Oracle® Solaris ZFS 관리 설명서

부품 번호: E38892–03
2013년 6월

ORACLE

130701@25097

목차

머리말

Oracle Solaris ZFS 관리 설명서는 Oracle Solaris ZFS 파일 시스템의 설정 및 관리에 대한 정보를 제공합니다.

본 설명서에서는 SPARC 기반 시스템과 x86 기반 시스템에 대한 정보를 모두 다룹니다.

주 – 본 Oracle Solaris 릴리스는 프로세서 아키텍처의 SPARC 및 x86 제품군을 사용하는 시스템을 지원합니다. 지원되는 시스템은 http://www.oracle.com/webfolder/technetwork/hcl/index.html에서 **Oracle Solaris 하드웨어 호환성 목록**을 참조하십시오. 이 설명서에서는 플랫폼 유형에 따른 구현 차이가 있는 경우 이에 대하여 설명합니다.

이 설명서의 대상

본 설명서는 Oracle Solaris ZFS 파일 시스템을 설정 및 관리하는 사용자를 대상으로 작성되었습니다. Oracle Solaris OS(운영 체제)나 다른 UNIX 버전의 사용 경험이 있어야 합니다.

이 설명서의 구성

다음 표는 이 설명서의 장을 설명합니다.

장	설명
1 장, “Oracle Solaris ZFS 파일 시스템(소개)”	ZFS 및 기능과 이점에 대한 개요를 제공합니다. 일부 기본 개념 및 용어도 다룹니다.
2 장, “Oracle Solaris ZFS 시작하기”	기본 풀과 파일 시스템으로 기본 ZFS 구성을 설정하는 방법에 대한 단계별 지침을 제공합니다. 또한 ZFS 파일 시스템을 만드는 데 필요한 하드웨어 및 소프트웨어를 제공합니다.
3 장, “Oracle Solaris ZFS 저장소 풀 관리”	ZFS 저장소 풀을 만들고 관리하는 방법에 대한 자세한 설명을 제공합니다.
4 장, “Oracle Solaris ZFS 루트 파일 시스템 설치 및 부트”	ZFS 파일 시스템을 설치 및 부트하는 방법을 설명합니다. Oracle Solaris Live Upgrade를 사용하여 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션하는 방법도 다룹니다.

장	설명
5 장, “Oracle Solaris ZFS 파일 시스템 관리”	ZFS 파일 시스템 관리에 대한 자세한 정보를 제공합니다. 계층적 파일 시스템 레이아웃, 등록 정보 상속, 자동 마운트 지점 관리 및 공유 상호 작용과 같은 개념을 다룹니다.
6 장, “Oracle Solaris ZFS 스냅샷 및 복제 작업”	ZFS 스냅샷 및 복제를 만들고 관리하는 방법을 설명합니다.
7 장, “ACL 및 속성을 사용하여 Oracle Solaris ZFS 파일 보호”	액세스 제어 목록(ACL)을 사용하여 표준 UNIX 권한보다 훨씬 세부적인 권한을 통해 ZFS 파일을 보호하는 방법을 설명합니다.
8 장, “Oracle Solaris ZFS 위임 관리”	ZFS 위임 관리를 사용하여 비권한 사용자에게 ZFS 관리 작업을 수행하도록 허용하는 방법을 설명합니다.
9 장, “Oracle Solaris ZFS 고급 주제”	ZFS 볼륨 사용, 영역이 설치된 Oracle Solaris 시스템에서 ZFS 사용, 대체 루트 풀 사용에 대한 정보를 제공합니다.
10 장, “Oracle Solaris ZFS 문제 해결 및 풀 복구”	ZFS 실패를 식별하고 이로부터 복구하는 방법을 설명합니다. 실패 예방을 위한 단계도 다룹니다.
11 장, “Oracle Solaris ZFS 권장 방법”	ZFS 저장소 풀과 파일 시스템을 만들고 모니터 및 유지 관리하는 권장 방법에 대해 설명합니다.
부록 A, “Oracle Solaris ZFS 버전 설명”	사용 가능한 ZFS 버전, 각 버전의 특징, 그리고 ZFS 버전 및 기능을 제공하는 Solaris OS를 설명합니다.

관련 설명서

다음 설명서에서 일반적인 Oracle Solaris 시스템 관리 항목에 관련된 정보를 찾을 수 있습니다.

- **시스템 관리 설명서: 고급 관리**
- **System Administration Guide: Devices and File Systems**
- **System Administration Guide: Security Services**

Oracle Support에 액세스

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

활자체 규약

다음 표는 이 책에서 사용되는 활자체 규약에 대해 설명합니다.

표 P–1 활자체 규약

활자체	설명	예
`AaBbCc123`	명령, 파일, 디렉토리 이름 및 컴퓨터 화면에 출력되는 내용입니다.	`.login` 파일을 편집하십시오. 모든 파일 목록을 보려면 `ls -a` 명령을 사용하십시오. `machine_name% you have mail.`
`AaBbCc123`	사용자가 입력하는 내용으로 컴퓨터 화면의 출력 내용과 대조됩니다.	`machine_name%` **`su`** `Password:`
aabbcc123	새로 나오는 용어, 강조 표시할 용어입니다. 명령줄 변수를 실제 이름이나 값으로 바꾸십시오.	`rm` *filename* 명령을 사용하여 파일을 제거합니다.
AaBbCc123	책 제목, 장, 절	**사용자 설명서**의 6장을 읽으십시오. **캐시**는 로컬로 저장된 복사본입니다. 파일을 저장하면 **안됩니다**. **주:** 일부 강조된 항목은 온라인에서 굵은체로 나타납니다.

명령 예의 셸 프롬프트

다음 표에는 Oracle Solaris OS에 포함된 셸의 UNIX 시스템 프롬프트 및 수퍼유저 프롬프트가 나와 있습니다. 명령 예에서 셸 프롬프트는 명령을 일반 사용자가 실행해야 하는지 아니면 권한이 있는 사용자가 실행해야 하는지를 나타냅니다.

표 P–2 셸 프롬프트

셸	프롬프트
Bash 셸, Korn 셸 및 Bourne 셸	`$`
수퍼유저용 Bash 셸, Korn 셸 및 Bourne 셸	`#`
C 셸	`machine_name%`
수퍼유저용 C 셸	`machine_name#`

◆ ◆ ◆ 1 장

1 Oracle Solaris ZFS 파일 시스템(소개)

이 장에서는 Oracle Solaris ZFS 파일 시스템 및 해당 기능과 이점에 대한 개요를 제공합니다. 이 장에서는 또한 이 설명서의 남은 부분에서 일반적으로 사용되는 일부 기본 용어에 대해서도 설명합니다.

이 장에서는 다음과 같은 내용을 다룹니다.

ZFS의 새로운 기능

이 절에서는 ZFS 파일 시스템의 새 기능을 요약해서 보여 줍니다.

ZFS 명령 사용법 향상된 기능

Oracle Solaris 10 1/13: `zfs` 및 `zpool` 명령에는 `zfs` 및 `zpool` 하위 명령과 지원되는 옵션에 대한 자세한 정보를 제공하기 위해 사용할 수 있는 `help` 하위 명령이 포함되어 있습니다. 예를 들면 다음과 같습니다.

```
# zfs help
The following commands are supported:
allow       clone       create      destroy     diff        get
groupspace  help        hold        holds       inherit     list
mount       promote     receive     release     rename      rollback
send        set         share       snapshot    unallow     unmount
unshare     upgrade     userspace
For more info, run: zfs help <command>
# zfs help create
usage:
        create [-p] [-o property=value] ... <filesystem>
        create [-ps] [-b blocksize] [-o property=value] ... -V <size> <volume>

# zpool help
The following commands are supported:
add      attach   clear    create   destroy  detach   export   get
help     history  import   iostat   list     offline  online   remove
replace  scrub    set      split    status   upgrade
For more info, run: zpool help <command>
# zpool help attach
usage:
        attach [-f] <pool> <device> <new-device>
```

자세한 내용은 `zfs(1M)` 및 `zpool(1M)`을 참조하십시오.

ZFS 스냅샷 향상된 기능

Oracle Solaris 10 1/13: 이 릴리스에는 다음과 같이 ZFS 스냅샷에 대한 향상된 기능이 포함됩니다.

- `zfs snapshot` 명령에는 이 명령에 대한 축약 구문을 제공하는 `snap` 별칭이 포함됩니다. 예를 들면 다음과 같습니다.

  ```
  # zfs snap -r users/home@snap1
  ```

- `zfs diff` 명령은 두 스냅샷 사이에 추가되었거나 수정된 모든 파일을 식별하기 위한 열거형 옵션 `-e`를 제공합니다. 생성된 출력 결과는 추가된 모든 파일을 식별하지만 가능한 삭제 항목을 제공하지 않습니다. 예를 들면 다음과 같습니다.

  ```
  # zfs diff -e tank/cindy@yesterday tank/cindy@now
  +       /tank/cindy/
  +       /tank/cindy/file.1
  ```

  또한 `-o` 옵션을 사용하여 표시하도록 선택한 필드를 식별할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs diff -e -o size -o name tank/cindy@yesterday tank/cindy@now
+       7       /tank/cindy/
+       206695  /tank/cindy/file.1
```

ZFS 스냅샷 만들기에 대한 자세한 내용은 6 장, “Oracle Solaris ZFS 스냅샷 및 복제 작업”을 참조하십시오.

향상된 aclmode 등록 정보

Oracle Solaris 10 1/13: aclmode 등록 정보는 chmod 작업 중 파일에 대한 ACL 권한이 수정될 때마다 ACL(액세스 제어 목록) 동작을 수정합니다. aclmode 등록 정보는 다음 등록 정보 값을 사용하여 재도입되었습니다.

- discard – aclmode 등록 정보가 discard인 파일 시스템은 파일 모드를 나타내지 않는 ACL 항목을 모두 삭제합니다. 이것이 기본값입니다.
- mask – aclmode 등록 정보가 mask인 파일 시스템은 사용자 또는 그룹 권한을 줄입니다. 파일 또는 디렉토리의 소유자와 동일한 UID를 가진 사용자 항목이 아닌 경우 그룹 권한 비트보다 크지 않도록 권한이 감소합니다. 이 경우 소유자 권한 비트보다 크지 않도록 ACL 권한이 감소합니다. 또한 명시적 ACL 세트 작업이 수행되지 않은 경우 모드 변경 후에도 마스크 값이 ACL을 유지합니다.
- passthrough – aclmode 등록 정보가 passthrough인 파일 시스템은 파일 또는 디렉토리의 새 모드를 나타내는 데 필요한 ACL 항목의 생성 외에 ACL에 대한 변경 사항이 없음을 나타냅니다.

자세한 내용은 예 7–13을 참조하십시오.

Oracle Solaris ZFS 설치 기능

Oracle Solaris 10 8/11: 이 릴리스에서는 다음과 같은 새로운 설치 기능을 사용할 수 있습니다.

- 텍스트 모드 설치 방법을 사용하여 ZFS 플래시 아카이브로 시스템을 설치할 수 있습니다. 자세한 내용은 예 4–3을 참조하십시오.
- Oracle Solaris Live Upgrade luupgrade 명령을 사용하여 ZFS 루트 플래시 아카이브를 설치할 수 있습니다. 자세한 내용은 예 4–8을 참조하십시오.
- Oracle Solaris Live Upgrade lucreate 명령을 사용하여 별도의 /var 파일 시스템을 지정할 수 있습니다. 자세한 내용은 예 4–5를 참조하십시오.

ZFS 전송 스트림의 향상된 기능

Oracle Solaris 10 8/11: 이 릴리스에서는 스냅샷 스트림에서 전송 및 수신되는 파일 시스템 등록 정보를 설정할 수 있습니다. 이러한 향상된 기능은 전송 스트림의 파일

시스템 등록 정보를 수신 파일 시스템에 적용하거나 mountpoint 등록 정보 값과 같은 로컬 파일 시스템 등록 정보가 수신될 때 이를 무시할지 여부를 결정할 수 있는 유연성을 제공합니다.

자세한 내용은 217 페이지 “ZFS 스냅샷 스트림에 다른 등록 정보 값 적용”을 참조하십시오.

ZFS 스냅샷 차이(zfs diff)

Oracle Solaris 10 8/11: 이 릴리스에서는 zfs diff 명령을 사용하여 ZFS 스냅샷 차이를 확인할 수 있습니다.

예를 들어, 다음 두 스냅샷이 생성된다고 가정해 보겠습니다.

```
$ ls /tank/cindy
fileA
$ zfs snapshot tank/cindy@0913
$ ls /tank/cindy
fileA  fileB
$ zfs snapshot tank/cindy@0914
```

예를 들어, 두 스냅샷 간의 차이를 확인하려면 다음과 비슷한 구문을 사용하십시오.

