Oracle® Solaris 11.3 でのシステム管理のトラブルシューティング

Part No: E62621
2016 年 11 月

Part No: E62621

ドキュメントのアクセシビリティについて

オラクルのアクセシビリティについての詳細情報は、Oracle Accessibility ProgramのWeb サイト(http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc)を参照してください。

Oracle Supportへのアクセス

サポートをご契約のお客様には、My Oracle Supportを通して電子支援サービスを提供しています。詳細情報は(http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info) か、聴覚に障害のあるお客様は (http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs)を参照してください。

目次

このドキュメントの使用法

- **概要** – SPARC および x86 の両方のプラットフォームにおける Oracle Solaris のトラブルシューティングの問題について説明します。
- **対象読者** – Oracle Solaris 11 リリースを使用しているシステム管理者
- **前提知識** – UNIX システムの管理経験

製品ドキュメントライブラリ

この製品および関連製品のドキュメントとリソースは http://www.oracle.com/pls/topic/lookup?ctx=E62101-01 で入手可能です。

フィードバック

このドキュメントに関するフィードバックを http://www.oracle.com/goto/docfeedback からお聞かせください。

♦ ♦ ♦ 第 1 章

1 システムクラッシュのトラブルシューティング

この章では、Oracle Solaris OS のシステムクラッシュへの対処およびトラブルシューティングの方法について説明します。

この章では次の内容について説明します。

Oracle Solaris 11.3 でのリブート後まで遅延するようになったクラッシュファイル

Oracle Solaris 11.3 以降では、システムがクラッシュした場合、システムがリブートされるまでクラッシュダンプファイルがメモリーに保存されることがあります。システムのリブート中に、クラッシュダンプファイルはダンプ構成で定義されたファイルシステムにメモリーから抽出されます。これらのファイルが書き込まれたあと、システムは通常のマルチユーザー構成に自動的にリブートされます。このプロセスは遅延ダンプと呼ばれます。遅延ダンプを利用すると、システムはカーネルパニック後に短時間で実行状態に戻ることができます。また遅延ダンプは、ローカルディスクを搭載していない場合がある SPARC M7 シリーズサーバーなどのシステムに特にメリットがあります。

注意 - 最初のリブートで遅延ダンプが実行されます。x86 システムでは、メモリーのフラグメンテーションが原因で発生する可能性があるパフォーマンスの問題を排除するために、2 回目のリブートが自動的に行われることがあります。このプロセスは中断しないでください。システムは、2 回目のリブート後に使用する準備ができます。

遅延ダンプのシステムメッセージ

遅延ダンププロセスには、次のシステムメッセージが含まれています。

- システムがメモリーに対して遅延ダンプを実行しているときに、「`Preserving kernel image in RAM`」メッセージが進行状況バーとともに表示されます。
- システムがパニック後にリブートすると、「`Verified previous kernel image`」とその後に「`Reconciling deferred dump`」メッセージが進行状況バーとともに表示されます。
- `savecore` がクラッシュダンプファイルをファイルシステムに保存している間に、「`Extracting crash dump`」メッセージが進行状況バーとともに表示されます。

遅延ダンプのサポート

x86 システムでは、遅延ダンプは、Oracle Solaris でデフォルトの高速リブート機能とともに動作します。そのため、x86 システムの遅延ダンプは、高速リブートをサポートするプラットフォームでのみ機能できます。高速リブートの使用の詳細は、『Oracle Solaris 11.3 システムのブートとシャットダウン』 の 「リブートプロセスの高速化」を参照してください。

遅延ダンプは、少なくとも `Sys Firmware 8.7` または `Sys Firmware 9.4` 以降をサポートし、システムがリブートするまでクラッシュダンプファイルを保存するために十分なメモリーを持つ SPARC T4 システムではデフォルトで有効になっています。

遅延ダンプを使用した場合でも、構成されているダンプデバイスにクラッシュダンプ情報を格納するために十分な領域があることを確認する必要があります。これは、`savecore -L` を使用したライブシステムクラッシュダンプが必要になった場合に備えるためです。また、次の状況では、可能であればクラッシュダンプ情報が (メモリー内に保持されることなく) ダンプデバイスに自動的に書き込まれます。

- そのプラットフォームでは高速リブートがサポートされないか、高速リブートが失敗する。
- `savecore` ユーティリティーが無効になっているか失敗する。
- システムがカーネルゾーンを使用している。

システムのクラッシュの準備と対応の詳細は、この章と `dumpadm(1M)` のマニュアルページを参照してください。

システムクラッシュについて

ハードウェアの障害、入出力の問題、ソフトウェアエラーなどが原因でシステムがクラッシュすることがあります。システムがクラッシュする場合、コンソールにエラー

メッセージが表示され、その物理メモリーのコピーが RAM に保存されるか、物理メモリーのコピーがダンプデバイスに書き込まれます。その後、システムは自動的にリブートします。システムがリブートすると、savecore コマンドが実行され、メモリーまたはダンプデバイスのデータを取り出して、保存されたクラッシュダンプファイルを savecore ディレクトリに書き込みます。これらの保存されたファイルは、問題を診断する上で貴重な情報となります。

注記 - 「クラッシュダンプ」の用語は、クラッシュダンプファイルのセット、ファイルの配置場所、これらのファイルの編成および書式設定の方法を含む、このプロセスの全体の結果を指します。

システムクラッシュダンプファイル

savecore コマンドは、システムがクラッシュすると自動的に実行され、メモリーまたはダンプデバイスからクラッシュダンプ情報を取得し、その情報をファイルのセットに書き込みます。その後、savecore コマンドを同じシステムまたは別のシステムで呼び出し、クラッシュダンプファイルを展開または圧縮できます。

注記 -クラッシュダンプファイルはコアファイルと混同されることがあります。コアファイルは、アプリケーションが異常終了したときに書き込まれるユーザーアプリケーションのイメージです。

クラッシュダンプファイルは、あらかじめ決められたディレクトリに保存され、デフォルトは /var/crash/ です。クラッシュダンプファイルの保存はデフォルトで有効です。

再構築されたファイル

Oracle Solaris 11.2 リリース以降、カーネルクラッシュダンプファイルの内容は、その内容に基づいて複数の新しいファイルに分割されます。この方法では、構成の精度を高くできるため、ファイルにより簡単にアクセスして調べることができます。

クラッシュダンプ情報は、vmdump-*section*.*n* ファイルのセットに書き込まれます。セクション値は、特定の種類のダンプ情報を含むファイルセクションの名前です。*n* 値は、savecore が実行され、クラッシュダンプをコピーし、新しいクラッシュダンプがダンプデバイスに見つかるたびに増分される整数です。設定可能なファイルは次のとおりです。

- vmdump-proc.*n* – プロセスページが圧縮されたダンプファイル

- `vmdump-zfs.`*n*– ZFS メタデータが圧縮されたダンプファイル
- `vmdump-other.`*n* – ほかのページがあるダンプファイル

カーネルクラッシュダンプは、以前は `vmdump.n`、`unix.n`、および `vmcore.n` に格納されていました。

`vmdump.`*n* および `vmcore` ファイルにはカーネルページ (メタデータとデータ) がそれぞれ圧縮形式または非圧縮形式で格納されます。

詳細は、`dumpadm(1M)` および `savecore(1M)` のマニュアルページを参照してください。

dumpadm および savecore コマンド

`dumpadm` および `savecore` ユーティリティーは、次のようにクラッシュダンプの作成を構成および管理します。

1. `dumpadm` コマンドは、システム起動時に `svc:/system/dumpadm:default` サービスによって呼び出され、クラッシュダンプパラメータを構成します。`/dev/dump` インタフェースを通してダンプデバイスとダンプ内容を初期化します。
2. ダンプ構成の完了後に、`savecore` を呼び出して、RAM またはダンプデバイスにクラッシュダンプがあるかどうかを調べたり、クラッシュダンプディレクトリにある `minfree` ファイルの内容を確認したりします。`savecore` コマンドで生成されるシステムクラッシュダンプファイルは、デフォルトで保存されます。
3. ダンプデータは、圧縮した形式でダンプデバイスに格納されます。カーネルのクラッシュダンプイメージは 128G バイトを超える場合があります。データを圧縮することにより、ダンプが速くなり、ダンプデバイスのディスク領域も少なくてすみます。
4. デフォルトでは、インストールプログラムは専用のダンプサービスを作成します。システムは、`savecore` コマンドが完了するまで待ってから、次の段階に進みます。大容量のメモリーを搭載したシステムでは、`savecore` が完了する前にシステムが使用可能になります。

