Oracle® Solaris 11.2의 시스템 관리 문제 해결

부품 번호: E53852-02
2014년 9월

ORACLE®

목차

이 설명서 사용

- **개요** - SPARC 및 x86 플랫폼에서 문제를 해결하는 작업을 설명합니다.
- **대상** - Oracle Solaris 11 릴리스를 사용하는 시스템 관리자
- **필요한 지식** - UNIX 시스템 관리 경험

제품 설명서 라이브러리

이 제품에 대한 최신 정보 및 알려진 문제는 설명서 라이브러리(http://www.oracle.com/pls/topic/lookup?ctx=E56343)에서 확인할 수 있습니다.

Oracle 지원 액세스

Oracle 고객은 My Oracle Support를 통해 온라인 지원에 액세스할 수 있습니다. 자세한 내용은 http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info를 참조하거나, 청각 장애가 있는 경우 http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs를 방문하십시오.

피드백

http://www.oracle.com/goto/docfeedback에서 이 설명서에 대한 피드백을 보낼 수 있습니다.

♦ ♦ ♦ 1 장

1 시스템 충돌 문제 해결

이 장에서는 Oracle Solaris OS에서 시스템 충돌에 대비하고 문제를 해결하는 방법을 설명합니다.

이 장에서는 다음 내용을 다룹니다.

Oracle Solaris 11.2의 충돌 덤프 파일 관련 새로운 기능

이 절에서는 이 Oracle Solaris 릴리스에서 충돌 덤프 파일을 관리하기 위한 변경 사항에 대해 설명합니다.

충돌 덤프 파일이 재구성됨

이 릴리스에서 커널 충돌 덤프 파일은 더 빠른 초기 분석을 수행하고 구성을 더욱 세분화할 수 있도록 내용에 따라 여러 개의 새 파일로 나뉩니다. 자세한 내용은 “재구성된 파일” [8]을 참조하십시오.

`/var/crash` 디렉토리에 저장된 충돌 덤프 파일

Oracle Solaris 11 운영 체제에서 충돌 덤프 파일은 `/var/crash` 디렉토리에 저장됩니다. 이 변경은 원래 Oracle Solaris 10 1/13 릴리스에서 도입되었고 모든 Oracle Solaris 11 릴리스에 포함됩니다.

시스템 충돌 정보

시스템 충돌은 하드웨어 오작동, I/O 문제 및 소프트웨어 오류로 인해 발생할 수 있습니다. 시스템이 충돌하면 콘솔에 오류 메시지가 표시되고 덤프 장치에 물리적 메모리 복사본이 기록됩니다. 그런 다음 시스템이 자동으로 재부트됩니다. 시스템이 재부트될 때 savecore 명령이 실행되어 덤프 장치에서 데이터가 검색되고 저장된 충돌 덤프 파일이 savecore 디렉토리에 기록됩니다. 이러한 저장된 파일은 문제 진단에 도움이 되는 중요한 정보를 제공합니다.

참고 - 충돌 덤프는 충돌 덤프 파일 세트, 해당 파일 위치, 해당 파일 구성/형식 지정 방법을 포함하여 이 프로세스의 전체 결과를 나타냅니다.

시스템 충돌 덤프 파일

savecore 명령은 시스템 충돌 후에 자동으로 실행되어 덤프 장치에서 충돌 덤프 정보를 검색하고 해당 정보를 일련의 파일에 씁니다. 이후 같은 시스템이나 또 다른 시스템에서 savecore 명령을 호출하여 압축된 충돌 덤프 파일을 확장할 수 있습니다.

참고 - 응용 프로그램이 비정상적으로 종료될 때 작성되는 사용자 응용 프로그램 이미지인 코어 파일과 충돌 덤프 파일이 혼동되는 경우도 있습니다.

충돌 덤프 파일은 미리 정해진 디렉토리(기본적으로 /var/crash/)에 저장됩니다. 이전 릴리스에서는 물리적 메모리 이미지가 충돌 덤프 파일에 저장되도록 수동으로 설정하지 않은 경우 시스템이 재부트될 때 충돌 덤프 파일을 덮어썼습니다. 지금은 충돌 덤프 파일 저장이 기본적으로 사용으로 설정되어 있습니다.

재구성된 파일

Oracle Solaris 11.2 릴리스에서는 커널 충돌 덤프 파일이 재구성되었습니다. 이러한 파일의 내용은 더 빠른 초기 분석을 수행하고 구성을 더욱 세분화할 수 있도록 내용에 따라 여러 개의 새 파일로 나뉩니다. 파일을 더 손쉽게 액세스하고 조사할 수 있습니다.

이전에 커널 충돌 덤프는 다음 파일에 저장되었습니다.

- vmdump.N
- unix.N
- vmcore.N

vmdump.N 및 vmcore.N에서는 커널 페이지 메타 데이터 및 데이터를 각각 압축된 형식이나 압축되지 않은 형식으로 저장했습니다.

Oracle Solaris 11.2 릴리스부터 충돌 덤프 정보는 vmdump-*section*.*n* 파일 세트에 기록됩니다. 섹션 값은 특정 종류의 덤프 정보가 포함된 파일 섹션의 이름입니다. *n* 값은 충돌 덤프를 복사하기 위해 savecore가 실행될 때마다 증분되는 정수이며, 새 충돌 덤프는 덤프 장치에 있습니다. 가능한 파일은 다음과 같습니다.

vmdump-proc.N	압축된 프로세스 페이지가 있는 덤프 파일
vmdump-zfs.N	압축된 ZFS 메타 데이터가 있는 덤프 파일
vmdump-other.N	기타 페이지가 있는 덤프 파일

자세한 내용은 dumpadm(1M) 및 savecore(1M) 매뉴얼 페이지를 참조하십시오.

dumpadm 및 savecore 명령

dumpadm 및 savecore 유틸리티는 다음과 같이 충돌 덤프 만들기를 구성하고 관리합니다.

시스템 시작 시 svc:/system/dumpadm:default 서비스가 충돌 덤프 매개변수를 구성하기 위해 dumpadm 명령을 호출합니다. /dev/dump 인터페이스를 통해 덤프 장치 및 덤프 컨텐츠를 초기화합니다.

참고 - Oracle Solaris 11.2 릴리스의 dumpadm 명령에는 덤프 컨텐츠를 지정하고, 디스크 공간 추정값을 인쇄하고, 구문 분석 가능한 출력을 생성하기 위한 새로운 옵션이 있습니다. "충돌 덤프 구성 수정" [11]을 참조하십시오.

덤프 구성이 완료되면 savecore 스크립트가 충돌 덤프 파일 디렉토리의 위치를 찾습니다. 그런 다음 savecore가 호출되어 충돌 덤프가 확인되고 충돌 덤프 디렉토리에서 minfree 파일의 컨텐츠가 확인됩니다. savecore 명령으로 생성되는 시스템 충돌 덤프 파일은 기본적으로 저장됩니다.

덤프 데이터는 덤프 장치에 압축된 형식으로 저장됩니다. 커널 충돌 덤프 이미지는 4GB 이상일 수 있습니다. 데이터를 압축하면 덤프는 빨라지고 덤프 장치에 필요한 디스크 공간은 줄어듭니다.

스왑 영역이 아니라 전용 덤프 장치가 덤프 구성에 포함되면 충돌 덤프 파일 저장 기능이 백그라운드에서 실행됩니다. 부트 중인 시스템은 다음 단계로 이동하기 전에 savecore 명령이 완료될 때까지 기다리지 않습니다. 대용량 메모리 시스템에서는 savecore가 완료되기 전에 시스템을 사용할 수 있습니다.

savecore -L 명령을 사용하여 관리자는 현재 Oracle Solaris OS 실행의 충돌 덤프를 가져올 수 있습니다. 이 명령은 잘못된 상태(예: 일시적인 성능 문제 또는 서비스 중단)가 발생할 때 메모리 스냅샷을 만들어 실행 중인 시스템 문제를 해결하는 데 사용됩니다. 시스템이 작동하고 일부 명령을 실행할 수 있을 경우 savecore -L 명령을 실행하여 시스템 스냅샷을 덤프 장치에 저장한 다음 충돌 덤프 파일을 savecore 디렉토리에 바로 기록할 수 있습니다. 시스템이 계속 실행 중이므로 전용 덤프 장치를 구성한 경우에만 savecore -L 명령을 사용할 수 있습니다.

자세한 내용은 “충돌 덤프 구성 수정” [11], dumpadm(1M) 및 savecore(1M) 매뉴얼 페이지를 참조하십시오.

시스템에 대한 충돌 덤프 구성

이 절에서는 시스템에 대한 충돌 덤프 절차를 관리하기 위한 작업에 대해 설명합니다.

시스템 충돌 정보를 사용할 때는 다음 사항을 염두에 두어야 합니다.