```
$ zfs diff tank/cindy@0913 tank/cindy@0914
M       /tank/cindy/
+       /tank/cindy/fileB
```

출력 결과에서 M은 디렉토리가 수정되었음을 나타냅니다. +는 fileB가 나중 스냅샷에 존재함을 나타냅니다.

자세한 내용은 209 페이지 “ZFS 스냅샷 차이 식별(zfs diff)”을 참조하십시오.

ZFS 저장소 풀 복구 및 성능의 향상된 기능

Oracle Solaris 10 8/11: 이 릴리스에서는 다음과 같은 새 ZFS 저장소 풀 기능이 제공됩니다.

- zpool import -m 명령을 사용하여 누락된 로그로 풀을 가져올 수 있습니다. 자세한 내용은 97 페이지 “누락된 로그 장치가 있는 풀 가져오기”를 참조하십시오.
- 읽기 전용 모드로 풀을 가져올 수 있습니다. 이 기능은 주로 풀 복구에 사용됩니다. 기본 장치가 고장나서 손상된 풀에 액세스할 수 없는 경우 읽기 전용 풀을 가져와서 데이터를 복구할 수 있습니다. 자세한 내용은 98 페이지 “읽기 전용 모드로 풀 가져오기”를 참조하십시오.
- 이 릴리스에서 생성된 RAID-Z(raidz1, raidz2 또는 raidz3) 저장소 풀은 읽기 I/O 처리 성능을 향상시키기 위해 자동으로 미러링된 일부 지연 시간에 민감한 메타 데이터가 포함됩니다. 최소한 풀 버전 29로 업그레이드된 기존 RAID-Z 풀의 경우 새로 작성되는 모든 데이터에 대해 일부 메타 데이터가 미러링됩니다.

RAID-Z 풀에서 미러링된 메타 데이터는 미러링된 저장소 풀이 제공하는 것과 비슷한 하드웨어 오류에 대한 추가 보호를 제공하지 **않습니다**. 미러링된 메타 데이터로는 추가 공간이 소비되지만 RAID-Z 보호는 이전 릴리스와 동일하게 유지됩니다. 이러한 향상된 기능은 오직 성능에 한정됩니다.

ZFS 동기식 동작 조정

Solaris 10 8/11: 이 릴리스에서는 sync 등록 정보를 사용하여 ZFS 파일 시스템의 동기식 동작을 확인할 수 있습니다.

기본 동기식 동작은 모든 동기식 파일 시스템 트랜잭션을 의도한 로그에 쓰고 모든 장치를 비워서 데이터가 안정되도록 하는 것입니다. 기본 동기식 동작을 사용 안함으로 설정하는 것은 권장되지 않습니다. 동기식 지원에 의존하는 응용 프로그램에 영향을 줄 수 있으며 데이터 손실이 발생할 수 있습니다.

sync 등록 정보는 파일 시스템이 만들어지기 전 또는 후에 설정할 수 있습니다. 어느 경우에든 등록 정보 값이 즉시 적용됩니다. 예를 들면 다음과 같습니다.

```
# zfs set sync=always tank/neil
```

zil_disable 매개변수는 sync 등록 정보가 포함된 Oracle Solaris 릴리스에서 더 이상 사용할 수 없습니다.

자세한 내용은 표 5–1을 참조하십시오.

향상된 ZFS 풀 메시지

Oracle Solaris 10 8/11: 이 릴리스에서는 -T 옵션을 사용하여 zpool list 및 zpool status 명령에 대한 간격 및 개수 값을 제공하여 추가 정보를 표시할 수 있습니다.

또한 다음과 같이 추가로 풀 스크러빙 및 리실버링 정보가 zpool status 명령으로 제공됩니다.

- 리실버링 진행 중 보고입니다. 예를 들면 다음과 같습니다.

```
scan: resilver in progress since Thu Jun  7 14:41:11 2012
    3.83G scanned out of 73.3G at 106M/s, 0h11m to go
    3.80G resilvered, 5.22% done
```

- 스크러빙 진행 중 보고입니다. 예를 들면 다음과 같습니다.

```
scan: scrub in progress since Thu Jun  7 14:59:25 2012
    1.95G scanned out of 73.3G at 118M/s, 0h10m to go
    0 repaired, 2.66% done
```

- 리실버링 완료 메시지입니다. 예를 들면 다음과 같습니다.

```
resilvered 73.3G in 0h13m with 0 errors on Thu Jun  7 14:54:16 2012
```

- 스크러빙 완료 메시지입니다. 예를 들면 다음과 같습니다.

```
   scan: scrub repaired 512B in 1h2m with 0 errors on Thu Jun  7 15:10:32 2012
```

- 진행 중인 스크러빙 취소 메시지입니다. 예를 들면 다음과 같습니다.

```
   scan: scrub canceled on Thu Jun  7 15:19:20 MDT 2012
```

- 스크러빙 및 리실버링 완료 메시지는 시스템 재부트 시에도 지속됩니다.

다음 구문에서는 간격 및 카운트 옵션을 사용하여 지속적인 풀 리실버링 정보를 표시할 수 있습니다. `-T d` 값을 사용하여 표준 날짜 형식으로 정보를 표시하거나 `-T u`를 사용하여 내부 형식으로 정보를 표시할 수 있습니다.

```
# zpool status -T d tank 3 2
Wed Nov 14 15:44:34 MST 2012
  pool: tank
 state: DEGRADED
status: One or more devices is currently being resilvered.  The pool will
        continue to function in a degraded state.
action: Wait for the resilver to complete.
  scan: resilver in progress since Wed Nov 14 15:44:34 2012
    2.96G scanned out of 4.19G at 189M/s, 0h0m to go
    1.48G resilvered, 70.60% done
config:

        NAME                        STATE     READ WRITE CKSUM
        tank                        DEGRADED     0     0     0
          mirror-0                  ONLINE       0     0     0
            c0t5000C500335F95E3d0   ONLINE       0     0     0
            c0t5000C500335F907Fd0   ONLINE       0     0     0
          mirror-1                  DEGRADED     0     0     0
            c0t5000C500335BD117d0   ONLINE       0     0     0
            c0t5000C500335DC60Fd0   DEGRADED     0     0     0  (resilvering)

errors: No known data errors
```

ZFS ACL 상호 운용성의 향상된 기능

Oracle Solaris 10 8/11: 이 릴리스에서는 다음과 같은 ACL의 향상된 기능이 제공됩니다.

- 단순 ACL에는 비정상적 권한을 제외하고 deny ACE(액세스 제어 항목)가 필요하지 않습니다. 예를 들어, `0644`, `0755` 또는 `0664` 모드에는 deny ACE가 필요하지 않지만 `0705`, `0060` 등의 모드에는 deny ACE가 필요합니다.

  이전 동작에는 단순 ACL(예: `644`)에 deny ACE가 포함됩니다. 예를 들면 다음과 같습니다.

```
# ls -v file.1
-rw-r--r--   1 root     root      206663 Jun 14 11:52 file.1
     0:owner@:execute:deny
     1:owner@:read_data/write_data/append_data/write_xattr/write_attributes
         /write_acl/write_owner:allow
     2:group@:write_data/append_data/execute:deny
     3:group@:read_data:allow
     4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
         /write_acl/write_owner:deny
```

```
        5:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
            :allow
```

단순 ACL(예: 644)에 대한 새 동작에는 deny ACE가 포함되지 않습니다. 예를 들면 다음과 같습니다.

```
# ls -v file.1
-rw-r--r--   1 root     root      206663 Jun 22 14:30 file.1
     0:owner@:read_data/write_data/append_data/read_xattr/write_xattr
         /read_attributes/write_attributes/read_acl/write_acl/write_owner
         /synchronize:allow
     1:group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
     2:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

- ACL이 원래의 수정되지 않은 권한을 보존하기 위해 상속 도중 더 이상 여러 ACE로 분할되지 않습니다. 대신, 파일 만들기 모드를 강제하기 위해 필요에 따라 권한이 수정됩니다.
- aclinherit 등록 정보 동작에는 해당 등록 정보가 restricted로 설정된 경우 권한 감소가 포함됩니다. 즉, ACL이 상속 도중에 더 이상 여러 ACE로 분할되지 않습니다.
- 새 권한 모드 계산 규칙에 따르면, ACL에 파일 소유자이기도 한 *user* ACE가 있는 경우 해당 권한이 권한 모드 계산에 포함됩니다. *group* ACE가 파일의 그룹 소유자인 경우 동일한 규칙이 적용됩니다.

자세한 내용은 7 장, “ACL 및 속성을 사용하여 Oracle Solaris ZFS 파일 보호”를 참조하십시오.

미러링된 ZFS 저장소 풀 분할(zpool split)

Oracle Solaris 10 9/10: 이 릴리스에서는 zpool split 명령을 사용하여 미러링된 저장소 풀을 분할하고, 원래 미러링된 풀에서 디스크를 분리하여 다른 동일한 풀을 만들 수 있습니다.

자세한 내용은 66 페이지 “미러링된 ZFS 저장소 풀을 분할하여 새로운 풀 만들기”를 참조하십시오.

새 ZFS 시스템 프로세스

Oracle Solaris 10 9/10: 이 릴리스에서는 각 ZFS 저장소 풀에 연관된 프로세스인 zpool-*poolname*이 포함됩니다. 이 프로세스의 스레드는 압축 및 체크섬 검증과 같이 풀과 연관된 I/O 작업을 처리하기 위한 풀의 I/O 처리 스레드입니다. 이 프로세스의 목적은 각 저장소 풀의 CPU 사용량을 표시하기 위한 것입니다.

이러한 실행 중인 프로세스에 대한 정보는 ps 및 prstat 명령을 사용하여 검토할 수 있습니다. 이러한 프로세스는 전역 영역에서만 사용할 수 있습니다. 자세한 내용은 SDC(7)을 참조하십시오.

ZFS 장치 교체의 향상된 기능

Oracle Solaris 10 9/10: 이 릴리스에서는 풀의 디스크가 더 큰 디스크로 교체될 때 시스템 이벤트 또는 *sysevent*가 제공됩니다. ZFS는 이러한 이벤트를 인식하고 autoexpand 등록 정보 설정에 따라 새로운 디스크 크기를 기반으로 풀을 조정할 수 있도록 향상되었습니다. 큰 디스크로 작은 디스크를 교체할 경우 autoexpand 풀 등록 정보를 사용하여 자동 풀 확장을 사용 또는 사용 안함으로 설정할 수 있습니다.

이러한 향상된 기능을 통해서는 풀을 내보내고 가져오거나 시스템을 재부트할 필요 없이 풀 크기를 늘릴 수 있습니다.

예를 들어, tank 풀에서는 자동 LUN 확장이 사용으로 설정되어 있습니다.

```
# zpool set autoexpand=on tank
```

또는 autoexpand 등록 정보를 사용으로 설정하여 풀을 만들 수 있습니다.

```
# zpool create -o autoexpand=on tank c1t13d0
```

autoexpand 등록 정보는 기본적으로 사용 안함으로 설정되므로 큰 디스크가 작은 디스크를 대체할 때 풀 크기를 확장할지 여부를 결정할 수 있습니다.

또한 zpool online -e 명령을 사용하여 풀 크기를 확장할 수도 있습니다. 예를 들면 다음과 같습니다.

```
# zpool online -e tank c1t6d0
```

또는 큰 디스크가 연결되었거나 zpool replace 명령을 사용하여 설정된 다음 autoexpand 등록 정보를 재설정할 수 있습니다. 예를 들어, 다음 풀은 8GB 디스크 한 개(c0t0d0)를 포함하여 만들어졌습니다. 8GB 디스크가 16GB 디스크(c1t13d0)로 교체되지만 autoexpand 등록 정보가 사용으로 설정되기 전에는 풀 크기가 확장되지 않습니다.

```
# zpool create pool c0t0d0
# zpool list
NAME   SIZE   ALLOC  FREE    CAP   HEALTH  ALTROOT
pool   8.44G  76.5K  8.44G     0%  ONLINE  -
# zpool replace pool c0t0d0 c1t13d0
# zpool list
NAME   SIZE   ALLOC  FREE    CAP   HEALTH  ALTROOT
pool   8.44G  91.5K  8.44G     0%  ONLINE  -
# zpool set autoexpand=on pool
# zpool list
NAME   SIZE   ALLOC  FREE    CAP   HEALTH  ALTROOT
pool   16.8G   91.5K  16.8G    0%  ONLINE  -
```

autoexpand 등록 정보를 사용으로 설정하지 않고 디스크를 확장할 수 있는 또 다른 방법은 장치가 이미 온라인 상태일지라도 zpool online -e 명령을 사용하는 것입니다. 예를 들면 다음과 같습니다.

```
# zpool create tank c0t0d0
# zpool list tank
NAME   SIZE   ALLOC  FREE    CAP   HEALTH  ALTROOT
tank   8.44G  76.5K  8.44G     0%  ONLINE  -
# zpool replace tank c0t0d0 c1t13d0
# zpool list tank
NAME   SIZE   ALLOC  FREE    CAP   HEALTH  ALTROOT
tank   8.44G  91.5K  8.44G     0%  ONLINE  -
# zpool online -e tank c1t13d0
# zpool list tank
NAME   SIZE   ALLOC  FREE    CAP   HEALTH  ALTROOT
tank   16.8G    90K  16.8G     0%  ONLINE  -
```

이 릴리스에서는 다음과 같은 추가적인 장치 교체의 향상된 기능이 포함되었습니다.

- 이전 릴리스에서는 교체 디스크의 크기가 약간 다를 경우 ZFS가 기존 디스크를 다른 디스크로 교체하거나 디스크를 연결할 수 없었습니다. 이 릴리스에서는 풀이 완전히 꽉 차지 않는 한 크기가 약간 다르더라도 기존 디스크를 다른 디스크로 교체하거나 새 디스크를 연결할 수 있습니다.
- 이 릴리스에서는 풀 크기를 확장하기 위해 시스템을 재부트하거나 풀을 내보내고 가져올 필요가 없습니다. 앞에서 설명한 것처럼 `autoexpand` 등록 정보를 사용으로 설정하거나 `zpool online -e` 명령을 사용하여 풀 크기를 확장할 수 있습니다.