`savecore -L` コマンドを使用すると、管理者は、Oracle Solaris OS を現在実行中のシステムのクラッシュダンプを取得できます。たとえば、パフォーマンスに一時的な問題が発生しているときやサービスが停止しているときなどにメモリーのイメージをとって、実行中のシステムをトラブルシューティングするのに使用します。このメモリーのイメージは、システムの実行中に発生する変更が原因で不正確です。システムが稼働中で、一部のコマンドをまだ実行できる場合は、`savecore -L` コマンドを実行してシステムのイメージをダンプデバイスに保存し、クラッシュダンプファイルをただちに `savecore` ディレクトリに書き込むことができます。`savecore -L` コマンドは、専用のダンプデバイスを構成した場合にかぎり使用できます。

システムのクラッシュダンプの構成

このセクションでは、システムのクラッシュダンプの管理手順のタスクについて説明します。

システムクラッシュ情報を処理する場合には、次の点に注意してください。

- システムクラッシュ情報にアクセスして管理するには、root 役割になる必要があります。『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』 の「割り当てられている管理権利の使用」を参照してください。
- Oracle Solaris 11.3 リリース以降では、システムがクラッシュした場合、システムがリブートするまでその物理メモリーのコピーを RAM に保持してから、ファイルシステムに書き込むことができます。このプロセスは遅延ダンプと呼ばれます。遅延ダンプが可能ではない場合、物理メモリーはクラッシュ中にダンプデバイスに書き込まれます。『Oracle Solaris 11.3 でのシステム管理のトラブルシューティング』の 「Oracle Solaris 11.3 でのリブート後まで遅延するようになったクラッシュファイル」を参照してください。
- システム上でシステムクラッシュダンプファイルを保存するオプションを無効にしないでください。システムクラッシュファイルにより、システムクラッシュの原因を判断する非常に有効な方法が提供されます。
- スワップ領域とダンプ領域には専用の ZFS ボリュームが使用されます。手順については、『Oracle Solaris 11.3 での ZFS ファイルシステムの管理』 の 「ZFS スワップおよびダンプデバイスの管理」を参照してください。

現在のクラッシュダンプ構成の表示

現在のクラッシュダンプ構成を表示するには、root 役割になり、引数なしの dumpadm コマンドを発行します。

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

この出力例は、次の構成を示しています。

- ダンプの内容は、ZFS メタデータが含まれるカーネルメモリーページです。
- カーネルメモリーは、リブートまでメモリーに保持されるか、専用のダンプデバイス /dev/zvol/dsk/rpool/dump にダンプされます。
- システムクラッシュダンプファイルは /var/crash/ ディレクトリに保存される。
- システムクラッシュダンプファイルの保存は有効に設定されている

- クラッシュダンプファイルは圧縮形式で保存されます。

クラッシュダンプの構成の変更

クラッシュダンプ構成を変更するには、root 役割になり、dumpadm コマンドを使用します。

dumpadm コマンドの構文は次のとおりです。

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

-c *content-type*
: ダンプするデータの種類を指定する。使用可能な値は次のとおりです。
  - カーネルメモリーページのみをダンプする kernel
  - すべてのメモリーページをダンプする all
  - クラッシュの発生時にスレッドが実行されていたプロセスのカーネルメモリーとメモリーページをダンプする curproc
  - カーネルメモリーページとすべてのプロセスページをダンプする allproc
  - ZFS メタデータを格納するカーネルページをダンプする zfs

  デフォルトのダンプの内容は、ZFS メタデータが含まれるカーネルメモリーです。例:

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

-d *dump-device*
: システムがクラッシュしたときに、ダンプデータを一時的に保存するデバイスを指定します。デフォルトのダンプデバイスはプライマリダンプデバイスです。ダンプデバイスがスワップ領域でない場合は、savecore がバックグラウンドで実行されるため、ブートプロセスの速度が上がる

注記 - 遅延ダンプは、ダンプデバイスを使用する代わりに、ダンプデータを RAM に一時的に格納します。ただし、遅延ダンプが失敗した場合、遅延ダンプを実行できない場合、またはクラッシュダンプファイルを直接ファイルシステムに書き込むために十分な領域が savecore ディレクトリにない場合に備えて、ダンプデバイスを指定してください。

-e
: 圧縮されたクラッシュダンプの格納に必要なディスク容量の概算を出力します。この値は、現在の構成および現在実行中のシステムを使用して算出されます。

-m *min*k | *min*m | *min*%	現在の savecore ディレクトリに minfree ファイルを作成することにより、クラッシュダンプファイルを保存する最小限の空き容量を指定します。このパラメータは K バイト (mink)、M バイト (minm)、またはファイルシステムサイズのパーセント (min%) で指定できます。最小の空き容量を構成しないと、デフォルトで 1M バイトになります。 savecore コマンドは、クラッシュダンプファイルを書き込む前にこのファイルを調べる。クラッシュダンプファイルを書き込むと、サイズにより空き容量が minfree しきい値より少なくなる場合、ダンプファイルは書き込まれず、エラーメッセージが記録されます。このシナリオからの回復については、23 ページの「クラッシュダンプファイルディレクトリがいっぱいになった場合のデータの保存」を参照してください。
-n	システムがリブートするときに、savecore を自動的に実行しないように指定します。システムはクラッシュダンプイメージをメモリーに保存しようとするため、このダンプ構成は推奨されません。
-p	マシンが読める出力を生成します。
-s *savecore-dir*	クラッシュダンプファイルを保存する別のディレクトリを指定する。Oracle Solaris 11 では、デフォルトのディレクトリは /var/crash です。
-u	/etc/dumpadm.conf ファイルの内容に基づいてカーネルダンプ構成を強制的に更新します。
-y	リブート時に自動的に savecore コマンドを実行するようにダンプ構成を変更します。このダンプ設定では、このコマンドの自動実行がデフォルトです。
-z on | off	リブート時の savecore コマンドの動作を制御するために、ダンプ構成を変更します。on 設定では、コアファイルを圧縮形式で保存できます。off 設定では、クラッシュダンプファイルを自動的に圧縮解除します。クラッシュダンプファイルはサイズが非常に大きくなる場合があり、圧縮した形式で保存すれば必要なシステム領域が小さくなるため、デフォルトは on です。

例 1 クラッシュダンプ構成を変更する

次の例は、すべてのメモリーを専用のダンプデバイス /dev/zvol/dsk/rpool/dump にダンプし、クラッシュダンプファイルを保存したあとに残っていなければならない最小空き容量は、ファイルシステム容量の 10% です。

```
# dumpadm
      Dump content: kernel with ZFS metadata
       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 = 5935131KB)
  Savecore enabled: yes
   Save compressed: on

# dumpadm -n
       Dump content: all pages
       Dump device: /dev/zvol/dsk/rpool/dump (dedicated)
Savecore directory: /var/crash (minfree = 5935131KB)
  Savecore enabled: no
   Save compressed: on

# dumpadm -y
       Dump content: all pages
       Dump device: /dev/zvol/dsk/rpool/dump(dedicated)
Savecore directory: /var/crash (minfree = 5935131KB)
      Savecore enabled: yes
        Save compressed: on
```

例 2 クラッシュダンプの保存を無効にする

次の例は、システムでのクラッシュダンプの保存を無効にします。

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

注意 - Oracle Solaris では、クラッシュダンプの保存を無効にしないことを強く推奨しています。クラッシュダンプは、システムのクラッシュの原因を判断するための重要な方法です。

例 3 クラッシュダンプの保存を有効にする

次の例は、システムでのクラッシュダンプの保存を有効化する方法を示しています。

```
# 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
```