- 시스템 충돌 정보를 액세스 및 관리하려면 root 역할을 맡아야 합니다. “Oracle Solaris 11.2의 사용자 및 프로세스 보안 ”의 “지정된 관리 권한 사용”을 참조하십시오.
- 시스템에서 시스템 충돌 덤프 파일 저장 옵션을 사용 안함으로 설정하지 마십시오. 시스템 충돌 덤프 파일은 시스템 충돌 원인을 확인할 수 있는 유용한 방법을 제공합니다.
- 전용 ZFS 볼륨은 스왑 및 덤프 영역에 사용됩니다. 설치 후 스왑 및 덤프 장치 크기를 조정하거나 스왑 및 덤프 볼륨을 다시 만들어야 할 수 있습니다. 자세한 내용은 “Oracle Solaris 11.2의 ZFS 파일 시스템 관리 ”의 “ZFS 스왑 및 덤프 장치 관리”를 참조하십시오.

시스템 충돌이 발생하기 전에 시스템 충돌 덤프 프로세스를 사용자 정의하려면 다음을 참조하십시오.

- “현재 충돌 덤프 구성 표시” [10]
- “충돌 덤프 구성 수정” [11]
- “충돌 덤프 저장을 사용 안함 또는 사용으로 설정” [12]

현재 충돌 덤프 구성 표시

현재 충돌 덤프 구성을 표시하려면 root 역할을 수행하고 dumpadm 명령을 인수 없이 실행합니다.

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

이 예제 출력에서는 다음 구성을 보여줍니다.

- 덤프 컨텐츠가 커널 메모리 페이지입니다.
- 커널 메모리는 전용 덤프 장치 /dev/zvol/dsk/rpool/dump에 덤프됩니다.
- 시스템 충돌 덤프 파일이 /var/crash 디렉토리에 저장됩니다.
- 충돌 덤프 파일 저장이 사용으로 설정되어 있습니다.
- 충돌 덤프를 압축된 형식으로 저장합니다.

충돌 덤프 구성 수정

충돌 덤프 구성을 수정하려면 root 역할을 수행하고 dumpadm 명령을 사용합니다.

dumpadm 명령의 구문은 다음과 같습니다.

```
#  /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*

덤프할 데이터의 유형을 지정합니다. Oracle Solaris 11.2 릴리스의 경우 이 옵션 값이 변경되었습니다. 커널을 사용하여 커널 메모리 페이지만 덤프하거나, all을 사용하여 모든 메모리 페이지를 덤프하거나, curproc를 사용하여 충돌이 발생할 때 스레드가 실행 중이던 프로세스의 메모리 페이지 및 커널 메모리를 덤프하거나, allproc를 사용하여 커널 메모리 페이지 및 모든 프로세스 페이지를 덤프하거나, zfs를 사용하여 ZFS 메타 데이터를 저장하는 커널 페이지를 덤프합니다. 기본 덤프 컨텐츠는 커널 메모리입니다.

다음 -c 옵션 예를 참조하십시오.

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

-d *dump-device*

시스템 충돌 시 덤프 데이터를 임시로 저장하는 장치를 지정합니다. 기본 덤프 장치가 기본적으로 사용되는 덤프 장치입니다. 덤프 장치가 스왑 영역이 아닐 경우 savecore가 백그라운드로 실행되므로 부트 프로세스 속도가 빨라집니다.

-e

압축된 충돌 덤프를 저장하는 데 필요한 예상 디스크 공간을 인쇄합니다. 값은 현재 구성 및 현재 실행 중인 시스템을 사용하여 계산됩니다.

-m *min*k | *min*m | *min*%

현재 savecore 디렉토리에 minfree 파일을 만들어 충돌 덤프 파일 저장을 위한 최소 사용 가능 디스크 공간을 지정합니다. 이 매개변수는 KB(mink), MB(minm) 또는 파일 시스템 크기 퍼센트(min%)로 지정할 수 있습니다. 최소 사용 가능 공간이 구성되지 않은 경우 기본값은 1MB입니다.

savecore 명령은 충돌 덤프 파일을 작성하기 전에 이 파일을 참조합니다. 충돌 덤프 파일을 작성할 때 크기 때문에 사용 가능한 공간이 minfree 임계치 아래로 떨어지는 경우 덤프 파일이 작성되지 않고 오류 메시지가 기록됩니다. 이 시나리오 복구에 대한 자세한 내용은 "충돌 덤프 디렉토리가 가득 찼을 때 데이터 저장" [17]을 참조하십시오.

-n

시스템 재부트 시 savecore가 실행되지 않도록 지정합니다. 이 덤프 구성은 권장되지 않습니다. 시스템 충돌 정보가 스왑 장치에 기록되며 savecore가 사용으로 설정되지 않은 경우 시스템이 스왑을 시작할 때 충돌 덤프 정보를 덮어씁니다.

-p	시스템에서 구문 분석할 수 있는 출력을 생성합니다.
-s *savecore-dir*	충돌 덤프 파일을 저장할 대체 디렉토리를 지정합니다. Oracle Solaris 11에서 기본 디렉토리는 /var/crash입니다.
-u	/etc/dumpadm.conf 파일의 컨텐츠를 기반으로 커널 덤프 구성을 강제로 업데이트합니다.
-y	재부트 시 savecore 명령이 자동으로 실행(이 덤프 설정의 기본값임)되도록 덤프 구성을 수정합니다.
-z on | off	재부트 시 savecore 명령 작동이 제어되도록 덤프 구성을 수정합니다. on으로 설정하면 코어 파일이 압축된 형식으로 저장됩니다. off로 설정하면 자동으로 충돌 덤프 파일의 압축이 풀립니다. 충돌 덤프 파일은 매우 커질 수 있으므로 충돌 덤프 파일이 압축된 형식으로 저장되면 파일 시스템 공간이 줄어들므로 기본값은 on입니다.

예 1-1 충돌 덤프 구성 수정

이 예에서는 모든 메모리가 전용 덤프 장치인 /dev/zvol/dsk/rpool/dump에 덤프되며 충돌 덤프 파일 저장 후 사용할 수 있어야 하는 최소 사용 가능 공간이 파일 시스템 공간의 10%입니다.

```
# 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
```

충돌 덤프 저장을 사용 안함 또는 사용으로 설정

주의 - Oracle Solaris에서는 충돌 덤프 저장을 사용 안함으로 설정하지 않도록 적극 권장합니다. 충돌 덤프는 시스템 충돌 원인을 확인할 수 있는 유용한 방법을 제공합니다.

root 역할로서 충돌 덤프 저장을 사용 안함 또는 사용으로 설정할 수 있습니다.

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

-n 충돌 덤프 저장을 사용 안함으로 설정합니다.

-y 충돌 덤프 저장을 사용으로 설정합니다.

예 1-2 충돌 덤프 저장을 사용 안함으로 설정

다음 예에서는 시스템에서의 충돌 덤프 저장을 사용 안함으로 설정하는 방법을 보여줍니다.

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

예 1-3 충돌 덤프 저장을 사용으로 설정

다음 예에서는 시스템에서의 충돌 덤프 저장을 사용으로 설정하는 방법을 보여줍니다.

```
# 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
```

시스템 충돌 후 문제 해결

Oracle Solaris 시스템이 충돌하면 일반 시스템 정보 및 충돌 덤프 파일에 제공된 특정 정보를 포함하여 가능한 많은 정보를 서비스 공급자에게 제공해야 합니다.

기존 충돌 덤프 문제를 해결하려면 다음을 참조하십시오.

- "시스템이 충돌할 경우 수행할 작업" [13]
- "충돌 덤프 정보 검사" [14]
- "시스템 충돌 문제 해결 점검 목록" [16]
- "충돌 덤프 디렉토리가 가득 찼을 때 데이터 저장" [17]

시스템이 충돌할 경우 수행할 작업

다음 목록에는 시스템 충돌 시에 기억해야 할 가장 중요한 정보가 설명되어 있습니다.

1. 시스템 콘솔 메시지를 기록해 둡니다.
   - 시스템이 충돌할 경우 시스템이 다시 실행되도록 하는 것이 급선무인 것처럼 여겨질 수 있지만, 시스템을 재부트하기 전에 콘솔 화면에서 메시지를 검사하십시오. 해당 메

시지를 통해 충돌 원인을 파악할 수 있습니다. 시스템이 자동으로 재부트되고 콘솔 메시지가 화면에서 사라진 경우에도 시스템 오류 로그(/var/adm/messages 파일)에서 해당 메시지를 확인할 수 있습니다. 시스템 오류 로그 파일 확인에 대한 자세한 내용은 시스템 메시지 확인 방법 [39]을 참조하십시오.

- 충돌이 자주 발생하며 충돌 원인을 확인할 수 없는 경우 시스템 콘솔 또는 /var/adm/messages 파일에서 확인할 수 있는 모든 정보를 수집하여 고객 서비스 담당자가 검사할 수 있도록 하십시오. 서비스 공급자에게 제공하기 위해 수집할 문제 해결 정보의 전체 목록은 "시스템 충돌 문제 해결 점검 목록" [16]을 참조하십시오.

2. 시스템 충돌 후 시스템 충돌 덤프가 생성되었는지 여부를 확인하십시오.

주의 - 고객 서비스 담당자에게 보내기 전까지는 중요한 시스템 충돌 정보를 제거하지 마십시오.

3. 시스템 충돌 후 시스템이 부트되지 않으면 "Oracle Solaris 11.2 시스템 부트 및 종료 "의 "복구를 위한 시스템 종료 및 부트"에서 추가 지침을 확인하십시오.

충돌 덤프 정보 검사

다음 절차에 설명된 대로 mdb 유틸리티를 사용하여 제어 구조, 활성 테이블, 실시간 또는 충돌한 시스템 커널의 메모리 이미지 및 커널 작동에 대한 기타 정보를 검사할 수 있습니다.

참고 - 다음 절차에서는 mdb 유틸리티 사용 방법에 대한 제한적인 예만 제공합니다. mdb 유틸리티를 완전히 활용하려면 본 매뉴얼에서는 다루지 않는 커널에 대한 충분한 지식이 있어야 합니다. 이 유틸리티 사용에 대한 자세한 내용은 mdb(1) 매뉴얼 페이지를 참조하십시오.

▼ 충돌 덤프 정보 검사 방법

1. **root 역할을 맡습니다.**

   "Oracle Solaris 11.2의 사용자 및 프로세스 보안 "의 "지정된 관리 권한 사용"을 참조하십시오.

2. **충돌 덤프 정보가 저장된 디렉토리로 이동합니다.**

   예를 들면 다음과 같습니다.

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

   충돌 덤프의 위치를 모르면 dumpadm 명령을 사용하여 시스템이 커널 충돌 덤프 파일을 저장할 위치를 확인합니다. 예를 들면 다음과 같습니다.

   ```
   # /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. **모듈식 디버거 유틸리티(mdb)를 사용하여 충돌 덤프를 검사합니다.**