장치 교체에 대한 자세한 내용은 72 페이지 “저장소 풀의 장치 교체”를 참조하십시오.

ZFS 및 플래시 설치 지원

Solaris 10 10/09: 이 릴리스에서는 ZFS 루트 풀의 Flash 아카이브를 식별하기 위해 JumpStart 프로파일을 설정할 수 있습니다. 자세한 내용은 114 페이지 “ZFS 루트 파일 시스템 설치(Oracle Solaris Flash 아카이브 설치)”를 참조하십시오.

ZFS 환경의 영역 마이그레이션

Solaris 10 5/09: 이 릴리스에서는 Oracle Solaris Live Upgrade를 통해 ZFS 환경에서 영역 마이그레이션에 대한 지원이 확장되었습니다. 자세한 내용은 136 페이지 “Oracle Solaris Live Upgrade를 사용하여 영역이 있는 시스템 마이그레이션 또는 업그레이드(Solaris 10 5/09 이상)”를 참조하십시오.

이 릴리스에서 알려진 문제에 대한 목록을 보려면 Solaris 10 5/09 릴리스 노트를 참조하십시오.

ZFS 설치 및 부트 지원

Solaris 10 10/08: 이 릴리스에서는 ZFS 루트 파일 시스템을 설치하고 부트할 수 있습니다. 초기 설치 옵션 또는 JumpStart 기능을 사용하여 ZFS 루트 파일 시스템을 설치할 수

있습니다. Oracle Solaris Live Upgrade 기능을 사용하여 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션할 수 있습니다. 스왑 및 덤프 장치에 대한 ZFS 지원도 제공됩니다. 자세한 내용은 4 장, “Oracle Solaris ZFS 루트 파일 시스템 설치 및 부트”를 참조하십시오.

이 릴리스에서 알려진 문제에 대한 목록을 보려면 Solaris 10 10/08 릴리스 노트를 참조하십시오.

ZFS 웹 기반 관리

Solaris 10 6/06 릴리스: 웹 기반 ZFS 관리 도구인 ZFS 관리 콘솔을 사용하면 다음과 같은 관리 작업을 수행할 수 있습니다.

- 새 저장소 풀 만들기
- 기존 풀에 용량 추가
- 다른 시스템으로 저장소 풀 이동(내보내기)
- 이전에 내보낸 저장소 풀을 가져와서 다른 시스템에서 사용 가능하도록 지정
- 저장소 풀에 대한 정보 보기
- 파일 시스템 만들기
- 볼륨 만들기
- 파일 시스템 또는 볼륨에 대한 스냅샷 만들기
- 이전 스냅샷에 파일 시스템 롤백

보안 웹 브라우저를 통해 ZFS 관리 콘솔에 액세스할 수 있습니다.

```
https://system-name:6789/zfs
```

적합한 URL을 입력해도 ZFS 관리 콘솔에 연결할 수 없으면 서버가 시작되지 않았을 수 있습니다. 서버를 시작하려면 다음 명령을 실행합니다.

```
# /usr/sbin/smcwebserver start
```

시스템이 부트될 때 서버가 자동으로 실행되도록 하려면 다음 명령을 실행합니다.

```
# /usr/sbin/smcwebserver enable
```

주 – Solaris Management Console(`smc`)을 사용해서는 ZFS 저장소 풀이나 파일 시스템을 관리할 수 없습니다.

Oracle Solaris ZFS란?

ZFS 파일 시스템은 오늘날의 다른 모든 파일 시스템에서는 제공되지 않는 다양한 기능과 이점을 통해 파일 시스템의 관리 방식을 근본적으로 바꿔 놓은 파일 시스템입니다. ZFS는 강력하고, 확장성이 뛰어나며, 쉽게 관리할 수 있습니다.

ZFS 풀링된 저장소

ZFS에는 실제 저장소의 관리를 위해 **저장소 풀**이라는 개념이 사용됩니다. 지금까지 파일 시스템은 단일 물리적 장치를 기반으로 만들어졌습니다. 다중 장치를 나열하고 데이터 중복성을 제공하기 위해 도입된 **볼륨 관리자**라는 개념은 단일 장치 표시를 제공함으로써 다중 장치의 이점을 활용하기 위해 파일 시스템을 수정할 필요가 없었습니다. 이러한 설계는 파일 시스템이 가상화된 볼륨에서 데이터의 물리적 배치를 제어할 수 있는 수단이 없기 때문에 또 다른 복잡성을 추가하고 궁극적으로 특정 파일 시스템의 발전을 저해한 원인이 되었습니다.

ZFS에는 볼륨 관리자가 없습니다. 사용자에게 가상 볼륨을 만들도록 강제하는 대신 ZFS는 장치를 저장소 풀로 결합합니다. 저장소 풀은 저장소의 물리적 특성(장치 레이아웃, 데이터 중복성 등)을 기술하고 파일 시스템을 만들 수 있는 모든 데이터 저장소 역할을 수행합니다. 파일 시스템은 더 이상 개별 장치로 제한되지 않기 때문에 풀의 모든 파일 시스템이 디스크 공간을 공유할 수 있습니다. 저장소 풀에 할당된 디스크 공간 내에서 파일 시스템 크기가 자동으로 증가하기 때문에 더 이상 파일 시스템의 크기를 미리 결정할 필요도 없습니다. 새 저장소가 추가되면 추가 작업 없이도 풀에 있는 모든 파일 시스템이 즉시 추가 공간을 사용할 수 있습니다. 많은 경우에 저장소 풀은 가상 메모리 시스템과 비슷하게 작동합니다. 메모리 DIMM이 시스템에 추가되면 사용자가 운영 체제에서 메모리를 구성하고 이를 개별 프로세스에 지정하기 위한 명령을 수행할 필요가 없습니다. 모든 시스템 프로세스에서 추가 메모리가 자동으로 사용됩니다.

트랜잭션 개념

ZFS는 트랜잭션 파일 시스템입니다. 즉, 디스크에서 파일 시스템 상태는 항상 일관적인 상태입니다. 기존의 파일 시스템은 제자리에서 데이터를 겹쳐 씁니다. 즉, 데이터 블록이 할당되는 시간과 이를 디렉토리에 연결하는 시간 사이에 시스템 전원이 꺼지면 파일 시스템이 일관적이지 않은 상태가 됩니다. 지금까지 이러한 문제는 `fsck` 명령을 사용하여 해결되었습니다. 이 명령은 파일 시스템 상태를 검토 및 확인하고 프로세스 중에 발견된 모든 비일관성을 복구하려고 시도합니다. 이러한 비일관적인 파일 시스템 문제는 관리자에게 많은 부담을 주었으며, `fsck` 명령으로는 모든 발생 가능한 문제에 대한 해결책을 보장할 수 없었습니다. 보다 최근에는 파일 시스템에 **저널링**이라는 개념이 도입되었습니다. 저널링 프로세스는 별도의 저널에 작업을 기록하여, 시스템 문제가 발생할 경우 안전하게 **재생**할 수 있도록 하는 프로세스입니다. 이 프로세스는 데이터를 두 번 기록해야 하기 때문에 불필요한 오버헤드를 가져왔으며, 저널을 올바르게 재생할 수 없는 등의 새로운 문제도 자주 발생했습니다.

트랜잭션 파일 시스템에서는 **쓰기 작업 시 복사**라는 개념을 사용하여 데이터가 관리됩니다. 데이터는 겹쳐 쓸 필요가 없으며, 어떠한 작업 시퀀스라도 완전히 커밋되거나 완전히 무시됩니다. 따라서 정전이 되거나 시스템 중단이 발생하더라도 파일 시스템이 손상될 가능성이 전혀 없습니다. 가장 최근에 작성된 데이터 일부는 손실될 수도 있지만 파일 시스템 자체는 항상 일관적인 상태로 유지됩니다. 또한 동기 데이터(O_DSYNC 플래그를 사용하여 기록된 데이터)는 항상 반환 전에 기록하도록 보장되므로 절대로 손실되지 않습니다.

체크섬 및 자체 치유 데이터

ZFS에서는 모든 데이터 및 메타 데이터가 사용자가 선택 가능한 체크섬 알고리즘을 통해 확인됩니다. 체크섬 확인 기능을 제공하는 기존 파일 시스템에서는 볼륨 관리 계층과 기존의 파일 시스템 설계로 인한 필요로 블록별 기준으로 작업이 수행되었습니다. 따라서 이러한 기존 설계에서는 전체 블록을 잘못된 위치에 기록하는 등의 특정 문제로 인해 잘못되었지만 체크섬 오류가 없는 데이터가 발생할 수 있습니다. ZFS 체크섬은 이러한 오류를 감지하고 이를 올바르게 복구할 수 있는 방식으로 저장됩니다. 모든 체크섬 확인 및 데이터 복구는 파일 시스템 계층에서 수행되며 응용 프로그램에는 영향을 주지 않습니다.

또한 ZFS는 자체 치유 데이터를 제공합니다. ZFS는 저장소 풀에 다양한 레벨의 데이터 중복성을 지원합니다. 잘못된 데이터 블록이 감지되면 ZFS는 다른 중복 복사본에서 올바른 데이터를 가져와서 잘못된 데이터를 올바른 데이터로 교체합니다.

비교할 수 없는 확장성

ZFS 파일 시스템의 핵심적인 디자인 요소는 확장성입니다. 파일 시스템 자체는 128비트이며, 저장소에 대해 256 퀴드릴리온 제타바이트가 허용됩니다. 모든 메타 데이터는 동적으로 할당되므로 inode를 미리 할당하거나 파일 시스템을 처음 만들 때 파일 시스템의 확장성을 미리 제한할 필요도 없습니다. 모든 알고리즘은 확장성을 염두에 두고 작성되었습니다. 디렉토리에는 최대 2^{48}(256 트릴리온)개의 항목을 포함할 수 있으며 파일 시스템 수 또는 하나의 파일 시스템 내에 포함할 수 있는 파일 수에도 제한이 없습니다.

ZFS 스냅샷

스냅샷은 파일 시스템 또는 볼륨에 대한 읽기 전용 복사본입니다. 스냅샷은 쉽고 빠르게 만들 수 있습니다. 처음에 스냅샷은 풀 내에서 추가 디스크 공간을 소비하지 않습니다.

활성 데이터 세트 내의 데이터가 변경되면 이전 데이터를 계속 참조하기 때문에 스냅샷에서 디스크 공간이 소비됩니다. 결과적으로 스냅샷은 해당 데이터가 풀로 다시 돌아가지 않도록 방지합니다.

간소화된 관리

ZFS의 가장 중요한 특징은 매우 간소화된 관리 모델을 제공한다는 점입니다. 계층적인 파일 시스템 레이아웃 사용, 등록 정보 상속, 마운트 지점 및 NFS 공유 의미에 대한 자동 관리를 지원하는 ZFS에서는 여러 명령을 수행하거나 구성 파일을 편집할 필요 없이 파일 시스템을 쉽게 만들고 관리할 수 있습니다. 쿼터 또는 예약을 쉽게 설정하고, 압축을 설정 또는 해제하고, 여러 파일 시스템에 대한 마운트 지점을 관리하는 등의 작업을 단일 명령을 통해 쉽게 수행할 수 있습니다. 별도의 볼륨 관리자 명령을 배우지 않아도 장치를 검사하거나 교체할 수 있습니다. 파일 시스템 스냅샷 스트림을 전송 및 수신할 수 있습니다.

ZFS는 쿼터, 예약, 압축 및 마운트 지점과 같이 등록 정보에 대한 간소화된 관리를 지원하는 계층을 통해 파일 시스템을 관리합니다. 이 모델에서 파일 시스템은 중앙 제어 지점이 됩니다. 새로운 디렉토리를 만드는 것과 동일한 정도로 파일 시스템 자체가 매우 간단하므로 각 사용자, 프로젝트, 작업 공간 등에 따라 각각의 파일 시스템을 만드는 것이 좋습니다. 이러한 설계 덕분에 세밀하게 조정된 관리 지점을 정의할 수 있습니다.

ZFS 용어

이 절에서는 이 설명서 전체에 사용되는 기본적인 용어에 대해 설명합니다.

대체 부트 환경
: `lucreate` 명령으로 만들고 `luupgrade` 명령으로 업데이트할 수 있는 부트 환경이지만 활성 또는 기본 부트 환경이 아닙니다. 대체 부트 환경은 `luactivate` 명령을 실행하여 기본 부트 환경이 될 수 있습니다.

체크섬
: 파일 시스템 블록에서 데이터의 256비트 해시입니다. 체크섬 성능은 간단하고 속도가 빠른 fletcher4(기본값)로부터 SHA256과 같이 암호화 방식으로 강력한 해시까지 다양할 수 있습니다.

복제본
: 초기 컨텐츠가 스냅샷의 컨텐츠와 동일한 파일 시스템입니다.

  복제본에 대한 자세한 내용은 210 페이지 “ZFS 복제본 개요”를 참조하십시오.

데이터 세트
: 복제본, 파일 시스템, 스냅샷 및 볼륨과 같은 ZFS 구성 요소를 나타내는 일반적인 이름입니다.

  각 데이터 세트는 ZFS 이름 공간에서 고유한 이름으로 식별됩니다. 데이터 세트는 다음과 같은 형식을 사용하여 식별됩니다.

  pool/*path*[*@snapshot*]

  pool
  : 데이터 세트가 포함된 저장소 풀의 이름을 식별합니다.

  path
  : 데이터 세트 구성 요소에 대한 슬래시로 구분된 경로 이름입니다.

snapshot 데이터 세트의 스냅샷을 식별하는 선택적인 구성 요소입니다.

