crontab` File 92
EXAMPLE 35 Displaying the Default `root` `crontab` file. 92
EXAMPLE 36 Displaying the `crontab` File of Another User 93
EXAMPLE 37 Removing a `crontab` File 94
EXAMPLE 38 Limiting `crontab` Command Access to Specified Users 96
EXAMPLE 39 Creating an `at` Job 99
EXAMPLE 40 Displaying `at` Jobs 100
EXAMPLE 41 Removing `at` Jobs 101
EXAMPLE 42 Denying `at` Access 102
EXAMPLE 43 Enabling and Disabling Power Management 110
EXAMPLE 44 Setting and Displaying Power Management Parameters 110

Using This Documentation

- **Overview** – Describes how to manage system information, processes, and monitoring performance.
- **Audience** – System administrators using the Oracle Solaris 11.4 release.
- **Required knowledge** – Experience in administering UNIX systems.

Product Documentation Library

Documentation and resources for this product and related products are available at https://docs.oracle.com/cd/E37838_01/.

Feedback

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

♦ ♦ ♦ CHAPTER 1

Managing System Information

This chapter describes how to display and change basic system information:

- "Displaying System Information" on page 13
- "Changing System Information" on page 24

For information about resource management where you allocate, monitor, and control system resources in a flexible way, see *Administering Resource Management in Oracle Solaris 11.4*.

Displaying System Information

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

Commands That Display System Information

The following is an alphabetical list of system information commands shown in man page form, with short descriptions and examples or links to examples.

`date`(1) — Displays and sets date and time.

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

`hostid`(1) — Displays Host ID number.

```
$ hostid
84f9ae0a
```

`isainfo`(1) — Identifies various attributes of the instruction set architectures supported on the currently running system. Examples of questions that `isainfo` can

answer include whether 64-bit applications are supported, or whether the running kernel uses 32-bit or 64-bit device drivers.

See “Displaying Architecture Type” on page 14 and “Displaying Processor Type” on page 15.

`prtconf`(8) — Displays system configuration information, installed memory, device properties, and product name.

See “Displaying Product Name” on page 16, “Displaying Installed Memory” on page 16 and “Displaying Default and Customized Property Values for a Device” on page 16.

`prtdiag`(8) — Displays system configuration and diagnostic information, including any failed field replacement units (FRUs). See “Displaying System Diagnostic Information” on page 20.

`psrinfo`(8) — Displays processor information. See “Chip Multithreading Features” on page 23

```
$ psrinfo
0       on-line   since 08/22/2019 23:27:21
1       on-line   since 08/22/2019 23:27:22
...
```

`uname`(1) — Displays operating system name, release, version, node name, hardware name, processor type, and virtualization status. The virtualization status appears in the `uname -a` command output starting with the Oracle Solaris 11.4 SRU 36 release.

```
$ uname -a
SunOS example-server 5.11 11.4.36.15.0 sun4v sparc sun4v non-
virtualized
```

`cat /etc/release` — Displays release information.

```
$ cat /etc/release
                           Oracle Solaris 11.4 SPARC
  Copyright (c) 1983, 2018, Oracle and/or its affiliates.  All
 rights reserved.
                           Assembled 10 October 2018
```

Displaying Architecture Type

You can 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. An x86

based system would display `amd64 i386`, while a SPARC based system would display `sparcv9 sparc`.

Command options show more specific information. 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
        crc32c cbcond pause mont mpmul sha512 sha256 sha1 md5 camellia kasumi
        des aes ima hpc vis3 fmaf asi_blk_init vis2 vis popc
32-bit sparc applications
        crc32c cbcond pause mont mpmul sha512 sha256 sha1 md5 camellia kasumi
        des aes ima hpc vis3 fmaf asi_blk_init vis2 vis popc v8plus div32 mul32
```

On an x86 based system, the same command syntax would generate the following output:

```
$ isainfo -v
64-bit amd64 applications
        avx xsave pclmulqdq aes sse4_2 sse4_1 ssse3 popcnt tscp ahf
        cx16 sse3 sse2 sse fxsr mmx cmov amd_sysc cx8 tsc fpu
32-bit i386 applications
        avx xsave pclmulqdq aes sse4_2 sse4_1 ssse3 popcnt tscp ahf
        cx16 sse3 sse2 sse fxsr mmx cmov sep cx8 tsc fpu
```

Displaying Processor Type

You can use the `isainfo -x` command to display information about the processor of a system. The following sample output is from an x86 based system:

```
$ isainfo -x
amd64: avx xsave pclmulqdq aes sse4_2 sse4_1 ssse3 popcnt tscp ahf cx16 sse3
       sse2 sse fxsr mmx cmov amd_sysc cx8 tsc fpu
i386: avx xsave pclmulqdq aes sse4_2 sse4_1 ssse3 popcnt tscp ahf cx16 sse3
       sse2 sse fxsr mmx cmov sep cx8 tsc fpu
```

The following sample output is from a SPARC based system:

```
$ isainfo -x
sparcv9: ima fmaf vis2 vis popc
sparc: ima fmaf vis2 vis popc v8plus div32 mul32
```

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

Displaying Product Name

You can display the product name of your system using the `prtconf` command with the `-b` option. The following example shows verbose output on a SPARC based system:

```
$ /usr/sbin/prtconf -vb
name:  ORCL,SPARC-S7-2L
banner-name:  SPARC S7-2L
compatible: 'sun4v'
idprom:  01860010.e0bb7ce6.00000000.bb7ce677.00000000.00000000.00000000.00000000
openprom model:  SUNW,4.43.2
openprom version: 'OBP 4.43.2 2019/01/25 07:51'
```

Displaying Installed Memory

You can display the amount of memory that is installed on your system using the `prtconf` command piped to `grep Memory`. For example:

```
$ prtconf | grep Memory
Memory size: 260352 Megabytes
```

Displaying Default and Customized Property Values for a Device

You can use the `prtconf -u` command to display the default and customized property values for devices.

EXAMPLE 1 SPARC: Displaying Default and Custom Device Properties

This example shows the default and custom properties for the `e1000g.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.

```
$ /usr/sbin/prtconf -u
System Configuration:  Oracle Corporation  sun4v
Memory size: 260352 Megabytes
System Peripherals (Software Nodes):

ORCL,SPARC-S7-2L
    scsi_vhci, instance #0
```

```
        disk, instance #50
        disk, instance #51
...
        disk, instance #1 (driver not attached)
        disk, instance #23 (driver not attached)
...
    packages (driver not attached)
    chosen (driver not attached)
    openprom (driver not attached)
    options, instance #0
    aliases (driver not attached)
    memory (driver not attached)
    virtual-memory (driver not attached)
    iscsi-hba (driver not attached)
    reboot-memory (driver not attached)
    cpu (driver not attached)
...
    virtual-devices, instance #0
        flashprom (driver not attached)
        random-number-generator, instance #0
        dax, instance #0 (driver not attached)
        console, instance #0
        channel-devices, instance #0
            virtual-channel, instance #0
            virtual-channel, instance #3
            virtual-channel-client, instance #1
            virtual-channel-client, instance #2
            pciv-communication, instance #0
            virtual-domain-service, instance #0
...
        pci, instance #12
            pci, instance #16
                pci, instance #17
                pci, instance #18
                pci, instance #19
                    LSI,sas, instance #3
                        iport, instance #6
                            smp, instance #1 (driver not attached)
                            enclosure, instance #3 (driver not attached)
                        iport, instance #8
                            enclosure, instance #0 (driver not attached)
                            smp, instance #2 (driver not attached)
                pci, instance #20
                    LSI,sas, instance #0
                        iport, instance #9
                        iport, instance #1
    ramdisk-root, instance #0 (driver not attached)
    fcoe, instance #0
```

```
    iscsi, instance #0
    pseudo, instance #0
```

EXAMPLE 2 x86: Displaying Default and Custom Device Properties

```
$ /usr/sbin/prtconf -u
System Configuration:  Oracle Corporation  i86pc
Memory size: 131015 Megabytes
System Peripherals (Software Nodes):

i86pc
    scsi_vhci, instance #0
        disk, instance #5 (driver not attached)
        disk, instance #3
        disk, instance #4 (driver not attached)
...
    ioapics (driver not attached)
        ioapic, instance #0 (driver not attached)
        ioapic, instance #1 (driver not attached)
    pci, instance #1
        pci108e,484e (driver not attached)
        pci108e,484e, instance #0
        pci108e,484e (driver not attached)
...
    pci, instance #2
        pci8086,3c01 (driver not attached)
        pci8086,3c02, instance #5
            pci108e,484e, instance #0
            pci108e,484e, instance #1
...
    fw, instance #0
        sb, instance #1
            socket, instance #2
                cpu, instance #0
                cpu, instance #16
...
            socket, instance #3
                cpu, instance #8
                cpu, instance #24
...
    used-resources (driver not attached)
    fcoe, instance #0
    iscsi, instance #0
    agpgart, instance #0 (driver not attached)
    options, instance #0
    pseudo, instance #0
    vga_arbiter, instance #0 (driver not attached)
```

```
    xsvc, instance #0 (driver not attached)
    intel-iommu, instance #0
    intel-iommu, instance #1
```

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.

EXAMPLE 3 x86: Displaying System Configuration Information

This 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 the system. The output of this command displays `driver not attached` messages next to the device instances for which no device exists.