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

-k 파일을 운영 체제 충돌 덤프 파일로 가정하여 커널 디버깅 모드를 지정합니다.

crashdump-file 운영 체제 충돌 덤프 파일을 지정합니다.

예를 들면 다음과 같습니다.

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

다음과 같이 명령을 지정할 수도 있습니다.

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

4. **시스템 충돌 상태를 표시합니다.**

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

커널 충돌 덤프를 검사할 때 `::system dcmd` 명령을 사용하려면 코어 파일이 커널 충돌 덤프여야 하고 `mdb` 유틸리티를 시작할 때 `-k` 옵션이 지정되어 있어야 합니다.

5. **mdb 유틸리티를 종료합니다.**

```
> $quit
```

예 1-4 충돌 덤프 정보 검사

이 예에서는 시스템 정보를 포함하며 이 시스템의 `/etc/system` 파일에서 설정된 tunable을 식별하는 `mdb` 유틸리티의 샘플 출력을 보여줍니다.

```
# 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
```

시스템 충돌 문제 해결 점검 목록

시스템 문제를 격리하게 도와주고 지원 공급자에 대한 문의를 준비하기 위해 다음 점검 목록의 질문에 답하십시오.

항목	데이터
시스템 충돌 덤프를 사용할 수 있습니까?	
운영 체제 릴리스 및 적합한 소프트웨어 응용 프로그램 릴리스 레벨을 식별하십시오.	
시스템 하드웨어를 식별하십시오.	
SPARC 시스템의 경우 `prtdiag` 출력을 포함시키고, 다른 시스템의 경우 Explorer 출력을 포함시키십시오.	
패치가 설치되었습니까? 설치된 경우 `showrev -p` 출력을 포함시키십시오.	
문제를 재현할 수 있습니까? 재현 가능한 테스트 사례는 어려운 문제를 디버깅하는 데 필요한 경우가 많으므로 문제를 재현할 수 있어야 합니다. 문제를 재현하면 서비스 공급자가 특수한 계측으로 커널을 작성하여 버그를 트리거, 진단 및 수정할 수 있습니다.	
시스템에 타사 드라이버가 설치되었습니까? 드라이버는 모두 동일한 권한을 사용하여 커널과 동일한 주소 공간에서 실행되므로 버그가 있을 경우 시스템 충돌의 원인이 될 수 있습니다.	
시스템 충돌 전에 시스템에서 어떤 작업을 수행했습니까? 시스템에서 특수한 작업을 수행(예: 새로운 스트레스 테스트 실행 또는 평소보다 부하가 높은 작업 실행) 중이었던 경우 이로 인해 충돌이 발생한 것일 수 있습니다.	
시스템 충돌 직전 특수한 콘솔 메시지가 표시되었습니까? 시스템이 실제로 충돌하기 전에 원인을 나타내는 메시지가 표시되기도 하며, 이 정보는 유용한 경우가 많습니다.	
매개변수를 `/etc/system` 파일에 추가했습니까? 조정 매개변수(예: 시스템이 확보한 것보다 많은 메모리를 할당할 수 있도록 공유 메모리 세그먼트 증가)가 시스템 충돌의 원인일 수 있습니다.	
문제가 최근에 시작되었습니까?	

항목	데이터
최근에 시작된 경우 시스템 변경(예: 새 드라이버, 새 소프트웨어, 다른 작업 부하, CPU 업그레이드 또는 메모리 업그레이드)으로 인해 문제가 나타난 것일 수 있습니다.	

충돌 덤프 디렉토리가 가득 찼을 때 데이터 저장

savecore 디렉토리에 공간이 남아 있지 않아 시스템 충돌이 발생하는 경우 일부 중요 시스템 충돌 덤프 정보를 저장하려면 다음 방법의 하나를 사용합니다.

시스템 재부트 후에 root 역할로 로그인합니다. savecore 디렉토리에서 서비스 공급자에게 이미 보낸 기존 충돌 덤프 파일을 제거합니다.

참고 - savecore 디렉토리는 일반적으로 /var/crash입니다.

또는 시스템이 재부트한 후 root 역할로 로그인합니다. savecore 명령을 수동으로 실행하고 디스크 공간이 충분한 대체 디렉토리를 지정합니다.

```
# savecore directory
```

Oracle Enterprise Manager Ops Center를 사용하여 문제 관리

데이터 센터 내에서 물리적 및 가상 운영 체제, 서버, 저장 장치에 대한 문제를 관리해야 하면 개별 시스템 내에서 단지 문제를 모니터링하는 것이 아니라 Oracle Enterprise Manager Ops Center에서 제공되는 포괄적인 시스템 관리 솔루션을 사용할 수 있습니다.

Enterprise Manager Ops Center를 사용하여 데이터 센터의 일부가 예상대로 작동하지 않을 때 이를 알리는 경보를 설정하고, 이러한 문제 보고서를 관리하고, 복구를 시도할 수 있습니다.

자세한 내용은 http://www.oracle.com/pls/topic/lookup?ctx=oc122를 참조하십시오.

♦ ♦ ♦ 2 장

2 시스템 정지 또는 재부트 실패 시 문제 해결

이 장에서는 시스템이 정지되거나 시스템 재부트 문제가 있는 경우 수행할 수 있는 작업에 대해 설명합니다.

이 장에서는 다음 정보를 다룹니다.

재부트를 실패할 경우 수행할 작업

시스템이 완전히 재부트되지 않거나 재부트 후 다시 충돌하는 경우 시스템이 제대로 재부트되지 못하도록 하는 소프트웨어 또는 하드웨어 문제가 발생한 것일 수 있습니다.

시스템이 부트되지 않는 원인	문제 해결 방법
시스템이 `/platform/`uname -m`/kernel/sparcv9/unix`를 찾을 수 없습니다.	SPARC 기반 시스템의 PROM에서 `boot-device` 설정을 변경해야 할 수 있습니다. 기본 부트 장치 변경에 대한 자세한 내용은 “Oracle Solaris 11.2 시스템 부트 및 종료 ”의 “부트 속성 표시 및 설정”을 참조하십시오.
Oracle Solaris 부트 아카이브가 손상되었습니다. 또는 SMF 부트 아카이브 서비스가 실패했습니다. `svcs -x` 명령을 실행하는 경우 오류 메시지가 표시됩니다.	기본 부트 환경의 백업에 해당하는 두번째 부트 환경을 만듭니다. 기본 부트 환경을 부트할 수 없는 경우 백업 부트 환경을 부트합니다. 또는 Live CD 또는 USB 매체에서 부트할 수 있습니다.
`/etc/passwd` 파일에 잘못된 항목이 있습니다.	잘못된 `passwd` 파일에서 복구하는 방법에 대한 자세한 내용은 “Oracle Solaris 11.2 시스템 부트 및 종료 ”의 “매체에서 부트하여 알 수 없는 root 암호 문제를 해결하는 방법”을 참조하십시오.
x86 부트 로더(GRUB)가 손상되었습니다. 또는 GRUB 메뉴가 누락되었거나 손상되었습니다.	손상된 x86 부트 로더 또는 누락되었거나 손상된 GRUB 메뉴에서 복구하는 방법에 대한 자세한 내용은 “Oracle Solaris 11.2 시스템 부트 및 종료 ”의 “매체에서 부트하여 시스템 부트를 방해하는 GRUB 구성 관련 문제를 해결하는 방법”을 참조하십시오.
디스크 또는 다른 장치와 관련된 하드웨어 문제가 있습니다.	다음과 같이 하드웨어 연결을 확인하십시오. ■ 장비가 연결되어 있는지 확인합니다.