데이터 세트에 대한 자세한 내용은 5 장, “Oracle Solaris ZFS 파일 시스템 관리”를 참조하십시오.

파일 시스템 표준 시스템 이름 공간 내에 마운트되고 다른 파일 시스템과 같이 동작하는 `filesystem` 유형의 ZFS 데이터 세트입니다.

파일 시스템에 대한 자세한 내용은 5 장, “Oracle Solaris ZFS 파일 시스템 관리”를 참조하십시오.

미러 두 개 이상의 디스크에 데이터의 동일한 복사본을 저장하는 가상 장치입니다. 미러에서 특정 디스크가 실패하면 해당 미러의 다른 디스크가 동일한 데이터를 제공할 수 있습니다.

풀 사용 가능한 저장소의 레이아웃 및 물리적 특성을 설명하는 장치의 논리적 그룹입니다. 데이터 세트에 대한 디스크 공간은 풀에서 할당됩니다.

저장소 풀에 대한 자세한 내용은 3 장, “Oracle Solaris ZFS 저장소 풀 관리”를 참조하십시오.

기본 부트 환경 대체 부트 환경을 작성하기 위해 `lucreate` 명령에 사용되는 부트 환경입니다. 기본적으로 기본 부트 환경은 현재 부트 환경입니다. 이러한 기본값은 `lucreate -s` 옵션을 사용하여 대체할 수 있습니다.

RAID-Z 여러 디스크에 데이터 및 패리티를 저장하는 가상 장치입니다. RAID-Z에 대한 자세한 내용은 47 페이지 “RAID-Z 저장소 풀 구성”을 참조하십시오.

리실버링 하나의 장치에서 다른 장치로 데이터를 복사하는 프로세스를 **리실버링**이라고 부릅니다. 예를 들어, 미러 구성 요소가 교체되거나 오프라인 상태가 되면 최신 미러 장치의 데이터가 새로 복원된 미러 구성 요소로 복사됩니다. 기존의 볼륨 관리 제품에서는 이러한 프로세스를 **미러 재동기화**라고 부릅니다.

ZFS 리실버링에 대한 자세한 내용은 290 페이지 “리실버링 상태 보기”를 참조하십시오.

snapshot 지정된 시점의 파일 시스템 또는 볼륨의 읽기 전용 복사본입니다.

스냅샷에 대한 자세한 내용은 203 페이지 “ZFS 스냅샷 개요”를 참조하십시오.

가상 장치 실제 장치, 파일 또는 장치 모음일 수 있는 풀의 논리적 장치입니다.

가상 장치에 대한 자세한 내용은 55 페이지 “저장소 풀 가상 장치 정보 표시”를 참조하십시오.

volume — 블록 장치를 나타내는 데이터 세트입니다. 예를 들어 ZFS 볼륨을 스왑 장치로 만들 수 있습니다.

ZFS 볼륨에 대한 자세한 내용은 261 페이지 “ZFS 볼륨”을 참조하십시오.

ZFS 구성 요소 명명 요구 사항

데이터 세트 및 풀과 같은 각 ZFS 구성 요소는 다음 규칙에 따라 이름을 지정해야 합니다.

- 각 구성 요소는 다음 네 개의 특수 문자와 영숫자 문자만 포함할 수 있습니다.
  - 밑줄(_)
  - 하이픈(-)
  - 콜론(:)
  - 마침표(.)
- 풀 이름은 문자로 시작해야 하며, 영숫자 문자와 밑줄(_), 대시(-) 및 마침표(.)만 포함할 수 있습니다. 다음 풀 이름 제한을 확인하십시오.
  - 시작 시퀀스 `c[0-9]`는 허용되지 않습니다.
  - `log` 이름은 예약되어 있습니다.
  - `mirror`, `raidz`, `raidz1`, `raidz2`, `raidz3` 또는 `spare`로 시작하는 이름은 예약되어 있으므로 허용되지 않습니다.
  - 풀 이름은 퍼센트 기호(%)를 포함하지 않아야 합니다.
- 데이터 세트 이름은 영숫자 문자로 시작해야 합니다.
- 데이터 세트 이름은 퍼센트 기호(%)를 포함하지 않아야 합니다.

또한 비어 있는 구성 요소는 허용되지 않습니다.

Oracle Solaris ZFS와 전통적인 파일 시스템의 차이

- 30 페이지 “ZFS 파일 시스템 세분성”
- 30 페이지 “ZFS 디스크 공간 계산”
- 32 페이지 “ZFS 파일 시스템 마운트”
- 32 페이지 “기존 볼륨 관리”
- 32 페이지 “NFSv4 기반 Solaris ACL 모델”

ZFS 파일 시스템 세분성

이전에는 파일 시스템이 하나의 장치로, 그리고 이로 인해 해당 장치의 크기로 제약되었습니다. 파일 제약 조건으로 인해 기존 파일 시스템을 만들고 다시 만드는 작업은 시간이 오래 걸리고 어렵기도 했습니다. 기존의 볼륨 관리 제품이 이 프로세스를 관리하는 데 도움이 되었습니다.

ZFS 파일 시스템은 특정 장치로 제약되지 않으므로 디렉토리를 만드는 과정처럼 쉽고 빠르게 만들 수 있습니다. ZFS 파일 시스템은 상주하는 저장소 풀에 할당된 디스크 공간 내에서 자동으로 커집니다.

여러 사용자 하위 디렉토리를 관리하려는 경우 파일 시스템(예: `/export/home`)을 하나만 만들지 않고 사용자당 하나씩 만들 수 있습니다. 계층에 포함된 종속 파일 시스템이 상속할 수 있는 등록 정보를 적용하여 여러 파일 시스템을 간편하게 설정하고 관리할 수 있습니다.

파일 시스템 계층을 만드는 방법을 보여 주는 예는 38 페이지 "ZFS 파일 시스템 계층 만들기"를 참조하십시오.

ZFS 디스크 공간 계산

ZFS는 풀링된 저장소라는 개념을 기반으로 합니다. 물리적 저장소에 매핑되는 일반 파일 시스템과 달리, 풀의 모든 ZFS 파일 시스템은 풀에서 사용 가능한 저장소를 공유합니다. 따라서 특정 파일 시스템이 비활성 상태인 경우에도 풀의 다른 파일 시스템이 디스크 공간을 사용하거나 해제함에 따라 유틸리티(예: `df`)가 보고하는 사용 가능한 디스크 공간이 바뀔 수 있습니다.

쿼터를 사용하여 최대 파일 시스템 크기를 제한할 수 있습니다. 쿼터에 대한 자세한 내용은 195 페이지 "ZFS 파일 시스템에 대한 쿼터 설정"을 참조하십시오. 예약을 사용하여 파일 시스템이 지정된 디스크 공간을 사용하도록 보장할 수 있습니다. 예약에 대한 자세한 내용은 198 페이지 "ZFS 파일 시스템에 대한 예약 설정"을 참조하십시오. 이 모델은 여러 디렉토리가 동일한 파일 시스템(예: `/home`)에서 마운트되는 NFS 모델과 매우 유사합니다.

ZFS에서는 모든 메타 데이터가 동적으로 할당되는 반면, 대부분의 다른 파일 시스템에서는 많은 양의 메타 데이터가 미리 할당되므로 파일 시스템을 만들 때 이 메타 데이터에 대한 공간이 바로 계산되어야 합니다. 즉, 파일 시스템에서 지원하는 총 파일 수가 미리 정해집니다. 하지만 ZFS는 필요에 따라 메타 데이터를 할당하므로 초기 공간 계산이 필요하지 않고 파일 수도 사용 가능한 디스크 공간으로만 제한됩니다. ZFS의 경우 `df -g` 명령 출력 결과를 다른 파일 시스템과 다르게 해석해야 합니다. 보고되는 `total files`는 풀에서 사용 가능한 저장소의 양을 기반으로 한 예측일 뿐입니다.

ZFS는 트랜잭션 파일 시스템입니다. 대부분의 파일 시스템 수정 사항은 트랜잭션 그룹으로 번들되고 비동기적으로 디스크에 커밋됩니다. 디스크에 커밋되기 전까지의

수정 사항을 **보류 중인 변경 사항**이라고 합니다. 사용된 디스크 공간, 사용 가능한 디스크 공간 및 파일 또는 파일 시스템에서 참조하는 디스크 공간은 보류 중인 변경 사항으로 간주되지 않습니다. 일반적으로 변경 사항은 몇 초 동안 보류됩니다. `fsync(3c)` 또는 `O_SYNC`를 사용하여 디스크에 변경 사항을 커밋해도 디스크 공간 사용량 정보가 바로 업데이트되지는 않습니다.

UFS 파일 시스템에서 `du` 명령은 파일 내 데이터 블록의 크기를 보고합니다. ZFS 파일 시스템에서 `du`는 디스크에 저장된 파일의 실제 크기를 보고합니다. 이 크기에는 메타데이터와 압축이 포함됩니다. 실제로 이 보고 기능은 "이 파일을 제거할 경우 확보되는 추가 공간의 크기"를 확인하는 데 도움이 됩니다. 따라서 압축이 해제된 경우에도 ZFS와 UFS에서 다른 결과가 나타납니다.

`df` 명령에서 보고된 공간 사용을 `zfs list` 명령과 비교할 경우 `df`는 파일 시스템 크기만이 아니라 풀 크기를 보고한다는 것에 주의합니다. 또한 `df`는 종속 파일 시스템 또는 스냅샷 존재 여부를 인식하지 못합니다. 압축, 할당량 등의 ZFS 등록 정보가 파일 시스템에 설정된 경우 `df`에서 보고된 공간 사용을 조정하기 어려울 수 있습니다.

보고된 공간 사용에도 영향을 줄 수 있는 다음 시나리오를 고려해 보십시오.

- `recordsize`보다 큰 파일의 경우 파일의 마지막 블록은 일반적으로 약 1/2 정도 차 있습니다. 기본 `recordsize`가 128KB로 설정된 경우 파일당 약 64KB가 낭비되어 큰 영향을 줄 수 있습니다. RFE 6812608을 통합하면 이 시나리오가 해결됩니다. 압축을 사용으로 설정하면 이 문제를 해결할 수 있습니다. 데이터가 이미 압축된 경우에도 마지막 블록에서 사용되지 않은 부분은 채우기가 0이며 효과적으로 압축됩니다.
- RAIDZ-2 풀에서 각 블록은 2개 섹터(512바이트 청크) 이상의 패리티 정보를 사용합니다. 패리티 정보에 사용되는 공간은 보고되지 않지만 각각 다르고 작은 블록의 경우 비율이 훨씬 더 클 수 있으므로 공간 보고에 영향을 줄 수 있습니다. `recordsize`가 512바이트로 설정된 경우 이 영향이 극대화되며, 각 512바이트 논리 블록이 1.5KB(공간의 3배)를 사용합니다. 저장 중인 데이터에 관계없이 공간 효율성이 주요 관심사인 경우 `recordsize`를 기본값(128KB)으로 그대로 두고 압축을 사용으로 설정해야 합니다(기본값 `lzjb`).
- `df` 명령은 중복 제거된 파일 데이터를 인식하지 못합니다.

공간 부족 동작

파일 시스템 스냅샷은 ZFS에서 저렴한 비용으로 간편하게 만들 수 있습니다. 스냅샷은 대부분의 ZFS 환경에 공통됩니다. ZFS 스냅샷에 대한 자세한 내용은 6 장, “Oracle Solaris ZFS 스냅샷 및 복제 작업”을 참조하십시오.

스냅샷이 있으면 디스크 공간을 확보하려고 시도할 때 예상치 않은 동작이 발생할 수 있습니다. 일반적으로 적합한 권한이 부여된 경우 전체 파일 시스템에서 파일을 제거할 수 있으며 이 작업으로 인해 파일 시스템에서 사용 가능한 디스크 공간이 늘어납니다.

하지만 제거할 파일이 파일 시스템의 스냅샷에 존재할 경우 파일 삭제 작업으로도 디스크 공간이 확보되지 않습니다. 파일에 사용되는 블록이 스냅샷에서 계속 참조됩니다.

따라서 이름 공간의 새 상태가 반영되도록 디렉토리의 새 버전을 만들어야 하므로 파일 삭제 작업으로 인해 더 많은 디스크 공간이 사용될 수 있습니다. 즉, 파일을 제거하려고 시도할 때 예상치 않은 ENOSPC 또는 EDQUOT 오류가 발생할 수 있습니다.

ZFS 파일 시스템 마운트

ZFS는 복잡성을 줄여 관리 작업을 간소화합니다. 예를 들어, 기존 파일 시스템을 사용하는 경우 새 파일 시스템을 추가할 때마다 /etc/vfstab 파일을 편집해야 합니다. ZFS는 파일 시스템의 등록 정보에 따라 자동으로 파일 시스템을 마운트하고 마운트 해제하므로 이와 같은 편집 작업을 수행할 필요가 없습니다. /etc/vfstab 파일에서 ZFS 항목을 관리하지 않아도 됩니다.

ZFS 파일 시스템 마운트 및 공유에 대한 자세한 내용은 188 페이지 “ZFS 파일 시스템 마운트”를 참조하십시오.

