

Troubleshooting System Administration Issues in Oracle® Solaris 11.2

ORACLE

Part No: E36822-02
September 2014

Copyright © 1998, 2014, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT END USERS. Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Copyright © 1998, 2014, Oracle et/ou ses affiliés. Tous droits réservés.

Ce logiciel et la documentation qui l'accompagne sont protégés par les lois sur la propriété intellectuelle. Ils sont concédés sous licence et soumis à des restrictions d'utilisation et de divulgation. Sauf disposition de votre contrat de licence ou de la loi, vous ne pouvez pas copier, reproduire, traduire, diffuser, modifier, breveter, transmettre, distribuer, exposer, exécuter, publier ou afficher le logiciel, même partiellement, sous quelque forme et par quelque procédé que ce soit. Par ailleurs, il est interdit de procéder à toute ingénierie inverse du logiciel, de le désassembler ou de le décompiler, excepté à des fins d'interopérabilité avec des logiciels tiers ou tel que prescrit par la loi.

Les informations fournies dans ce document sont susceptibles de modification sans préavis. Par ailleurs, Oracle Corporation ne garantit pas qu'elles soient exemptes d'erreurs et vous invite, le cas échéant, à lui en faire part par écrit.

Si ce logiciel, ou la documentation qui l'accompagne, est concédé sous licence au Gouvernement des Etats-Unis, ou à toute entité qui délivre la licence de ce logiciel ou l'utilise pour le compte du Gouvernement des Etats-Unis, la notice suivante s'applique:

U.S. GOVERNMENT END USERS. Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

Ce logiciel ou matériel a été développé pour un usage général dans le cadre d'applications de gestion des informations. Ce logiciel ou matériel n'est pas conçu ni n'est destiné à être utilisé dans des applications à risque, notamment dans des applications pouvant causer des dommages corporels. Si vous utilisez ce logiciel ou matériel dans le cadre d'applications dangereuses, il est de votre responsabilité de prendre toutes les mesures de secours, de sauvegarde, de redondance et autres mesures nécessaires à son utilisation dans des conditions optimales de sécurité. Oracle Corporation et ses affiliés déclinent toute responsabilité quant aux dommages causés par l'utilisation de ce logiciel ou matériel pour ce type d'applications.

Oracle et Java sont des marques déposées d'Oracle Corporation et/ou de ses affiliés. Tout autre nom mentionné peut correspondre à des marques appartenant à d'autres propriétaires qu'Oracle.

Intel et Intel Xeon sont des marques ou des marques déposées d'Intel Corporation. Toutes les marques SPARC sont utilisées sous licence et sont des marques ou des marques déposées de SPARC International, Inc. AMD, Opteron, le logo AMD et le logo AMD Opteron sont des marques ou des marques déposées d'Advanced Micro Devices. UNIX est une marque déposée d'The Open Group.

Ce logiciel ou matériel et la documentation qui l'accompagne peuvent fournir des informations ou des liens donnant accès à des contenus, des produits et des services émanant de tiers. Oracle Corporation et ses affiliés déclinent toute responsabilité ou garantie expresse quant aux contenus, produits ou services émanant de tiers. En aucun cas, Oracle Corporation et ses affiliés ne sauraient être tenus pour responsables des pertes subies, des coûts occasionnés ou des dommages causés par l'accès à des contenus, produits ou services tiers, ou à leur utilisation.

Contents

Using This Documentation	5
1 Troubleshooting System Crashes	7
What's New in Crash Dump Files in Oracle Solaris 11.2	7
Crash Dump Files are Restructured	7
Crash Dump Files Saved to the /var/crash Directory	7
About System Crashes	8
System Crash Dump Files	8
Restructured Files	8
dumpadm and savecore Commands	9
Configuring Your System for Crash Dumps	10
Displaying the Current Crash Dump Configuration	10
Modifying the Configuration for Crash Dumps	11
Disabling or Enabling the Saving of Crash Dumps	13
Troubleshooting After a System Crashes	14
What to Do If the System Crashes	14
Examining Crash Dump Information	15
Checklist for Troubleshooting a System Crash	16
Saving Data When the Crash Dump Directory is Full	18
Managing Incidents With Oracle Enterprise Manager Ops Center	18
2 Troubleshooting When a System Hangs or Rebooting Fails	19
What to Do If Rebooting Fails	19
What to Do If You Forgot the Root Password or Cannot Boot the System	20
What to Do If a System Hang Occurs	20
3 Troubleshooting File System Problems	23
What to Do If a File System Fills Up	23
File System Fills Up Because a Large File or Directory Was Created	23
A TMPFS File System Is Full Because the System Ran Out of Memory	24

What to Do If File ACLs Are Lost After Copy or Restore	24
Troubleshooting File Access Problems	24
Solving Problems With Search Paths (Command not found)	24
Changing File and Group Ownerships	26
Solving File Access Problems	27
Recognizing Problems With Network Access	27
4 Prepare for Possible Process Failures by Using Core Files	29
About Process Failures and Core Files	29
Parameters for Core File Creation	30
Administering Your Core File Specifications	31
Displaying the Current Core Dump Configuration	32
Setting the Core File Name Pattern	32
Enabling File Paths	33
Enabling setuid Programs to Produce Core Files	33
Reverting to the Default Core File Settings	34
Correcting Obsolete Core File Parameter	34
Examining Core Files After a Process Failure	35
5 Managing System Logs and Messaging	37
Extended System Logging With rsyslogd	37
▼ Installing and Enabling rsyslog	37
Managing System Messages	38
Viewing System Messages	38
System Log Rotation	40
Customizing System Message Logging	41
Enabling Remote Console Messaging	43
Index	49

Using This Documentation

- **Overview** – Describes troubleshooting issues on both SPARC and x86 platforms
- **Audience** – System administrators using the Oracle Solaris 11 release
- **Required knowledge** – Experience administering UNIX systems

Product Documentation Library

Late-breaking information and known issues for this product are included in the documentation library at <http://www.oracle.com/pls/topic/lookup?ctx=E36784>.

Access to Oracle Support

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

Feedback

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

◆◆◆ 1 CHAPTER 1

Troubleshooting System Crashes

This chapter describes how to prepare for and troubleshoot a system crash in the Oracle Solaris OS.

This chapter covers the following:

- [“What’s New in Crash Dump Files in Oracle Solaris 11.2” on page 7](#)
- [“About System Crashes” on page 8](#)
- [“Configuring Your System for Crash Dumps” on page 10](#)

What's New in Crash Dump Files in Oracle Solaris 11.2

This section describes changes for managing crash dump files in this Oracle Solaris release.

Crash Dump Files are Restructured

In this release, the kernel crash dump files are divided into multiple new files based on their contents in order to perform quicker initial analysis and enable more granularity in configuration. For information, see [“Restructured Files” on page 8](#).

Crash Dump Files Saved to the /var/crash Directory

In the Oracle Solaris 11 operating system, the crash dump files are saved in the /var/crash directory. This change was originally introduced in the Oracle Solaris 10 1/13 release and is included in all the Oracle Solaris 11 releases.

About System Crashes

System crashes can occur due to hardware malfunctions, I/O problems, and software errors. If the system crashes, it will display an error message on the console and then write a copy of its physical memory to the dump device. The system will then reboot automatically. When the system reboots, the `savecore` command is executed to retrieve the data from the dump device and write the saved crash dump files to your `savecore` directory. These saved files provide invaluable information to aid in diagnosing the problem.

Note - The term, a *crash dump*, refers to the overall result of this process, including the set of crash dump files, where they are located, and how these files are organized and formatted.

System Crash Dump Files

The `savecore` command runs automatically after a system crash to retrieve the crash dump information from the dump device and writes the information into a set of files. Afterwards, the `savecore` command can be invoked on the same system or another system to expand the compressed crash dump files.

Note - Crash dump files are sometimes confused with *core* files, which are images of user applications that are written when the application terminates abnormally.