시스템이 부트되지 않는 원인	문제 해결 방법
	■ 모든 스위치가 제대로 설정되어 있는지 확인합니다. ■ 이더넷 케이블을 비롯하여 모든 커넥터와 케이블을 확인합니다. ■ 이러한 모든 단계가 효과가 없을 경우 시스템 전원을 껐다가 10-20초 후에 전원을 다시 켜십시오.

위 제안 조치를 통해 문제가 해결되지 않을 경우 현지 서비스 공급자에게 문의하십시오.

루트 암호를 잊어버렸거나 시스템을 부트할 수 없는 경우 수행할 작업

root 암호를 잊어버렸거나 시스템을 부트하지 못하는 문제가 발생한 경우 다음을 수행합니다.

- 시스템을 중지합니다.
- “Oracle Solaris 11.2 시스템 부트 및 종료 ”의 “매체에서 부트하여 알 수 없는 root 암호 문제를 해결하는 방법”의 지침을 따르십시오.
- root 암호가 문제인 경우 /etc/shadow 파일에서 루트 암호를 제거합니다.
- 시스템을 재부트합니다.
- 로그인하여 root 암호를 설정합니다.

시스템이 정지될 경우 수행할 작업

일부 소프트웨어 프로세스가 멈춘 경우 시스템이 충돌하는 것이 아니라 멈추거나 정지될 수 있습니다. 다음 단계에 따라 정지된 시스템을 복구하십시오.

1. 시스템에서 윈도우 환경이 실행되고 있는지 여부를 확인하고 다음 제안 사항을 따릅니다. 이러한 제안 사항으로 문제가 해결되지 않을 경우 2단계로 이동합니다.
   - 포인터가 명령을 입력 중인 창에 있는지 확인합니다.
   - 사용자가 실수로 Ctrl-s를 누른 경우 Ctrl-q를 눌러 화면을 고정합니다. Ctrl-s를 누르면 전체 화면이 고정되는 것이 아니라 창만 고정됩니다. 창이 고정된 경우 다른 창을 사용해 봅니다.
   - 가능한 경우 네트워크의 다른 시스템에서 원격으로 로그인합니다. pgrep 명령을 사용하여 정지된 프로세스를 찾습니다. 윈도우 시스템이 정지된 것 같으면 프로세스를 식별하여 강제 종료합니다.
2. 실행 중인 프로그램을 강제로 종료하고 core 파일을 기록하려면 **Ctrl-₩**을 누릅니다.
3. 실행 중일 수 있는 프로그램을 중단하려면 **Ctrl-c**를 누릅니다.

4. 원격으로 로그인하여 시스템 정지 원인이 되는 프로세스를 식별하여 강제 종료합니다.
5. 원격으로 로그인하여 `root` 역할을 맡고 시스템을 재부트합니다.
6. 시스템이 계속 응답하지 않을 경우 충돌 덤프를 강제로 실행하고 재부트합니다. 충돌 덤프 및 부트를 강제 수행하는 방법에 대한 자세한 내용은 "Oracle Solaris 11.2 시스템 부트 및 종료 "의 "시스템의 충돌 덤프 및 재부트 강제 수행"을 참조하십시오.
7. 시스템이 계속 응답하지 않을 경우 전원을 껐다가 1분 정도 후에 전원을 다시 켭니다.
8. 시스템이 전혀 응답하지 않는 경우 현지 서비스 공급자에게 문의하여 도움을 받으십시오.

♦ ♦ ♦ 3 장

3 파일 시스템 문제 해결

이 장에서는 다음을 포함하여 파일 시스템 문제를 해결하는 방법을 설명합니다.

파일 시스템이 가득 찬 경우 수행할 작업

루트(/) 파일 시스템 또는 기타 파일 시스템이 가득 찬 경우 콘솔 창에 다음 메시지가 표시됩니다.

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

파일 시스템이 가득 차는 원인은 여러 가지입니다. 다음 단원에서는 가득 찬 파일 시스템을 복구할 수 있는 여러 가지 시나리오에 대해 설명합니다.

큰 파일 또는 디렉토리가 만들어져 파일 시스템이 가득 참

오류 발생 원인	문제 해결 방법
사용자가 실수로 파일 또는 디렉토리를 잘못된 위치에 복사했습니다. 응용 프로그램이 충돌하고 큰 `core` 파일을 파일 시스템에 작성한 경우에도 이 오류가 발생합니다.	로그인하고 `root` 역할을 맡은 후 특정 파일 시스템에서 `ls -tl` 명령을 사용하여 새로 생성된 큰 파일을 식별한 후 제거합니다.

시스템 메모리 부족으로 인해 TMPFS 파일 시스템이 가득 참

오류 발생 원인	문제 해결 방법
TMPFS가 허용된 수를 초과하여 작성을 시도하거나 현재 프로세스 중 일부에서 많은 양의 메모리를 사용 중인 경우 이 오류가 발생할 수 있습니다.	tmpfs 관련 오류 메시지 복구에 대한 자세한 내용은 tmpfs(7FS) 매뉴얼 페이지를 참조하십시오.

복사 또는 복원 후 파일 ACL이 손실된 경우 수행할 작업

오류 발생 원인	문제 해결 방법
ACL이 있는 파일 또는 디렉토리가 /tmp 디렉토리에 복사되거나 복원되는 경우 ACL 속성이 손실됩니다. 일반적으로 /tmp 디렉토리는 UFS 파일 시스템 속성(예: ACL)을 지원하지 않는 임시 파일 시스템으로 마운트됩니다.	대신 /var/tmp 디렉토리에 파일을 복사하거나 복원합니다.

파일 액세스 문제 해결

사용자들은 종종 이전에 사용할 수 있었던 프로그램, 파일 또는 디렉토리에 액세스할 수 없어 문제가 발생할 경우 시스템 관리자에게 도움을 요청합니다.

이와 같은 문제가 발생할 경우 항상 다음 세 가지 측면 중 하나를 조사하십시오.

- 사용자의 검색 경로가 변경되었거나 검색 경로 내 디렉토리의 순서가 부적절한 것일 수 있습니다.
- 파일 또는 디렉토리의 권한이나 소유권이 부적절한 것일 수 있습니다.
- 네트워크를 통해 액세스된 시스템의 구성이 변경된 것일 수 있습니다.

이 장에서는 이와 같은 세 가지 측면에서 각각 문제를 인식하는 방법에 대해 간략하게 설명하고 가능한 해결 방법을 제안합니다.

검색 경로 문제 해결(Command not found)

Command not found 메시지는 다음 중 하나를 나타냅니다.

- 시스템에서 명령을 사용할 수 없습니다.
- 검색 경로에 명령 디렉토리가 없습니다.

검색 경로 문제를 해결하려면 명령이 저장된 디렉토리의 경로 이름을 알아야 합니다.

잘못된 버전의 명령이 있을 경우 동일한 이름의 명령이 있는 디렉토리가 검색 경로에 포함됩니다. 이 경우 적절한 디렉토리가 검색 경로의 뒷부분에 포함되거나 아예 표시되지 않을 수 있습니다.

`echo $PATH` 명령을 사용하여 현재 검색 경로를 표시할 수 있습니다.

`type` 명령을 사용하여 잘못된 버전의 명령을 실행 중인지 여부를 확인할 수 있습니다. 예를 들면 다음과 같습니다.

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

▼ 검색 경로 문제 진단 및 해결 방법

1. **현재 검색 경로를 표시하여 명령에 대한 디렉토리가 경로에 없거나 디렉토리의 철자가 잘못되었는지 확인합니다.**

   ```
   $ echo $PATH
   ```

2. **다음 사항을 확인합니다.**

   - 검색 경로가 올바른지 여부
   - 검색 경로가 다른 버전의 명령이 있는 다른 검색 경로 앞에 나열되는지 여부
   - 명령이 검색 경로 중 하나에 포함되어 있는지 여부

   경로를 수정해야 할 경우 3단계로 이동합니다. 그렇지 않을 경우 4단계로 이동합니다.

3. **다음 테이블과 같이 경로를 적합한 파일에 추가합니다.**

셸	파일	구문	주
bash 및 ksh93	`$HOME/.profile`	`$ PATH=`*`$HOME/bin:/sbin:/usr/local /bin ...`* `$ export PATH`	경로 이름은 콜론으로 구분합니다.

4. **다음과 같이 새 경로를 활성화합니다.**

셸	경로 위치	경로를 활성화하는 명령
bash 및 ksh93	`.profile`	`. $HOME/.profile`
	`.login`	`hostname$ source $HOME/.login`

5. **새 경로를 확인합니다.**

   ```
   $ which command
   ```

예 3-1 검색 경로 문제 진단 및 해결

다음 예에서는 type 명령을 사용할 때 mytool 실행 파일이 검색 경로 내 디렉토리에 없음을 보여줍니다.

```
$ 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
```

명령을 찾을 수 없는 경우 매뉴얼 페이지에서 해당 디렉토리 경로를 찾으십시오.

파일 및 그룹 소유권 변경

파일 및 디렉토리 소유권 변경은 다른 사용자가 관리자로 파일을 편집했기 때문인 경우가 많습니다. 새 사용자에 대한 홈 디렉토리를 만들 때 사용자를 홈 디렉토리 내 dot(.) 파일의 소유자로 설정해야 합니다. "." 파일을 소유하지 않은 사용자는 고유 홈 디렉토리에 파일을 만들 수 없습니다.