システムクラッシュ後のトラブルシューティング

Oracle Solaris システムがクラッシュした場合は、一般的なシステム情報や、クラッシュダンプファイルの具体的な情報など、できるだけ多くの情報を提供してください。

既存のクラッシュダンプをトラブルシューティングするには、次を参照してください。

- 19 ページの「システムがクラッシュした場合の対処方法」
- 20 ページの「クラッシュダンプ情報を検査する」
- 22 ページの「システムクラッシュのトラブルシューティングのチェックリスト」
- 23 ページの「クラッシュダンプファイルディレクトリがいっぱいになった場合のデータの保存」

システムがクラッシュした場合の対処方法

システムクラッシュが発生した場合:

1. システムのコンソールメッセージを書き取ります。

   システムがクラッシュした場合は、システムを再稼働させるのを最優先に考えがちです。しかし、システムをリブートする前に、コンソール画面のメッセージを確認してください。 これらのメッセージは、クラッシュした原因を解明するのに役立ちます。システムが自動的にリブートして、コンソールメッセージが画面から消えた場合でも、`/var/adm/messages` システムエラーログファイルを表示すれば、メッセージをチェックできます。システムエラーログファイルの詳細は、45 ページの「システムメッセージとロギングの表示」を参照してください。
2. クラッシュが頻繁に発生し、その原因を特定できない場合は、システムコンソールまたは `/var/adm/messages` ファイルからすべての情報を収集して、カスタマサポートの担当者に問い合わせます。

   サービスプロバイダ用に収集するトラブルシューティング情報の完全なリストについては、22 ページの「システムクラッシュのトラブルシューティングのチェックリスト」を参照してください。
3. また、システムクラッシュダンプがシステムのクラッシュ後に生成されたかどうかを確認してください。

注意 - 重要なシステムクラッシュ情報は、カスタマサポート担当者に送信するまでは削除しないでください。

4. システムクラッシュ後にブートが失敗する場合は、『Oracle Solaris 11.3 システムのブートとシャットダウン』 の 「復旧目的のシステムのシャットダウンおよびブート」を参照してください。

クラッシュダンプ情報を検査する

制御構造体、アクティブなテーブル、動作中またはクラッシュしたシステムカーネルのメモリーのイメージなど、カーネルの動作状況についての情報を調べるには、mdb ユーティリティーを使用します。

注記 - 次の手順は、mdb ユーティリティーの使用方法の制限された例にすぎません。mdb ユーティリティーを完全に使いこなすには、カーネルについての詳細な知識が必要ですが、このマニュアルでは説明を省きます。このユーティリティーの使用の詳細は、『Oracle Solaris Modular Debugger Guide』および mdb(1) マニュアルページを参照してください。

▼ クラッシュダンプ情報を検査する方法

1. **root 役割になります。**

   『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』 の 「割り当てられている管理権利の使用」を参照してください。

2. **クラッシュダンプ情報が保存されているディレクトリに移動します。**

   たとえば、デフォルトディレクトリに変更するには:

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

   クラッシュダンプの場所が不明な場合は、dumpadm コマンドを使用して、システムがカーネルクラッシュダンプファイルを格納するように構成されている場所を特定します。次のサンプル出力は、デフォルトのディレクトリの場所が変更されていないことを示しています。

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

3. **クラッシュダンプを検査するには、モジュラデバッガユーティリティー (mdb) を使用します。**

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

-k	オペレーティングシステムのクラッシュダンプファイルの場合のカーネルデバッグモードを指定します。
crashdump-file	オペレーティングシステムのクラッシュダンプファイルを指定します。

例:

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

コマンドは次のように指定することもできます。

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

4. **システムクラッシュのステータスを表示します。**

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

カーネルクラッシュダンプを検査するときに `::system dcmd` コマンドを使用するには、コアファイルがカーネルクラッシュダンプである必要があり、`mdb` ユーティリティーの起動時に `-k` オプションを指定しておく必要があります。

5. **`mdb` ユーティリティーを終了します。**

```
> $quit
```

例 4 クラッシュダンプ情報を検査する

次の例は、`mdb` ユーティリティーの出力例を示します。この例にはシステムのシステム情報が含まれシステムの `/etc/system` ファイルに設定されているチューニング可能値を識別します。

```
# 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 ncsize=0x4e20 [0t20000]
set pt_cnt=0x400 [0t1024]
> $q
```

システムクラッシュのトラブルシューティングのチェックリスト

システム問題を特定し、Oracle サポートプロバイダへの問い合わせを準備するには、次のチェックリストの質問に回答します。

項目	ユーザーのデータ
システムクラッシュダンプがあるか	
オペレーティングシステムのリリースと適切なソフトウェアアプリケーションのリリースレベルを確認する ユーザーは、`more/etc/release` または `pkg info entire` に関する情報を受信します。	
システムのハードウェアを確認する。	
SPARC および x86 システムの `prtdiag` 出力を含めます。通常すべてのシステムの保守で要求されるエクスプローラ出力を含めます。	
パッチはインストールされているか。 Oracle Solaris 11 では `showrev -p` の出力は表示されないため、インストールされている SRU & IDR に関する情報を代わりに含めます。	
問題を再現できるか 困難な問題をデバッグするには多くの場合、再現可能なテストケースが重要です。購入先では、特殊な計測機構を使用してカーネルを構築して問題を再現し、バグを引き起こし、診断、および修正できる	
Sun 以外のドライバをシステムで使用しているか ドライバは、カーネルと同じアドレス空間で実行されます。すべてが同じ特権を持っていると、バグがある場合にシステムがクラッシュする原因となる可能性があります。	
クラッシュ直前のシステムの動作は 新しいストレステストの実行や通常より高い負荷の発生などの特殊な状況では、クラッシュが発生する可能性があります。	
システムのクラッシュ直前に、異常なコンソールメッセージが表示されたか システムがクラッシュする前には、なんらかの兆候を示していることがある。この情報は重要	
`/etc/system` ファイルにパラメータを追加したか	

項目	ユーザーのデータ
Oracle Solaris 11.2 以降では、`/etc/system` は `/etc/system.d/*` で補足されるようになりました。	
問題は最近発生するようになったか その場合、問題の開始はシステムの変更と一致しましたか。例: 新しいドライバ、新しいソフトウェア、異なる作業負荷、CPU のアップグレード、またはメモリーのアップグレード。	

クラッシュダンプファイルディレクトリがいっぱいになった場合のデータの保存

- システムがクラッシュし、`savecore` ディレクトリの容量がない場合に、重要なシステムクラッシュダンプ情報を保存するには、次のいずれかの方法を使用します。

  システムのリブート後、root役割としてログインします。サービスプロバイダに送信済みの既存のクラッシュダンプファイルを `savecore` ディレクトリから削除します。

  注記 - savecore ディレクトリは通常は `/var/crash` です。

- または、システムのリブート後、root 役割としてログインします。`savecore` コマンドを手作業で実行し、十分なディスク容量がある代替ディレクトリを指定します。

  ```
  # savecore directory
  ```

Oracle Enterprise Manager Ops Center を使用したインシデントの管理

大規模な配備内の物理および仮想オペレーティングシステム、サーバー、およびストレージデバイスのインシデントを管理する必要があります。単に個々のシステム内でインシデントをモニターするのではなく、Oracle Enterprise Manager Ops Center で使用可能な包括的なシステム管理ソリューションを使用できます。

Enterprise Manager Ops Center を使用すると、データセンターの一部が予想どおりに実行されていない場合に通知するアラートの設定、これらのインシデントレポートの管理、および修復が可能です。

詳細は、Oracle Solaris Enterprise Manager を参照してください。

♦ ♦ ♦ 第 2 章

2 システムハングまたはリブート失敗時のトラブルシューティング

この章では、システムのハングが発生した場合、またはシステムのリブート中に問題が発生した場合の対処について説明します。

ここには、次の情報が含まれています。

リブートが失敗した場合の対処

システムが完全にリブートしない場合、またはシステムがリブートしたが再度クラッシュした場合は、ソフトウェアまたはハードウェアの問題がシステムの正常なブートを妨害している可能性があります。