Crash dump files are saved in a predetermined directory, which by default is `/var/crash/`. In previous releases, crash dump files were overwritten when a system rebooted unless you manually enabled the system to save the images of physical memory in a crash dump file. Now, the saving of crash dump files is enabled by default.

Restructured Files

In the Oracle Solaris 11.2 release, the kernel crash dump files have been restructured. The contents of these files are divided into multiple new files based on their contents in order to perform quicker initial analysis and enable more granularity in configuration. The files can more easily be accessed and studied.

Kernel crash dumps were previously stored in the following files:

- `vmdump.N`

- `unix.N`
- `vmcore.N`

The `vmdump.N` and `vmcore.N` stored kernel pages metadata and data in compressed or uncompressed form, respectively.

Beginning with the Oracle Solaris 11.2 release, the crash dump information is written to the set of `vmdump-section.n` files. The section value is the name of a file section that contains a specific kind of dump information. The *n* value is an integer which increments every time `savecore` is run to copy a crash dump and a new crash dump is found on the dump device. Possible files include:

<code>vmdump-proc.N</code>	Dump file with compressed process pages
<code>vmdump-zfs.N</code>	Dump file with compressed ZFS metadata
<code>vmdump-other.N</code>	Dump file with other pages

For further information, see the `dumpadm(1M)` and `savecore(1M)` man pages.

dumpadm and savecore Commands

The `dumpadm` and `savecore` utilities configure and manage the creation of a crash dump as follows:

During system startup, the `dumpadm` command is invoked by the `svc:/system/dumpadm:default` service to configure crash dump parameters. It initializes the dump device and the dump content through the `/dev/dump` interface.

Note - In the Oracle Solaris 11.2 release, the `dumpadm` command has new options for specifying the dump contents, printing disk space estimates, and producing parseable output. See [“Modifying the Configuration for Crash Dumps” on page 11](#).

After the dump configuration is complete, the `savecore` script looks for the location of the crash dump file directory. Then, `savecore` is invoked to check for crash dumps and check the content of the `minfree` file in the crash dump directory. System crash dump files generated by the `savecore` command are saved by default.

Dump data is stored in a compressed format on the dump device. Kernel crash dump images can be as large as 4 Gbytes or more. Compressing the data means faster dumping and less disk space required for the dump device.

When a dedicated dump device, not the swap area, is part of the dump configuration, the saving of crash dump files is run in the background. A system that is booting does not wait for the

savecore command to complete before going to the next step. On large memory systems, the system can be available before savecore completes.

The savecore -L command enables an administrator to get a crash dump of a currently running the Oracle Solaris OS. This command is intended for troubleshooting a running system by taking a snapshot of memory during some bad state, such as a transient performance problem or service outage. If the system is up and you can still run some commands, you can execute the savecore -L command to save a snapshot of the system to the dump device and then immediately write out the crash dump files to your savecore directory. Because the system is still running, you can use the savecore -L command only if you have configured a dedicated dump device.

For more information, see [“Modifying the Configuration for Crash Dumps” on page 11](#), the `dumppadm(1M)`, and the `savecore(1M)` man pages.

Configuring Your System for Crash Dumps

This section describes tasks for managing crash dump procedures for your system.

Keep the following points in mind when you are working with system crash information:

- You must assume the root role to access and manage system crash information. See [“Using Your Assigned Administrative Rights” in “Securing Users and Processes in Oracle Solaris 11.2”](#).
- Do not disable the option of saving system crash dumps files on the system. System crash dump files provide an invaluable way to determine what is causing the system to crash.
- Dedicated ZFS volumes are used for swap and dump areas. After an installation, you might need to adjust the size of swap and dump devices or possibly recreate the swam and dump volumes. For instructions, see [“Managing Your ZFS Swap and Dump Devices” in “Managing ZFS File Systems in Oracle Solaris 11.2”](#).

To customize your system's crash dump processes before a system crash occurs, see the following:

- [“Displaying the Current Crash Dump Configuration” on page 10](#)
- [“Modifying the Configuration for Crash Dumps” on page 11](#)
- [“Disabling or Enabling the Saving of Crash Dumps” on page 13](#)

Displaying the Current Crash Dump Configuration

To display the current crash dump configuration, assume the root role and issue the `dumppadm` command with no arguments.

```
# dumpadm
Dump content: kernel pages
    Dump device: /dev/zvol/dsk/rpool/dump (dedicated)
Savecore directory: /var/crash
    Savecore enabled: yes
    Save compressed: on
```

This example output shows the following configuration:

- The dump content is kernel memory pages.
- Kernel memory will be dumped on a dedicated dump device, `/dev/zvol/dsk/rpool/dump`.
- System crash dump files will be saved in the `/var/crash` directory.
- Saving crash dump files is enabled.
- Save crash dumps in compressed format.

Modifying the Configuration for Crash Dumps

To modify the crash dump configuration, assume the root role and use the `dumpadm` command.

The syntax for the `dumpadm` command is as follows:

```
# /usr/sbin/dumpadm [-nuy] [-c content-type] [-d dump-device] [-m mink | minm | min%]
[-s savecore-dir] [-r root-dir] [-z on | off]
```

`-c content-type` Specifies the type of data to dump. The values for this option have changed for the Oracle Solaris 11.2 release. Use `kernel` to dump the kernel memory pages only, `all` to dump all memory pages, `curproc`, to dump kernel memory and the memory pages of the process whose thread was executing when the crash occurred, `allproc`, to dump kernel memory pages and all process pages, or `zfs`, to dump kernel pages which store ZFS metadata. The default dump content is kernel memory.

See the following examples of the `-c` option:

```
# dumpadm -c kernel
# dumpadm -c +zfs
# dumpadm -c -zfs
# dumpadm -c curproc+zfs
```

`-d dump-device` Specifies the device that stores dump data temporarily when the system crashes. The primary dump device is the default dump device. When the dump device is not the swap area, `savecore` runs in the background, which speeds up the boot process.

`-e` Prints estimate of disk space required for storing compressed crash dump. The value is computed using current configuration and currently running system.

<p><code>-m <i>mink</i> <i>minm</i> <i>min%</i></code></p>	<p>Specifies the minimum free disk space for saving crash dump files by creating a <code>minfree</code> file in the current <code>savecore</code> directory. This parameter can be specified in Kbytes (<code>mink</code>), Mbytes (<code>minm</code>) or file system size percentage (<code>min%</code>). If no minimum free space has been configured, the default is 1 Mbyte.</p> <p>The <code>savecore</code> command consults this file prior to writing the crash dump files. If writing the crash dump files would decrease the amount of free space below the <code>minfree</code> threshold due to their size, the dump files are not written and an error message is logged. For information about recovering from this scenario, see “Saving Data When the Crash Dump Directory is Full” on page 18.</p>
<p><code>-n</code></p>	<p>Specifies that <code>savecore</code> should not be run when the system reboots. This dump configuration is not recommended. If system crash information is written to the swap device and <code>savecore</code> is not enabled, the crash dump information is overwritten when the system begins to swap.</p>
<p><code>-p</code></p>	<p>Produces machine-parsable output.</p>
<p><code>-s <i>savecore-dir</i></code></p>	<p>Specifies an alternate directory for storing crash dump files. In Oracle Solaris 11, the default directory is <code>/var/crash</code>.</p>
<p><code>-u</code></p>	<p>Forcibly updates the kernel dump configuration based on the contents of the <code>/etc/dumpadm.conf</code> file.</p>
<p><code>-y</code></p>	<p>Modifies the dump configuration to automatically execute the <code>savecore</code> command upon reboot, which is the default for this dump setting.</p>
<p><code>-z on off</code></p>	<p>Modifies the dump configuration to control the operation of the <code>savecore</code> command upon reboot. The <code>on</code> setting enables the saving of core file in a compressed format. The <code>off</code> setting automatically uncompresses the crash dump file. Because crash dump files can be extremely large and therefore require less file system space if they are saved in a compressed format, the default is <code>on</code>.</p>

EXAMPLE 1-1 Modifying a Crash Dump Configuration

In this example, all of memory is dumped to the dedicated dump device, `/dev/zvol/dsk/rpool/dump`, and the minimum free space that must be available after the crash dump files are saved is 10% of the file system space.

```
# dumpadm
  Dump content: kernel pages
  Dump device: /dev/zvol/dsk/rpool/dump (dedicated)
  Savecore directory: /var/crash
```

```

Savecore enabled: yes
Save compressed: on

# dumpadm -c all -d /dev/zvol/dsk/rpool/dump -m 10%
Dump content: all pages
Dump device: /dev/zvol/dsk/rpool/dump (dedicated)
Savecore directory: /var/crash (minfree = 5697105KB)
Savecore enabled: yes
Save compressed: on

```

Disabling or Enabling the Saving of Crash Dumps



Caution - Oracle Solaris strongly recommends that you do not disable the saving of crash dumps. Crash dumps provide invaluable way to determine what causes your system to crash.