그룹 소유권이 변경되거나 사용자가 속한 그룹이 /etc/group 데이터베이스에서 삭제된 경우에도 액세스 문제가 발생할 수 있습니다.

액세스 문제가 발생한 파일의 권한 또는 소유권을 변경하는 방법에 대한 자세한 내용은 “Oracle Solaris 11.2의 파일 보안 및 파일 무결성 확인 ”의 1 장, “파일에 대한 액세스 제어”를 참조하십시오.

파일 액세스 문제 해결

사용자가 이전에 액세스할 수 있었던 파일 또는 디렉토리에 액세스할 수 없을 경우 파일 또는 디렉토리의 권한이나 소유권이 변경된 것일 수 있습니다.

네트워크 액세스 문제 인식

사용자가 rcp 원격 복사 명령을 사용하여 네트워크를 통해 파일을 복사할 때 문제가 발생할 경우 원격 시스템의 디렉토리 및 파일이 권한 설정을 통해 액세스를 제한한 것일 수 있습니다. 원격 시스템 및 로컬 시스템이 액세스를 허용하도록 구성되지 않은 경우에도 문제가 발생할 수 있습니다.

네트워크 문제 및 AutoFS를 통해 시스템에 액세스할 때 발생하는 문제에 대한 자세한 내용은 “Oracle Solaris 11.2의 네트워크 파일 시스템 관리 ”의 “NFS 문제 해결 전략”을 참조하십시오.

♦ ♦ ♦ 4 장

4 코어 파일을 사용하여 가능한 프로세스 실패 대비

이 장에서는 프로세스가 실패할 때 시스템에서 생성하는 코어 파일에 대한 사양을 설정하는 방법과 실패 이후 이러한 코어 파일을 검사하는 방법을 설명합니다.

다음은 이 장에 포함된 정보 목록입니다.

프로세스 실패 및 코어 파일 정보

프로세스나 응용 프로그램이 비정상적으로 종료되면 시스템에서는 자동으로 일련의 파일을 생성합니다. 이 프로세스를 코어 덤프로 설명할 수 있습니다. 생성되는 파일은 코어 파일입니다. 코어 파일은 프로세스 상태에 대한 추가 정보와 함께 종료 시 프로세스 주소 공간 내용의 디스크 복사본입니다. 일반적으로 코어 파일은 해당하는 응용 프로그램의 버그로 인해 발생하는 프로세스의 비정상적인 종료 후에 생성됩니다. 코어 파일은 프로세스 실패를 초래하는 문제를 진단하는 데 사용할 수 있는 중요한 정보를 제공합니다. "코어 파일 만들기 매개변수" [29]를 참조하십시오.

시스템 관리를 진행하면서 `coreadm` 명령을 사용하여 코어 파일에 대한 만들기 사양을 제어할 수 있습니다. 예를 들어 `coreadm` 명령을 사용하여 모든 프로세스 코어 파일이 단일 시스템 디렉토리에 지정되도록 시스템을 구성할 수 있으므로, 문제를 더욱 간편하게 추적할 수 있습니다. "코어 파일 사양 관리" [31]를 참조하십시오.

프로세스가 비정상적으로 종료되면 `mdb` 등의 디버거를 사용하거나 `proc` 도구를 사용하여 생성된 코어 파일을 검사할 수 있습니다. "프로세스 실패 후 코어 파일 검사" [34]를 참조하십시오.

코어 파일 만들기 매개변수

프로세스가 실패하면 시스템에서는 전역 코어 파일 이름 패턴과 프로세스별 코어 파일 이름 패턴을 사용하여 각 코어 파일 이름을 만드는 방식으로 실패한 각 프로세스에 대한 두 개의

코어 파일을 만들려고 합니다. coreadm 명령은 이러한 이름 패턴을 제어하고 코어 파일 위치를 지정합니다. 이 절에서는 일부 파일 경로 및 파일 이름 매개변수에 대해 설명합니다. 코어 덤프 프로세스에 대한 자세한 내용은 core(4) 매뉴얼 페이지를 참조하십시오. coreadm 옵션에 대한 자세한 내용은 coreadm(1M) 매뉴얼 페이지를 참조하십시오.

구성 가능한 코어 파일 경로

프로세스가 비정상적으로 종료될 때 기본적으로 현재 디렉토리에 코어 파일이 생성됩니다. 전역 코어 파일 경로를 사용으로 설정할 경우 프로세스가 비정상적으로 종료될 때마다 현재 작업 디렉토리와 전역 코어 파일 위치에 파일이 하나씩 생성될 수 있습니다. 사용되는 파일 경로는 구성 가능한 매개변수입니다.

두 개의 구성 가능한 코어 파일 경로는 다음과 같이 상호 독립적으로 사용 또는 사용 안함으로 설정할 수 있습니다.

- 프로세스별 코어 파일 경로: 기본값은 core이며 기본적으로 사용으로 설정되어 있습니다. 사용으로 설정할 경우 프로세스가 비정상적으로 종료될 때 프로세스별 코어 파일 경로로 인해 core 파일이 생성됩니다. 새 프로세스는 상위 프로세스에서 프로세스별 경로를 상속합니다.

  생성되는 프로세스별 코어 파일은 소유자 액세스에 대해 읽기/쓰기 권한을 가지는 프로세스 소유자가 소유합니다. 소유 사용자만 이 파일을 볼 수 있습니다.
- 전역 코어 파일 경로: 기본값은 core이며 기본적으로 사용 안함으로 설정되어 있습니다. 사용으로 설정할 경우 전역 코어 파일 경로를 사용하여 프로세스별 코어 파일과 컨텐츠가 동일한 추가 코어 파일이 생성됩니다.

  생성되는 전역 코어 파일은 루트에 대해서만 읽기/쓰기 권한을 가지는 루트가 소유합니다. 권한이 없는 사용자는 이 파일을 볼 수 없습니다.

참고 - 기본적으로 setuid 프로세스는 전역 또는 프로세스별 경로를 사용하여 코어 파일을 생성하지 않습니다.

확장된 코어 파일 이름

코어 파일의 이름에는 실패한 프로세스에 대한 정보 필드가 포함됩니다. 코어 파일 이름 필드에 대한 자세한 내용은 coreadm(1M) 매뉴얼 페이지를 참조하십시오. 이 절에서는 전역 변수를 중점적으로 설명합니다.

전역 core 파일 디렉토리를 사용으로 설정할 경우 다음 표의 설명에 따라 변수를 사용하여 core 파일을 서로 구별할 수 있습니다.

%d	최대 MAXPATHLEN자의 실행 파일 디렉토리 이름
%f	최대 MAXCOMLEN자의 실행 파일 이름
%g	유효 그룹 ID

%m	시스템 이름(uname -m)
%n	시스템 노드 이름(uname -n)
%p	프로세스 ID
%t	시간의 십진수 값(2)
%u	유효 사용자 ID
%z	프로세스가 실행된 영역의 이름(zonename)
%%	리터럴 %

예를 들어 /var/core/core.%f.%p가 전역 코어 파일 경로로 설정된다고 가정합니다. PID가 12345인 sendmail 프로세스가 비정상적으로 종료되면 /var/core/core.sendmail.12345가 core 파일로 생성됩니다.

코어 파일 덤프 성능 향상

코어 덤프에서 프로세스 이진 이미지의 일부분을 제외하면 시스템에서 코어 파일 덤프의 성능을 향상할 수 있습니다. coreadm 명령을 입력하여 코어 덤프 사양을 사용자 정의하면 코어 덤프에서 DISM 매핑, ISM 매핑 또는 시스템 V 공유 메모리 등을 제외하게 지정할 수 있습니다. 자세한 내용은 coreadm(1M) 매뉴얼 페이지를 참조하십시오.

코어 파일 사양 관리

다음과 같이 코어 파일을 관리할 수 있습니다.

현재 코어 덤프 구성 표시

옵션 없이 coreadm 명령을 사용하여 현재 코어 덤프 구성을 표시합니다.

```
$ 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
```

코어 파일 이름 패턴 설정

전역, 영역 또는 프로세스별 기준으로 코어 파일 이름 패턴을 설정할 수 있습니다. 또한 시스템을 재부트해도 지속되는 프로세스별 기본값을 설정할 수 있습니다.

예를 들어 다음 `coreadm` 명령을 사용하여 `init` 프로세스가 시작한 모든 프로세스에 대한 기본 프로세스별 코어 파일 패턴을 설정할 수 있습니다. 이 설정은 기본 코어 파일 패턴을 명시적으로 무시하지 않은 모든 프로세스에 적용됩니다. 이 설정은 시스템 재부트 시 지속됩니다.

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

다음 `coreadm` 명령을 사용하여 모든 프로세스에 대한 프로세스별 코어 파일 이름 패턴을 설정할 수 있습니다.