```
$ /usr/sbin/prtconf -v | more
System Configuration:  Oracle Corporation  i86pc
Memory size: 131015 Megabytes
System Peripherals (Software Nodes):

i86pc
    Device Hold:
        mod='genunix' id=1
            value='root nexus.'
        mod='devinfo' id=191
            value='Device snapshot.'
    Driver properties:
        name='fm-accchk-capable' type=boolean dev=none
        name='fm-dmachk-capable' type=boolean dev=none
        name='fm-ereport-capable' type=boolean dev=none
        name='fm-errcb-capable' type=boolean dev=none
        name='pm-hardware-state' type=string items=1 dev=none
            value='needs-suspend-resume'
...
        disk, instance #3
            Device Hold:
                mod='specfs' id=7
                    value='Device opened.'
            Driver properties:
                name='ddi-dpofua-supported' type=boolean dev=none
                name='ddi-failfast-supported' type=boolean dev=none
                name='ddi-kernel-ioctl' type=boolean dev=none
                name='fm-ereport-capable' type=boolean dev=none
                name='inquiry-rpm' type=int items=1 dev=none
                    value=00002724
                name='inquiry-serial-no' type=string items=1 dev=none
                    value='001205PNARLB        PVGNARLB'
```

```
...
        disk, instance #5 (driver not attached)
        disk, instance #4 (driver not attached)
...
    ioapics (driver not attached)
...
    used-resources (driver not attached)
    agpgart, instance #0 (driver not attached)
    vga_arbiter, instance #0 (driver not attached)
    xsvc, instance #0 (driver not attached)
...
    intel-iommu, instance #1
        Driver properties:
            name='fm-ereport-capable' type=boolean dev=none
...
```

For more information, see the `driver`(5), `driver.conf`(5), and `prtconf`(8) 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 11.4*.

Displaying System Diagnostic Information

You can 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, send this output to `syslogd`(8) only.

EXAMPLE 4 SPARC: Displaying System Diagnostic Information

This example shows the output for the `prtdiag -v` command on a SPARC based system.

```
$ /usr/sbin/prtdiag -v | more
System Configuration:  Oracle Corporation  sun4v SPARC S7-2L
Memory size: 260352 Megabytes

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


CPU ID Frequency Implementation         Status
```

```
------ --------- --------------------- -------
0      4267 MHz  SPARC-S7              on-line
1      4267 MHz  SPARC-S7              on-line
2      4267 MHz  SPARC-S7              on-line
3      4267 MHz  SPARC-S7              on-line
...
======================== Physical Memory Configuration ========================
Segment Table:
------------------------------------------------------------
Base               Segment  Interleave  Bank     Contains
Address            Size     Factor      Size     Modules
------------------------------------------------------------
0x0                128 GB   0           128 GB   /SYS/MB/CMP0/MCU0/CH0/D0
                                                 /SYS/MB/CMP0/MCU0/CH0/D1
...
0x400000000000     128 GB   0           128 GB   /SYS/MB/CMP1/MCU0/CH0/D0
                                                 /SYS/MB/CMP1/MCU0/CH0/D1
...
======================================== IO Devices
 ========================================
Slot +             Bus   Name +                              Model      Max Speed  Cur
 Speed
Status             Type  Path                                           /Width     /Width
-------------------------------------------------------------------------------------------------
/SYS/MB/XGBE       PCIE  network-pciex8086,1589                         --         --
                         /pci@300/pci@1/pci@0/pci@1/network@0
/SYS/MB/NET1       PCIE  network-pciex8086,1589                         --         --
                         /pci@300/pci@1/pci@0/pci@1/network@0,1
...
============================ Environmental Status ============================
Fan sensors:
------------------------------------------------------------
Location                          Sensor            Status
------------------------------------------------------------
SYS/MB/FM0/F0                     TACH              ok
SYS/PS1/F0                        TACH              ok

Temperature sensors:
------------------------------------------------------------
Location                          Sensor            Status
------------------------------------------------------------
SYS/MB/CMP0/MCU0/CH0/D0           T_AMB             ok
SYS/MB/CMP0/MCU0/CH0/D1           T_AMB             ok
============================ Environmental Status ============================
Fan sensors:
------------------------------------------------------------
Location                          Sensor            Status
------------------------------------------------------------
```

```
SYS/FANBD/F0                      TACH              ok
SYS/FANBD/F1                      TACH              ok

...
Current sensors:
...
Voltage sensors:
...
LEDs:
...
============================ FRU Status ============================
Location                          Name      Status
------------------------------------------------------
SYS                               DBP       enabled
SYS/DBP                           HDD0      enabled
...
============================ FW Version ============================
Version
------------------------------------------------------------
Sun System Firmware 9.9.2.a 2019/02/06 13:17

====================== System PROM revisions =======================
Version
------------------------------------------------------------
OBP 4.43.2 2019/01/25 07:51

Chassis Serial Number
---------------------
AK00312345
```

EXAMPLE 5 x86: Displaying System Diagnostic Information

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

```
$ /usr/sbin/prtdiag -l
System Configuration: Oracle Corporation SUN FIRE X4270 M3
BIOS Configuration: American Megatrends Inc. 18110500 12/22/2014
BMC Configuration: IPMI 2.0 (KCS: Keyboard Controller Style)

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

Version                          Location Tag
-------------------------------- --------------------------
Intel(R) Xeon(R) CPU E5-2690 0 @ 2.90GHz P0
Intel(R) Xeon(R) CPU E5-2690 0 @ 2.90GHz P1

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

```
Type        Status Set Device Locator      Bank Locator
----------- ------ --- ------------------- ----------------
DDR3        in use 0   D7                  /SYS/MB/P0
unknown     empty  0   D6                  /SYS/MB/P0
DDR3        in use 0   D5                  /SYS/MB/P0
unknown     empty  0   D1                  /SYS/MB/P1
DDR3        in use 0   D2                  /SYS/MB/P1

==== On-Board Devices =====================================
 Onboard Video
 X540 10GbE Controller
 Intel C600 Series - SAS
 Intel C600 Series - SATA

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

ID  Status    Type             Description
--- --------- ---------------- ----------------------------
1   available PCI Express Gen3 /SYS/MB/PCIE1
2   in use    PCI Express Gen3 /SYS/MB/PCIE2
3   available PCI Express Gen3 /SYS/MB/PCIE3
4   in use    PCI Express Gen3 /SYS/MB/PCIE4
5   available PCI Express Gen3 /SYS/MB/PCIE5
6   in use    PCI Express Gen3 /SYS/MB/PCIE6
```

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 processors of the system and their associated socket, core, and CPU IDs.