As the root role, you do have the ability to disable or enable the saving of crash dumps:

```
# dumpadm -n | -y
```

-n Disables the saving of crash dumps

-y Enables the saving of crash dumps

EXAMPLE 1-2 Disabling the Saving of Crash Dumps

This example illustrates how to disable the saving of crash dumps on your system.

```

# Dump content: all pages
Dump device: /dev/zvol/dsk/rpool/dump (dedicated)
Savecore directory: /var/crash (minfree = 5697105KB)
Savecore enabled: no
Save compressed: on

```

EXAMPLE 1-3 Enabling the Saving of Crash Dumps

This example illustrates how to enable the saving of crash dumps on your system.

```

# dumpadm -y
Dump content: all pages
Dump device: /dev/zvol/dsk/rpool/dump (dedicated)
Savecore directory: /var/crash (minfree = 5697105KB)
Savecore enabled: yes
Save compressed: on

```

Troubleshooting After a System Crashes

If an Oracle Solaris system crashes, you will want to provide your service provider with as much information as possible, including general system information and the specific information provided in the crash dump files.

To troubleshoot an existing crash dump, see the following:

- [“What to Do If the System Crashes” on page 14](#)
- [“Examining Crash Dump Information” on page 15](#)
- [“Checklist for Troubleshooting a System Crash” on page 16](#)
- [“Saving Data When the Crash Dump Directory is Full” on page 18](#)

What to Do If the System Crashes

The following list describes the most important information to remember in the event of a system crash:

1. Write down the system console messages.
 - If a system crashes, making it run again might seem like your most pressing concern. However, before you reboot the system, examine the console screen for messages. These messages can provide some insight about what caused the crash. Even if the system reboots automatically and the console messages have disappeared from the screen, you might be able to check these messages by viewing the system error log, the `/var/adm/messages` file. For more information about viewing system error log files, see [“How to View System Messages” on page 39](#).
 - If you have frequent crashes and cannot determine the cause, gather all of the information you can from the system console or the `/var/adm/messages` file and have it ready for a customer service representative to examine. For a complete list of troubleshooting information to gather for your service provider, see [“Checklist for Troubleshooting a System Crash” on page 16](#).
2. Check to see if a system crash dump was generated after the system crash.



Caution - Do not remove important system crash information until it has been sent to your customer service representative.

3. If the system fails to boot after a system crash, see [“Shutting Down and Booting a System for Recovery Purposes”](#) in [“Booting and Shutting Down Oracle Solaris 11.2 Systems”](#) for further instructions.

Examining Crash Dump Information

You can examine the control structures, active tables, memory images of a live or crashed system kernel, and other information about the operation of the kernel by using the `mdb` utility as described in the following procedure.

Note - The following procedure provides only a limited example of how to use the `mdb` utility. Using the `mdb` utility to its full potential requires a detailed knowledge of the kernel, and, is beyond the scope of this manual. For further information about using this utility, see the [mdb\(1\)](#) man page.

▼ How to Examine Crash Dump Information

1. Assume the root role.

See “Using Your Assigned Administrative Rights” in “Securing Users and Processes in Oracle Solaris 11.2”.

2. Change to the directory where the crash dump information has been saved.

For example:

```
# cd /var/crash
```

If you are unsure of the location of the crash dump, use the `dumpadm` command to determine where the system has been configured to store kernel crash dump files. For example:

```
# /usr/sbin/dumpadm
  Dump content: kernel pages
  Dump device: /dev/zvol/dsk/rpool/dump (dedicated)
Savecore directory: /var/crash
  Savecore enabled: yes
  Save compressed: on
```

3. Examine the crash dump by using the modular debugger utility (`mdb`).

```
# /usr/bin/mdb [-k] crashdump-file
```

`-k` Specifies kernel debugging mode by assuming the file is an operating system crash dump file.

`crashdump-file` Specifies the operating system crash dump file.

For example:

```
# /usr/bin/mdb -K vmcore.0
```

The command can also be specified as follows:

```
# /usr/bin/mdb -k 0
```

4. Display the system crash status.

```
> ::status
.
.
.
> ::system
.
.
.
```

To use the `::system dcmd` command when examining a kernel crash dump, the core file must be a kernel crash dump, and the `-k` option must have been specified when starting the `mdb` utility.

5. Quit the `mdb` utility.

```
> $quit
```

Example 1-4 Examining Crash Dump Information

This example shows sample output from the `mdb` utility, which includes system information and identifies the tunables that are set in this system's `/etc/system` file.

```
# cd /var/crash
# /usr/bin/mdb -k unix.0
Loading modules: [ unix krtld genunix ip nfs ipc ptm ]
> ::status
debugging crash dump /dev/mem (64-bit) from ozlo
operating system: 5.10 Generic sun4v
> ::system
set ufs_ninode=0x9c40 [0t40000]
set ncsiz=0x4e20 [0t20000]
set pt_cnt=0x400 [0t1024]
> $q
```

Checklist for Troubleshooting a System Crash

Answer the questions in the following checklist to help isolate the system problem and to prepare to consult with your support providers.

Item	Your Data
Is a system crash dump available?	

Item	Your Data
Identify the operating system release and appropriate software application release levels.	
Identify system hardware.	
Include <code>prtdiag</code> output for SPARC systems. Include Explorer output for other systems.	
Are patches installed? If so, include <code>showrev -p</code> output.	
Is the problem reproducible?	<p>This is important because a reproducible test case is often essential for debugging really hard problems. By reproducing the problem, the service provider can build kernels with special instrumentation to trigger, diagnose, and fix the bug.</p>
Does the system have any third-party drivers?	<p>Drivers run in the same address space as the kernel, with all the same privileges, so they can cause system crashes if they have bugs.</p>
What was the system doing before it crashed?	<p>If the system was doing anything unusual like running a new stress test or experiencing higher-than-usual load, that might have led to the crash.</p>
Were there any unusual console messages right before the system crashed?	<p>Sometimes the system will show signs of distress before it actually crashes; this information is often useful.</p>
Did you add any parameters to the <code>/etc/system</code> file?	<p>Sometimes tuning parameters, such as increasing shared memory segments so that the system tries to allocate more than it has, can cause the system to crash.</p>
Did the problem start recently?	<p>If so, did the onset of problems coincide with any changes to the system, for example, new drivers, new software, different workload, CPU upgrade, or a memory upgrade.</p>

Saving Data When the Crash Dump Directory is Full

If system crashes with no room left in the `savecore` directory and you want to save some critical system crash dump information, use one of the following methods.

After the system reboots, log in as the root role. Remove existing crash dump files that have already been sent to your service provider from the `savecore` directory.

Note - The `savecore` directory is typically `/var/crash`.