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

`$$` 기호는 현재 실행 중인 셸의 프로세스 ID에 대한 위치 표시자입니다. 모든 하위 프로세스는 프로세스별 코어 파일 이름 패턴을 상속합니다.

```
Here's another example:

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

또는 `root` 역할을 수행하고 전역 파일 이름 패턴을 설정합니다.

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

전역 또는 프로세스별 코어 파일 이름 패턴을 설정한 후에는 `coreadm -e` 명령을 통해 사용으로 설정해야 합니다.

사용자의 초기화 파일(예: `.profile`)에 명령을 삽입하여 사용자의 로그인 세션 중 실행되는 모든 프로세스에 대해 코어 파일 이름 패턴을 설정할 수 있습니다.

파일 경로 사용으로 설정

프로세스별 또는 전역 코어 파일 경로를 사용으로 설정할 수 있습니다.

- 프로세스별 코어 파일 경로를 사용으로 설정하려면 root 역할을 수행하고 다음 명령을 실행합니다.

```
# coreadm -e process
```

구성을 확인하려면 현재 프로세스 코어 파일 경로를 표시합니다.

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

- 전역 코어 파일 경로를 사용으로 설정하려면 root 역할을 수행하고 다음 명령을 실행합니다.

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

구성을 확인하려면 현재 프로세스 코어 파일 경로를 표시합니다.

```
# 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
```

코어 파일을 생성하도록 setuid 프로그램을 사용으로 설정

coreadm 명령으로 다음 경로를 설정하여 모든 시스템 프로세스에 대해 코어 파일을 생성하거나 프로세스별로 코어 파일을 생성하도록 setuid 프로그램을 사용 또는 사용 안함으로 설정할 수 있습니다.

- 전역 setuid 옵션을 사용으로 설정할 경우 전역 코어 파일 경로를 통해 시스템의 모든 setuid 프로그램이 코어 파일을 생성할 수 있습니다.
- 프로세스별 setuid 옵션을 사용으로 설정할 경우 프로세스별 코어 파일 경로를 통해 특정 setuid 프로세스가 코어 파일을 생성할 수 있습니다.

기본적으로 두 플래그는 사용 안함으로 설정되어 있습니다. 보안상 전역 코어 파일 경로는 /로 시작하는 전체 경로 이름이어야 합니다. 루트가 프로세스별 코어 파일을 사용 안함으로 설정할 경우 개별 사용자가 코어 파일을 얻을 수 없습니다.

setuid 코어 파일은 루트 액세스에 대해서만 읽기/쓰기 권한을 가지는 루트가 소유합니다. 일반 사용자가 setuid 코어 파일을 생성한 프로세스를 소유한 경우에도 일반 사용자는 해당 파일에 액세스할 수 없습니다.

자세한 내용은 coreadm(1M) 매뉴얼 페이지를 참조하십시오.

기본 코어 파일 설정으로 되돌리기

root로서 다음 명령의 하나를 실행하여 코어 파일 경로를 사용 안함으로 설정하고 코어 파일 이름 패턴을 제거합니다.

- 전역 코어 파일 설정:

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

참고 - ""는 공백 없는 빈 문자열입니다.

- 프로세스별 코어 파일 설정:

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

-d 옵션은 코어 파일 경로를 사용 안함으로 설정합니다. -g 옵션을 빈 문자열 변수와 함께 사용하면 코어 파일 이름 패턴이 제거됩니다. 코어 파일 경로 및 코어 파일 이름 패턴이 원래 기본 설정으로 돌아갑니다.

오래된 코어 파일 매개변수 수정

오류 메시지

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

원인

/etc/system 파일에서 setuid 코어 파일을 허용하는 오래된 매개변수가 있습니다.

해결 방법

allow_setid_core=1을 /etc/system 파일에서 제거합니다. 그런 다음 coreadm 명령을 사용하여 전역 setuid 코어 파일 경로를 사용으로 설정하십시오.

프로세스 실패 후 코어 파일 검사

proc 도구를 사용하면 실시간 프로세스뿐만 아니라 프로세스 코어 파일을 검사할 수 있습니다. proc 도구는 /proc 파일 시스템의 기능을 조작할 수 있는 유틸리티입니다.

해당 명령에 대한 프로세스 ID를 지정하는 것과 유사한 방법으로 명령줄에 코어 파일 이름을 지정하여 /usr/proc/bin/pstack, pmap, pldd, pflags 및 pcred 도구를 코어 파일에 적용할 수 있습니다.

proc 도구를 사용하여 코어 파일을 검사하는 방법에 대한 자세한 내용은 proc(1)을 참조하십시오.

예 4-1 proc 도구로 코어 파일 검사

```
$ ./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
```

♦ ♦ ♦ 5 장

5 시스템 로그 및 메시징 관리

이 장에서는 시스템 로그와 시스템 메시지를 보고 관리하는 방법을 설명합니다.

이 장에서는 다음 정보를 다룹니다.

rsyslogd로 확장된 시스템 로깅

이 Oracle Solaris 릴리스에는 시스템 로깅 관리를 위해 rsrsyslog 서비스를 설치하고 사용하는 옵션이 포함됩니다. rsyslog는 안정적인 확장된 syslog 데몬 구현으로, 필터링, TCP, 암호화, 고정밀 시간 기록, 출력 제어와 같은 여러 기능을 지원하는 모듈식 디자인입니다.

syslog SMF 서비스 svc:/system/system-log:기본값이 계속해서 기본 로깅 서비스로 사용됩니다. rsyslog 서비스를 사용하려면 rsyslog 패키지를 설치하고 rsyslog 서비스를 사용으로 설정해야 합니다.

▼ rsyslog 설치 및 사용으로 설정

1. **다음과 같이 서비스를 사용으로 설정하여 rsyslog 패키지가 시스템에 이미 설치되어 있는지 확인할 수 있습니다.**

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

rsyslog 패키지가 설치되어 있지 않으면 다음 메시지가 표시됩니다.

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

2. **rsyslog 패키지가 설치되어 있지 않으면 설치하십시오.**

```
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. **rsyslog의 인스턴스가 있는지 확인합니다.**

```
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
```

이 출력은 rsyslog 인스턴스가 있지만 사용 안함으로 설정되어 있음을 확인합니다.

4. **rsyslog 서비스로 전환합니다.**

```
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.
```

다음 순서 rsyslog가 설치되고 사용으로 설정된 후 /etc/rsyslog.conf에서 syslog를 구성할 수 있습니다.

시스템 메시지 관리

다음 절에서는 Oracle Solaris의 시스템 메시징 기능에 대해 설명합니다.

시스템 메시지 확인

시스템 메시지는 콘솔 장치에 표시됩니다. 대부분의 시스템 메시지 텍스트는 다음과 같이 표시됩니다.

[ID *msgid facility*.]

예를 들면 다음과 같습니다.

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

메시지가 커널에서 시작된 경우 커널 모듈 이름이 표시됩니다. 예를 들면 다음과 같습니다.

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

시스템이 충돌하면 시스템 콘솔에 다음과 같은 메시지가 표시될 수 있습니다.

```
panic: error message
```

가끔 비상 메시지 대신 다음 메시지가 표시될 수도 있습니다.

```
Watchdog reset !
```

오류 로깅 데몬(syslogd)이 자동으로 메시지 파일에 다양한 시스템 경고 및 오류를 기록합니다. 기본적으로 이러한 시스템 메시지는 시스템 콘솔에 표시되고 /var/adm 디렉토리에 저장되는 경우가 많습니다. 시스템 메시지 로깅을 설정하여 이러한 메시지가 저장되는 위치를 지정할 수 있습니다. 자세한 내용은 “시스템 메시지 로깅 사용자 정의” [41]를 참조하십시오. 이러한 메시지는 시스템 문제(예: 실패 예상 장치)를 알리는 것입니다.

/var/adm 디렉토리에는 여러 메시지 파일이 들어 있습니다. 가장 최근 메시지는 /var/adm/messages 파일(및 messages.*)에 있으며 가장 오래된 메시지는 messages.3 파일에 있습니다. 특정 기간이 경과되면(일반적으로 10일마다) 새 messages 파일이 생성됩니다. messages.0 파일의 이름은 messages.1로, messages.1의 이름은 messages.2로, messages.2의 이름은 messages.3으로 바뀝니다. 현재 /var/adm/messages.3 파일은 삭제됩니다.

/var/adm 디렉토리에는 메시지, 충돌 덤프 및 기타 데이터를 포함하는 큰 파일이 저장되므로 이 디렉토리는 많은 양의 디스크 공간을 사용할 수 있습니다. /var/adm 디렉토리가 너무 커지지 않도록 하고 다음 충돌 덤프가 저장될 수 있도록 하려면 주기적으로 불필요한 파일을 제거해야 합니다. crontab 파일을 사용하여 이 작업을 자동화할 수 있습니다. 이 작업을 자동화하는 방법에 대한 자세한 내용은 “Oracle Solaris 11.2의 장치 관리 ”의 “덤프 파일 제거” 및 “Oracle Solaris 11.2의 시스템 정보, 프로세스, 성능 관리 ”의 4 장, “시스템 작업 예약”을 참조하십시오.

▼ 시스템 메시지 확인 방법

- **dmesg 명령을 사용하여 시스템 충돌 또는 재부트로 생성된 최근 메시지를 표시합니다.**

```
$ dmesg
```

또는 more 명령을 사용하여 메시지 화면을 한 번에 하나씩 표시합니다.