Using the `psrinfo -pv` command lists the physical processors in the system and the virtual processors that are associated with each physical processor. The following example applies the command to an x86 based system:

```
$ /usr/sbin/psrinfo -pv
The physical processor has 8 cores and 16 virtual processors (0-7,16-23)
  The core has 2 virtual processors (0,16)
  The core has 2 virtual processors (1,17)
  The core has 2 virtual processors (2,18)
  The core has 2 virtual processors (3,19)
  The core has 2 virtual processors (4,20)
```

```
  The core has 2 virtual processors (5,21)
  The core has 2 virtual processors (6,22)
  The core has 2 virtual processors (7,23)
    x86 (GenuineIntel 206D7 family 6 model 45 step 7 clock 2893 MHz)
      Intel(r) Xeon(r) CPU E5-2690 0 @ 2.90GHz
The physical processor has 8 cores and 16 virtual processors (8-15,24-31)
 ...
```

Changing System Information

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

▼ How to Manually Set the Date and Time of a System

1. **Become an administrator.**

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

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

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

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

Example 6 Manually Setting the Date and Time of a System

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

```
# 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. **Become an administrator who is assigned the `Administrator Message Edit` rights profile.**

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

2. **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 the Identity of a System

1. **Become an administrator.**

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

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(8)`, and `nodename(5)` 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" on page 27
- "Tasks and Commands for Managing System Processes" on page 28
- "Displaying and Managing Process Class Information" on page 38
- "Troubleshooting Problems With System Processes" on page 44

System Processes That Do Not Require Administration

The following Oracle Solaris 11 system processes do not require any administration.

`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
`vmtasks`	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

Tasks and Commands for Managing System Processes

This section summarizes the various tasks for managing system processes. It also lists commands for managing these processes. The various tasks you can perform on system processes include:

- "How to List Processes" on page 32
- "How to Display Information About Processes" on page 33
- "How to Control Processes" on page 34
- "How to Terminate a Process With the `pkill` Command" on page 35
- "How to Terminate a Process With the `kill` Command" on page 36

The following list describes system management commands. The commands are shown in man page form.

`dispadmin`(8)	Assigns processes to a priority class and manages process priorities. See "dispadmin Command" in *Oracle Solaris 11.4 Programming Interfaces Guide*.
`nice`(1)	Changes the priority of a timesharing process. See "Changing the Priority of a Process" on page 43.
`pargs`(1) and `preap`(1)	Assist with process debugging. See "Debugging a Process" on page 37.
`pgrep`(1), `ps`(1), and `prstat`(8)	Check the status of active processes on a system, and also display detailed information about the processes. See Example 7, "Listing Processes," on page 32.
`pkill`(1)	With `pgrep`(1), finds or signals processes by name or other attribute, and terminates the process. Instead of having the process ID printed, each matching process is signaled similar to the `kill`(1) command. See "How to Terminate a Process With the `pkill` Command" on page 35.
`priocntl`(1)	Lists default process scheduling policies. See "Changing the Scheduling Priority of Processes" on page 40.

psrset(8)	Binds specific process groups to a group of processors rather than to just a single processor.

Commands for Managing System Processes

This section describes how to use the commands listed in “Tasks and Commands for Managing System Processes” on page 28.

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.

The ps command includes options to report the following information:

Command name and arguments	Memory used	Process ID
CPU time used	Parent process ID	Scheduling class
Current status of the process	Priority	User ID

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.

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.
CMD	The command that generated the process.
FMRI	The SMF FMRI that corresponds to the contract ID for the process.
NI	The nice number of the process, which contributes to its scheduling priority. Making a process "nicer" means lowering its priority.
PID	The process ID.

`PPID`	The parent process ID.
`PRI`	The scheduling priority of the kernel thread. Higher numbers indicate a higher priority.
`STIME`	The starting time of the process in hours, minutes, and seconds.
`SZ`	The virtual address size of the process.
`TIME`	The total amount of CPU time used by the process since it began.
`TTY`	The terminal from which the process, or its parent, was started. A question mark indicates that there is no controlling terminal.
`UID`	The effective user ID of the owner of the process.
`ZONE`	The zone in which the process is running.

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 `/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.

The following process commands display details about a process in the `/proc` directory. The `prun` and `pstop` commands start and stop a process:

`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
`prun`	Starts each process
`psig`	Lists the signal actions and handlers of 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

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 perform the following tasks:

- 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 terminate or resume them

The `pldd` and `pstack` commands stop target processes to inspect and extract information.

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

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. For a list of some of the process commands, see “Using the `/proc` File System and Commands” on page 30.

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 examples, see “Terminating a Process” on page 35.

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' 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`(5) 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 7 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
    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. For a description of all the process commands, see "Using the /proc File System and Commands" on page 30.

PID — The process ID.

Example 8 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. **If you do not own the process, assume the `root` role.**

   ```
   $ su root
   root password: password
   #
   ```

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

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

pcommand — The process command that you want to run. For a description of all the process commands, see "Using the /proc File System and Commands" on page 30.

PID — The process ID.

4. **Verify the process status.**

```
# ps -ef | grep PID
```

Terminating a Process

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 With the pkill Command

1. **To terminate a process that you do not own, assume the root role.**

2. **Obtain the process ID for the process that you want to terminate.**

```
# pgrep process
```

For example:

```
# pgrep firefox
587
566
```

The process ID is displayed in the output.

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, do not use the -9 signal to kill certain processes such as a database process or an LDAP server process because data might be lost.

PID — The process ID of the process to terminate.

Tip - When trying to terminate a process, first try using the `pkill` 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 not be listed.

How to Terminate a Process With the `kill` Command

1. **To terminate a process that you do not own, 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, do not use the -9 signal to kill certain processes such as a database process or an LDAP server process because data might be lost.

PID The process ID of the process that you want to terminate.

Tip - When trying to terminate a process, first try using the `kill` command by itself without 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 -ef | grep PID
```

The process you terminated should not be listed.

Debugging a Process

The `pargs` and `preap` commands 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 `pargs` and `preap` to examine any process that you have the privileges to examine. With the appropriate rights, you can examine any process.

For more information, see the `pargs(1)`, `preap(1)`, and `proc(1)` man pages.

EXAMPLE 9 Debugging a Process With `pargs` and `pgrep`

The `pargs` command is unable to display all the arguments that are passed to a process with the `ps` command. 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`)
- Interactive (`IA`)
- Real-time (`RT`)
- System (`SYS`)
- 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 Priority Information

You can use the `priocntl -l` command to display process scheduling classes and priority ranges. 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

You can use the `ps -ecl` command to display the global priority of a process.

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
```

. . .

Changing the Scheduling Priority of Processes

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`(8) 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

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

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

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*	Specifies the maximum amount you can raise or lower your priority, when you use the `-p`option with this option.
`-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 to execute.

3. **Verify the process status.**

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

Example 10 Designating a Process Priority

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

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

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

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*	Specifies the maximum amount you can raise or lower the priority, when you use the `-p` option with this option.
`-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 11 Changing the Scheduling Parameters of a Timesharing Process

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

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.

   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

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 12 Changing the Class of a Process

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

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 scheduling class of processes, its CPU time consumption, 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

As a user, you can only lower the priority of a process. An administrator can 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 troubleshoot are as follows:

- Several identical jobs 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.
- A process has accumulated a large amount of CPU time.

  You can identify this problem by checking the `TIME` field in the `ps` output. This value can indicate that the process is in an endless loop.
- A process 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 running 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.
- A runaway process 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 your computer systems. 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:

- "System Resources That Affect System Performance" on page 45
- "Managing Performance Using Oracle Enterprise Manager Ops Center" on page 46
- "About Processes and System Activities" on page 46
- "System Activities That Are Monitored" on page 47
- "Displaying System Performance Information" on page 49
- "Monitoring System Activities" on page 60

Information about monitoring system performance is discussed in several guides:

- Chapter 2, "Managing System Processes"
- *Oracle Solaris 11.4 Tunable Parameters Reference Manual*
- Chapter 2, "About Projects and Tasks" in *Administering Resource Management in Oracle Solaris 11.4*
- Chapter 8, "About Fair Share Scheduler" in *Administering Resource Management in Oracle Solaris 11.4*
- *Using Oracle Solaris 11.4 StatsStore and System Web Interface*

System Resources That Affect System Performance

The performance of a computer system depends on how the system uses and allocates its resources. Monitor the performance of your system regularly so that you know how it behaves under normal conditions.

The following system resources affect performance:

- **Central processing unit (CPU)** – The CPU processes instructions by fetching instructions from memory and runs them
- **Input/output (I/O) devices** – The CPU processes instructions by fetching instructions from memory and running them.
- **Memory** – Physical (or main) memory is the amount of random access memory (RAM) on the system.

Managing Performance Using Oracle Enterprise Manager Ops Center

If you need to monitor, analyze, and improve performance of 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 Oracle 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 Oracle Enterprise Manager Ops Center 12c Release 2 library.

About Processes and System Activities

The following terms are related to processes:

- **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 the address space of a user. Application threads can be multiplexed on top of LWPs.

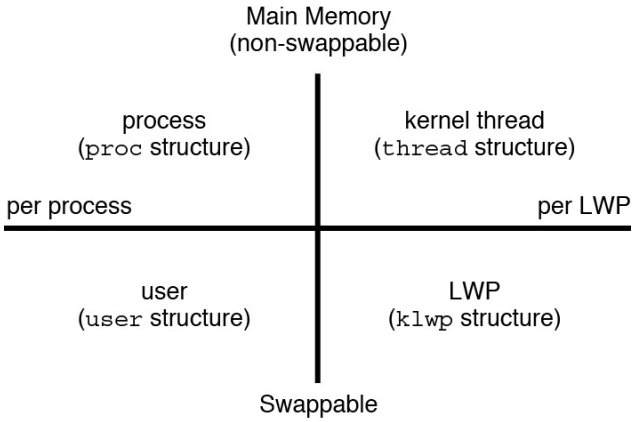
A process can consist of multiple LWPs and multiple application threads. The kernel schedules a kernel-thread structure, which is the scheduling entity in the Oracle Solaris environment.

Various process structures are as follows:

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

System Activities That Are Monitored

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

The following system activities are tracked:

- 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)

The Oracle Solaris software provides several commands to help you track how your system is performing. The following lists selected commands in man page format with short descriptions.

`cpustat`(8) and `cputrack`(1)
: Monitors performance of a system or a process using CPU performance counters.

  See "cpc Probe and Existing Tools" in *Oracle Solaris 11.4 DTrace (Dynamic Tracing) Guide*.

`kstat2`(8) and `mpstat`(8)
: 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.

  See "kstat2 Utility" in *Oracle Solaris 11.4 Tunable Parameters Reference Manual* and "Kernel Statistics" in *Writing Device Drivers in Oracle Solaris 11.4*.

`netstat`(8) and `nfsstat`(8)
: Displays information about network performance.

  See "Using the netstat Command" in *Administering TCP/IP Networks, IPMP, and IP Tunnels in Oracle Solaris 11.4*, "Networking Dcmds" in *Oracle Solaris Modular Debugger Guide*, "nfsstat Command" in *Managing Network File Systems in Oracle Solaris 11.4*, and "Administering Network File Systems" in *Managing Network File Systems in Oracle Solaris 11.4*.

`ps`(1) and `prstat`(8)
: Displays information about active processes.

  See "Using the `ps` Command" on page 29.

`sar`(8) and `sadc`
: Collects and reports on system activity data.

  See "Monitoring System Activities With the `sar` Command" on page 60.

`swap`(8)
: Displays information about available swap space on your system.

  See "Managing ZFS Swap and Dump Devices" in *Managing ZFS File Systems in Oracle Solaris 11.4*.

`vmstat`(8) and `iostat`(8)	Summarizes system activity data, such as virtual memory statistics, disk usage, and CPU activity. See "Displaying System Performance Information" on page 49.

Displaying System Performance Information

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

In addition to traditional command line tools such as `iostat` and `mpstat`, Oracle Solaris includes a browser-based performance visualization tool. The Oracle Solaris System Web Interface enables you to visualize both historical and current performance data and relationships among data from different sources and system events. For more information, see *Using Oracle Solaris 11.4 StatsStore and System Web Interface*.

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.

`kthr`	Reports the number of kernel threads in the following states:	
	`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` Kilobytes paged in.
- `po` Kilobytes paged out.
- `fr` Kilobytes 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.

EXAMPLE 13 Displaying VM Statistics at 5-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
```