Alternately, after the system reboots, log in as the root role. Manually run the `savecore` command and specify an alternate directory that has sufficient disk space:

```
# savecore directory
```

Managing Incidents With Oracle Enterprise Manager Ops Center

If you need to manage incidents for physical and virtual operating systems, servers, and storage devices within a data center, rather than just monitoring incidents within individual systems, you can use the comprehensive system management solutions available in the Oracle Enterprise Manager Ops Center.

Using the Enterprise Manager Ops Center, you can setup alerts that tell you when part of your data center is not performing as expected, manage these incident reports, and attempt repairs.

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

Troubleshooting When a System Hangs or Rebooting Fails

This chapter describes what you can do when a system hangs or if you have problems rebooting a system..

This chapter covers the following information:

- [“What to Do If Rebooting Fails” on page 19](#)
- [“What to Do If You Forgot the Root Password or Cannot Boot the System” on page 20](#)
- [“What to Do If a System Hang Occurs” on page 20](#)

What to Do If Rebooting Fails

If the system does not reboot completely, or if the system reboots and then crashes again, there might be a software or hardware problem that is preventing the system from booting successfully.

Cause of System Not Booting	How to Fix the Problem
The system can't find <code>/platform/`uname -m`/kernel/sparcv9/unix</code> .	You may need to change the <code>boot-device</code> setting in the PROM on a SPARC based system. For information about changing the default boot device, see “Displaying and Setting Boot Attributes” in “Booting and Shutting Down Oracle Solaris 11.2 Systems” .
The Oracle Solaris boot archive has become corrupted. Or, the SMF boot archive service has failed. An error message is displayed if you run the <code>svcs -x</code> command.	Create a second boot environment that is a backup of the primary boot environment. In the event the primary boot environment is not bootable, boot the backup boot environment. Alternatively, you can boot from the live CD or USB media.
There is an invalid entry in the <code>/etc/passwd</code> file.	For information about recovering from an invalid <code>passwd</code> file, see “How to Boot From Media to Resolve an Unknown root Password” in “Booting and Shutting Down Oracle Solaris 11.2 Systems” .

Cause of System Not Booting	How to Fix the Problem
The x86 boot loader (GRUB) is damaged. Or, the GRUB menu is missing or has become corrupt.	For information about recovering from a damaged x86 boot loader or a missing or corrupt GRUB menu, see “How to Boot From Media to Resolve a Problem With the GRUB Configuration That Prevents the System From Booting” in “Booting and Shutting Down Oracle Solaris 11.2 Systems ” .
There's a hardware problem with a disk or another device.	Check the hardware connections: <ul style="list-style-type: none">■ Make sure the equipment is plugged in.■ Make sure all the switches are set properly.■ Look at all the connectors and cables, including the Ethernet cables.■ If all these steps fail, turn off the power to the system, wait 10 to 20 seconds, and then turn on the power again.

If none of the above suggestions solve the problem, contact your local service provider.

What to Do If You Forgot the Root Password or Cannot Boot the System

If you forget the root password or experience another problem that prevents the system from booting, do the following:

- Stop the system.
- Follow the directions in [“How to Boot From Media to Resolve an Unknown root Password”](#) in [“Booting and Shutting Down Oracle Solaris 11.2 Systems ”](#).
- If the root password is the problem, remove the root password from the `/etc/shadow` file.
- Reboot the system.
- Log in and set the root password.

What to Do If a System Hang Occurs

A system can freeze or hang rather than crash completely if some software process is stuck. Follow these steps to recover from a hung system.

1. Determine whether the system is running a window environment and follow these suggestions. If these suggestions do not solve the problem, go to step 2.
 - Make sure the pointer is in the window where you are typing the commands.

- Press Control-q in case the user accidentally pressed Control-s, which freezes the screen. Control-s freezes only the window, not the entire screen. If a window is frozen, try using another window.
 - If possible, log in remotely from another system on the network. Use the `pgrep` command to look for the hung process. If it looks like the window system is hung, identify the process and kill it.
2. Press **Control-** to force quit the running program and (probably) write out a core file.
 3. Press **Control-c** to interrupt the program that might be running.
 4. Log in remotely and attempt to identify and kill the process that is hanging the system.
 5. Log in remotely, assume the root role and then reboot the system.
 6. If the system still does not respond, force a crash dump and reboot. For information about forcing a crash dump and booting, see [“Forcing a Crash Dump and Reboot of the System” in “Booting and Shutting Down Oracle Solaris 11.2 Systems”](#).
 7. If the system still does not respond, turn the power off, wait a minute or so, then turn the power back on.
 8. If you cannot get the system to respond at all, contact your local service provider for help.

◆◆◆ 3 CHAPTER 3

Troubleshooting File System Problems

This chapter describes how to fix file system issues including the following:

- [“What to Do If a File System Fills Up” on page 23](#)
- [“What to Do If File ACLs Are Lost After Copy or Restore” on page 24](#)
- [“Troubleshooting File Access Problems” on page 24](#)

What to Do If a File System Fills Up

When the root (/) file system or any other file system fills up, you will see the following message in the console window:

```
.... file system full
```

There are several reasons why a file system fills up. The following sections describe several scenarios for recovering from a full file system.

File System Fills Up Because a Large File or Directory Was Created

Reason Error Occurred	How to Fix the Problem
Someone accidentally copied a file or directory to the wrong location. This also happens when an application crashes and writes a large <code>core</code> file to the file system.	Log in and assume the <code>root</code> role, then use the <code>ls -tl</code> command in the specific file system to identify which large file is newly created and then remove it.

A TMPFS File System Is Full Because the System Ran Out of Memory

Reason Error Occurred	How to Fix the Problem
This can occur if TMPFS is trying to write more than it is allowed or some current processes are using a lot of memory.	For information about recovering from tmpfs-related error messages, see the tmpfs(7FS) man page.

What to Do If File ACLs Are Lost After Copy or Restore

Reason Error Occurred	How to Fix the Problem
If files or directories with ACLs are copied or restored into the /tmp directory, the ACL attributes are lost. The /tmp directory is usually mounted as a temporary file system, which doesn't support UFS file system attributes such as ACLs.	Copy or restore files into the /var/tmp directory instead.

Troubleshooting File Access Problems

Users frequently experience problems, and call on a system administrator for help, because they cannot access a program, a file, or a directory that they could previously use.

Whenever you encounter such a problem, investigate one of three areas:

- The user's search path may have been changed, or the directories in the search path may not be in the proper order.
- The file or directory may not have the proper permissions or ownership.
- The configuration of a system accessed over the network may have changed.

This chapter briefly describes how to recognize problems in each of these three areas and suggests possible solutions.

Solving Problems With Search Paths (Command not found)

A message of `Command not found` indicates one of the following:

- The command is not available on the system.
- The command directory is not in the search path.

To fix a search path problem, you need to know the pathname of the directory where the command is stored.

If the wrong version of the command is found, a directory that has a command of the same name is in the search path. In this case, the proper directory may be later in the search path or may not be present at all.

You can display your current search path by using the `echo $PATH` command.

Use the `type` command to determine whether you are running the wrong version of the command. For example:

```
$ type acroread
acroread is /usr/bin/acroread
```

▼ How to Diagnose and Correct Search Path Problems

1. **Display the current search path to verify that the directory for the command is not in your path or that it isn't misspelled.**

```
$ echo $PATH
```

2. **Check the following:**

- Is the search path correct?
- Is the search path listed before other search paths where another version of the command is found?
- Is the command in one of the search paths?

If the path needs correction, go to step 3. Otherwise, go to step 4.

3. **Add the path to the appropriate file, as shown in this table.**

Shell	File	Syntax	Notes
bash and ksh93	<code>\$HOME/.profile</code>	<code>\$ PATH=\$HOME/bin:/sbin:/usr/local/bin ...</code> <code>\$ export PATH</code>	A colon separates path names.