기존 볼륨 관리

25 페이지 “ZFS 풀링된 저장소”에 설명된 대로 ZFS는 별도의 Volume Manager를 필요로 하지 않습니다. ZFS는 원시 장치에서 작동하므로 소프트웨어 또는 하드웨어에 논리적 볼륨으로 구성된 저장소 풀을 만들 수 있습니다. ZFS는 원시 물리적 장치를 사용할 때 최적으로 작동하므로 이 구성은 권장되지 않습니다. 논리적 볼륨을 사용하면 성능 또는 안정성이 저하되거나 성능과 안정성이 모두 저하될 수 있으므로 논리적 볼륨은 사용하지 않아야 합니다.

NFSv4 기반 Solaris ACL 모델

이전 버전의 Solaris OS에서는 주로 POSIX ACL 드래프트 사양을 기반으로 한 ACL 구현을 지원했습니다. POSIX 드래프트 기반 ACL은 UFS 파일을 보호하는 데 사용됩니다. NFSv4 사양을 기반으로 하는 새로운 Solaris ACL 모델이 ZFS 파일을 보호하는 데 사용됩니다.

새로운 Solaris ACL 모델의 주요 차이점은 다음과 같습니다.

- 이 모델은 NFSv4 사양을 기반으로 하며 NT 스타일 ACL과 유사합니다.
- 이 모델은 보다 세분화된 액세스 권한 세트를 제공합니다.
- ACL은 setfacl 및 getfacl 명령이 아닌 chmod 및 ls 명령으로 설정되고 표시됩니다.

- 다양한 상속 의미에 따라 디렉토리의 액세스 권한이 하위 디렉토리에 적용되는 방식 등이 지정됩니다.

ZFS 파일에 ACL 사용에 대한 자세한 내용은 7 장, “ACL 및 속성을 사용하여 Oracle Solaris ZFS 파일 보호”를 참조하십시오.

◆ ◆ ◆ 2 장

2 Oracle Solaris ZFS 시작하기

이 장에서는 기본적인 Oracle Solaris ZFS 구성 설정에 대한 단계별 지침을 제공합니다. 이 장을 숙지하면 ZFS 명령 작동 방식의 기본 사항을 파악하고 기본 풀 및 파일 시스템을 만들 수 있습니다. 이 장에서는 포괄적인 개요를 제공하지 않으므로 자세한 내용은 후속 장을 참조하십시오.

이 장에서는 다음과 같은 내용을 다룹니다.

ZFS 권한 프로파일

수퍼유저(루트) 계정을 사용하지 않고 ZFS 관리 작업을 수행하려면 다음 프로파일 중 하나로 ZFS 관리 작업을 수행하는 역할이라고 가정합니다.

- ZFS 저장소 관리 – ZFS 저장소 풀 내에서 장치를 만들고 삭제, 조작하는 권한을 제공합니다.
- ZFS 파일 시스템 관리 – ZFS 파일 시스템을 만들고 삭제, 수정하는 권한을 제공합니다.

역할 만들기 또는 지정에 대한 자세한 내용은 **System Administration Guide: Security Services**를 참조하십시오.

ZFS 파일 시스템 관리를 위해 RBAC 역할을 사용하는 것 외에도, 분산된 ZFS 관리 작업에 ZFS 위임 관리의 사용을 고려해 볼 수 있습니다. 자세한 내용은 8 장, "Oracle Solaris ZFS 위임 관리"를 참조하십시오.

ZFS 하드웨어 및 소프트웨어 요구 사항과 권장 사항

ZFS 소프트웨어를 사용하기 전에 다음과 같은 하드웨어 및 소프트웨어 요구 사항과 권장 사항을 검토해야 합니다.

- 지원되는 Oracle Solaris 릴리스를 실행하는 SPARC 또는 x86 기반 시스템을 사용합니다.
- 저장소 풀에 필요한 최소 디스크 공간은 64MB입니다. 최소 디스크 크기는 128MB입니다.
- ZFS 성능을 최적화하려면 작업 부하를 기준으로 메모리 요구 사항의 크기를 지정합니다.
- 미러링된 풀 구성을 만드는 경우 컨트롤러를 여러 개 사용합니다.

기본 ZFS 파일 시스템 만들기

ZFS 관리는 간소화에 중점을 두고 설계되었습니다. 설계 목표 중 하나가 사용 가능한 파일 시스템을 만드는 데 필요한 명령 수를 줄이는 것입니다. 예를 들어, 새 풀을 만들 때 ZFS 파일 시스템이 새로 만들어지고 자동으로 마운트됩니다.

다음 예에서는 tank라는 이름의 기본적인 미러링된 저장소 풀과 tank라는 이름의 ZFS 파일 시스템을 하나의 명령으로 만드는 방법을 보여 줍니다. 전체 디스크 /dev/dsk/c1t0d0 및 /dev/dsk/c2t0d0을 사용할 수 있다고 가정합니다.

```
# zpool create tank mirror c1t0d0 c2t0d0
```

중복 ZFS 풀 구성에 대한 자세한 내용은 47 페이지 “ZFS 저장소 풀의 복제 기능”을 참조하십시오.

새 ZFS 파일 시스템인 tank는 필요에 따라 사용 가능한 디스크 공간을 사용할 수 있으며 /tank에 자동으로 마운트됩니다.

```
# mkfile 100m /tank/foo
# df -h /tank
Filesystem             size   used  avail capacity  Mounted on
tank                   80G   100M    80G     1%    /tank
```

풀 내에서 추가 파일 시스템을 만들 수도 있습니다. 파일 시스템은 동일한 풀에서 여러 데이터 세트를 관리할 수 있도록 관리 지점을 제공합니다.

다음 예에서는 저장소 풀 tank에서 fs라는 이름의 파일 시스템을 만드는 방법을 보여 줍니다.

```
# zfs create tank/fs
```

새 ZFS 파일 시스템인 tank/fs는 필요에 따라 사용 가능한 디스크 공간을 사용할 수 있으며 /tank/fs에 자동으로 마운트됩니다.

```
# mkfile 100m /tank/fs/foo
# df -h /tank/fs
Filesystem             size   used  avail capacity  Mounted on
tank/fs                 80G   100M    80G     1%    /tank/fs
```

일반적으로 조직 요구 사항에 맞는 파일 시스템 계층을 만들고 구성할 수 있습니다. ZFS 파일 시스템 계층을 만드는 방법은 38 페이지 "ZFS 파일 시스템 계층 만들기"를 참조하십시오.

기본 ZFS 저장소 풀 만들기

이전 예에서는 ZFS의 간소화에 대해 설명했습니다. 이 장의 나머지 부분에서는 사용자 환경에서 발생하는 것과 유사한 보다 완전한 예를 제공합니다. 첫번째 작업은 저장소 요구 사항을 식별하고 저장소 풀을 만드는 것입니다. 풀은 저장소의 물리적 특성을 기술하므로 파일 시스템 생성 전에 만들어져야 합니다.

▼ ZFS 저장소 풀에 대한 저장소 요구 사항 식별 방법

1 저장소 풀에서 사용 가능한 장치를 확인합니다.

저장소 풀을 만들기 전에 데이터를 저장할 장치를 결정해야 합니다. 해당 장치는 128MB 이상의 디스크여야 하며 운영 체제의 다른 부분에서 사용되고 있지 않아야 합니다. 장치는 미리 포맷된 디스크의 개별 슬라이스 또는 ZFS가 하나의 큰 슬라이스로 포맷한 전체 디스크일 수 있습니다.

38 페이지 "ZFS 저장소 풀을 만드는 방법"의 저장소 예에서는 전체 디스크 `/dev/dsk/c1t0d0` 및 `/dev/dsk/c2t0d0`을 사용할 수 있다고 가정합니다.

디스크에 대한 자세한 내용과 디스크 사용 및 레이블 지정 방법은 43 페이지 "ZFS 저장소 풀의 디스크 사용"을 참조하십시오.

2 데이터 복제를 선택합니다.

ZFS는 풀에서 발생 가능한 하드웨어 오류 유형을 결정하는 데이터 복제의 여러 유형을 지원합니다. ZFS는 비중복(스트라이프) 구성을 비롯하여 미러링과 RAID-Z(RAID-5 변형)를 지원합니다.

38 페이지 "ZFS 저장소 풀을 만드는 방법"의 예에서는 사용 가능한 두 개 디스크의 기본 미러링이 사용됩니다.

ZFS 복제 기능에 대한 자세한 내용은 47 페이지 "ZFS 저장소 풀의 복제 기능"을 참조하십시오.

▼ ZFS 저장소 풀을 만드는 방법

1 **루트로 로그인하거나 적합한 ZFS 권한 프로파일을 가진 동등한 역할을 수락합니다.**

ZFS 권한 프로파일에 대한 자세한 내용은 35 페이지 "ZFS 권한 프로파일"을 참조하십시오.

2 **저장소 풀에 대한 이름을 선택합니다.**

이 이름은 `zpool` 및 `zfs` 명령을 사용할 때 저장소 풀을 식별하는 데 사용됩니다. 원하는 풀 이름을 선택합니다. 단, 29 페이지 "ZFS 구성 요소 명명 요구 사항"의 명명 요구 사항을 충족해야 합니다.

3 **풀을 만듭니다.**

예를 들어, 다음 명령은 `tank`라는 이름이 지정된 미러링된 풀을 만듭니다.

```
# zpool create tank mirror c1t0d0 c2t0d0
```

하나 이상의 장치가 다른 파일 시스템을 포함하거나 사용 중인 경우 명령을 통해 풀을 만들 수 없습니다.

저장소 풀 만들기에 대한 자세한 내용은 49 페이지 "ZFS 저장소 풀 만들기"를 참조하십시오. 사용 중인 장치 확인 방법에 대한 자세한 내용은 56 페이지 "사용 중인 장치 감지"를 참조하십시오.

4 **결과를 확인합니다.**

`zpool list` 명령을 사용하여 성공적으로 풀을 만들었는지 여부를 확인할 수 있습니다.

```
# zpool list
NAME                    SIZE    ALLOC   FREE    CAP  HEALTH     ALTROOT
tank                     80G    137K     80G     0%  ONLINE     -
```

풀 상태 확인에 대한 자세한 내용은 82 페이지 "ZFS 저장소 풀 상태 질의"를 참조하십시오.

ZFS 파일 시스템 계층 만들기

데이터를 저장할 저장소 풀을 만든 후에는 파일 시스템 계층을 만들 수 있습니다. 계층은 간단하면서도 강력한 정보 구성 방식이며, 파일 시스템을 사용했던 기존 사용자에게도 매우 친숙합니다.

ZFS를 통해 각 파일 시스템에 상위가 하나만 포함되는 계층으로 파일 시스템을 구성할 수 있습니다. 계층의 루트는 항상 풀 이름입니다. ZFS는 전체 파일 시스템 트리에서 공통 등록 정보를 쉽고 빠르게 설정할 수 있도록 등록 정보 상속을 지원하는 방식으로 이 계층을 활용합니다.

▼ ZFS 파일 시스템 계층 확인 방법

1 **파일 시스템 세분성을 선택합니다.**

ZFS 파일 시스템은 중앙 관리 지점으로, 경량형이며 쉽게 만들 수 있습니다. 사용자 또는 프로젝트당 하나의 파일 시스템을 설정할 모델을 사용하는 것이 좋습니다. 이 모델에서는 등록 정보, 스냅샷 및 백업을 사용자 또는 프로젝트 단위로 제어할 수 있습니다.

39 페이지 "ZFS 파일 시스템을 만드는 방법"에서는 두 개의 ZFS 파일 시스템인 `jeff`와 `bill`이 만들어집니다.

파일 시스템 관리에 대한 자세한 내용은 5 장, "Oracle Solaris ZFS 파일 시스템 관리"를 참조하십시오.

2 **유사한 파일 시스템을 그룹화합니다.**

ZFS를 통해 유사한 파일 시스템이 그룹화될 수 있도록 파일 시스템을 계층으로 구성할 수 있습니다. 이 모델은 등록 정보 제어 및 파일 시스템 관리를 위한 중앙 관리 지점을 제공합니다. 유사한 파일 시스템은 공통 이름으로 만들어져야 합니다.

39 페이지 "ZFS 파일 시스템을 만드는 방법"의 예에서는 두 개의 파일 시스템이 `home`이라는 이름의 파일 시스템에 배치됩니다.

3 **파일 시스템 등록 정보를 선택합니다.**

대부분의 파일 시스템 특성은 등록 정보로 제어됩니다. 이러한 등록 정보는 파일 시스템 마운트 위치, 파일 시스템 공유 방식, 파일 시스템의 압축 사용 여부, 쿼터 적용 여부 등 다양한 동작을 제어합니다.

39 페이지 "ZFS 파일 시스템을 만드는 방법"의 예에서는 `/export/zfs/` *user*에 마운트된 모든 홈 디렉토리가 NFS를 통해 공유되며 압축이 사용으로 설정됩니다. 또한 사용자 `jeff`에 대해 10GB의 쿼터가 적용됩니다.

등록 정보에 대한 자세한 내용은 169 페이지 "ZFS 등록 정보 소개"를 참조하십시오.

▼ ZFS 파일 시스템을 만드는 방법

1 **루트로 로그인하거나 적합한 ZFS 권한 프로파일을 가진 동등한 역할을 수락합니다.**

ZFS 권한 프로파일에 대한 자세한 내용은 35 페이지 "ZFS 권한 프로파일"을 참조하십시오.

2 **원하는 계층을 만듭니다.**

이 예에서는 개별 파일 시스템의 컨테이너로 사용되는 파일 시스템이 만들어집니다.

```
# zfs create tank/home
```

3 상속된 등록 정보를 설정합니다.