For more information, see the vmstat(8) man page.

Displaying System Event Information

You can use 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

You can use the `vmstat -S` command 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 "Displaying Virtual Memory Statistics" on page 49.

`si` Average number of LWPs that are swapped 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

You can use the `vmstat -i` command to show the number of interrupts per device.

```
$ 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

You can 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, see the `iostat(8)` man page.

Displaying Disk Utilization With the `iostat` Command

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 list describes the fields in the output of the `iostat` *n* command:

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

You can use the `iostat -xt` command to display extended disk statistics.

```
$ iostat -xt
                    extended device statistics                      tty
device     r/s    w/s   kr/s   kw/s wait actv wsvc_t asvc_t  %w  %b  tin tout
sd0        0.0    0.0    0.0    0.0  0.0  0.0    0.0    0.0   0   0    1  449
sd1        0.0    0.0    0.0    0.0  0.0  0.0    0.0    0.0   0   0
sd2        0.3    0.6   84.9  205.9  0.0  0.0    0.0    6.6   0   0
sd3        2.2    7.3   54.4  187.2  0.0  0.1    0.0    5.7   0   2
...
nfs3102    0.0    0.0    0.0    0.0  0.0  0.0    0.0    0.0   0   0
nfs3104    0.0    0.0    0.0    0.0  0.0  0.0    0.0    0.5   0   0
nfs3106    0.0    0.0    0.1    0.0  0.0  0.0    0.0    0.6   0   0
nfs3107    0.0    0.0    0.0    0.0  0.0  0.0    0.0    0.0   0   0
nfs3108    0.0    0.0    0.0    0.0  0.0  0.0    0.0    0.0   0   0
nfs3109    0.0    0.0    0.0    0.0  0.0  0.0    0.0  117.1   0   0
```

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

You can 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 can transfer files to a disk that is not as full by using the `cp` command. Alternately, you can transfer files to a tape by using the `tar` or `cpio` commands or you can remove the files.

You can use the `df -k` command to display disk space information in kilobytes.

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

EXAMPLE 14 Displaying File System Information

The following example shows output from the `df -k` command on a SPARC based 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 15 Displaying File System Information by Using the `df` Command With No Options

The following example shows a list of all the mounted file systems, when the `df` command is used without any options or operands.

```
$ 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
```

For more information, see the `df`(8) man page.

Displaying Data Analytics Accelerator Statistics

The `daxstat` command reports the utilization and performance statistics for Data Analytics Accelerator (DAX) on systems that have the SPARC M7, SPARC M8, SPARC T7, or SPARC T8 chip. You must be in the `root` role to use the `daxstat` command. This command reports DAX statistics per-DAX, per-CPU, and per-queue in a tabular form. The first table summarizes all activity since boot. Each subsequent table summarizes the activity for the preceding interval. The values are reported as rates (events per second), unless mentioned otherwise.

The `daxstat` command has the following syntax:

```
/usr/bin/daxstat [[-T u | d] [-c processor_id] | [-d dax_id [-q queue_id]]] \
        | [-[x]d dax_id]] [interval [count]]

/usr/bin/daxstat -a [[-T u | d] [-c] | [-d [-q ]]] [-[x]d]] [interval [count]]
```

The following attributes are supported by the `daxstat` command.

`-a` The output is aggregated into a single value for all the CPUs, queues, and DAX units. Do not enter any *processor_id*, *dax_id*, or *queue_id* with this option.

`-c` Displays CPU statistics for specified CPUs.

`-d` Displays DAX statistics for specified DAX units.

`-q` Displays queue statistics for specified queues in specified DAX units.

`-x` Displays per-dax extended statistics for specified DAX units. This option is only valid when used with the `-d` option.

`-T u|d` Prints a time stamp before each report, in either standard date format, `d`, or the internal representation of time, `u`. For more information, see the `time(2)` and `date(1)` man pages.

interval Reports once each *interval* second.

count Prints only *count* reports.

EXAMPLE 16 Using the `daxstat` Command to Display Per-DAX Statistics

The following example shows DAX statistics over a three-second interval in two reports.

```
$ daxstat -ad 3 2
DAX    commands fallbacks    input    output %busy
ALL      201757    194975    74.0M      0.0M     0
ALL       53388     52066     9.0G     31.0M     0
```

The output fields of the `daxstat` command for statistics per-DAX are as follows:

`DAX` DAX ID

`commands` Number of commands completed by DAX

`fallbacks` Number of commands completed by the software but not completed by DAX

`input` Total input processed by DAX in megabytes per second (M) or gigabytes per second (G)

`output` Total output produced by DAX in megabytes per second (M) or gigabytes per second (G)

`%busy` Percentage of DAX cycles spent processing a command

EXAMPLE 17 Using the `daxstat` Command to Display Per-CPU Statistics

The following example shows DAX statistics for CPUs `0` and `1` in one report.

```
$ daxstat -c 0-1
CPU        calls          time    success  fallbacks
```

```
  0      6486811    9276154005    6486811          0
  1      6553856    9272570130    6553856          0
```

The output fields of the `daxstat` command for DAX statistics per-CPU are as follows:

`CPU`	CPU ID
`calls`	Number of `dax_submit` fast trap calls
`time`	Total kernel and hypervisor time in seconds spent submitting DAX commands
`success`	Number of `ccb_submit` hyper-calls that returned `success`
`fallbacks`	Number of `ccb_submit` hyper-calls that returned an error

EXAMPLE 18 Using the `daxstat` Command to Display Per-Queue Statistics

The following example shows DAX statistics for queues `0-3` in DAX unit `4`.