4. **Activate the new path as follows:**

Shell	Path Location	Command to Activate The Path
bash and ksh93	.profile	. \$HOME/.profile
	.login	hostname\$ source \$HOME/.login

5. Verify the new path.

```
$ which command
```

Example 3-1 Diagnosing and Correcting Search Path Problems

This example shows that the `mytool` executable is not in any of the directories in the search path using the `type` command.

```
$ mytool
-bash: mytool: command not found
$ type mytool
-bash: type: mytool: not found
$ echo $PATH
/usr/bin:
$ vi $HOME/.profile
(Add appropriate command directory to the search path)
$ . $HOME/.profile
$ mytool
```

If you cannot find a command, look at the man page for its directory path.

Changing File and Group Ownerships

Frequently, file and directory ownerships change because someone edited the files as an administrator. When you create home directories for new users, be sure to make the user the owner of the dot (.) file in the home directory. When users do not own “.” they cannot create files in their own home directory.

Access problems can also arise when the group ownership changes or when a group of which a user is a member is deleted from the `/etc/group` database.

For information about how to change the permissions or ownership of a file that you are having problems accessing, see [Chapter 1, “Controlling Access to Files,”](#) in [“Securing Files and Verifying File Integrity in Oracle Solaris 11.2”](#).

Solving File Access Problems

When users cannot access files or directories that they previously could access, the permissions or ownership of the files or directories probably has changed.

Recognizing Problems With Network Access

If users have problems using the `rcp` remote copy command to copy files over the network, the directories and files on the remote system may have restricted access by setting permissions. Another possible source of trouble is that the remote system and the local system are not configured to allow access.

See [“Strategies for NFS Troubleshooting”](#) in [“Managing Network File Systems in Oracle Solaris 11.2”](#) for information about problems with network access and problems with accessing systems through AutoFS.

◆◆◆ CHAPTER 4

Prepare for Possible Process Failures by Using Core Files

This chapter describes how to set up the specifications for the core files that a system produces when a process fails and how to examine these core files after a failure.

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

- [“About Process Failures and Core Files” on page 29](#)
- [“Parameters for Core File Creation” on page 30](#)
- [“Administering Your Core File Specifications” on page 31](#)
- [“Examining Core Files After a Process Failure” on page 35](#)

About Process Failures and Core Files

When a process or application terminates abnormally, the system automatically generates a set of files. This process can be described as a *core dump*. The files that are created are *core files*. A core file is a disk copy of the contents of the process address space at the time of its termination, along with additional information about the state of the process. Typically, core files are produced following abnormal termination of a process resulting from a bug in the corresponding application. A core file provides invaluable information for you to use in diagnosing the problem that cause the process failure. See [“Parameters for Core File Creation” on page 30](#).

As part of your ongoing system administration, you can use the `coreadm` command to control the creation specifications for core files. For example, you can use the `coreadm` command to configure a system so that all process core files are placed in a single system directory so you can more easily track problems. See [“Administering Your Core File Specifications” on page 31](#).

When a process terminates abnormally, you can inspect the core files that are created using a debugger such as `mdb` or by using a `proc` tool. See [“Examining Core Files After a Process Failure” on page 35](#).

Parameters for Core File Creation

When a process fails, the system tries to create up to two core files for each failed process, using a global core file name pattern and a per-process core file name pattern to create each core file name. The `coreadm` command controls these name patterns and specifies the location of the core files. This section describes some of the file path and file name parameters. For a full description of the core dump process, see the `core(4)` man page. For the full description of the `coreadm` options, see the `coreadm(1M)` man page.

Configurable Core File Paths

When a process terminates abnormally, it produces a core file in the current directory by default. If the global core file path is enabled, each abnormally terminating process might produce two files, one in the current working directory, and one in the global core file location. The file paths that are used are configurable parameters.

Two configurable core file paths can be enabled or disabled independently of each other as follows:

- A per-process core file path, which defaults to `core` and is enabled by default. If enabled, the per-process core file path causes a core file to be produced when the process terminates abnormally. The per-process path is inherited by a new process from its parent process.

When generated, a per-process core file is owned by the owner of the process with read/write permissions for the owner. Only the owning user can view this file.

- A global core file path, which defaults to `core` and is disabled by default. If enabled, an *additional* core file with the same content as the per-process core file is produced by using the global core file path.

When generated, a global core file is owned by root, with read/write permissions for root *only*. Non-privileged users cannot view this file.

Note - By default, a `setuid` process does not produce core files using either the global or per-process path.

Expanded Core File Names

The name of a core file contains fields with information about the failed process. For a full description of the core file name fields, see the `coreadm(1M)` man page. This section focuses on the global variables.

If a global core file directory is enabled, core files can be distinguished from one another by using the variables that are described in the following table.

%d	Executable file directory name, up to a maximum of MAXPATHLEN characters
%f	Executable file name, up to a maximum of MAXCOMLEN characters
%g	Effective group ID
%m	Machine name (<code>uname -m</code>)
%n	System node name (<code>uname -n</code>)
%p	Process ID
%t	Decimal value of <code>time(2)</code>
%u	Effective user ID
%z	Name of the zone in which process is executed (<code>zonename</code>)
%%	Literal %

For example, suppose `/var/core/core.%f.%p` is set as the global core file path. If a `sendmail` process with PID 12345 terminates abnormally, it would produce `/var/core/core.sendmail.12345` as the core file.

Improving Core File Dump Performance

You can improve the performance of core file dumping on a system by excluding some parts of a process's binary image from the core dump. When you enter the `coreadm` command to customize your core dump specifications, you could specify exclusion of, for example, DISM mappings, or ISM mappings, or System V shared memory from a core dump. For instructions, see the `coreadm(1M)` man page.

Administering Your Core File Specifications

You can manage the core files as follows:

- [“Displaying the Current Core Dump Configuration” on page 32](#)

- [“Setting the Core File Name Pattern” on page 32](#)
- [“Enabling File Paths” on page 33](#)
- [“Enabling setuid Programs to Produce Core Files” on page 33](#)
- [“Reverting to the Default Core File Settings” on page 34](#)
- [“Correcting Obsolete Core File Parameter” on page 34](#)

Displaying the Current Core Dump Configuration

Use the `coreadm` command without any options to display the current core dump configuration.

```
$ coreadm
    global core file pattern:
global core file content: default
    init core file pattern: core
    init core file content: default
    global core dumps: disabled
    per-process core dumps: enabled
    global setid core dumps: disabled
per-process setid core dumps: disabled
    global core dump logging: disabled
```

Setting the Core File Name Pattern

You can set a core file name pattern on a global, zone, or per-process basis. In addition, you can set per-process defaults that persist across a system reboot.

For example, you can use the following `coreadm` command to set the default per-process core file pattern for all processes that are started by the `init` process. This setting applies to all processes that have not explicitly overridden the default core file pattern. This setting persists across system reboots.

```
# coreadm -i /var/core/core.%f.%p
```

You can use the following `coreadm` command to set the per-process core file name pattern for any processes:

```
# coreadm -p /var/core/core.%f.%p $$
```

The `$$` symbols represent a placeholder for the process ID of the currently running shell. The per-process core file name pattern is inherited by all child processes.

Here's another example:

```
$ coreadm -p $HOME/corefiles/%f.%p $$
```


Alternately, assume the root role and set a global file name pattern:

```
# coreadm -g /var/corefiles/%f.%p
```

After a global or per-process core file name pattern is set, it must be enabled with the `coreadm -e` command.

You can set the core file name pattern for all processes that are run during a user's login session by putting the command in a user's initialization file, for example, `.profile`.

Enabling File Paths

You can enable a per-process or global core file path.