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

예 5-1 시스템 메시지 확인

다음 예에서는 Oracle Solaris 10 시스템에 대한 dmesg 명령의 출력을 보여줍니다.

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

참조 자세한 내용은 dmesg(1M) 매뉴얼 페이지를 참조하십시오.

시스템 로그 교체

시스템 로그 파일을 교체하려면 루트 crontab 파일의 항목에서 logadm 명령을 사용하십시오. /usr/lib/newsyslog 스크립트는 더 이상 사용되지 않습니다.

시스템 로그 교체는 /etc/logadm.conf 파일에서 정의됩니다. 이 파일에는 syslogd 등의 프로세스에 대한 로그 교체 항목이 들어 있습니다. 예를 들어, /etc/logadm.conf 파일의 한 항목은 파일이 비어 있지 않은 경우 /var/log/syslog 파일이 매주 교체되도록 지정합니다. 가장 최근 syslog 파일은 syslog.0이 되고 다음 번 가장 최근 파일은 syslog.1이 됩니다. 여덟 개의 이전 syslog 로그 파일이 보관됩니다.

/etc/logadm.conf 파일에는 마지막 로그 교체가 발생한 시간 기록도 포함되어 있습니다.

logadm 명령을 사용하여 시스템 로깅을 사용자 정의하고 필요에 따라 /etc/logadm.conf 파일에서 로깅을 더 추가할 수 있습니다.

예를 들어, Apache 액세스 및 오류 로그를 교체하려면 다음 명령을 사용하십시오.

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

이 예에서 Apache access_log 파일은 크기가 100MB에 도달할 때 .0, .1 등의 접미어를 사용하여 교체되며, 열 개의 이전 access_log 파일 복사본이 보관됩니다. error_log는 크기가 10MB에 도달할 때 access_log 파일과 동일한 접미어 및 복사본 수를 사용하여 교체됩니다.

앞선 Apache 로그 교체 예에 대한 /etc/logadm.conf 항목은 다음과 유사하게 표시됩니다.

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

```
/var/apache/logs/access_log -s 100m
```

자세한 내용은 logadm(1M)을 참조하십시오.

root가 아닌 사용자에게 로그 파일 유지 관리 권한을 부여하려면 해당 사용자에 대한 권한 프로파일인 로그 관리를 허가할 수 있습니다. -P 옵션을 useradd 명령(새 사용자용) 또는 usermod 명령(기존 사용자용)과 함께 사용하여 해당 허가를 수행할 수 있습니다. 자세한 내용은 useradd(1M) 및 usermod(1M) 매뉴얼 페이지를 참조하십시오.

시스템 메시지 로깅 사용자 정의

/etc/syslog.conf 파일을 수정하여 다양한 시스템 프로세스로 생성된 추가 오류 메시지를 캡처할 수 있습니다. 기본적으로 /etc/syslog.conf 파일은 여러 시스템 프로세스 메시지를 /var/adm/messages 파일로 전달합니다. 충돌 및 부트 메시지도 여기에 저장됩니다. /var/adm 메시지를 확인하려면 시스템 메시지 확인 방법 [39]을 참조하십시오.

/etc/syslog.conf 파일의 두 열은 다음과 같이 탭으로 구분됩니다.

facility.level ... action

facility.level 메시지 또는 상태의 *facility*나 시스템 소스. 쉼표로 구분되는 기능 목록일 수 있습니다. 기능 값은 표 5-1. "syslog.conf 메시지의 소스 기능"에 나열되어 있습니다. *level*은 기록 중인 상태의 심각도 또는 우선 순위를 나타냅니다. 우선 순위 레벨은 표 5-2. "syslog.conf 메시지의 우선 순위 레벨"에 나열되어 있습니다.

항목이 다른 우선 순위에 대한 것일 경우 동일한 줄의 동일한 기능에 대해 두 개의 항목을 삽입하지 마십시오. syslog 파일에 우선 순위를 삽입하면 해당 우선 순위 이상의 모든 메시지가 기록되며 마지막 메시지가 우선합니다. 제공된 기능 및 레벨에 대해 syslogd가 해당 레벨 이상의 모든 메시지와 일치됩니다.

action action 필드는 메시지 전달 위치를 나타냅니다.

다음 예에서는 기본 /etc/syslog.conf 파일의 샘플 행을 보여줍니다.

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

이 경우 다음과 같은 사용자 메시지가 자동으로 기록됩니다.

- 사용자 오류는 콘솔에 출력되고 /var/adm/messages 파일에도 기록됩니다.
- 즉각적인 조치가 필요한 사용자 메시지(alert)는 root 및 operator 사용자에게 전송됩니다.

- 사용자 긴급 메시지는 개별 사용자에게 전송됩니다.

참고 - 로그 대상이 `/etc/syslog.conf` 파일에서 두 번 이상 지정된 경우 별도의 행에 항목을 지정하면 메시지가 잘못 기록될 수 있습니다. 각각 세미콜론으로 구분하여 한 행 항목에 여러 선택 항목을 지정할 수 있습니다.

다음 표에서는 가장 일반적인 오류 상태 소스를 보여줍니다. 표 5-2, "`syslog.conf` 메시지의 우선 순위 레벨"에서는 가장 일반적인 우선 순위를 심각도순으로 보여줍니다.

표 5-1 `syslog.conf` 메시지의 소스 기능

소스	설명
`kern`	커널
`auth`	인증
`daemon`	모든 데몬
`mail`	메일 시스템
`lp`	스풀링 시스템
`user`	사용자 프로세스

참고 - `/etc/syslog.conf` 파일에서 활성화할 수 있는 `syslog` 기능 수는 무제한입니다.

표 5-2 `syslog.conf` 메시지의 우선 순위 레벨

우선 순위	설명
`emerg`	시스템 긴급
`alert`	즉각적인 수정이 필요한 오류
`crit`	심각한 오류
`err`	기타 오류
`info`	정보 메시지
`debug`	디버깅에 사용되는 출력
`none`	이 설정은 출력을 기록하지 않습니다.

▼ 시스템 메시지 로깅 사용자 정의 방법

1. **`root` 역할 또는 `solaris.admin.edit/etc/syslog.conf` 권한이 부여된 역할을 맡습니다.**

   "Oracle Solaris 11.2의 사용자 및 프로세스 보안 "의 "지정된 관리 권한 사용"을 참조하십시오.

2. **`syslog.conf(4)`에 설명된 구문에 따라 `pfedit` 명령을 사용하여 `/etc/syslog.conf` 파일을 편집해서 메시지 소스, 우선 순위, 메시지 위치를 추가하거나 변경합니다.**

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

3. **변경 사항을 저장합니다.**

예 5-2 시스템 메시지 로깅 사용자 정의

이 샘플 `/etc/syslog.conf` `user.emerg` 기능은 사용자 긴급 메시지를 `root` 및 개별 사용자에게 전송합니다.

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

원격 콘솔 메시지를 사용으로 설정

다음과 같은 새로운 콘솔 기능을 통해 원격 시스템 문제 해결 성능이 향상되었습니다.

- `consadm` 명령을 사용하여 직렬 장치를 보조(또는 원격) 콘솔로 선택할 수 있습니다. 시스템 관리자는 `consadm` 명령을 사용하여 시스템이 실행 레벨 간에 전환될 때 `sulogin` 세션을 호스트하고 재지정된 콘솔 메시지를 표시하도록 직렬 포트를 하나 이상 구성할 수 있습니다. 이 기능을 사용하면 모뎀을 이용한 전화 접속을 통해 직렬 포트에 연결하여 콘솔 메시지를 모니터링하고 `init` 상태 전환에 참여할 수 있습니다. 자세한 내용은 `sulogin`(1M) 및 뒷부분의 단계별 절차를 참조하십시오.

  보조 콘솔로 구성된 포트를 사용하여 시스템에 로그인할 수 있는 상태에서는 출력 장치에 표시되는 정보가 기본 콘솔에도 표시됩니다. 부트 스크립트 또는 기타 응용 프로그램이 기본 콘솔에 대한 읽기 및 쓰기를 수행하는 경우 쓰기 출력은 모든 보조 콘솔에 표시되지만 입력 읽기는 기본 콘솔에서만 수행됩니다. 대화식 로그인 세션 중 `consadm` 명령을 사용하는 방법은 "대화식 로그인 세션 중 `consadm` 명령 사용 지침" [44]을 참조하십시오.
- 이제 콘솔 출력은 커널 및 새 의사 장치(`/dev/sysmsg`)에 기록된 `syslog` 메시지로 구성됩니다. 또한 `rc` 스크립트 시작 메시지가 `/dev/msglog`에 기록됩니다. 이전에는 이러한 메시지가 모두 `/dev/console`에 기록되었습니다.

  보조 콘솔에 표시되는 스크립트 메시지를 확인하려면 콘솔 출력을 `/dev/console`로 전달하는 스크립트를 `/dev/msglog`로 변경해야 합니다. 메시지가 보조 장치로 재지정되도록 하려면 `/dev/console`을 참조하는 프로그램을 명시적으로 수정하여 `syslog()` 또는 `strlog()`를 사용하도록 해야 합니다.
- `consadm` 명령은 데몬을 실행하여 보조 콘솔 장치를 모니터링합니다. 보조 콘솔로 지정된 디스플레이 장치(반송파 연결 해제, 정지 또는 손실)가 보조 콘솔 장치 목록에서 제거되고 더 이상 활성화되지 않습니다. 하나 이상의 보조 콘솔을 사용으로 설정하면 기본 콘솔에 메시지가 표시되지 않습니다. `/dev/console`에는 메시지가 계속 표시됩니다.