システムがブートしない原因	問題の解決方法
システムが /platform/`uname -m`/kernel/sparcv9/unix を見つけられない。	SPARC システムの PROM 内の boot-device 設定を変更します。デフォルトのブートデバイスの変更については、『Oracle Solaris 11.3 システムのブートとシャットダウン』 の 「ブート属性の表示と設定」を参照してください。
Oracle Solaris ブートアーカイブが破損したか、SMF ブートアーカイブサービスが失敗した。svcs -x コマンドを実行すると、エラーメッセージが表示される	プライマリブート環境のバックアップとなるセカンダリブート環境を作成します。プライマリブート環境をブートできなくなった場合は、バックアップのブート環境をブートします。あるいは、Live CD または USB メディアからブートすることもできます。
無効なエントリが /etc/passwd ファイル内に存在する。	無効な passwd ファイルからの回復については、『Oracle Solaris 11.3 システムのブートとシャットダウン』 の 「メディアからブートして、不明な

システムがブートしない原因	問題の解決方法
	root パスワードを解決する方法」を参照してください。
x86 ブートローダー (GRUB) が破損しているか、GRUB メニューが欠落しているか破損した。	破損した x86 ブートローダーまたは失われたか壊れた GRUB メニューからの回復については、『Oracle Solaris 11.3 システムのブートとシャットダウン』 の 「メディアからブートして、システムのブートを妨げている GRUB 構成の問題を解決する方法」を参照してください。
ディスクなどのデバイスにハードウェアの問題が存在する。	ハードウェアの接続を確認します。 ■ 装置が接続されていることを確認します。 ■ すべてのスイッチが適切に設定されていることを確認します。 ■ すべてのコネクタおよびケーブル (Ethernet ケーブルも含む) を検査します。 ■ これらの手順がすべて失敗した場合は、システムの電源を切り、10 秒 - 20 秒ほど待って、もう一度電源を投入します。

上記のリストで問題が解決できない場合は、ご購入先にお問い合わせください。

root パスワードを忘れた場合またはシステムをブートできない場合の対処方法

root パスワードを忘れた場合や、システムのブートを妨害する別の問題が発生した場合は、次を実行します。

1. システムを停止します。
2. 『Oracle Solaris 11.3 システムのブートとシャットダウン』 の 「メディアからブートして、不明な root パスワードを解決する方法」の指示に従います。
3. root パスワードが問題である場合は、/etc/shadow ファイルから root パスワードを削除します。
4. システムをリブートします。
5. ログインして root パスワードを設定します。

システムのハングが発生した場合の対処

ソフトウェアプロセスに問題がある場合、システムは完全にクラッシュせずに凍結、つまりハングすることがあります。ハングしたシステムから回復するには、次の手順に従ってください。

1. システムがウィンドウ環境を実行している場合は、次の推奨事項に従ってください。これらのリストで問題が解決できなかった場合は、手順 2 に進みます。
   - コマンドを入力しているウィンドウの中に、ポインタがあることを確認します。
   - 間違って Control-S キー (画面を凍結する) を押した場合は、Control-Q キーを押します。Control-S キーはウィンドウだけを凍結し、画面全体は凍結しません。ウィンドウが凍結している場合は、他のウィンドウを試します。
   - 可能であれば、ネットワーク上の他のシステムからリモートでログインします。`pgrep` コマンドを使用して、ハングしているプロセスを見つけます。ウィンドウシステムがハングしているように見える場合は、そのプロセスを特定して強制終了します。
2. Control-\ キーを押すことによって、強制的にプログラムの実行を終了し、(場合によっては) コアファイルを書き込みます。
3. Control-C キーを押すことによって、実行されている可能性があるプログラムを中断します。
4. リモートからログインして、システムをハングさせているプロセスを特定して強制終了します。
5. リモートからログインして、`root` 役割になってからシステムをリブートします。
6. システムがまだ応答しない場合は、強制的にクラッシュダンプしてリブートします。詳細は、『Oracle Solaris 11.3 システムのブートとシャットダウン』 の 「クラッシュダンプを強制してシステムをリブートする」を参照してください。
7. システムがまだ応答しない場合は、電源を切ってから数分待ち、もう一度電源を入れます。
8. システムがまったく応答しない場合は、ご購入先にお問い合わせください。

♦ ♦ ♦ 第 3 章

3 ファイルシステムの問題のトラブルシューティング

この章では、次のようなファイルシステムの問題の修正方法について説明します。

ファイルシステムが一杯になった場合の対処

ルート (/) ファイルシステムや他のファイルシステムが一杯になると、次のようなメッセージがコンソールウィンドウに表示されます。

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

ファイルシステムが一杯になる原因はいくつかあります。次のセクションでは、一杯になったファイルシステムを回復する方法をいくつか説明します。

大規模ファイルまたはディレクトリを作成したために、ファイルシステムが一杯になる

原因: だれかが誤ってファイルまたはディレクトリを間違った場所にコピーしたか、アプリケーションがクラッシュして大規模な `core` ファイルをファイルシステムに書き込みました。

解決策: ログインして `root` 役割になり、ファイルシステムで `ls -tl` コマンドを使用して、新しく作成された大きなファイルを特定して削除します。

システムのメモリーが不足したために、ZFS ファイルシステムがいっぱいになる

原因: ZFS ファイルシステムは、前のスナップショットにファイルを保存するときに変更を記録しようとするために、許可されるよりも多くのメモリーを使用します。

解決策: ZFS ファイルシステムからファイルを削除して、ディスク容量を解放します。詳細は、『Oracle Solaris 11.3 での ZFS ファイルシステムの管理』 の 「ZFS の領域の問題を解決する」 を参照してください。

システムのメモリーが不足したために、TMPFSファイルシステムが一杯になる

原因: TMPFS ファイルシステムが、許可されるより多くを書き込もうとしたか、現在のプロセスが多くのメモリーを使用しています。

解決策: `tmpfs` に関連するエラーメッセージから回復する方法については、`tmpfs(7FS)` のマニュアルページを参照してください。

コピーまたは復元後にファイルの ACL が消失した場合の対処

原因: ACL を持つファイルまたはディレクトリを `/tmp` ディレクトリにコピーしたり復元して、ACL 属性が失われました。`/tmp` ディレクトリは、通常、一時ファイルシステムとしてマウントされ、ACL などの UFS ファイルシステム属性はサポートしない

解決策: ファイルを代わりに `/var/tmp` ディレクトリにコピーするか復元します。

ファイルアクセスでの問題のトラブルシューティング

以前は使用できていたプログラム、ファイル、またはディレクトリにアクセスできないユーザーは多くの場合、システム管理者に問い合わせます。

このような問題が発生したときは、次の 3 点のいずれかを調べてください。

- ユーザーの検索が変更されているかどうか、または検索パス内のディレクトリが適切な順序になっていないかどうか。
- ファイルまたはディレクトリに適切なアクセス権や所有権があるかどうか。
- ネットワーク経由でアクセスするシステムの構成が変更されているかどうか。

この章では、これらの 3 点を確認する方法を簡単に説明して、可能な解決策を提案します。

検索パスの問題の解決

コマンドにアクセスしようとすると、無効なバージョンのコマンドを取得するか、メッセージ「コマンドが見つかりません」が表示されることがあります。

検索パスの問題を修正するには、コマンドが格納されているディレクトリのパス名を知る必要があります。コマンドのマニュアルページを調べて、標準の場所を確認します。

無効なバージョンのコマンドへのアクセス

間違ったバージョンのコマンドが見つかった場合、同じコマンド名が含まれているディレクトリが検索パス内に存在します。正しいディレクトリがあとで検索パスに表示されるか、まったく含まれていないことがあります。

▼ 検索パスの問題を診断して解決する方法

1. **使用しているコマンドのバージョンを判別します。**

   例:

   ```
   $ which acroread
   /usr/bin/acroread
   ```

2. **現在の検索パスを表示します。**

   ```
   $ echo $PATH
   ```

3. **現在の検索パスを調べて、正しいディレクトリが含まれているかどうか、および同じコマンド名が含まれているほかのディレクトリが正しいディレクトリの前に一覧表示されるかどうかを確認します。**