- To enable a per-process core file path, assume the root role and issue the following command:

```
# coreadm -e process
```

If you want to verify the configuration, display the current process core file path:

```
# coreadm $$
1180: /home/kryten/corefiles/%f.%p
```

- To enable a global core file path, assume the root role and issue the following command:

```
# coreadm -e global -g /var/core/core.%f.%p
```

If you want to verify the configuration, display the current process core file path:

```
# coreadm
  global core file pattern: /var/core/core.%f.%p
  global core file content: default
  init core file pattern: core
  init core file content: default
    global core dumps: enabled
  per-process core dumps: enabled
  global setid core dumps: disabled
  per-process setid core dumps: disabled
  global core dump logging: disabled
```

Enabling setuid Programs to Produce Core Files

You can use the `coreadm` command to enable or disable `setuid` programs to produce core files for all system processes or on a per-process basis by setting the following paths:

- If the global `setuid` option is enabled, a global core file path allows all `setuid` programs on a system to produce core files.
- If the per-process `setuid` option is enabled, a per-process core file path allows specific `setuid` processes to produce core files.

By default, both flags are disabled. For security reasons, the global core file path must be a full pathname starting with a leading `/`. If root disables per-process core files, individual users cannot obtain core files.

The `setuid` core files are owned by root, with read/write permissions for root only. Regular users cannot access these files even if the process that produced the `setuid` core file is owned by an ordinary user.

For more information, see the [coreadm\(1M\)](#) man page.

Reverting to the Default Core File Settings

As root, run one of the following commands to disable the core file path and remove the core file name pattern:

- For global core file settings:

```
# coreadm -d global -g ""
```

Note - "" is an empty string with no space.

- For per-process core file settings:

```
# coreadm -d process -g ""
```

The `-d` option disables the core file path. The `-g` option with the empty string variable removes the core file name pattern. The core file path and core file name pattern are returned to the original default settings.

Correcting Obsolete Core File Parameter

Error Message

```
NOTICE: 'set allow_setid_core = 1' in /etc/system is obsolete
NOTICE: Use the coreadm command instead of 'allow_setid_core'
```

Cause

You have an obsolete parameter that allows `setuid` core files in your `/etc/system` file.

Solution

Remove `allow_setuid_core=1` from the `/etc/system` file. Then use the `coreadm` command to enable global setuid core file paths.

Examining Core Files After a Process Failure

The `proc` tools enable you to examine process core files as well as live processes. The `proc` tools are utilities that can manipulate features of the `/proc` file system.

The `/usr/proc/bin/pstack`, `pmap`, `pldd`, `pflags`, and `pcrd` tools can be applied to core files by specifying the name of the core file on the command line, similar to the way you specify a process ID to these commands.

For more information about using the `proc` tools to examine core files, see [proc\(1\)](#).

EXAMPLE 4-1 Examining Core Files With the `proc` Tools

```
$ ./a.out
Segmentation Fault(coredump)
$ /usr/proc/bin/pstack ./core
core './core' of 19305: ./a.out
000108c4 main      (1, ffbef5cc, ffbef5d4, 20800, 0, 0) + 1c
00010880 _start    (0, 0, 0, 0, 0, 0) + b8
```


Managing System Logs and Messaging

This chapter covers viewing and managing system logs and system messages.

This chapter cover the following information:

- [“Extended System Logging With rsyslogd” on page 37](#)
- [“Viewing System Messages” on page 38](#)
- [“System Log Rotation” on page 40](#)
- [“Customizing System Message Logging” on page 41](#)
- [“Enabling Remote Console Messaging” on page 43](#)

Extended System Logging With rsyslogd

This Oracle Solaris release includes the option of installing and using the `rsyslog` service for managing system logging. `rsyslog` is a reliable and extended `syslog` daemon implementation with a modular design that supports several features, for example, filtering, TCP, encryption, and high-precision time-stamps, as well as output control.

The `syslog` SMF service, `svc:/system/system-log:default`, continues to be the default logging service. To use the `rsyslog` service, you need to install the `rsyslog` package and enable the `rsyslog` service.

▼ Installing and Enabling rsyslog

1. **You can check whether the `rsyslog` package is already installed on your system by trying to enable the service as follows:**

```
root@pcclone: ~# svcadm enable svc:/system/system-log:rsyslog
```

If the `rsyslog` package is not installed, the following message displays:

```
svcadm: Pattern 'svc:/system/system-log:rsyslog' doesn't match any instance.
```

2. If the rsyslog package is not installed, install it.

```

root@pcclone:~# pkg install rsyslog
    Packages to install: 3
    Services to change: 1
    Create boot environment: No
    Create backup boot environment: No

DOWNLOAD                                PKGS      FILES    XFER (MB)  SPEED
Completed                                3/3       68/68     1.7/1.7    354k/s

PHASE                                    ITEMS
Installing new actions                    147/147
Updating package state database           Done
Updating package cache                    0/0
Updating image state                      Done
Creating fast lookup database             Done
    
```

3. Confirm that there is an instance of rsyslog.

```

root@pcclone:~# svcs -a | grep "system-log"
disabled      18:27:16 svc:/system/system-log:rsyslog
online        18:27:21 svc:/system/system-log:default
    
```

This output confirms that the rsyslog instance exists, but that it is disabled.

4. Switch to the rsyslog service.

```

root@pcclone:~# svcadm disable svc:/system/system-log:default
root@pcclone:~# svcadm enable svc:/system/system-log:rsyslog
root@pcclone:~# svcs -xv
    
```

These commands disable the default service, enable rsyslog and report on status.

Next Steps After rsyslog is installed and enabled, you can configure syslog in /etc/rsyslog.conf.

Managing System Messages

The following sections describe system messaging features in Oracle Solaris.

Viewing System Messages

System messages display on the console device. The text of most system messages look like this:

```
[ID msgid facility.]
```

For example:

```
[ID 672855 kern.notice] syncing file systems...
```

If the message originated in the kernel, the kernel module name is displayed. For example:

```
Oct 1 14:07:24 mars ufs: [ID 845546 kern.notice] alloc: /: file system full
```

When a system crashes, it might display a message on the system console like this:

```
panic: error message
```

Less frequently, this message might be displayed instead of the panic message:

```
Watchdog reset !
```

The error logging daemon, `syslogd`, automatically records various system warnings and errors in message files. By default, many of these system messages are displayed on the system console and are stored in the `/var/adm` directory. You can direct where these messages are stored by setting up system message logging. For more information, see [“Customizing System Message Logging” on page 41](#). These messages can alert you to system problems, such as a device that is about to fail.

The `/var/adm` directory contains several message files. The most recent messages are in `/var/adm/messages` file (and in `messages.*`), and the oldest are in the `messages.3` file. After a period of time (usually every ten days), a new messages file is created. The `messages.0` file is renamed `messages.1`, `messages.1` is renamed `messages.2`, and `messages.2` is renamed `messages.3`. The current `/var/adm/messages.3` file is deleted.

Because the `/var/adm` directory stores large files containing messages, crash dumps, and other data, this directory can consume lots of disk space. To keep the `/var/adm` directory from growing too large, and to ensure that future crash dumps can be saved, you should remove unneeded files periodically. You can automate this task by using the `crontab` file. For more information about automating this task, see [“Removing Dump Files” in “Managing Devices in Oracle Solaris 11.2”](#) and [Chapter 4, “Scheduling System Tasks,” in “Managing System Information, Processes, and Performance in Oracle Solaris 11.2”](#).

▼ How to View System Messages

- **Display recent messages generated by a system crash or reboot by using the `dmesg` command.**

```
$ dmesg
```

Or, use the `more` command to display one screen of messages at a time.

```
$ more /var/adm/messages
```

Example 5-1 Viewing System Messages

The following example shows output from the `dmesg` command on an Oracle Solaris 10 system.

```
$ dmesg
Mon Sep 13 14:33:04 MDT 2010
Sep 13 11:06:16 sr1-ubrm-41 svc.startd[7]: [ID 122153 daemon.warning] ...
Sep 13 11:12:55 sr1-ubrm-41 last message repeated 398 times
Sep 13 11:12:56 sr1-ubrm-41 svc.startd[7]: [ID 122153 daemon.warning] ...
Sep 13 11:15:16 sr1-ubrm-41 last message repeated 139 times
Sep 13 11:15:16 sr1-ubrm-41 xscreensaver[25520]: ...,
Sep 13 11:15:16 sr1-ubrm-41 xscreensaver[25520]: ...
Sep 13 11:15:17 sr1-ubrm-41 svc.startd[7]: [ID 122153 daemon.warning]...
.
.
.
```

See Also For more information, see the [dmesg\(1M\)](#) man page.