실행 레벨 전환 중 보조 콘솔 메시지 사용

실행 레벨 전환 중 보조 콘솔 메시지를 사용할 때는 다음 사항을 염두에 두십시오.

- 시스템이 부트될 때 실행되는 rc 스크립트에 대한 사용자 입력이 필요한 경우 보조 콘솔에서 입력을 가져올 수 없습니다. 입력은 기본 콘솔에서 가져와야 합니다.
- 실행 레벨 간의 전환 시 루트 암호에 대한 프롬프트를 표시하기 위해 init에 의해 호출되는 sulogin 프로그램이 루트 암호 프롬프트를 기본 콘솔 장치와 각 보조 장치에 전송하도록 수정되었습니다.
- 사용자는 sulogin을 직접 호출하지 않아야 합니다. 이 유틸리티를 사용하려면 사용자에게 solaris.system.maintenance authorization이 있어야 합니다.
- 시스템이 단일 사용자 모드이며 consadm 명령을 통해 하나 이상의 보조 콘솔이 사용으로 설정된 경우 올바른 루트 암호를 sulogin 프롬프트에 제공하기 위해 첫번째 장치에서 콘솔 로그인 세션이 실행됩니다. 콘솔 장치에서 올바른 암호가 수신되면 sulogin이 기타 모든 콘솔 장치의 입력을 사용 안함으로 설정합니다.
- 콘솔 중 하나가 단일 사용자 권한을 사용하는 경우 기본 콘솔 및 기타 보조 콘솔에 메시지가 표시됩니다. 이 메시지는 올바른 루트 암호를 승인하여 특정 장치가 콘솔로 지정되었음을 나타냅니다. 단일 사용자 셸이 실행되는 보조 콘솔에서 반송파 손실이 있을 경우 다음 두 가지 작업 중 하나가 발생할 수 있습니다.
  - 보조 콘솔이 실행 레벨 1의 시스템을 나타내는 경우 시스템이 계속 기본 실행 레벨로 실행됩니다.
  - 보조 콘솔이 실행 레벨 S의 시스템을 나타내는 경우 셸에서 init s 또는 shutdown 명령이 입력된 장치에 ENTER RUN LEVEL (0-6, s or S): 메시지가 표시됩니다. 해당 장치에 반송파가 없을 경우 반송파를 재설정하고 올바른 실행 레벨을 입력해야 합니다. init 또는 shutdown 명령은 실행 레벨 프롬프트를 다시 표시하지 않습니다.
- 직렬 포트를 사용하여 시스템에 로그인하고 다른 실행 레벨로의 전환을 위해 init 또는 shutdown 명령이 실행된 경우 이 장치가 보조 콘솔인지 여부에 관계없이 로그인 세션이 끊깁니다. 보조 콘솔 기능이 없는 릴리스에서도 이와 동일한 상황이 발생합니다.
- consadm 명령을 사용하여 보조 콘솔로 선택된 장치는 시스템이 재부트되거나 보조 콘솔의 선택이 해제될 때까지 보조 콘솔로 유지됩니다. 단, consadm 명령에는 시스템 재부트 시 장치를 보조 콘솔로 설정할 수 있는 옵션이 포함되어 있습니다. 단계별 지침은 뒷부분의 절차를 참조하십시오.

대화식 로그인 세션 중 consadm 명령 사용 지침

직렬 포트에 연결된 터미널을 사용하여 시스템에 로그인한 후 consadm 명령을 사용하여 터미널의 콘솔 메시지를 확인하는 방식으로 대화식 로그인 세션을 실행하려면 다음 동작에 유의하십시오.

- 보조 콘솔이 활성화된 상태에서 대화식 로그인 세션에 터미널을 사용하면 콘솔 메시지가 /dev/sysmsg 또는 /dev/msglog 장치로 전송됩니다.
- 터미널에서 명령을 실행하는 동안에는 입력이 기본 콘솔(/dev/console)이 아닌 대화형 세션으로 전달됩니다.
- init 명령을 실행하여 실행 레벨을 변경하면 원격 콘솔 소프트웨어가 대화형 세션을 강제 종료하고 sulogin 프로그램을 실행합니다. 이 단계에서 입력은 터미널에서 가져온 것만 승인되고 콘솔 장치에서 가져온 것처럼 처리됩니다. 따라서 실행 레벨 전환 중 보조 콘솔 메시지 사용의 설명에 따라 “실행 레벨 전환 중 보조 콘솔 메시지 사용” [43] 프로그램에 암호를 입력할 수 있습니다.

그런 다음 (보조) 터미널에서 올바른 암호를 입력하면 보조 콘솔이 대화식 `sulogin` 세션을 실행하고 기본 콘솔 및 기타 보조 콘솔을 잠급니다. 즉, 터미널이 시스템 콘솔로 작동합니다.

- 여기서 실행 레벨 3으로 변경하거나 다른 실행 레벨로 이동할 수 있습니다. 실행 레벨을 변경하면 `sulogin`이 다시 모든 콘솔 장치에서 실행됩니다. 종료하거나 시스템이 실행 레벨 3에 도달하도록 지정하면 모든 보조 콘솔의 입력 제공 기능이 손실됩니다. 해당 보조 콘솔은 콘솔 메시지용 디스플레이 장치로 되돌아갑니다.

  시스템이 시작되면 기본 콘솔 장치에서 `rc` 스크립트에 정보를 제공해야 합니다. 시스템이 다시 시작되면 `login` 프로그램이 직렬 포트에서 실행되므로 다른 대화식 세션에 다시 로그인할 수 있습니다. 장치를 보조 콘솔로 지정한 경우 터미널의 콘솔 메시지는 계속 표시되지만 터미널의 모든 입력은 대화형 세션으로 전달됩니다.

▼ 보조(원격) 콘솔을 사용으로 설정하는 방법

`consadm` 데몬은 `consadm` 명령을 통해 보조 콘솔이 추가되기 전까지 포트 모니터링을 시작하지 않습니다. 보안 기능으로 콘솔 메시지는 반송파가 끊기거나 보조 콘솔 장치의 선택이 해제될 때까지만 재지정됩니다. 따라서 포트에서 반송파를 설정해야만 `consadm` 명령을 성공적으로 사용할 수 있습니다.

보조 콘솔을 사용으로 설정하는 방법에 대한 자세한 내용은 `consadm`(1m) 매뉴얼 페이지를 참조하십시오.

1. **시스템에 로그인하고 `root` 역할을 맡습니다.**

   "Oracle Solaris 11.2의 사용자 및 프로세스 보안 "의 "지정된 관리 권한 사용"을 참조하십시오.

2. **보조 콘솔을 사용으로 설정합니다.**

   ```
   # consadm -a devicename
   ```

3. **현재 연결이 보조 콘솔인지 확인합니다.**

   ```
   # consadm
   ```

예 5-3 보조(원격) 콘솔을 사용으로 설정

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

▼ 보조 콘솔 목록 표시 방법

1. **시스템에 로그인하고 `root` 역할을 맡습니다.**

"Oracle Solaris 11.2의 사용자 및 프로세스 보안 "의 "지정된 관리 권한 사용"을 참조하십시오.

2. **다음 단계 중 하나를 선택합니다.**

   a. **보조 콘솔 목록을 표시합니다.**

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

   b. **영구 보조 콘솔 목록을 표시합니다.**

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

▼ 시스템 재부트 시 보조(원격) 콘솔을 사용으로 설정하는 방법

1. **시스템에 로그인하고 root 역할을 맡습니다.**
   "Oracle Solaris 11.2의 사용자 및 프로세스 보안 "의 "지정된 관리 권한 사용"을 참조하십시오.

2. **시스템 재부트 시 보조 콘솔을 사용으로 설정합니다.**

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

   그러면 장치가 영구 보조 콘솔 목록에 추가됩니다.

3. **장치가 영구 보조 콘솔 목록에 추가되었는지 확인합니다.**

   ```
   # consadm
   ```

예 5-4 시스템 재부트 시 보조(원격) 콘솔을 사용으로 설정

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

▼ 보조(원격) 콘솔을 사용 안함으로 설정하는 방법

1. **시스템에 로그인하고 root 역할을 맡습니다.**
   "Oracle Solaris 11.2의 사용자 및 프로세스 보안 "의 "지정된 관리 권한 사용"을 참조하십시오.

2. **다음 단계 중 하나를 선택합니다.**

   a. **보조 콘솔을 사용 안함으로 설정합니다.**

```
# consadm -d devicename
```

또는

b. **보조 콘솔을 사용 안함으로 설정하고 영구 보조 콘솔 목록에서 제거합니다.**

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

3. **보조 콘솔이 사용 안함으로 설정되었는지 확인합니다.**

```
# consadm
```

예 5-5 보조(원격) 콘솔을 사용 안함으로 설정

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

색인