4. **ホームディレクトリ内の `.profile` ファイルに一覧表示されるパスで、正しいディレクトリを追加するか、パスの前半に正しいディレクトリを移動します。**

   パス名を区切るには、コロンを使用します。

5. **(オプション) 次のシステムのログイン前にコマンドを使用する必要がある場合、新しいパスをアクティブ化します。**

   ```
   $ . $HOME/.profile
   ```

6. **(オプション) 新しいパスをアクティブ化した場合、正しいパスからコマンドにアクセスできることを確認します。**

   ```
   $ which command
   ```

見つからないコマンドへのアクセス

エラーメッセージ「コマンド見つかりません」は、次のいずれかの理由で表示されます。

- そのコマンドはシステムでは使用できない
- コマンドのディレクトリが検索パスに存在しない

コマンドがシステムで使用できない場合は、システム管理者に連絡してください。

▼ パスに検索パスを含める方法

1. **現在の検索パスを表示して、コマンドが入っているディレクトリがユーザーのパス内に存在しない (あるいはパスのスペルが間違っている) ことを確認します。**

   ```
   echo $PATH
   ```

2. **コマンドのディレクトリを `$HOME/.profile` ファイル内の `PATH` エントリに追加します。**

   パス名を区切るには、コロンを使用します。

3. **新しいパスをアクティブ化します。**

   ```
   $ . $HOME/.profile
   ```

4. **コマンドの正しいパスが表示されるようになったことを確認します。**

   ```
   $ which command
   ```

ファイルとグループの所有権の変更

ファイルとディレクトリの所有権は、だれかが管理者としてファイルを編集したために、変更されることが頻繁にあります。新しいユーザーのホームディレクトリを作成するときには、必ず、そのユーザーをホームディレクトリ内のドット (.) ファイルの所有者にしてください。ユーザーをドット (.) ファイルの所有者にしなかった場合、そのユーザーは自分のホームディレクトリにファイルを作成できません。

アクセスに関する問題は、グループの所有権が変更されたとき、またはユーザーの属するグループが `/etc/group` データベースから削除されたときにも発生します。

アクセスに問題のあるファイルのアクセス権または所有権を変更する方法については、『Oracle Solaris 11.3 でのファイルのセキュリティー保護とファイル整合性の検証』 の 第 1 章,「ファイルアクセスの制御」を参照してください。

ファイルアクセスの問題を解決する

以前はアクセスできていたファイルまたはディレクトリにアクセスできない場合は、そのファイルまたはディレクトリのアクセス権または所有権が変更されていることがあります。

ネットワークアクセスで発生する問題の把握

リモートコピーコマンド `rcp` を使用してネットワーク上でファイルをコピーするときに問題が発生した場合、リモートシステム上のディレクトリやファイルは、アクセス権の設定によりアクセスが制限されている可能性があります。他に考えられる問題の原因は、リモートシステムとローカルシステムがアクセスを許可するように構成されていないことです。

ネットワークアクセスに伴う問題、および AutoFS を通じたシステムへのアクセスでの問題については、『Oracle Solaris 11.3 でのネットワークファイルシステムの管理』の 「NFS のトラブルシューティングの方法」を参照してください。

♦ ♦ ♦ 第 4 章

コアファイルを使用したプロセス障害への対処

この章では、プロセスに障害が発生した場合にシステムで生成されるコアファイルの仕様を設定する方法と、障害の発生後にこれらのコアファイルを調べる方法について説明します。次の情報が含まれます。

プロセス障害およびコアファイルについて

プロセスまたはアプリケーションが異常終了すると、システムでファイルのセットが自動的に生成されます。このプロセスを コアダンプと呼びます。作成されるファイルはコアファイルです。コアファイルとは、終了時のプロセスアドレス空間の内容と、プロセスの状態の追加情報のディスクコピーです。通常、コアファイルは、対応するアプリケーションのバグに起因するプロセスの異常終了後に生成されます。コアファイルには、プロセス障害の原因となった問題の診断に使用する重要な情報が提供されます。36 ページの「コアファイル作成のパラメータ」を参照してください。

実行中のシステム管理の一部として、`coreadm` コマンドを使用して、コアファイルの作成仕様を制御できます。たとえば、`coreadm` コマンドを使用して、プロセスコアファイルをすべて 1 つのシステムディレクトリに置き、問題を簡単に追跡できるようにシステムを構成できます。38 ページの「コアファイル仕様の管理」を参照してください。

プロセスが異常終了した場合は、`mdb` などのデバッガや、`proc` ツールを使用して、作成されたコアファイルを調べることができます。41 ページの「プロセスの障害発生後にコアファイルを調べる」を参照してください。

コアファイル作成のパラメータ

プロセスに障害が発生すると、システムは、グローバルコアファイル名パターンおよびプロセス別コアファイル名パターンを使用して、障害の発生したプロセスごとに最大 2 つのコアファイルの作成を試み、それぞれのコアファイル名を作成します。`coreadm` コマンドでは、これらの名前パターンを制御し、コアファイルの場所を指定します。このセクションでは、ファイルパスとファイル名のパラメータについて説明します。コアダンププロセスの詳細は、`core(4)` のマニュアルページを参照してください。`coreadm` オプションの詳細は、`coreadm(1M)` のマニュアルページを参照してください。

構成可能なコアファイルのパス

プロセスが異常終了すると、コアファイルがデフォルトで現在のディレクトリに作成されます。グローバルコアファイルのパスが有効になっていると、プロセスが異常終了するたびにコアファイルが 2 つ、1 つは現在の作業ディレクトリに、1 つはグローバルコアファイルの場所にそれぞれ作成されます。使用されるファイルパスは、構成可能なパラメータです。

次の 2 つの構成可能なコアファイルのパスは、個別に有効または無効にできます。

- プロセス別コアファイルのパスは、デフォルトで**コア**に設定されており、デフォルトで有効になっています。プロセス別コアファイルのパスが有効になっていると、プロセスが異常終了したときにコアファイルが生成されます。プロセス別のパスは、親プロセスから新しいプロセスに継承されます。

  プロセス別コアファイルは生成されるとプロセスの所有者によって所有され、所有者には読み取り/書き込み権が与えられます。所有者だけがこのファイルを表示できます。
- グローバルコアファイルのパスは、デフォルトで**コア**に設定されており、デフォルトで無効になっています。このパスが有効になっていると、プロセス別コアファイルのパスと同じ内容のコアファイルがグローバルコアファイルのパスに追加で作成されます。

  グローバルコアファイルは生成されると root によって所有され、root だけに読み取り/書き込み権が与えられます。アクセス権のないユーザーはこのファイルを表示できません。

注記 - デフォルトでは、`setuid` プロセスは、グローバルの設定やプロセス別のパスを使ってコアファイルを生成することはありません。

拡張されたコアファイル名

コアファイルの名前には、障害の発生したプロセスの情報のフィールドが含まれます。コアファイル名のフィールドの詳細は、coreadm(1M) のマニュアルページを参照してください。このセクションでは、グローバル変数について重点的に説明します。

グローバルコアファイルディレクトリが有効な場合、次の変数を使用して、コアファイルを相互に区別できます。

%d	実行ファイルのディレクトリ名。最大文字数は MAXPATHLEN
%f	実行ファイルの名前。最大文字数は MAXCOMLEN
%g	実効グループ ID
%m	マシン名 (uname -m)
%n	システムノード名 (uname -n)
%p	プロセス ID
%t	time(2) の 10 進数
%u	実効ユーザー ID
%z	プロセスが実行されているゾーン名 (zonename)
%%	リテラル %

たとえば、/var/core/core.%f.%p がグローバルコアファイルパスとして設定されているとします。PID 12345 の sendmail プロセスが異常終了した場合は、コアファイルとして /var/core/core.sendmail.12345 が生成されます。

コアファイルダンプのパフォーマンスの改善

一部のプロセスのバイナリイメージをコアダンプから除外することで、システムのコアファイルダンプのパフォーマンスを改善できます。coreadm コマンドを使用して、コアダンプの仕様をカスタマイズすると、DISM マッピング、ISM マッピング、System V 共有メモリーなどをコアダンプから除外するように指定できます。詳細は、coreadm(1M) のマニュアルページを参照してください。