```
$ daxstat -d 4 -q 0-3
DAX   QUEUE    commands
  4       0         105
          1         196
          2         496
          3         196
```

where `QUEUE` is the queue ID.

For more information, see the `daxstat(8)` man page.

For more information about DAX, see the Breaking New Ground with Software in Silicon" blog and *Accelerate Analytics with Oracle Database In-Memory and Software in Silicon* at the Oracle Developer Community site.

Displaying DAX Information

You can use the `daxinfo` command to display the static configuration of DAX hardware available on systems that have the SPARC M7, SPARC M8, SPARC T7, or SPARC T8 chip. Information available includes DAX version, DAX operation codes, and the number of DAX instances on the system.

EXAMPLE 19 Displaying DAX Information

This example shows how to display information related to DAX by using the `daxinfo` command.

```
$ daxinfo
Version: DAX1
Opcodes: Extract Scan Select Translate
Enabled: 6
Disabled:0
```

where `Version` is the DAX version, `Opcodes` is the list of DAX operation codes that are supported, `Enabled` is the number of available DAX instances, and `Disabled` is the number of DAX instances with hardware error.

To display some of the selected fields, use `-o` and `-p` options with the `daxinfo` command as shown in the following example:

```
$ daxinfo -p -o version,enabled
DAX2:8
```

where `-p` displays the output in machine-readable format.

For more information, see the `daxinfo`(8) man page.

Monitoring System Activities

This section describes how to monitor system activities by using the `sar` command.

- "Monitoring System Activities With the `sar` Command" on page 60
- "Collecting System Activity Data Automatically" on page 76

Monitoring System Activities With the `sar` Command

You can 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" on page 76.

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

Monitoring File Access

You can display file access operation statistics with the `sar -a` command.

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 to view whether an application is disk-dependent.

The following list describes the file access routines that are displayed by the `-a` option.

`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.

EXAMPLE 20 Reporting File Access

```
$ sar -a
SunOS example-server 5.11 11.4.0.15.0 sun4v    09/07/2019

00:00:00  iget/s namei/s dirbk/s
01:00:01       0     112       0
02:00:00       0     136       0
...
09:20:00       0     156       0
09:40:01       0      96       0
10:00:00       0      83       0
10:20:00       0     119       0

Average        0     115       0
```

Monitoring Buffer Activity

You can 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.

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

`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 21 Reporting Buffer Activity

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 example-server 5.11 11.4.0.15.0 sun4v    09/07/2019

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

```
03:00:00       0       0     100       0       0     100       0       0

Average        0       0     100       0       0     100       0       0
```

Monitoring System Call Statistics

You can display system call statistics by using the `sar -c` command. 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.

The following list describes the system call categories that are reported by the `-c` option.

`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 22 Reporting System Call Statistics

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

```
$ sar -c
SunOS example-server 5.11 11.4.0.15.0 sun4v    09/07/2019

00:00:00 scall/s sread/s swrit/s  fork/s  exec/s rchar/s wchar/s
01:00:01    1276      24       3    0.68    0.71  170648  123527
02:00:00    1390      30       4    0.97    1.03  202074  136693
03:00:00    1272      25       4    0.66    0.69  196961  152584
```

```
04:00:00    1267      24       3    0.67    0.69  207036  161820
05:00:00    1315      26       4    0.73    0.77  232693  175446
06:00:01    1290      25       4    0.67    0.70  234286  188378
07:00:00    1306      26       4    0.76    0.80  195116  143241
08:00:00    1265      25       3    0.66    0.69  130554   83942
08:20:00    1626      44       9    1.21    1.27  175683   94947

Average     1297      26       4    0.71    0.75  187263  137889
```

Monitoring Disk Activity

You can display disk activity statistics with the `sar -d` command.

The following list describes the system call categories that the `-d` option reports.

`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 23 Reporting Disk Activity

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

```
$ sar -d
SunOS example-server 5.11 11.4.0.15.0 sun4v    09/07/2019

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

01:00:01   iscsi_se          0     0.1       0       1  4432.8     1.2
           sd0.t1.i          0     0.0       0       1     0.0     1.3
```

```
          nfs4              0     0.0       0       0     0.8     0.9
          nfs11             0     0.0       0       0     0.0     0.0
          nfs1694           0     0.0       0       0     0.0     0.0
          nfs2981           0     0.0       0       0     0.0     0.0
          nfs5471           0     0.0       0       0     0.0     0.0
          nfs20629          0     0.0       0       0     0.0     0.6
          nfs20631          0     0.0       0       0     0.0     0.7
          sd0               0     0.0       0       1     0.0     1.3
          sd0,a             0     0.0       0       1     0.0     1.3
          sd0,h             0     0.0       0       0     0.0     0.0
          vdc0              2     0.1       8     529     0.0    11.4

02:00:00  iscsi_se          0     0.1       0       0     0.0     0.0
          sd0.t1.i          0     0.0       0       0     0.0     0.0
          nfs4              0     0.0       0       0     0.8     0.7
...
Average   iscsi_se          0     0.1       0       2  1479.7     3.9
          sd0.t1.i          0     0.0       0       2     0.0     4.0
          nfs4              0     0.0       0       0     0.0     0.6
...
```

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.

Monitoring Page-Out and Memory

You can use the `sar -g` command to display page-out and memory freeing acCtivities in averages.

The output displayed by the `sar -g` command is a good indicator memory needs. 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 is 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 need to be increased.

EXAMPLE 24 Reporting Page-Out and Memory

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

```
$ sar -g
SunOS example-server 5.11 11.4.0.15.0 sun4v    09/07/2019

00:00:00  pgout/s ppgout/s pgfree/s pgscan/s %ufs_ipf
01:00:01     0.00     0.00     0.00     0.00     0.00
02:00:00     0.00     0.00     0.00     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
06:00:01     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
08:20:00     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
09:20:00     0.00     0.00     0.00     0.00     0.00
09:40:01     0.00     0.00     0.00     0.00     0.00
10:00:00     0.00     0.00     0.00     0.00     0.00
10:20:00     0.00     0.00     0.00     0.00     0.00

Average      0.00     0.00     0.00     0.00     0.00
```

Monitoring 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.

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

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 25 Reporting Kernel Memory Allocation

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

```
$ sar -k
SunOS example-server 5.11 11.4.0.15.0 sun4v    09/07/2019

00:00:00 sml_mem   alloc  fail  lg_mem   alloc  fail  ovsz_alloc  fail
01:00:01 2334107392 2634568704 1860748 2826780672 3027058432 31666  1714692096     0
02:00:00 2334107392 2639957504 1860748 3328139264 3561145088 31666  1714692096     0
03:00:00 2334148352 2644719616 1860748 3881467904 4150487808 31666  1714692096     0
04:00:00 2334213888 2650413568 1860748 185196544 489396352 31666  1714692096     0
05:00:00 2334287616 2655426560 1860748 826310656 1153170304 31666  1714692096     0
06:00:01 2334369536 2660379136 1860748 1518903296 1869017728 31666  1714692096     0
07:00:00 2334426880 2665492480 1860748 2046361600 2417197056 31666  1714692096     0
08:00:00 2334459648 2669230080 1860748 2361794560 2749543168 31666  1714692096     0
08:20:00 2334476032 2671023616 1860748 2479095808 2869149184 31666  1714692096     0
08:40:00 2334492416 2672560640 1860748 2603687936 3001746688 31666  1714692096     0
09:00:00 2334500608 2673795072 1860748 2733891584 3140963072 31666  1714692096     0
09:20:00 2334516992 2675175680 1860748 2865938432 3275326208 31666  1714692096     0
09:40:01 2334533376 2677078016 1860748 3005399040 3428043776 31666  1714692096     0
10:00:00 2334549760 2678332160 1860748 3146645504 3579805952 31666  1714692096     0
10:20:00 2334566144 2679967232 1860748 3301113856 3741610240 31666  1714692096     0

Average  2334383616 2663207936 1860748 2474048512 2830244096 31666  1714692096     0
```

Monitoring Interprocess Communication

You can use the `sar -m` command to report interprocess communication activities.

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

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

`msg/s` The number of message operations (`send` and `receive`) 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 example-server 5.11 11.4.0.15.0 sun4v    09/07/2019

00:00:00   msg/s  sema/s
01:00:01    0.00    0.00
02:00:00    0.00    0.00
...
10:20:00    0.00    0.00

Average     0.00    0.00
```

Monitoring Page-In Activity

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

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 26 Reporting Page-In Activity

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

```
$ sar -p
SunOS example-server 5.11 11.4.0.15.0 sun4v    09/07/2019

00:00:00  atch/s  pgin/s ppgin/s  pflt/s  vflt/s slock/s
01:00:01   56.31    0.00    0.00  167.66  299.39    0.02
02:00:00   66.17    0.01    0.03  218.07  349.68    0.03
03:00:00   56.92    0.00    0.01  156.83  299.16    0.02
04:00:00   56.49    0.00    0.00  157.43  300.03    0.02
05:00:00   59.62    0.21    0.85  171.29  313.03    0.02
06:00:01   57.71    0.03    0.05  150.49  303.41    0.02
07:00:00   60.85    0.14    0.21  177.63  306.82    0.02
08:00:00   57.45    0.00    0.00  156.70  274.23    0.02
08:20:00   77.47    0.01    0.02  312.96  391.30    0.04
08:40:00   46.90    0.01    0.01   93.34  226.69    0.01
09:00:00   43.93    0.00    0.00   67.28  209.63    0.00
09:20:00   75.66    0.00    0.00  321.04  390.62    0.04
09:40:01   46.68    0.01    0.01   93.83  229.32    0.02
10:00:00   44.07    0.00    0.01   67.41  219.60    0.00
10:20:00   59.93    0.00    0.00  161.27  303.64    0.04

Average    58.36    0.04    0.11  167.27  300.26    0.02
```

Monitoring Queue Activity

You can 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.