파일 시스템 계층이 설정되면 모든 사용자와 공유할 등록 정보를 설정합니다.

```
# zfs set mountpoint=/export/zfs tank/home
# zfs set sharenfs=on tank/home
# zfs set compression=on tank/home
# zfs get compression tank/home
NAME             PROPERTY       VALUE                      SOURCE
tank/home        compression    on                         local
```

파일 시스템이 만들어진 경우 파일 시스템 등록 정보를 설정할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs create -o mountpoint=/export/zfs -o sharenfs=on -o compression=on tank/home
```

등록 정보 및 등록 정보 상속에 대한 자세한 내용은 169 페이지 “ZFS 등록 정보 소개”를 참조하십시오.

다음으로 tank 풀의 home 파일 시스템 아래에 개별 파일 시스템이 그룹화됩니다.

4 개별 파일 시스템을 만듭니다.

파일 시스템이 만들어지고 등록 정보가 home 레벨에서 변경되었을 수 있습니다. 파일 시스템이 사용되고 있는 동안에는 모든 등록 정보를 동적으로 변경할 수 있습니다.

```
# zfs create tank/home/jeff
# zfs create tank/home/bill
```

이러한 파일 시스템은 상위에서 등록 정보 값을 상속하므로 /export/zfs/ *user*에 자동으로 마운트되며 NFS를 통해 공유됩니다. /etc/vfstab 또는 /etc/dfs/dfstab 파일을 편집할 필요가 없습니다.

파일 시스템을 만드는 방법은 166 페이지 “ZFS 파일 시스템 만들기”를 참조하십시오.

파일 시스템 마운트 및 공유에 대한 자세한 내용은 188 페이지 “ZFS 파일 시스템 마운트”를 참조하십시오.

5 파일 시스템 관련 등록 정보를 설정합니다.

이 예에서는 사용자 jeff에게 10GB의 쿼터가 지정됩니다. 이 등록 정보는 풀에서 사용 가능한 공간에 관계없이 사용자가 사용할 수 있는 공간을 제한합니다.

```
# zfs set quota=10G tank/home/jeff
```

6 결과를 확인합니다.

zfs list 명령을 통해 사용 가능한 파일 시스템 정보를 확인합니다.

```
# zfs list
NAME                   USED  AVAIL  REFER  MOUNTPOINT
tank                  92.0K  67.0G   9.5K  /tank
tank/home             24.0K  67.0G     8K  /export/zfs
tank/home/bill           8K  67.0G     8K  /export/zfs/bill
tank/home/jeff           8K  10.0G     8K  /export/zfs/jeff
```

사용자 jeff가 사용 가능한 공간은 10GB뿐인 반면, 사용자 bill은 전체 풀(67GB)을 사용할 수 있습니다.

파일 시스템 상태 확인에 대한 자세한 내용은 181 페이지 “ZFS 파일 시스템 정보 질의”를 참조하십시오.

디스크 공간 사용 및 계산 방법에 대한 자세한 내용은 30 페이지 “ZFS 디스크 공간 계산”을 참조하십시오.

◆ ◆ ◆ 3 장

3 Oracle Solaris ZFS 저장소 풀 관리

이 장에서는 Oracle Solaris ZFS에서 저장소 풀을 만들고 관리하는 방법을 설명합니다.

이 장에서는 다음과 같은 내용을 다룹니다.

ZFS 저장소 풀의 구성 요소

다음 절에서는 다음 저장소 풀 구성 요소에 대한 자세한 정보를 제공합니다.

ZFS 저장소 풀의 디스크 사용

저장소 풀의 가장 기본적인 요소는 물리적 저장소입니다. 물리적 저장소는 128MB 이상의 모든 블록 장치가 될 수 있습니다. 일반적으로 이 장치는 시스템의 `/dev/dsk` 디렉토리에서 볼 수 있는 하드 드라이브입니다.

저장 장치는 전체 디스크(`c1t0d0`) 또는 개별 슬라이스(`c0t0d0s7`)가 될 수 있습니다. 권장되는 작업 모드는 전체 디스크를 사용하는 것이며, 이 경우 특수한 포맷이 필요하지

않습니다. ZFS는 EFI 레이블을 사용하여 단일 대형 슬라이스를 포함하도록 디스크를 포맷합니다. 이 방식으로 사용할 때 `format` 명령으로 표시되는 파티션 테이블은 다음과 유사합니다.

```
Current partition table (original):
Total disk sectors available: 143358287 + 16384 (reserved sectors)

Part      Tag    Flag     First Sector         Size         Last Sector
  0        usr    wm               256       68.36GB          143358320
  1 unassigned    wm                 0           0               0
  2 unassigned    wm                 0           0               0
  3 unassigned    wm                 0           0               0
  4 unassigned    wm                 0           0               0
  5 unassigned    wm                 0           0               0
  6 unassigned    wm                 0           0               0
  8   reserved    wm         143358321        8.00MB          143374704
```

ZFS 저장소 풀의 전체 디스크를 사용하는 경우 다음 고려 사항을 검토합니다.

- 전체 디스크를 사용하는 경우 일반적으로 `/dev/dsk/cNtNdN` 이름 지정 규약을 사용하여 디스크 이름을 지정합니다. 일부 타사 드라이버는 다른 명명 규칙을 사용하거나 `/dev/dsk` 디렉토리 이외의 다른 위치에 디스크를 둘 수 있습니다. 이러한 디스크를 사용하려면 수동으로 디스크 레이블을 지정하고 ZFS에 슬라이스를 제공해야 합니다.
- x86 기반 시스템에서는 디스크에 유효한 Solaris `fdisk` 분할 영역이 있어야 합니다. Solaris `fdisk` 분할 영역 만들기 또는 변경에 대한 자세한 내용은 **System Administration Guide: Devices and File Systems**의 "Setting Up Disks for ZFS File Systems (Task Map)"을 참조하십시오.
- 단일 디스크로 저장소 풀을 만들 경우 ZFS는 EFI 레이블을 적용합니다. EFI 레이블에 대한 자세한 내용은 **System Administration Guide: Devices and File Systems**의 "EFI (GPT) Disk Label"을 참조하십시오.

디스크는 전체 경로(예: `/dev/dsk/c1t0d0`) 또는 `/dev/dsk` 디렉토리 내의 장치 이름으로 구성된 단축 이름(예: `c1t0d0`)을 사용하여 지정할 수 있습니다. 예를 들어, 다음은 유효한 디스크 이름입니다.

- `c1t0d0`
- `/dev/dsk/c1t0d0`
- `/dev/foo/disk`

ZFS 저장소 풀에서 슬라이스 사용

디스크 슬라이스로 저장소 풀을 만들 때 Solaris VTOC(SMI) 레이블로 디스크 레이블을 지정할 수 있지만 디스크 슬라이스 관리가 보다 어려워지기 때문에 풀에 대한 디스크 슬라이스 사용은 권장되지 않습니다.

다음 `format` 출력 결과에 나온 대로 SPARC 기반 시스템에서 72GB 디스크에는 68GB의 사용 가능한 공간이 슬라이스 0에 있습니다.

```
# format
.
.
.
Specify disk (enter its number): 4
selecting c1t1d0
partition> p
Current partition table (original):
Total disk cylinders available: 14087 + 2 (reserved cylinders)

Part      Tag    Flag     Cylinders         Size            Blocks
  0       root    wm       0 - 14086       68.35GB    (14087/0/0) 143349312
  1 unassigned    wm       0                0         (0/0/0)             0
  2     backup    wm       0 - 14086       68.35GB    (14087/0/0) 143349312
  3 unassigned    wm       0                0         (0/0/0)             0
  4 unassigned    wm       0                0         (0/0/0)             0
  5 unassigned    wm       0                0         (0/0/0)             0
  6 unassigned    wm       0                0         (0/0/0)             0
  7 unassigned    wm       0                0         (0/0/0)             0
```

다음 format 출력 결과에 나온 대로 x86 기반 시스템에서 72GB 디스크에는 68GB의 사용 가능한 디스크 공간이 슬라이스 0에 있습니다. 작은 양의 부트 정보가 슬라이스 8에 포함되어 있습니다. 슬라이스 8은 관리가 필요하지 않으며 변경할 수 없습니다.

```
# format
.
.
.
selecting c1t0d0
partition> p
Current partition table (original):
Total disk cylinders available: 49779 + 2 (reserved cylinders)

Part      Tag    Flag     Cylinders         Size            Blocks
  0       root    wm       1 - 49778       68.36GB    (49778/0/0) 143360640
  1 unassigned    wu       0                0         (0/0/0)             0
  2     backup    wm       0 - 49778       68.36GB    (49779/0/0) 143363520
  3 unassigned    wu       0                0         (0/0/0)             0
  4 unassigned    wu       0                0         (0/0/0)             0
  5 unassigned    wu       0                0         (0/0/0)             0
  6 unassigned    wu       0                0         (0/0/0)             0
  7 unassigned    wu       0                0         (0/0/0)             0
  8       boot    wu       0 -     0        1.41MB    (1/0/0)          2880
  9 unassigned    wu       0                0         (0/0/0)             0
```

x86 기반 시스템에는 fdisk 분할 영역도 존재합니다. fdisk 파티션은 /dev/dsk/cN[tN]dNpN 장치 이름으로 표시되고, 디스크의 사용 가능한 슬라이스에 대한 컨테이너로 동작합니다. 이 구성은 테스트되지 않았으며 지원되지 않으므로 ZFS 저장소 풀 구성 요소에 대해 cN[tN]dNpN 장치를 사용하지 마십시오.

ZFS 저장소 풀에서 파일 사용

ZFS에서는 파일을 저장소 풀의 가상 장치로 사용할 수도 있습니다. 이 기능은 운용 목적이 아닌 단순 실험 테스트 및 사용을 목적으로 합니다.

- UFS 파일 시스템의 파일로 지원되는 ZFS 풀을 만들 경우 정확성 및 동기 의미를 보장하기 위해 암묵적으로 UFS를 사용하게 됩니다.
- 다른 ZFS 풀에서 만들어진 파일 또는 볼륨으로 지원되는 ZFS 풀을 만들 경우 시스템 교착 상태 또는 패닉이 발생할 수 있습니다.

하지만 처음으로 ZFS를 사용하거나 충분한 물리적 장치가 없을 때 좀더 복잡한 구성으로 실험하려는 경우 파일은 꽤 유용할 수 있습니다. 모든 파일은 전체 경로로 지정해야 하며 크기가 64MB 이상이어야 합니다.

ZFS 저장소 풀 고려 사항

ZFS 저장소 풀을 만들고 관리할 때는 다음 고려 사항을 검토하십시오.

- 전체 물리적 디스크를 사용하는 것이 ZFS 저장소 풀을 만드는 가장 쉬운 방법입니다. 디스크 슬라이스, 하드웨어 RAID 어레이의 LUN 또는 소프트웨어 기반 볼륨 관리자가 제공하는 볼륨에서 풀을 만들 경우 관리, 안정성 및 성능 측면에서 ZFS 구성이 매우 복잡해질 수 있습니다. 다음 고려 사항은 다른 하드웨어나 소프트웨어 저장소 솔루션으로 ZFS를 구성하는 방법을 결정하는 데 도움이 될 것입니다.
  - 하드웨어 RAID 어레이의 LUN 기반으로 ZFS 구성을 만들 경우 ZFS 중복성 기능과 어레이가 제공하는 중복성 기능 사이의 관계를 이해해야 합니다. 일부 구성에서는 충분한 중복성과 성능을 제공하지만, 다른 구성에서는 그렇지 않을 수 있습니다.
  - Solaris Volume Manager(SVM) 또는 Veritas Volume Manager(VxVM) 등 소프트웨어 기반 볼륨 관리자가 제공하는 볼륨을 사용하여 ZFS에 대한 논리적 장치를 만들 수 있습니다. 하지만 이러한 구성은 권장되지 않습니다. ZFS는 이러한 장치에서도 제대로 작동하지만 최적 성능에 못 미치는 결과가 나타날 수 있습니다.

    저장소 풀 권장 사항에 대한 자세한 내용은 11 장, “Oracle Solaris ZFS 권장 방법”을 참조하십시오.
- 디스크는 경로 및 장치 ID 모두로 식별됩니다(사용 가능한 경우). 장치 ID 정보를 사용할 수 있는 시스템에서는 이 식별 방법을 통해 ZFS를 업데이트하지 않고 장치를 인식할 수 있습니다. 장치 ID 생성 및 관리는 시스템마다 다를 수 있으므로 한 컨트롤러에서 다른 컨트롤러로 디스크 이동과 같이 장치를 이동하기 전에 먼저 풀을 내보내기해야 합니다. 펌웨어 업데이트 또는 기타 하드웨어 변경과 같은 시스템 이벤트는 ZFS 저장소 풀에서 장치 ID를 변경하여 장치를 사용하지 못하게 될 수 있습니다.

ZFS 저장소 풀의 복제 기능

ZFS는 미러링 및 RAID-Z 구성에서 자가 치료 등록 정보와 함께 데이터 중복성을 제공합니다.

미러링된 저장소 풀 구성

미러링된 저장소 풀 구성을 위해서는 가능하면 별도의 컨트롤러에 둘 이상의 디스크가 필요합니다. 미러링 구성에서는 많은 디스크를 사용할 수 있습니다. 또한 각 풀에서 둘 이상의 미러를 만들 수도 있습니다. 개념적으로 기본적인 미러링 구성은 다음과 같습니다.