コアファイル仕様の管理

このセクションには、コアファイルの管理に関する次の情報が記載されています。

現在のコアダンプ構成の表示

現在のコアダンプ構成を表示するには、オプションを指定しないで coreadm コマンドを使用します。

```
$ 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
```

コアファイル名パターンの設定

コアファイル名パターンは、グローバル、ゾーン別、またはプロセス別に設定できます。また、システムのリブート後も有効なプロセス別デフォルトを設定できます。

たとえば、次の coreadm コマンドで、init プロセスによって開始されるすべてのプロセスのデフォルトのプロセス別コアファイルパターンを設定します。この設定は、デフォルトのコアファイルパターンを明示的に上書きしていないプロセスに対して適用されます。この設定はシステムリブート後も有効です。

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

次の coreadm コマンドでは、任意のプロセスに対しプロセス別コアファイル名パターンを設定します。

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

$$ 記号には、現在実行中のシェルのプロセス ID を指定します。プロセス別コアファイル名パターンは、すべての子プロセスに継承されます。

例:

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

または、root 役割になり、グローバルファイル名パターンを設定することもできます。

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

グローバルまたはプロセス別のコアファイル名パターンを設定したら、これを coreadm -e コマンドで有効にする必要があります。

このコマンドをユーザーの初期設定ファイル (たとえば .profile) に入れておけば、ユーザーのログインセッションで実行するすべてのプロセスに対しコアファイル名パターンを設定できます。

ファイルパスの有効化

プロセス別またはグローバルコアファイルパスを有効にできます。

- プロセス別コアファイルパスを有効にするには、root 役割になり、次のコマンドを発行します。

  ```
  # coreadm -e process
  ```

  構成を確認する場合は、現在のプロセスコアファイルパスを表示します。

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

- グローバルコアファイルパスを有効にするには、root 役割になり、次のコマンドを発行します。

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

  構成を確認する場合は、現在のプロセスコアファイルパスを表示します。

  ```
  # 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
  ```

setuid プログラムがコアファイルを作成できるようにする

coreadm コマンドを使って setuid プログラムを有効または無効にすれば、適切なパスを設定することによって、すべてのシステムプロセスに対して、または各プロセスに対してコアファイルを作成できます。

- グローバル setuid オプションが有効になっていると、グローバルコアファイルパスに従って、システムのすべての setuid プログラムがコアファイルを作成します。
- プロセス別 setuid オプションが有効になっていると、プロセス別コアファイルパスに従って、特定の setuid プロセスがコアファイルを作成します。

デフォルトでは、両方のフラグが無効になっています。セキュリティー上の理由により、グローバルコアファイルパスは、/ で始まるフルパス名にする必要があります。root がプロセス別コアファイルを無効にすると、個別のユーザーがコアファイルを得ることはできなくなります。

setuid コアファイルは root によって所有され、root だけに読み取り/書き込み権が与えられます。通常ユーザーは、ユーザーが setuid コアファイルを生成したプロセスを所有していても、それらのファイルにアクセスできません。

詳細については、coreadm(1M) のマニュアルページを参照してください。

デフォルトのコアファイル設定に戻す

root として、次のいずれかのコマンドを実行し、コアファイルパスを無効にして、コアファイル名パターンを削除します。

- グローバルコアファイル設定の場合:

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

注記 - "" は空白なしの空の文字列です。

- プロセス別コアファイル設定の場合:

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

-d オプションは、コアファイルパスを無効にします。-g オプションに空の文字列変数を付けると、コアファイル名パターンが削除されます。コアファイルパスおよびコアファイル名パターンは、元のデフォルト設定に戻ります。

非推奨のコアファイルパラメータの修正

/etc/system ファイル内で setuid コアファイルを許可する非推奨のパラメータがある場合、次のメッセージが表示されます。

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

この問題を修正するには、/etc/system ファイルから allow_setid_core=1 を削除します。次に coreadm コマンドを使って、 グローバル setuid コアファイルパスを有効にします。

プロセスの障害発生後にコアファイルを調べる

proc ツールを使用して、プロセスのコアファイルやライブプロセスを調べることができます。proc ツールは、/proc ファイルシステムの機能を操作するユーティリティーです。

プロセス ID をこれらのコマンドに指定する方法と類似する方法でコアファイルの名前をコマンド行に指定することで、/usr/proc/bin/pstack、pmap、pldd、pflags、および pcred ツールをコアファイルに適用できます。

proc ツールを使用してコアファイルを調べる方法については、proc(1) を参照してください。

例 5 proc ツールを使用してコアファイルを調べる

```
$ ./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
```

♦ ♦ ♦ 第 5 章

5 システムログとメッセージングの管理

この章では、システムログとシステムメッセージの表示および管理を行う方法について説明します。ここには、次の情報が含まれています。

システムログとメッセージングについて

syslogd コマンドを使用すると、システムメッセージの表示機能と転送機能が有効になります。これは、メッセージの優先順位とそのシステム機能の場所に基づいて、システムメッセージを読み取って、適切なログファイルまたはユーザーに転送します。syslog 構成ファイルはメッセージの転送先を制御するのに対して、syslogd コマンドは、すべてのメッセージのタイムスタンプを分間隔で記録します。デフォルトの間隔時間は 20 分に設定されます。

root 以外のユーザーにログファイルを管理する権限を付与する場合は、そのユーザーの権利プロファイルとして**ログ管理**を付与できます。その付与は、新しいユーザーの場合は useradd コマンド、既存のユーザーの場合は usermod コマンドとともに -P オプションを使用することによって実行できます。手順については、useradd(1M) および usermod(1M) のマニュアルページを参照してください。

システムメッセージとロギングの管理

このセクションでは、Oracle Solaris のシステムメッセージング機能について説明します。

システムメッセージの形式

システムのメッセージはコンソールデバイスに表示されます。ほとんどのシステムメッセージのテキストの形式は次のとおりです。

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

例:

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

カーネルから出されるメッセージには、カーネルモジュール名が次のように表示されます。例:

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

システムがクラッシュした場合、次のような形式のパニックメッセージがシステムコンソールに表示されることがあります。

`panic:` *error message*

まれに、パニックメッセージではなく次のメッセージが表示されることがあります。

```
Watchdog reset !
```

システムメッセージのロギング

エラーロギングデーモン `syslogd` は、さまざまなシステムの警告やエラーをメッセージファイルに自動的に記録します。デフォルトでは、これらのシステムメッセージの多くは、システムコンソールに表示されて、`/var/adm` ディレクトリに格納されます。システムメッセージ記録を設定することによって、これらのメッセージを格納する場所を指示できます。詳しくは、46 ページの「システムのメッセージ記録のカスタマイズ」を参照してください。これらのメッセージは、失敗の予兆のあるデバイスなど、システム障害をユーザーに警告できます。

`/var/adm` ディレクトリには、いくつかのメッセージファイルが含まれています。もっとも新しいメッセージは、`/var/adm/messages` (および `messages.*`) ファイルにあり、もっとも古いメッセージは、`messages.3` ファイルにあります。一定の期間 (通常は 10 日) ごとに、新しい `messages` ファイルが作成されます。`messages.0` のファイル名は `messages.1` に、`messages.1` は `messages.2` に、`messages.2` は `messages.3` にそれぞれ変更されます。その時点の `/var/adm/messages.3` ファイルは削除されます。

`/var/adm` ディレクトリは、メッセージやクラッシュダンプなどのデータを含む大きなファイルを格納するため、多くのディスク容量を消費します。`/var/adm` ディレクトリが大きくならないようにするために、そして将来のクラッシュダンプが保存できるようにするために、不要なファイルを定期的に削除しなければなりません。`crontab` ファイルを使用すれば、このタスクは自動化できます。このタスクの自

動化の詳細は、『Oracle Solaris 11.3 でのデバイスの管理』 の 「ダンプファイルの削除」および『Oracle Solaris 11.3 でのシステム情報、プロセス、およびパフォーマンスの管理』 の 第 4 章,「システムタスクのスケジュール設定」を参照してください。