System Log Rotation

System log files are rotated by the `logadm` command from an entry in the root crontab file. The `/usr/lib/newsyslog` script is no longer used.

The system log rotation is defined in the `/etc/logadm.conf` file. This file includes log rotation entries for processes such as `syslogd`. For example, one entry in the `/etc/logadm.conf` file specifies that the `/var/log/syslog` file is rotated weekly unless the file is empty. The most recent `syslog` file becomes `syslog.0`, the next most recent becomes `syslog.1`, and so on. Eight previous `syslog` log files are kept.

The `/etc/logadm.conf` file also contains time stamps of when the last log rotation occurred.

You can use the `logadm` command to customize system logging and to add additional logging in the `/etc/logadm.conf` file as needed.

For example, to rotate the Apache access and error logs, use the following commands:

```
# logadm -w /var/apache/logs/access_log -s 100m
# logadm -w /var/apache/logs/error_log -s 10m
```

In this example, the Apache `access_log` file is rotated when it reaches 100 MB in size, with a `.0`, `.1`, (and so on) suffix, keeping 10 copies of the old `access_log` file. The `error_log` is rotated when it reaches 10 MB in size with the same suffixes and number of copies as the `access_log` file.

The `/etc/logadm.conf` entries for the preceding Apache log rotation examples look similar to the following:

```
# cat /etc/logadm.conf
.
.
```



```

/var/apache/logs/error_log -s 10m
/var/apache/logs/access_log -s 100m

```

For more information, see [logadm\(1M\)](#).

If you want to grant non-root users the privilege of maintaining log files, you can grant Log Management as a rights profile for that user. You can make that grant by using `-P` option with either the `useradd` command for new users or the `usermod` command for an existing user. For instructions, see the `useradd(1M)` and the `usermod(1M)` man pages.

Customizing System Message Logging

You can capture additional error messages that are generated by various system processes by modifying the `/etc/syslog.conf` file. By default, the `/etc/syslog.conf` file directs many system process messages to the `/var/adm/messages` files. Crash and boot messages are stored here as well. To view `/var/adm` messages, see [“How to View System Messages” on page 39](#).

The `/etc/syslog.conf` file has two columns separated by tabs:

facility.level ... action

facility.level A *facility* or system source of the message or condition. May be a comma-separated listed of facilities. Facility values are listed in [Table 5-1](#). A *level*, indicates the severity or priority of the condition being logged. Priority levels are listed in [Table 5-2](#).

Do not put two entries for the same facility on the same line, if the entries are for different priorities. Putting a priority in the syslog file indicates that all messages of that at least that priority are logged, with the last message taking precedence. For a given facility and level, `syslogd` matches all messages for that level and all higher levels.

action The action field indicates where the messages are forwarded.

The following example shows sample lines from a default `/etc/syslog.conf` file.

```

user.err                            /dev/sysmsg
user.err                            /var/adm/messages
user.alert                          `root, operator'
user.emerg                          *

```

This means the following user messages are automatically logged:

- User errors are printed to the console and also are logged to the `/var/adm/messages` file.

- User messages requiring immediate action (alert) are sent to the root and operator users.
- User emergency messages are sent to individual users.

Note - Placing entries on separate lines might cause messages to be logged out of order if a log target is specified more than once in the `/etc/syslog.conf` file. Note that you can specify multiple selectors in a single line entry, each separated by a semicolon.

The most common error condition sources are shown in the following table. The most common priorities are shown in [Table 5-2](#) in order of severity.

TABLE 5-1 Source Facilities for `syslog.conf` Messages

Source	Description
kern	The kernel
auth	Authentication
daemon	All daemons
mail	Mail system
lp	Spooling system
user	User processes

Note - The number of `syslog` facilities that can be activated in the `/etc/syslog.conf` file is unlimited.

TABLE 5-2 Priority Levels for `syslog.conf` Messages

Priority	Description
emerg	System emergencies
alert	Errors requiring immediate correction
crit	Critical errors
err	Other errors
info	Informational messages
debug	Output used for debugging
none	This setting doesn't log output

▼ How to Customize System Message Logging

1. **Assume the root role or a role that has the `solaris.admin.edit/etc/syslog.conf` authorization assigned to it.**

See “Using Your Assigned Administrative Rights” in “Securing Users and Processes in Oracle Solaris 11.2”.

2. **Use the `pfedit` command to edit the `/etc/syslog.conf` file, adding or changing message sources, priorities, and message locations according to the syntax described in [syslog.conf\(4\)](#).**

```
$ pfedit /etc/syslog.conf
```

3. **Save the changes.**

Example 5-2 Customizing System Message Logging

This sample `/etc/syslog.conf` `user.emerg` facility sends user emergency messages to root and individual users.

```
user.emerg                                `root, *'
```

Enabling Remote Console Messaging

The following new console features improve your ability to troubleshoot remote systems:

- The `consadm` command enables you to select a serial device as an *auxiliary* (or remote) console. Using the `consadm` command, a system administrator can configure one or more serial ports to display redirected console messages and to host `sulogin` sessions when the system transitions between run levels. This feature enables you to dial in to a serial port with a modem to monitor console messages and participate in `init` state transitions. For more information, see [sulogin\(1M\)](#) and the step-by-step procedures that follow.

While you can log in to a system using a port configured as an auxiliary console, it is primarily an output device displaying information that is also displayed on the default console. If boot scripts or other applications read and write to and from the default console, the write output displays on all the auxiliary consoles, but the input is only read from the default console. For more information about using the `consadm` command during an interactive login session, see “Guidelines for Using the `consadm` Command During an Interactive Login Session” on page 45.

- Console output now consists of kernel and `syslog` messages written to a new pseudo device, `/dev/sysmsg`. In addition, `rc` script startup messages are written to `/dev/msglog`. Previously, all of these messages were written to `/dev/console`.

Scripts that direct console output to `/dev/console` need to be changed to `/dev/msglog` if you want to see script messages displayed on the auxiliary consoles. Programs referencing `/dev/console` should be explicitly modified to use `syslog()` or `strlog()` if you want messages to be redirected to an auxiliary device.

- The `consadm` command runs a daemon to monitor auxiliary console devices. Any display device designated as an auxiliary console that disconnects, hangs up or loses carrier, is removed from the auxiliary console device list and is no longer active. Enabling one or more auxiliary consoles does not disable message display on the default console; messages continue to display on `/dev/console`.

Using Auxiliary Console Messaging During Run Level Transitions

Keep the following in mind when using auxiliary console messaging during run level transitions:

- Input cannot come from an auxiliary console if user input is expected for an `rc` script that is run when a system is booting. The input must come from the default console.
- The `sulogin` program, invoked by `init` to prompt for the root password when transitioning between run levels, has been modified to send the root password prompt to each auxiliary device in addition to the default console device.
- The user should never directly invoke `sulogin`. The user must have the `solaris.system.maintenance` authorization to use this utility.
- When the system is in single-user mode and one or more auxiliary consoles are enabled using the `consadm` command, a console login session runs on the first device to supply the correct root password to the `sulogin` prompt. When the correct password is received from a console device, `sulogin` disables input from all other console devices.
- A message is displayed on the default console and the other auxiliary consoles when one of the consoles assumes single-user privileges. This message indicates which device has become the console by accepting a correct root password. If there is a loss of carrier on the auxiliary console running the single-user shell, one of two actions might occur:
 - If the auxiliary console represents a system at run level 1, the system proceeds to the default run level.
 - If the auxiliary console represents a system at run level S, the system displays the `ENTER RUN LEVEL (0-6, s or S):` message on the device where the `init s` or `shutdown` command had been entered from the shell. If there isn't any carrier on that device either, you will have to reestablish carrier and enter the correct run level. The `init` or `shutdown` command will not re-display the run-level prompt.
- If you are logged in to a system using a serial port, and an `init` or `shutdown` command is issued to transition to another run level, the login session is lost whether this device is the auxiliary console or not. This situation is identical to releases without auxiliary console capabilities.