```
mirror c1t0d0 c2t0d0
```

개념적으로 더 복잡한 미러링 구성은 다음과 같습니다.

```
mirror c1t0d0 c2t0d0 c3t0d0 mirror c4t0d0 c5t0d0 c6t0d0
```

미러링 저장소 풀 만들기에 대한 자세한 내용은 50 페이지 “미러된 저장소 풀 만들기”를 참조하십시오.

RAID-Z 저장소 풀 구성

미러링된 저장소 풀 구성과 함께 ZFS는 단일, 이중 또는 삼중 패리티 내결함성을 갖춘 RAID-Z 구성을 제공합니다. 단일 패리티 RAID-Z(`raidz` 또는 `raidz1`)는 RAID-5와 유사합니다. 이중 패리티 RAID-Z(`raidz2`)는 RAID-6과 유사합니다.

RAIDZ-3(`raidz3`)에 대한 자세한 내용은 다음 블로그를 참조하십시오.

`http://blogs.oracle.com/ahl/entry/triple_parity_raid_z`

모든 기존 RAID-5 유사 알고리즘(예: RAID-4, RAID-6, RDP 및 EVEN-ODD)에서 *RAID-5* **쓰기 허점**이라고 알려진 문제가 발생할 수 있습니다. RAID-5 스트라이프 중 일부만 쓰여지고 모든 블록이 디스크에 기록되기 전에 전원이 끊어질 경우 패리티가 데이터와 비동기화된 상태로 남게 되므로 영원히 쓸모 없게 됩니다(다음 전체 스트라이프 쓰기로 덮어쓰는 경우 제외). RAID-Z에서 ZFS는 모든 쓰기가 전체 스트라이프 쓰기가 되도록 가변 너비 RAID 스트라이프를 사용합니다. 이 설계는 파일 시스템의 메타 데이터가 가변 너비 RAID 스트라이프를 처리할 수 있는 기본 데이터 중복성 모델에 대한 충분한 정보를 가지도록 ZFS에서 파일 시스템과 장치 관리를 통합하기 때문에 가능합니다. RAID-Z는 RAID-5 쓰기 허점에 대한 세계 최초의 소프트웨어 전용 솔루션입니다.

X 크기의 N개 디스크와 P개의 패리티 디스크를 갖춘 RAID-Z 구성은 약 (N-P)*X바이트를 유지할 수 있으며 데이터 무결성이 침해되기 전 P개의 장치 결함을 견뎌낼 수 있습니다. 단일 패리티 RAID-Z 구성의 경우 2개 이상의 디스크, 이중 패리티 RAID-Z 구성의 경우 3개 이상의 디스크가 필요합니다. 예를 들어, 단일 패리티 RAID-Z 구성에서 3개의 디스크가 있을 경우 패리티 데이터는 3개의 디스크 중 하나에 해당하는 디스크 공간을 차지합니다. 그렇지 않은 경우 RAID-Z 구성을 만들기 위해 필요한 특수한 하드웨어는 없습니다.

개념적으로 3개의 디스크를 갖춘 RAID-Z 구성은 다음과 같습니다.

```
raidz c1t0d0 c2t0d0 c3t0d0
```

개념적으로 더 복잡한 RAID-Z 구성은 다음과 같습니다.

```
raidz c1t0d0 c2t0d0 c3t0d0 c4t0d0 c5t0d0 c6t0d0 c7t0d0
raidz c8t0d0 c9t0d0 c10t0d0 c11t0d0c12t0d0 c13t0d0 c14t0d0
```

많은 디스크로 RAID-Z 구성을 만들 경우 디스크를 여러 그룹으로 나눌 수 있습니다. 예를 들어, 14개의 디스크를 갖춘 RAID-Z 구성은 7개의 디스크로 이루어진 두 그룹으로 나누는 것이 좋습니다. 단일 숫자 그룹의 디스크를 갖춘 RAID-Z 구성이 더 좋은 성능을 발휘합니다.

RAID-Z 저장소 풀 만들기에 대한 자세한 내용은 51 페이지 “RAID-Z 저장소 풀 만들기”를 참조하십시오.

성능 및 디스크 공간 고려 측면에서 미러링 구성 또는 RAID-Z 구성 중에서 선택해야 하는 경우 자세한 내용은 다음 블로그 항목을 참조하십시오.

http://blogs.oracle.com/roch/entry/when_to_and_not_to

RAID-Z 저장소 풀 권장 사항에 대한 자세한 내용은 11 장, “Oracle Solaris ZFS 권장 방법”을 참조하십시오.

ZFS 하이브리드 저장소 풀

Oracle의 Sun Storage 7000 제품 시리즈에서 사용할 수 있는 ZFS 하이브리드 저장소 풀은 DRAM, SSD 및 HDD가 결합된 특수한 저장소 풀로 성능을 높이고 용량을 늘리면서 전력 소모를 줄일 수 있습니다. 이 제품의 관리 인터페이스에서 저장소 풀의 ZFS 중복성 구성을 선택하고 기타 구성 옵션도 쉽게 선택할 수 있습니다.

이 제품에 대한 자세한 내용은 **Sun Storage Unified Storage System Administration Guide**를 참조하십시오.

중복 구성에서 데이터 자가 치료

ZFS는 미러링 또는 RAID-Z 구성에서 데이터 자가 치료 기능을 제공합니다.

잘못된 데이터 블록이 발견되면 ZFS가 다른 중복 복사본에서 올바른 데이터를 인출할 뿐 아니라 정상 복사본으로 바꾸어 잘못된 데이터를 복구합니다.

저장소 풀의 동적 스트라이프

ZFS는 모든 최상위 레벨 가상 장치에 걸쳐 동적으로 데이터를 스트라이프합니다. 데이터를 어디에 둘 것인지에 대한 결정은 쓰기 시 이루어지므로 할당 시 고정 너비 스트라이프는 생성되지 않습니다.

새 가상 장치가 풀에 추가되면 ZFS는 성능 및 디스크 공간 할당 정책을 유지하기 위해 점진적으로 데이터를 새 장치에 할당합니다. 각 가상 장치는 다른 디스크 장치나 파일을 포함하는 미러 또는 RAID-Z 장치가 될 수도 있습니다. 이 구성은 풀의 결함 특성을 제어하는 데 있어 유연성을 제공합니다. 예를 들어, 4개의 디스크에서 다음 구성을 만들 수 있습니다.

- 동적 스트라이프를 사용하는 4개의 디스크
- 1개의 사중 RAID-Z 구성
- 2개의 동적 스트라이프를 사용하는 이중 미러

ZFS는 동일 풀 내에서 서로 다른 유형의 가상 장치 결합을 지원하지만 이 방식은 피하십시오. 예를 들어, 이중 미러와 삼중 RAID-Z 구성을 갖춘 풀을 만들 수 있습니다. 하지만 이 경우 내결함성은 최악의 가상 장치인 RAID-Z와 같습니다. 최상의 방식은 각 장치에서 동일 중복성 레벨로 동일 유형의 최상위 레벨 가상 장치를 사용하는 것입니다.

ZFS 저장소 풀 만들기 및 삭제

다음 절에서는 ZFS 저장소 풀을 만들고 삭제하는 여러 시나리오를 설명합니다.

- 49 페이지 “ZFS 저장소 풀 만들기”
- 55 페이지 “저장소 풀 가상 장치 정보 표시”
- 56 페이지 “ZFS 저장소 풀 만들기 오류 처리”
- 59 페이지 “ZFS 저장소 풀 삭제”

풀 만들기 및 삭제는 빠르고 쉽습니다. 하지만 이러한 작업을 수행할 때 주의하십시오. 새 풀에서 사용될 것으로 알려진 장치 사용을 막기 위한 확인이 수행되지만 ZFS에서 장치가 이미 사용 중인지 항상 알 수 있는 것은 아닙니다. 풀 삭제는 풀 만들기보다 쉽습니다. `zpool destroy`를 주의해서 사용하십시오. 이 단순한 명령이 막대한 결과를 가져올 수 있습니다.

ZFS 저장소 풀 만들기

저장소 풀을 만들려면 `zpool create` 명령을 사용합니다. 이 명령은 풀 이름 및 가상 장치 수를 인수로 사용합니다. 풀 이름은 29 페이지 “ZFS 구성 요소 명명 요구 사항”의 조건을 충족해야 합니다.

기본 저장소 풀 만들기

다음 명령은 c1t0d0 및 c1t1d0 디스크로 구성된 tank 이름의 새 풀을 만듭니다.

```
# zpool create tank c1t0d0 c1t1d0
```

전체 디스크를 나타내는 장치 이름은 /dev/dsk 디렉토리에서 찾을 수 있으며, 단일 대형 슬라이스를 포함하도록 ZFS에 의해 알맞게 레이블이 지정됩니다. 데이터는 두 디스크에 걸쳐 동적으로 스트라이프됩니다.

미러된 저장소 풀 만들기

미러된 풀을 만들려면 mirror 키워드 다음에 미러를 구성할 저장 장치 수를 사용합니다. 명령줄에 mirror 키워드를 반복하면 여러 미러를 지정할 수 있습니다. 다음 명령은 2개의 이중 미러를 갖춘 풀을 만듭니다.

```
# zpool create tank mirror c1d0 c2d0 mirror c3d0 c4d0
```

두번째 mirror 키워드는 새 최상위 레벨 가상 장치가 지정됨을 나타냅니다. 데이터는 두 미러에 걸쳐 동적으로 스트라이프되어, 두 디스크 사이에 알맞게 데이터가 중복됩니다.

권장되는 미러링된 구성에 대한 자세한 내용은 11 장, “Oracle Solaris ZFS 권장 방법”을 참조하십시오.

현재 ZFS 미러된 구성에서는 다음 작업이 지원됩니다.

- 기존 미러된 구성에 추가 최상위 레벨 가상 장치(vdev)를 위한 다른 디스크 모음 추가. 자세한 내용은 60 페이지 “저장소 풀에 장치 추가”를 참조하십시오.
- 기존 미러된 구성에 추가 디스크 연결. 또는 복제되지 않는 구성에 추가 디스크를 연결하여 미러된 구성 만들기. 자세한 내용은 65 페이지 “저장소 풀에서 장치 연결 및 분리”를 참조하십시오.
- 기존 미러된 구성에서 디스크 교체(교체 디스크가 교체될 디스크의 크기보다 크거나 같은 경우에만). 자세한 내용은 72 페이지 “저장소 풀의 장치 교체”를 참조하십시오.
- 미러된 구성에서 디스크 분리(나머지 장치가 구성에 대한 충분한 중복성을 제공하는 경우에만). 자세한 내용은 65 페이지 “저장소 풀에서 장치 연결 및 분리”를 참조하십시오.
- 디스크 중 하나를 분리하여 미러된 구성을 분할하고 새로운 동일 풀 만들기. 자세한 내용은 66 페이지 “미러링된 ZFS 저장소 풀을 분할하여 새로운 풀 만들기”를 참조하십시오.

미러링된 저장소 풀에서 스페어, 로그 장치 또는 캐시 장치가 아닌 장치는 절대로 제거할 수 없습니다.

ZFS 루트 풀 만들기

다음 루트 풀 구성 요구 사항을 고려해 보십시오.

- 루트 풀은 미러된 구성 또는 단일 디스크 구성으로 만들어야 합니다. `zpool add` 명령을 사용하여 디스크를 추가함으로써 여러 미러된 최상위 레벨 가상 장치를 만들 수 없지만, `zpool attach` 명령을 사용하여 미러된 가상 장치를 확장할 수는 있습니다.
- RAID-Z 또는 스트라이프 구성은 지원되지 않습니다.
- 루트 풀은 별도의 로그 장치를 가질 수 없습니다.
- 루트 풀에 대해 지원되지 않는 구성을 사용하려고 시도할 경우 다음과 유사한 메시지가 나타납니다.

```
ERROR: ZFS pool <pool-name> does not support boot environments
```

```
# zpool add -f rpool log c0t6d0s0
cannot add to 'rpool': root pool can not have multiple vdevs or separate logs
```

ZFS 루트 파일 시스템 설치 및 부트에 대한 자세한 내용은 4 장, "Oracle Solaris ZFS 루트 파일 시스템 설치 및 부트"를 참조하십시오.

RAID-Z 저장소 풀 만들기

단일 패리티 RAID-Z 풀을 만드는 것은 `raidz` 또는 `raidz1` 키워드가 `mirror` 대신 사용된다는 점을 제외하고 미러된 풀을 만드는 것과 동일합니다. 다음 예는 5개의 디스크로 구성된 단일 RAID-Z 장치의 풀을 만드는 방법을 보여줍니다.