The following list describes the queue information from the `-q` option.

`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 27 Reporting 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 example-server 5.11 11.4.0.15.0 sun4v    09/07/2019

00:00:00 runq-sz %runocc swpq-sz %swpocc
01:00:01     1.2       7     0.0       0
02:00:00     1.2       5     0.0       0
03:00:00     1.2       6     0.0       0
04:00:00     1.2       6     0.0       0
05:00:00     1.2       5     0.0       0
06:00:01     1.2       4     0.0       0
07:00:00     1.2       3     0.0       0
08:00:00     1.2       8     0.0       0
08:20:00     1.2       6     0.0       0
08:40:00     1.2       4     0.0       0

Average      1.2       5     0.0       0
```

Monitoring Unused Memory

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

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

`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 28 Reporting Unused Memory

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

```
$ sar -r
SunOS example-server 5.11 11.4.0.15.0 sun4v    09/07/2019
```

```
00:00:00 freemem freeswap
01:00:01 2033446 143413667
02:00:00 1976436 142455652
03:00:00 1913957 141472967
04:00:00 1843041 140330892
05:00:00 1767369 138958248
06:00:01 1684910 137870401
07:00:00 1602032 136537114
08:00:00 1558301 135857941
08:20:00 1530449 135420182
08:40:00 1514288 135150442
09:00:00 1497581 134858710
09:20:00 1481513 134649032
09:40:01 1465218 134331511
10:00:00 1446124 134037258
10:20:00 1426441 133654337

Average  1725814 138477241
```

Monitoring CPU Utilization

You can use the `sar -u` command to display CPU utilization statistics.

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 following list describes the CPU utilization statistics from the `-u` option.

`%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 29 Reporting CPU Utilization

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
```

Monitoring System Table Status

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

The following list describes the system table statuses from the `-v` option.

`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 30 Reporting System Table Status

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
```

Monitoring Swapping Activity

You can use the `sar -w` command to report swapping and switching activity.

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

`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 31 Reporting Swap Activity

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
```

```
...
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
```

Monitoring Terminal Activity

You can use the `sar -y` command to monitor terminal device activities.If you have a lot of terminal I/O, you can use this report to determine whether any bad lines exist.

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

`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 32 Reporting Terminal Activity

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
...
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

Average        0       0       1       0       0       0
```

Monitoring Overall System Performance

You can 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

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 syntax:

```
# /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 perform the following functions:

- 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

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 syntax:

`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 is 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. **Become an administrator.**

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

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 using the Oracle Solaris Service Management Facility (SMF), the `crontab` command, and the `at` command.

It covers the following topics:

- "Overview of Scheduled System Tasks" on page 81
- "Scheduling Repetitive System Tasks With SMF" on page 85
- "Scheduling a Repetitive System Task With `crontab`" on page 87
- "Scheduling a Single System Task With the `at` Command" on page 97

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.

SMF

In general, SMF provides 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". You must be assigned the `solaris.smf.modify` and the `solaris.smf.manage` authorizations to configure a periodic service.

A *scheduled* service in SMF always runs according to an assigned start time. The `scheduled/schedule` property sets the frequency to run. Other `scheduled` properties specify the time to run.

`crontab` command	To execute repetitive or routine jobs (at a specific start time), use the `crontab` command. Its interface is `/var/spool/cron/crontabs`.
`at` command	To schedule a single job or to execute a task once (at a specific time), use the `at` command. Its interface is `/var/spool/cron/atjobs`.

Scheduling a Periodic or Scheduled Task With SMF

You can manage running applications or services with the Oracle Solaris Service Management Framework (SMF). Services are represented in the SMF framework by service objects, 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, see the `smf(7)` 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 about scheduled services, see Chapter 4, “Creating a Service to Run on a Specific Schedule” in *Developing System Services in Oracle Solaris 11.4*.

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. Thus, the scheduled task will begin only at specified intervals for the duration of time that such instances are online. For more information about periodic services, see Chapter 3, “Creating a Service to Run Periodically” in *Developing System Services in Oracle Solaris 11.4*.

The many advantages of using SMF to schedule tasks include:

- 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 is not 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, the `cron` or `at` commands are the only options for users without administrative privileges.

For step-by-step instructions, see "Scheduling Repetitive System Tasks With SMF" on page 85.

Scheduling a Routine System Task With the `crontab` Command

`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 must be executed more than one time, whereas the `at` command only allows for a one-time run. Note that `cron` never exits, so it should be executed only one time.

For step-by-step instructions on scheduling `crontab` jobs, see "How to Create or Edit a `crontab` File" on page 90.

Scheduling a Single System Task With `at`

The `at` command enables you to schedule a one-time task or an 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 one time. 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 in order to schedule the time at which your job executes. For more information about submitting `at` jobs, see "Submitting an `at` Job" on page 97.

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 98.

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 any one of the scheduling tools mentioned in "Overview of Scheduled System Tasks" on page 81.

Daily system administration tasks might include the following:

- Removing files more than a few days old from temporary directories
- Running 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 With SMF

This 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 85
- "Scheduling Method and Time Values for SMF" on page 85
- "How SMF Handles Scheduling" on page 85

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(7)` 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. For a complete description of the method conventions for `svc.startd` and similar restarters, see the `smf_method(7)` man page. Administrative methods, such as for the capture of legacy configuration information into the repository, are discussed in the `svccfg(8)` 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.

The following are the acceptable numerical values for SMF scheduling:

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

For more information about constraints and how to specify the `scheduled_method` element, see "Specifying the scheduled_method Element" in *Developing System Services in Oracle Solaris 11.4*.

Sample SMF Manifest

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 examples, see Chapter 3, “Creating a Service to Run Periodically” in *Developing System Services in Oracle Solaris 11.4*.

Scheduling a Repetitive System Task With crontab

This section describes how to schedule routine system tasks by using the crontab command.

- “How to Create or Edit a crontab File” on page 90
- “Verifying That a crontab File Exists” on page 91
- “Displaying a crontab File” on page 92
- “How to Remove a crontab File” on page 93
- “How to Deny crontab Command Access” on page 95
- “How to Limit crontab Command Access to Specified Users” on page 95

About the 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 is 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 the Oracle Solaris software installation. The file's contents include the following 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 on an x86 based system only.

- The fourth line locates and removes duplicate entries in the Generic Security Service table, /etc/gss/gsscred_db, at 3:30 a.m. daily.

For more information about crontab syntax, see "Syntax of crontab File Entries" on page 89.

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 bblack, mjane, dsmith, or jdoe.

To access crontab files that belong to root or other users requires that you assume the root role.

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 following tasks are performed by the cron daemon 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.

The following are the acceptable numerical values for crontab time entry:

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

The following special characters are used in crontab time fields:

- *Space* separates each field
- *Comma* separates multiple values
- *Hyphen* designates a range of values
- *Asterisk* (*) is a wildcard to include all possible values
- *Comment mark* (#) at the beginning of a line indicates 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 ed editor.

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 are in the root role.

▼ 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 11.4*.

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. This 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 89. The crontab file will be placed in the /var/spool/cron/crontabs directory.