- Once a device is selected as an auxiliary console using the `consadm` command, it remains the auxiliary console until the system is rebooted or the auxiliary console is unselected. However, the `consadm` command includes an option to set a device as the auxiliary console across system reboots. (See the following procedure for step-by-step instructions.)

Guidelines for Using the `consadm` Command During an Interactive Login Session

If you want to run an interactive login session by logging in to a system using a terminal that is connected to a serial port, and then using the `consadm` command to see the console messages from the terminal, note the following behavior:

- If you use the terminal for an interactive login session while the auxiliary console is active, the console messages are sent to the `/dev/sysmsg` or `/dev/msglog` devices.
- While you issue commands on the terminal, input goes to your interactive session and not to the default console (`/dev/console`).
- If you run the `init` command to change run levels, the remote console software kills your interactive session and runs the `sulogin` program. At this point, input is accepted only from the terminal and is treated like it's coming from a console device. This allows you to enter your password to the `sulogin` program as described in [“Using Auxiliary Console Messaging During Run Level Transitions” on page 44](#).

Then, if you enter the correct password on the (auxiliary) terminal, the auxiliary console runs an interactive `sulogin` session, locks out the default console and any competing auxiliary console. This means the terminal essentially functions as the system console.

- From here you can change to run level 3 or go to another run level. If you change run levels, `sulogin` runs again on all console devices. If you exit or specify that the system should come up to run level 3, then all auxiliary consoles lose their ability to provide input. They revert to being display devices for console messages.

As the system is coming up, you must provide information to `rc` scripts on the default console device. After the system comes back up, the `login` program runs on the serial ports and you can log back into another interactive session. If you've designated the device to be an auxiliary console, you will continue to get console messages on your terminal, but all input from the terminal goes to your interactive session.

▼ How to Enable an Auxiliary (Remote) Console

The `consadm` daemon does not start monitoring the port until after you add the auxiliary console with the `consadm` command. As a security feature, console messages are only redirected until carrier drops, or the auxiliary console device is unselected. This means carrier must be established on the port before you can successfully use the `consadm` command.

For more information about enabling an auxiliary console, see the [consadm\(1m\)](#) man page.

1. Log in to the system and assume the root role.

See “Using Your Assigned Administrative Rights” in “Securing Users and Processes in Oracle Solaris 11.2”.

2. Enable the auxiliary console.

```
# consadm -a devicename
```

3. Verify that the current connection is the auxiliary console.

```
# consadm
```

Example 5-3 Enabling an Auxiliary (Remote) Console

```
# consadm -a /dev/term/a
# consadm
/dev/term/a
```

▼ How to Display a List of Auxiliary Consoles

1. Log in to the system and assume the root role.

See “Using Your Assigned Administrative Rights” in “Securing Users and Processes in Oracle Solaris 11.2”.

2. Select one of the following steps:

a. Display the list of auxiliary consoles.

```
# consadm
/dev/term/a
```

b. Display the list of persistent auxiliary consoles.

```
# consadm -p
/dev/term/b
```

▼ How to Enable an Auxiliary (Remote) Console Across System Reboots

1. Log in to the system and assume the root role.

See [“Using Your Assigned Administrative Rights”](#) in [“Securing Users and Processes in Oracle Solaris 11.2”](#).

2. Enable the auxiliary console across system reboots.

```
# consadm -a -p devicename
```

This adds the device to the list of persistent auxiliary consoles.

3. Verify that the device has been added to the list of persistent auxiliary consoles.

```
# consadm
```

Example 5-4 Enabling an Auxiliary (Remote) Console Across System Reboots

```
# consadm -a -p /dev/term/a
# consadm
/dev/term/a
```

▼ How to Disable an Auxiliary (Remote) Console

1. Log in to the system and assume the root role.

See [“Using Your Assigned Administrative Rights”](#) in [“Securing Users and Processes in Oracle Solaris 11.2”](#).

2. Select one of the following steps:

a. Disable the auxiliary console.

```
# consadm -d devicename
```

or

b. Disable the auxiliary console and remove it from the list of persistent auxiliary consoles.

```
# consadm -p -d devicename
```

3. Verify that the auxiliary console has been disabled.

```
# consadm
```

Example 5-5 Disabling an Auxiliary (Remote) Console

```
# consadm -d /dev/term/a
# consadm
```


Index

A

alert message priority (for `syslogd`), 42
auxiliary (remote) console, 43

B

booting
 displaying messages generated during, 39, 39
 troubleshooting, 20
 system hangs, 20

C

Command not found error message, 24
`consadm` command, 45
 disabling an auxiliary console, 47
 displaying list of auxiliary consoles, 46
 enabling an auxiliary console, 45
 across system reboots, 46
console
 auxiliary
 enabling across system reboots, 46
core dump configuration
 displaying with `coreadm`, 32
core file name pattern
 setting with `coreadm`, 32
core files
 administering, 31
 examining with `proc` tools, 35
 managing with `coreadm`, 29
`coreadm` command, 29
 displaying core dump configuration, 32
 managing core files, 29
 setting a core file name pattern, 32
crash dump

 disable or enable saving, 13
 displaying current configuration, 10
 examining crash dump information, 15
 files, 8
 changes, 7, 8
 modifying the configuration, 11
 restructured files, 7
 saved in `/var/crash` directory, 7
 saving data with full directory, 18
crashes, 41
 customer service and, 14
 displaying system information generated by, 16, 39
 examining crash dumps, 15, 16
 overview, 8, 9
 procedure following, 14
 rebooting fails after, 19
 saving crash dump information, 8
 saving other system information, 39
 troubleshooting, 10
`crontab` command
 `/var/adm` maintenance and, 39, 39
customer service
 sending crash information, 14
customizing
 system message logging, 41, 43

D

disabling
 an auxiliary console with the `consadm`
 command, 47
displaying
 booting messages, 39, 39
 core dump configuration with `coreadm`, 32
 crash information, 16, 39
`dmesg` command, 39, 39

dumpadm command, 9

E

enabling

- an auxiliary console with consadm command, 45
- auxiliary console across system reboots, 46

error messages

- crash messages, 39
- crash related, 39
- customizing logging of, 41, 41
- log file for, 14, 39
- priorities for, 42, 42
- sources of, 41, 41
- specifying storage location for, 39, 41, 41

/etc/syslog.conf file, 41, 41

examining a core file

- with proc tools, 35

F

file or group ownership

- solving file access problems, 26

files

- for setting search path, 25

G

global core file path

- setting with coreadm, 30

M

mdb utility, 15, 15, 16

messages file, 14, 41

messages.n file, 39

N

networks

- recognizing access problems, 27

O

Ops Center, 18

P

panic messages, 39

per-process core file path

- setting with coreadm, 30

proc tools

- examining a core file, 35

process failures

- troubleshooting, 29

R

rebooting

- fails after crash, 19

recognizing network access problems, 27

rsyslog service, 37

S

savecore command, 9

- changing directories, 18

saving data when crash dump directory is full, 18

search path

- files for setting, 25

setting

- a core file name pattern with coreadm, 32

syslog.conf file, 41, 41

syslogd daemon, 39

system crashes *See* crashes

system messages

- customizing logging, 41, 43
- shows file system is full, 23
- specifying storage location for, 39

T

technical support

- sending crash information, 14

U

UNIX systems (crash information), 8

/usr/adm/messages file, 14

/usr/bin/mdb utility, 15

V

/var/adm/messages file, 14, 41

/var/adm/messages.*n* file, 39

W

Watchdog reset ! message, 39