システムメッセージとロギングの表示

システムクラッシュまたはリブートによって生成された最近のメッセージを表示するには、dmesg コマンドを使用します。

```
$ dmesg
```

大きなメッセージログの場合は、more コマンドを使用して、メッセージを 1 回につき 1 画面のみ表示します。

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

例 6 システムメッセージの表示

次の例は、Oracle Solaris 11 システムでの dmesg コマンドからの出力を示します。

```
$ 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]...
.
.
.
```

詳細は、dmesg(1M) のマニュアルページを参照してください。

システムログローテーション

システムログファイルは、root の crontab ファイルのエントリから logadm コマンドによって実行されます。/usr/lib/newsyslog スクリプトは使用されません。

このシステムログローテーションは、/etc/logadm.conf ファイルに定義されます。このファイルには、syslogd などのプロセスのログローテーションエントリが含まれています。たとえば、/etc/logadm.conf ファイルにある 1 つのエントリは、/var/log/syslog ファイルが空でなければローテーションが毎週実行されることを示しています。つまり、最新の syslog ファイルが syslog.0 になり、その次に新しい syslog ファイルが syslog.1 になります。最新からさかのぼって 8 つまでの syslog ファイルが保存されます。

また、/etc/logadm.conf ファイルには、最後のログローテーション実行時のタイムスタンプも含まれます。

logadm コマンドを使用して、必要に応じてシステムログをカスタマイズしたり、/etc/logadm.conf ファイルにログを追加したりすることができます。最初に root、または syslog.conf 承認を持つ役割になる必要があります。

たとえば、Apache アクセスとエラーログのローテーションを実行するには、次のコマンドを使用します。

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

この例では、Apache の access_log ファイルのローテーションは、そのサイズが 100M バイトに達したときに実行され、そのファイル名に .0、.1 などのように接尾辞が付けられます。また、古い access_log ファイルのコピーが 10 個保存されます。また、error_log のローテーションは、そのサイズが 10M バイトに達したときに実行され、access_log ファイルと同様に、接尾辞が付けられ、コピーが保存されます。

Apache ログローテーションの例における /etc/logadm.conf エントリの例は、次のようになります。

```
# cat /etc/logadm.conf
.
.
.
/var/apache/logs/error_log -s 10m
/var/apache/logs/access_log -s 100m
```

詳細は、logadm(1M) を参照してください。

システムのメッセージ記録のカスタマイズ

/etc/syslog.conf ファイルを変更すると、さまざまなシステムプロセスが生成するその他のエラーメッセージを記録できます。デフォルトでは、/etc/syslog.conf ファイルは、多くのシステムプロセスのメッセージが、クラッシュメッセージとブートメッセージも含んでいる /var/adm/messages ファイルに格納されるように指示します。/var/adm メッセージを表示するには、45 ページの「システムメッセージとロギングの表示」を参照してください。

/etc/syslog.conf ファイルには、タブで区切られた 2 つの列が含まれています。

facility.level ... action

facility.level

facility は、そのメッセージまたは状態のシステムソースです。コンマで区切られた機能のリスト。*level* は、記録する状態の重要度や優先順位を示します。

同じ機能の 2 つのエントリは、それぞれの優先順位が異なる場合、同じ行に入力しないでください。syslog ファイルに優先順位

を入力すると、この優先順位以上のすべてのメッセージが記録され、最後のメッセージが優先されます。指定の機能とレベルに対し、`syslogd` はそのレベル以上のすべてのメッセージを記録します。

action メッセージが転送される場所を示します。

次の例は、デフォルトの `/etc/syslog.conf` ファイルのサンプル行を示します。

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

これらのエントリによって、次のユーザーメッセージが自動的に記録されます。

- ユーザーエラーはコンソールに出力され、`/var/adm/messages` ファイルにも記録されます。
- 早急な対応が必要なユーザーメッセージ (`alert`) は、`root` ユーザーと `operator` ユーザーに送信されます。
- ユーザー緊急メッセージは、各ユーザーに送信されます。

注記 - エントリを個別の行に入力すると、`/etc/syslog.conf` ファイルでログの対象が複数回指定された場合に、メッセージのログ順が変わることがあります。単独行のエントリに複数のセレクタを指定できます。その際、セレクタはセミコロンで区切ります。

もっとも一般的なエラー状態ソースは次のとおりです。

kern カーネル

auth 認証

daemon すべてのデーモン

mail メールシステム

lp スプールシステム

user ユーザープロセス

注記 - `/etc/syslog.conf` ファイルで有効化できる `syslog` 機能の数に制限はありません。

`syslog.conf` メッセージのもっとも一般的な優先順位レベルは、優先度順に次のとおりです。

emerg	システムの緊急事態
alert	すぐに修正が必要なエラー
crit	クリティカルなエラー
err	その他のエラー
info	情報メッセージ
debug	デバッグ用の出力
none	この設定は出力を記録しない

▼ システムのメッセージ記録をカスタマイズする方法

1. **root 役割になるか、solaris.admin.edit/etc/syslog.conf 承認が割り当てられている役割になります。**

   『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』 の 「割り当てられている管理権利の使用」を参照してください。

2. **pfedit コマンドを使用して、/etc/syslog.conf ファイルを編集します。**

   メッセージソース、優先順位、およびメッセージの場所を追加または変更できます。詳細は、syslog.conf(4) を参照してください。

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

3. **変更を保存します。**

例 7 システムのメッセージ記録のカスタマイズ

この例は、ユーザー緊急メッセージを root ユーザーおよび個別のユーザーに送信する /etc/syslog.conf user.emerg 機能のエントリを示しています。

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

rsyslogd コマンドを使用した拡張システムロギング

この Oracle Solaris リリースには、システムロギングを管理するための rsyslogd パッケージをインストールして使用するオプションが含まれています。rsyslogd は、フィルタリング、TCP、暗号化、高精度のタイムスタンプ、出力制御などの複数の機

能をサポートするモジュール設計を備えた、syslogd デーモンの実装から取得されます。

syslog SMF サービス、svc:/system/system-log:default は、引き続きデフォルトのロギングサービスになります。rsyslog サービスを使用するには、rsyslog パッケージをインストールし、rsyslogd サービスを有効にする必要があります。

▼ rsyslog をインストールして有効にする方法

1. **次のようにサービスを有効化することで、rsyslog パッケージがすでにシステムにインストールされているかどうかを確認します。**

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

rsyslog パッケージがインストールされていない場合は、次のメッセージが表示されます。

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

2. **rsyslog パッケージがインストールされていない場合は、インストールします。**

```
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. **rsyslog インスタンスを確認します。**