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

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

Example 33 Creating a `crontab` File

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

```
# crontab -e mjane
```

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/mjane/*.log > /dev/null 2>&1
```

Listing `crontab` Files and Entries

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. 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 mjane
-rw-------   1 root     un           437 Oct  8  2012 jdoe
-r--------   1 root     root         453 Apr 28  2012 lp
-rw-------   1 root     sparccad      63 Sep 10 10:39 mja2
-rw-------   1 root     sparccad     387 Oct 14 15:15 jdoe2
-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 mja3
-r--------   1 root     sys          404 Jan 24  2013 uucp
```

Displaying a crontab File

The crontab -l command displays the contents of a crontab file 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 (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.

The crontab command can be used 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 34 Displaying a crontab File

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

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

EXAMPLE 35 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 36 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 jdoe
13 13 * * * cp /home/jdoe/work_files /usr/backup/. > /dev/null 2>&1
```

Removing crontab Files

By default, crontab file protections are set up such 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 Become an administrator to remove a crontab file that belongs to root or another user. Roles contain authorizations and privileged commands. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

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 37 Removing a `crontab` File

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

```
$ ls /var/spool/cron/crontabs
adm     jdoe     root    dsmith    sys     uucp
$ crontab -r
$ ls /var/spool/cron/crontabs
adm     jdoe 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 following user names are a part of the `cron.deny` file, which is created during the Oracle Solaris software installation.

```
$ cat /etc/cron.d/cron.deny
daemon
bin
smtp
nuucp
listen
```

```
nobody
noaccess
```

The user names in the default `cron.deny` file are denied access the `crontab` command. You can edit this file to deny other users 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 11.4*.

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
   ...
   ```

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 11.4*.

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 38 Limiting `crontab` Command Access to Specified Users

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

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

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

```
$ cat /etc/cron.d/cron.allow
root
jdoe
dsmith
```

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. The following message is displayed if the user can access the crontab command but no crontab file exists.

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

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

The following message is displayed if the user cannot access the crontab command, 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 With the at Command

This section describes how to schedule routine system tasks by using the at command. This section contains the following topics:

- "Submitting an at Job" on page 97
- "How to Create an at Job" on page 98
- "Displaying the at Queue" on page 99
- "Verifying an at Job" on page 99
- "Displaying at Jobs" on page 100
- "How to Remove at Jobs" on page 100
- "Denying Access to the at Command" on page 102

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

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 the following 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 dsmith near midnight on the last day of July.

```
$ at 11:45pm July 31
at> rm /home/dsmith/*core*
at> Press Control-d
commands will be executed using /bin/csh
job 933486300.a at Wed Jul 31 23:45:00 2019
```

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 at which to execute your job.

```
$ 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 39 Creating an `at` Job

The following example shows the `at` job that user `jdoe` 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/jdoe/*.backup
at> Press Control-D
job 897355800.a at Sun  Sep 8 19:30:00 2019
```

The following email message confirmed the execution of her `at` job.

```
Your "at" job rm /home/jdoe/*.backup" completed.
```

The following example shows how `jdoe` 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/jdoe/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 the `at` jobs that belong to `jdoe` have been submitted to the queue.

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

```
1st   Sep  7, 2019 23:45   jdoe  897543600.a     a     stdin
2nd   Sep  8, 2019 19:30   jdoe  897355800.a     a     stdin
3rd   Sep 10, 2019 04:00   jdoe  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 40 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
897543600.a Sat Sep  7 23:45:00 2019
897355800.a Sun Sep  8 19:30:00 2019
897732000.a Tue Sep 10 04:00:00 2019
```

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 Sep 10 04:00:00 2019
```

▼ How to Remove at Jobs

Before You Begin Assume the root role to remove an at job owned by any user. See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

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 executes.**

   # **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 41 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
897543600.a Sat Sep  7 23:45:00 2019
897355800.a Sun Sep  8 19:30:00 2019
897732000.a Tue Sep 10 04:00:00 2019
$ at -r 897732000.a
$ at -l 897732000.a
at: 897732000.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 following user names are a part of the `at.deny` file, which is created during the Oracle Solaris software installation.

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

In the `root` role, 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 that you want to prevent from using the at commands. Add only one user name per line.

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

EXAMPLE 42 Denying at Access

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

```
$ cat at.deny
daemon
bin
smtp
nuucp
listen
nobody
noaccess
jdoe
dsmith
```

Verifying That 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 dsmith cannot access the at command, the following message is displayed:

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

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

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.

♦ ♦ ♦ 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:

- "SMF Services That Manage the System Console and Terminal Devices" on page 105
- "Managing System Power Services" on page 108

SMF Services That Manage the System Console and 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(8)` 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 initiate `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. **Become an administrator.**

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

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 console speeds are supported on a SPARC based system:

- `9600 bps`

- 19200 bps
- 38400 bps

1. **Become an administrator.**

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

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 is an SMF service. 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 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 the time of installation, 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 preceding properties.

To display a brief summary of power management status, use 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 the preceding output, the values of `active_control/administrative-authority` indicate the source of the configuration.

- `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(8) man page.

EXAMPLE 43 Enabling and Disabling Power Management

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

```
# 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 44 Setting and Displaying Power Management Parameters

The following example shows how to set time-to-full-capacity to 300 microseconds, set time-to-minimum-responsiveness to 500 milliseconds, and inform the Oracle Solaris instance 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 description of the administrative-authority property at the beginning of this section. See also the poweradm(8) man page.

How to Recover from Power Service in Maintenance Mode

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

1. **Become an administrator.**

   See "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

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

A

access control
  at command, 101
  crontab files, 95
address space map
  displaying, 31
administering *See* managing
application threads, 46
architecture type
  displaying, 14
at command
  automatic job scheduling, 97
  controlling access to, 97
  deleting job files, 101
  denying access, 102
  displaying job queue, 99, 100
  error messages, 102
  job files, 97
  overview, 83, 97
  scheduling, 97
  sending email confirmation, 98
  submitting job files, 97
  system tasks, 81
at job files
  creating, 98
  description, 83
  location of, 83
at.deny file, 102
atjobs directory, 89
automatic system activity
  data collection, 76
  reporting, 76, 77

B

baud rate
  setting on ttymon console, 106
  setting with eeprom, 106
buffers
  monitoring activity, 62

C

changing
  baud rate of system console, 106
  date and time, 24
  host name, 25
  priority, 41, 43
  scheduling classes, 42
  timesharing process priority, 43
chip multithreading
  displaying with psrinfo, 23
configuration
  reporting with daxinfo, 59
controlling
  access to at command, 97
  processes, 34
CPU (central processing unit)
  displaying time usage, 30, 44
  high-usage processes, 44
  monitoring utilization, 72
creating
  at jobs, 98
  crontab files, 91
  message of the day (MOTD), 25
cron daemon, 83, 88
cron.allow file, 94, 96

`cron.deny` file, 94, 95
`crontab` command
  controlling access to, 94, 95, 96
  creating, 90
  editing, 90
  error messages, 96
  examples, 84
  files used by, 88
  quitting without saving changes, 90
  repetitive system tasks, 87
  scheduling of, 88
  scheduling periodic tasks, 83
  scheduling repetitive tasks, 83
  scheduling system tasks, 87
  system tasks, 81
`crontab` files
  creating, 91
  defaults, 88
  deleting, 93
  denying access, 95
  description, 88, 89
  listing, 91
  listing entries, 92
  location of, 88
  removing, 93
  syntax, 89

D

daily *See* repetitive
Data Analytics Accelerator *See* DAX
date and time
  changing, 24
  displaying, 13
DAX
  displaying HW configuration, 59
  statistics, 57
`daxinfo` command, 59
`daxstat` command, 57
deleting
  `at` jobs, 101
  `crontab` files, 93
devices
  displaying interrupts, 52
  displaying property values, 16
`df` command
  examples, 55
  overview, 55
directories
  current working directory for processes, 31
disk drives
  displaying information about
    free disk space, 55
disk space
  displaying information about, 55
disks
  displaying space, 55
  displaying statistics, 54
  monitoring activity, 64
`dispadmin` command, 40
displaying
  architecture type, 14
  basic system information, 14
  buffer activity, 62
  chip multithreading, 23
  CPU utilization, 72
  `crontab` file entries, 91
  date and time, 13
  DAX HW configuration, 59
  DAX statistics, 57
  diagnostic information, 20
  disk activity, 64
  disk space statistics, 55
  disk utilization information, 53
  extended disk statistics, 54
  file access, 61
  host ID, 13
  information about processes, 32, 32
  interprocess communication, 68
  interrupts per device, 52
  kernel memory allocation, 66
  linked libraries, 31
  page-in activity, 69
  page-out activity, 65
  physical processor type, 23
  priority information, 30, 39

process information, 30, 31, 33
processor type, 15
product name, 16
property values for a device, 16
queue activity, 70
release information, 14
scheduling class information, 30, 38, 39
swapping activity, 74
swapping statistics, 52
system activity information, 60, 77
system call activity, 63
system information
commands for, 13
system performance, 49
system table status, 73
system's installed memory, 16
terminal activity, 75
unused memory, 71
VM statistics, 49

E

`eeprom` command
setting baud rate on `ttymon`, 106
error messages
`at` command, 102
`/etc/cron.d/at.deny` file, 102
`/etc/cron.d/cron.allow` file, 94, 96
`/etc/cron.d/cron.deny` file, 94, 95
/etc/release file, 14

F

`fcntl()` system call, 31, 34
field entries
SMF instances, 85
file systems
displaying capacity and use, 55
files
checking access operations, 61
displaying `fstat` and `fcntl` output, 31
`fstat` and `fcntl` information display, 34
`fstat()` system call, 31, 34

G

global priorities for process classes, 39

H

host ID
displaying, 13
host name
changing, 25

I

interprocess communication
monitoring, 68
`iostat` command, 53
`isainfo` command, 15

K

kernel memory
monitoring allocation, 66
kernel thread
scheduling and, 29
kernel thread structure, 47
killing processes, 31, 35
`klwp` structure, 47
`kthread` structure, 47

L

listing
processes, 32
processes being executed, 32
LWPs (lightweight processes)
defined, 46
processes and, 46, 47
structures for, 47

M

managing
  power services, 108
  system processes, 29
memory
  displaying installed, 16
  kernel monitoring, 66
  monitoring page-out and, 65
  monitoring unused, 71
  process structures and, 47
  shared virtual memory, 47
  virtual processes and, 47
message of the day (MOTD) facility, 25
monitoring
  buffer activity, 62
  CPU utilization, 72
  disk activity, 64
  disk use, 54
  file access, 61
  interprocess communication, 68
  kernel memory, 66
  Oracle Enterprise Manager Ops Center and, 46
  overall system performance, 76
  page-in activity, 69
  page-out activity, 65
  queue activity, 70
  swapping activity, 74
  system activity, 60
  system calls, 63
  system performance, 45
  system table status, 73
  terminal activity, 75
  tools for, 48
  unused memory, 71
monthly *See* periodic
`motd` file, 25

N

new features
  `sar:default` log file at boot, 76
`nice` command, 43, 43
`nice` number, 29, 43

O

Oracle Enterprise Manager Ops Center, 46

P

page-in activity
  monitoring, 69
page-out activity
  monitoring, 65
`perf` file, 76
performance
  activities that are tracked and, 47
  automatic collection of activity data and, 76
  buffering and, 62
  CPU utilization and, 72
  disk activity and, 64
  displaying disk space statistics, 55
  displaying extended disk statistics, 54
  displaying interrupts per device, 52
  displaying statistics, 49
  displaying swapping statistics, 52
  file access and, 61
  interprocess communication and, 68
  kernel memory and, 66
  manual collection of activity data and, 77
  monitoring overall, 75, 76
  monitoring system, 45
  monitoring with Oracle Enterprise Manager Ops Center, 46
  page-in and, 69
  page-out and memory, 65
  process management and, 31, 43, 46
  queue activity and, 70
  reporting with `daxstat`, 57
  reports using `sar`, 60
  swapping and, 74
  system activities and, 46
  system calls and, 63
  system resources and, 45
  system tables and, 73
  tools for monitoring, 48
  unused memory and, 71
periodic system tasks

scheduling using `crontab`, 83
scheduling using SMF, 82
`pflags` command, 31
physical processor type
displaying, 23
`pkill` command, 31, 35
`pldd` command, 31
`pmap` command, 31
power management *See* power services
power services
managing, 108
troubleshooting, 111
`priocntl` command
overview, 40
syntax, 38
priority (process)
changing, 41, 43
changing timesharing processes, 41, 43
designating, 41
displaying global, 39
displaying information about, 30, 39
global, 38
overview, 38
scheduling classes and, 41
user-mode priority, 38
`/proc` directory, 30
`proc` structure, 47
`proc` tool commands, 31
process file system (PROCFS), 30
processes
application threads and, 46
changing priority, 41, 43
commands for managing, 28, 29
controlling, 34
current working directory for, 31, 34
defined, 46
displaying address space map, 31
displaying `fstat` and `fcntl` about open files, 31, 34
displaying information about, 33
displaying priority, 30, 39
global priorities, 38
killing, 31, 35
libraries linked into, 31
`nice` number of, 29, 43
no administration needed, 27
priority, 38, 43, 43
priority of scheduling classes, 41
`proc` tool commands, 30
runaway, 44
scheduling class priority, 38
scheduling classes, 38, 38, 41
setting priority, 41
signal actions, 31
stack trace, 31
structures for, 47
threads and, 47
tracing flags, 31
trees, 31, 34
troubleshooting, 44, 44
user-mode priority, 38
processor type
displaying, 15
PROCFS (process file system), 30
product name
displaying, 16
programs
disk access and, 61
threads, 46
`prtconf` command
displaying a system's product name, 16
displaying installed memory, 16
`ps` command
displaying full information about processes, 32
displaying global priority, 39
displaying scheduling class information, 30, 44
information reported, 29
`psig` command, 31
`psrinfo` command, 23
`pstack` command, 31
`ptime` command, 31
`ptree` command, 31, 34
`pwait` command, 31
`pwdx` command, 31, 34

Q

queue activity
  monitoring, 70

R

real-time processes
  changing class of, 42
release information
  displaying, 14
removing
  `at` jobs, 101
  `crontab` files, 93
repetitive system tasks
  scheduling, 87
  scheduling with `crontab`, 83
  SMF instances, 85
reporting *See* displaying
runaway processes, 44

S

`sa1` command, 76
`sa2` command, 76, 77
`sadc` command
  automatic collection of system data, 76
  collecting data at boot, 76
`sa`*dd* file, 77
`sar` command
  all options of, 78
  displaying system activity, 60
  options, 77
  reporting collected data, 77
`sar:default`
  `SMF` service, 76
`scheduled_method`
  element, 85
scheduling, 82, 82, 83
  See also `crontab`
  See also SMF
  See also using SMF
  one-time system tasks, 83
  periodic system tasks, 82, 83
  repetitive system tasks, 83
  SMF instances, 85
scheduling classes
  changing, 42
  changing priority, 41, 43
  designating, 41
  displaying information about, 30, 39
  managing, 38
  priority levels and, 38, 41
Service Management Facility (SMF) *See* SMF
shared memory
  process virtual memory, 47
SMF
  creating and editing manifest, 86
  creating and editing scheduled services, 86
  displaying instance, 86
  examples of repetitive task execution, 84
  handling scheduling, 85
  managing console and tty, 105
  managing power services, 108
  running system tasks, 81
  `sar:default` service, 76
  scheduling periodic tasks, 82
  scheduling repetitive tasks, 85
  scheduling system tasks, 85
  scheduling time values, 85
  `system/console-login` services, 105
  troubleshooting power services, 111
swapping activity
  monitoring, 74
`sys` crontab, 77
system activities
  affecting performance, 46
  automatic collection of data on, 76
  changing date and time, 24
  changing host name, 25
  commands for, 13
  list of activities tracked, 47
  manual collection of data on, 77
  processes and, 46
  scheduled, 81
system calls

monitoring, 63
system console
managing with SMF, 105
system identity *See* host name
system information
displaying, 14
system processes *See* processes
system resources
affecting performance, 45
system tasks, 82, 83
See also `crontab` command
See also SMF
scheduling one-time, 83
scheduling one-time tasks, 97
`system/console-login`
`SMF` service, 106

T

tables
monitoring status of system, 73
terminal devices
managing with SMF, 105
monitoring activity, 75
process controlling, 30
setting baud rate of system console, 106
setting up login services, 106
time
CPU usage, 30, 44
processes accumulating large amounts of CPU time, 44
timesharing processes
changing priority, 41, 43
changing scheduling parameters, 41
priority of range of, 38
setting priority, 41
tracing flags, 31
troubleshooting
power services, 111
processes, 44, 44

U

user processes
changing priority, 43
priority, 38
user structure, 47
user-mode priority, 38
`/usr/proc/bin` directory, 30, 31

V

`/var/adm/sa/sa`*dd* file, 77
`/var/spool/cron/atjobs` directory, 83, 89, 97
`/var/spool/cron/crontabs` directory, 88
`/var/spool/cron/crontabs/root` file, 87
`/var/spool/cron/crontabs/sys` crontab, 77
`crontabs/root` file, 87
`vmstat` command, 49