```
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
```

この出力では、rsyslog インスタンスが存在しても、無効化されていることを確認します。

4. **rsyslog サービスに切り替えます。**

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

これらのコマンドは、デフォルトのサービスの無効化、rsyslog の有効化、およびステータスの報告を行います。

次の手順 rsyslog をインストールして有効化すると、syslog を /etc/rsyslog.conf ファイルに構成できます。詳細は、rsyslogd(1m) を参照してください。

リモートコンソールメッセージングを有効にする

次のコンソール機能により、Oracle Solaris リモートシステムをトラブルシューティングする機能が向上します。

- consadm コマンドでは、補助 (またはリモート) コンソールとしてシリアルデバイスを選択できます。consadm コマンドを使用すると、システム管理者は 1 つまたは複数のシリアルポートを構成して、出力先が変更されたコンソールメッセージを表示したり、システムの実行レベルが変わったときに sulogin セッションをホストしたりできます。この機能を使用して、モデム付きのシリアルポートにダイヤルインしてコンソールメッセージをモニターし、init 状態の変更を表示できます詳細は、sulogin(1M) と、次の詳しい手順を参照してください。

  補助コンソールとして構成されたポートからシステムにログインできますが、このポートは主に、デフォルトコンソールにも表示される情報を表示する出力デバイスです。ブートスクリプトやその他のアプリケーションがデフォルトコンソールに対して読み取りおよび書き込みを行う場合、書き込み出力はすべての補助コンソールに表示されます。ただし、入力はデフォルトコンソールからのみ読み取られます。詳細は、52 ページの「対話型ログインセッションで consadm コマンドを使用するためのガイドライン」を参照してください。
- コンソール出力は、新しい仮想デバイス /dev/sysmsg に書き込まれる、カーネルメッセージと syslog メッセージからなります。さらに、rc スクリプト起動メッセージが /dev/msglog に書き込まれます。以前のリリースでは、これらのメッセージはすべて /dev/console に書き込まれていました。

  スクリプトメッセージを補助コンソールに表示したい場合は、コンソール出力を /dev/console に出力しているスクリプトで出力先を /dev/msglog に変更する必要があります。メッセージ出力先を補助デバイスに変更する場合は、/dev/console を参照しているプログラムで syslog() または strlog() を使用するように明示的に変更してください。
- consadm コマンドは、デーモンを実行して補助コンソールデバイスをモニターします。補助コンソールに指定された表示デバイスがハングアップしたりキャリア信号がなくなって切り離されると、そのデバイスは補助コンソールデバイスのリストから削除され、アクティブでなくなります。1 つまたは複数の補助コンソールを有効にしても、メッセージがデフォルトコンソールに表示されなくなるわけではありません。メッセージは引き続き /dev/console に表示されます。
- consadm デーモンは、consadm コマンドで補助コンソールを追加するまでポートのモニタリングを開始しません。セキュリティー機能として、コンソールメッセージは、キャリア信号が失われるまで、または補助コンソールデバイスの選択が解除されるまでの間だけ出力変更されます。そのため、consadm コマンドを正常に使うには、そのポートでキャリア信号が確立されている必要があります。

実行レベルの変更中に補助コンソールメッセージングを使用する

実行レベルの変更中に補助コンソールメッセージングを使う場合は、次の点を確認してください。

- システムのブート時に実行する rc スクリプトにユーザーの入力がある場合は、補助コンソールから入力を行うことはできません。入力はデフォルトコンソールから行う必要があります。
- 実行レベルの変更中に、root パスワード入力を要求するために sulogin プログラムが init によって呼び出されます。このプログラムは、デフォルトのコンソールデバイスだけでなく各補助デバイスにも root パスワードの入力要求を送信するように変更されています。
- ユーザーは、sulogin を直接起動しないでください。ユーザーがこのユーティリティーを使用するには、solaris.system.maintenance authorization が必要です。
- システムがシングルユーザーモードで動作し、1 つまたは複数の補助コンソールが consadm コマンドによって有効になっていると、最初のデバイスでコンソールログインセッションが実行され、正確な root パスワードを要求する sulogin プロンプトが表示されます。コンソールデバイスから正しいパスワードを受け取ると、sulogin は他のすべてのコンソールデバイスからの入力を受信できないようにします。
- コンソールの 1 つがシングルユーザー特権を取得すると、デフォルトコンソールとその他の補助コンソールにメッセージが出力されます。このメッセージは、どのデバイスから正しい root パスワードが入力され、コンソールになったかを示します。シングルユーザーシェルが動作する補助コンソールのキャリア信号が失われると、次のどちらかのアクションが起ることがあります。
  - 補助コンソールが実行レベル 1 のシステムを表している場合は、システムはデフォルトの実行レベルに移行します。
  - 補助コンソールが実行レベル S のシステムを表している場合は、シェルから init s または shutdown コマンドが入力されたデバイスに「ENTER RUN LEVEL (0-6, s or S):」というメッセージが表示されます。そのデバイスのキャリア信号も失われている場合、キャリア信号を再確立して、正しい実行レベルを入力する必要があります。init コマンドや shutdown コマンドを実行しても、実行レベルプロンプトが再表示されることはありません。
- シリアルポートを使用してシステムにログインしている場合には、init または shutdown コマンドを使用して別の実行レベルに移行すると、このデバイスが補助コンソールかどうかに関係なくログインセッションは失われます。この状況は、補助コンソール機能がない リリースと同じです。
- consadm コマンドを使って補助コンソールにするデバイスを選択すると、システムをリブートするか補助コンソールの選択を解除するまで、そのデバイスは補助コンソールのままになります。ただし、consadm コマンドには、システムリブー

ト後も同じデバイスを補助コンソールとして使用するオプションがあります。詳細は、次の手順を参照してください。

対話型ログインセッションで consadm コマンドを使用するためのガイドライン

シリアルポートに接続された端末を使用してシステムにログインしてから、consadm コマンドを使ってこの端末にコンソールメッセージを表示して、対話型ログインセッションを実行できます。次の動作を確認してください。

- この端末で対話型ログインセッションを行う場合、補助コンソールがアクティブだと、コンソールメッセージは /dev/sysmsg デバイスまたは /dev/msglog デバイスに送られます。
- この端末からコマンドを発行すると、入力はデフォルトコンソール (/dev/console) ではなく対話型セッションに送られます。
- init コマンドを実行して実行レベルを変更すると、リモートコンソールソフトウェアは対話型セッションを終了し、sulogin プログラムを実行します。この時点では、入力はこの端末からだけ可能で、入力はコンソールデバイスから行われたかのように扱われます。このプロセスによって、51 ページの「実行レベルの変更中に補助コンソールメッセージングを使用する」の説明のとおりに、sulogin プログラムにパスワードを入力できます。

  補助端末から正しいパスワードを入力すると、補助コンソールは、対話型 sulogin セッションを実行し、デフォルトコンソールおよび競合する補助コンソールは使用できなくなります。この動作は、その端末は実質的にシステムコンソールとして機能することを意味します。
- この端末から実行レベル 3 または別の実行レベルに変更できます。実行レベルを変更すると、すべてのコンソールデバイスで sulogin が再び実行されます。終了したり、システムが実行レベル 3 で起動されるように指定すると、どの補助コンソールからも入力を行えなくなります。すべての補助コンソールはコンソールメッセージを表示するだけのデバイスに戻ります。

  システムが起動する際には、デフォルトのコンソールデバイスから rc スクリプトに情報を入力する必要があります。システムがふたたび起動すると login プログラムがシリアルポートで実行されるため、別の対話型セッションを開始できます。そのデバイスを補助コンソールに指定していれば、コンソールメッセージはその端末に引き続き出力されます。ただし、端末からの入力はすべて対話型セッションに送られます。

▼ 補助コンソールのリストを表示する方法

1. **引数を付けずに、root として consadm コマンドを発行します。**

『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』 の 「割り当てられている管理権利の使用」を参照してください。

2. 次のどちらかの手順に従います。

   a. 補助コンソールのリストを表示します。

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

   b. 永続的な補助コンソールのリストを表示するには、-p オプションを使用します。

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

▼ 補助 (リモート) コンソールを有効にする方法

1. システムにログインし、root 役割になります。

   『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』 の 「割り当てられている管理権利の使用」を参照してください。

2. 補助コンソールを有効にします。

   ```
   # consadm -a device name
   ```

3. 現在の接続が補助コンソールであることを確認します。

   ```
   # consadm
   ```

例 8 補助 (リモート) コンソールを有効にする

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

▼ システムリブート後も補助 (リモート) コンソールを有効にする方法

1. システムにログインし、root 役割になります。

   『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』 の 「割り当てられている管理権利の使用」を参照してください。

2. 複数のシステムリブート後も補助コンソールを有効にします。

   ```
   # consadm -a -p device name
   ```

このコマンドによって、このデバイスが永続的な補助コンソールのリストに追加されます。

3. **デバイスが持続的な補助コンソールのリストに追加されているか確認します。**

```
# consadm
```

例 9 システムリブート後も補助 (リモート) コンソールを有効にする

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

▼ 補助 (リモート) コンソールを無効にする方法

1. **システムにログインし、root 役割になります。**

   『Oracle Solaris 11.3 でのユーザーとプロセスのセキュリティー保護』 の 「割り当てられている管理権利の使用」を参照してください。

2. **補助コンソールを無効にします。**

```
# consadm -d devicename
```

   また、補助コンソールを永続的な補助コンソールのリストから削除するには、**-p** オプションを追加します。

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

3. **補助コンソールが無効になっていることを確認します。**

```
# consadm
```

例 10 補助 (リモート) コンソールを無効にする

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

索引

U

V

W