```
# zpool create tank raidz c1t0d0 c2t0d0 c3t0d0 c4t0d0 /dev/dsk/c5t0d0
```

이 예는 단축 장치 이름 또는 전체 장치 이름을 사용하여 디스크를 지정할 수 있다는 것을 보여줍니다. `/dev/dsk/c5t0d0` 및 `c5t0d0`은 모두 동일 디스크를 가리킵니다.

풀을 만들 때 `raidz2` 또는 `raidz3` 키워드를 사용하여 이중 패리티 또는 삼중 패리티 RAID-Z 구성을 만들 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool create tank raidz2 c1t0d0 c2t0d0 c3t0d0 c4t0d0 c5t0d0
# zpool status -v tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          raidz2-0  ONLINE       0     0     0
            c1t0d0  ONLINE       0     0     0
            c2t0d0  ONLINE       0     0     0
            c3t0d0  ONLINE       0     0     0
            c4t0d0  ONLINE       0     0     0
            c5t0d0  ONLINE       0     0     0

errors: No known data errors

# zpool create tank raidz3 c0t0d0 c1t0d0 c2t0d0 c3t0d0 c4t0d0
c5t0d0 c6t0d0 c7t0d0 c8t0d0
# zpool status -v tank
```

```
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          raidz3-0  ONLINE       0     0     0
            c0t0d0  ONLINE       0     0     0
            c1t0d0  ONLINE       0     0     0
            c2t0d0  ONLINE       0     0     0
            c3t0d0  ONLINE       0     0     0
            c4t0d0  ONLINE       0     0     0
            c5t0d0  ONLINE       0     0     0
            c6t0d0  ONLINE       0     0     0
            c7t0d0  ONLINE       0     0     0
            c8t0d0  ONLINE       0     0     0
errors: No known data errors
```

현재 ZFS RAID-Z 구성에서는 다음 작업이 지원됩니다.

- 기존 RAID-Z 구성에 추가 최상위 레벨 가상 장치를 위한 다른 디스크 모음 추가. 자세한 내용은 60 페이지 "저장소 풀에 장치 추가"를 참조하십시오.
- 기존 RAID-Z 구성에서 디스크 교체(교체 디스크가 교체될 디스크의 크기보다 크거나 같은 경우에만). 자세한 내용은 72 페이지 "저장소 풀의 장치 교체"를 참조하십시오.

현재 RAID-Z 구성에서는 다음 작업이 지원되지 **않습니다**.

- 기존 RAID-Z 구성에 추가 디스크 연결
- RAID-Z 구성에서 디스크 분리(스페어 디스크로 교체되는 디스크를 분리하거나 스페어 디스크를 분리해야 하는 경우는 제외)
- RAID-Z 구성에서 로그 장치 또는 캐시 장치가 아닌 장치는 절대로 제거할 수 없습니다. RFE가 이 기능을 위해 마련되었습니다.

RAID-Z 구성에 대한 자세한 내용은 47 페이지 "RAID-Z 저장소 풀 구성"을 참조하십시오.

로그 장치를 사용하여 ZFS 저장소 풀 만들기

동기식 트랜잭션에 대한 POSIX 요구 사항을 충족하기 위해 ZIL(ZFS 계획 로그)이 제공됩니다. 예를 들어, 데이터베이스의 트랜잭션이 시스템 호출에서 반환될 때 안정된 저장 장치에서 이루어져야 할 경우가 자주 있습니다. NFS 및 기타 응용 프로그램은 `fsync()`를 사용하여 데이터 안정성을 보장할 수도 있습니다.

기본적으로 ZIL은 기본 풀 내의 블록에서 할당됩니다. 하지만 NVRAM 또는 전용 디스크와 같은 별도의 의도 로그 장치를 사용하면 성능을 높일 수도 있습니다.

ZFS 로그 장치 설정이 해당 환경에 적합한지 여부를 확인하려면 다음과 같은 요소를 고려하십시오.

- ZFS 의도 로그용 로그 장치는 데이터베이스 로그 파일과 관련이 없습니다.

- 별도의 로그 장치를 구현하여 얻을 수 있는 성능 향상은 장치 유형, 풀의 하드웨어 구성 및 응용 프로그램 작업 부하에 따라 달라집니다. 기본적인 성능 정보를 보려면 다음 블로그를 참조하십시오.

  http://blogs.oracle.com/perrin/entry/slog_blog_or_blogging_on
- 로그 장치는 복제를 취소하거나 미러링할 수 있지만 RAID-Z는 로그 장치에 지원되지 않습니다.
- 별도의 로그 장치가 미러링되지 않았고 장치에 로그 오류가 포함된 경우 로그 블록을 저장하면 저장소 풀로 복구됩니다.
- 로그 장치는 더 큰 저장소 풀의 일부로 추가, 교체, 제거, 연결, 분리, 가져오기 및 내보내기를 수행할 수 있습니다.
- 로그 장치를 기존 로그 장치에 연결하여 미러된 로그 장치를 만들 수 있습니다. 이 작업은 미러링되지 않은 저장소 풀에서 장치를 연결하는 것과 동일합니다.
- 로그 장치의 최소 크기는 풀에 있는 각 장치의 최소 크기(64MB)와 동일합니다. 로그 장치에 저장할 수 있는 in-play 데이터의 양은 비교적 적습니다. 로그 블록은 로그 트랜잭션(시스템 호출)이 커밋될 때 비워집니다.
- 로그 장치의 최대 크기는 저장 가능한 in-play 데이터의 최대 크기이므로 실제 메모리 크기의 약 1/2이어야 합니다. 예를 들어 시스템의 실제 메모리가 16GB인 경우 최대 로그 장치 크기는 8GB가 적당합니다.

저장소 풀을 만들 때 또는 해당 풀이 만들어진 후 ZFS 로그 장치를 설정할 수 있습니다.

다음 예는 미러된 로그 장치가 있는 미러된 저장소 풀을 만드는 방법을 보여줍니다.

```
# zpool create datap mirror c0t5000C500335F95E3d0 c0t5000C500335F907Fd0 mirror
c0t5000C500335BD117d0 c0t5000C500335DC60Fd0 log mirror c0t5000C500335E106Bd0 c0t5000C500335FC3E7d0
# zpool status datap
  pool: datap
 state: ONLINE
 scrub: none requested
config:

        NAME                       STATE     READ WRITE CKSUM
        datap                      ONLINE       0     0     0
          mirror-0                 ONLINE       0     0     0
            c0t5000C500335F95E3d0  ONLINE       0     0     0
            c0t5000C500335F907Fd0  ONLINE       0     0     0
          mirror-1                 ONLINE       0     0     0
            c0t5000C500335BD117d0  ONLINE       0     0     0
            c0t5000C500335DC60Fd0  ONLINE       0     0     0
        logs
          mirror-2                 ONLINE       0     0     0
            c0t5000C500335E106Bd0  ONLINE       0     0     0
            c0t5000C500335FC3E7d0  ONLINE       0     0     0

errors: No known data errors
```

로그 장치 실패에서 복구에 대한 자세한 내용은 예 10–2를 참조하십시오.

캐시 장치를 사용하여 ZFS 저장소 풀 만들기

캐시 장치에서 주 메모리와 디스크 간에 추가 캐싱 계층을 제공합니다. 캐시 장치를 사용하면 대부분 정적 컨텐츠로 구성된 모든 읽기 작업 부하에 대한 성능이 최대한 향상됩니다.

캐시 장치가 있는 저장소 풀을 만들어 저장소 풀 데이터를 캐시에 저장할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool create tank mirror c2t0d0 c2t1d0 c2t3d0 cache c2t5d0 c2t8d0
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c2t0d0  ONLINE       0     0     0
            c2t1d0  ONLINE       0     0     0
            c2t3d0  ONLINE       0     0     0
        cache
          c2t5d0    ONLINE       0     0     0
          c2t8d0    ONLINE       0     0     0

errors: No known data errors
```

캐시 장치가 추가되면 해당 장치가 점차적으로 주 메모리의 컨텐츠로 채워집니다. 캐시 장치의 크기에 따라 장치가 채워지는 시간이 1시간 이상 걸릴 수 있습니다. 다음과 같이 `zpool iostat` 명령을 사용하여 용량 및 읽기 작업을 모니터할 수 있습니다.

```
# zpool iostat -v pool 5
```

풀을 만든 후 풀에서 캐시 장치를 추가하거나 제거할 수 있습니다.

캐시 장치가 있는 ZFS 저장소 풀을 만들지 여부를 결정할 때 다음 사항을 고려하십시오.

- 캐시 장치를 사용하면 대부분 정적 컨텐츠로 구성된 모든 읽기 작업 부하에 대한 성능이 최대한 향상됩니다.
- `zpool iostat` 명령을 사용하여 용량 및 읽기를 모니터할 수 있습니다.
- 풀을 만들 때 단일 또는 여러 캐시 장치를 추가할 수 있습니다. 풀을 만든 후 추가하고 제거할 수도 있습니다. 자세한 내용은 예 3–4를 참조하십시오.
- 캐시 장치는 미러링하거나 RAID-Z 구성의 일부가 될 수 없습니다.
- 캐시 장치에서 읽기 오류가 발생할 경우 해당 읽기 I/O는 미러된 구성 또는 RAID-Z 구성의 일부일 수 있는 원래 저장소 풀 장치에 다시 내려집니다. 캐시 장치의 내용은 다른 시스템 캐시와 마찬가지로 휘발성으로 간주됩니다.

저장소 풀을 만들 때 주의 사항

ZFS 저장소 풀을 만들고 관리할 때는 다음 주의 사항을 검토하십시오.

- 기존 저장소 풀에 속하는 디스크의 분할 영역을 재지정하거나 레이블을 재지정하지 마십시오. 루트 풀 디스크의 분할 영역을 재지정하거나 레이블을 재지정하려고 시도하면 OS를 다시 설치해야 할 수 있습니다.
- 다른 저장소 풀(예: 파일 또는 볼륨)의 구성 요소를 포함하는 저장소 풀을 만들지 마십시오. 이와 같이 지원되지 않는 구성에서는 교착 상태가 발생할 수 있습니다.
- 단일 슬라이스 또는 단일 디스크로 만들어진 풀의 경우 중복성이 없고 데이터 손실의 위험이 있습니다. 중복성 없이 여러 슬라이스로 만들어진 풀의 경우에도 데이터 손실의 위험이 있습니다. 여러 디스크에 걸쳐 있는 여러 슬라이스로 만들어진 풀은 전체 디스크로 만들어진 풀보다 관리하기가 어렵습니다.
- ZFS 중복성(RAIDZ 또는 미러) 없이 만들어진 풀은 데이터 불일치를 보고할 수만 있습니다. 데이터 불일치를 복구할 수는 없습니다.
- ZFS 중복성을 사용하여 만들어진 풀은 하드웨어 고장으로 인한 작동 중지 시간을 줄이는 데 도움이 되지만 하드웨어 고장, 정전 또는 연결 해제된 케이블의 영향을 받습니다. 정기적으로 데이터를 백업해야 합니다. 비엔터프라이즈급 하드웨어에서는 일상적인 풀 데이터 백업을 수행해야 합니다.
- 풀은 시스템 전체에서 공유될 수 없습니다. ZFS는 클러스터 파일 시스템이 아닙니다.

저장소 풀 가상 장치 정보 표시

각 저장소 풀에는 하나 이상의 가상 장치가 포함됩니다. **가상 장치**는 물리적 저장소의 레이아웃 및 저장소 풀의 결함 특성을 설명하는 저장소 풀의 내부 표현입니다. 따라서 가상 장치는 저장소 풀을 만드는 데 사용된 디스크 장치나 파일을 나타냅니다. 풀에는 구성 최상위에 많은 수의 가상 장치(**최상위 레벨** *vdev*라고 함)가 있을 수 있습니다.

최상위 레벨 가상 장치에 둘 이상의 물리적 장치가 포함되어 있을 경우 구성은 미러 또는 RAID-Z 가상 장치로 데이터 중복성을 제공합니다. 이러한 가상 장치는 디스크, 디스크 슬라이스 또는 파일로 구성됩니다. 스페어는 풀에 대해 사용 가능한 핫 스페어를 추적하는 특수한 가상 장치입니다.

다음 예는 각각 두 디스크의 미러인 2개의 최상위 레벨 가상 장치로 구성된 풀을 만드는 방법을 보여줍니다.

```
# zpool create tank mirror c1d0 c2d0 mirror c3d0 c4d0
```

다음 예는 4개의 디스크가 있는 하나의 최상위 레벨 가상 장치로 구성된 풀을 만드는 방법을 보여줍니다.

```
# zpool create mypool raidz2 c1d0 c2d0 c3d0 c4d0
```

`zpool add` 명령을 사용하면 이 풀에 다른 최상위 레벨 가상 장치를 추가할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool add mypool raidz2 c2d1 c3d1 c4d1 c5d1
```

중복되지 않은 풀에서 사용되는 디스크, 디스크 슬라이스 또는 파일은 최상위 레벨 가상 장치로 동작합니다. 저장소 풀은 일반적으로 여러 최상위 레벨 가상 장치를 포함합니다. ZFS는 풀의 모든 최상위 레벨 가상 장치 사이에 동적으로 데이터를 스트라이프합니다.

ZFS 저장소 풀에 포함된 가상 장치 및 물리적 장치는 `zpool status` 명령으로 표시됩니다. 예를 들면 다음과 같습니다.

```
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
          mirror-1  ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
          mirror-2  ONLINE       0     0     0
            c0t3d0  ONLINE       0     0     0
            c1t3d0  ONLINE       0     0     0

errors: No known data errors
```

ZFS 저장소 풀 만들기 오류 처리

풀 만들기 오류는 여러 가지 원인으로 발생할 수 있습니다. 지정된 장치가 존재하지 않는 경우와 같이 일부 원인은 분명하지만, 기타 원인은 좀더 미묘할 수 있습니다.

사용 중인 장치 감지

장치를 포맷하기 전에 ZFS는 먼저 디스크가 ZFS 또는 운영 체제의 다른 부분에서 사용되고 있는지 여부를 확인합니다. 디스크가 사용 중인 경우 다음과 같은 오류가 나타날 수 있습니다.

```
# zpool create tank c1t0d0 c1t1d0
invalid vdev specification
use '-f' to override the following errors:
/dev/dsk/c1t0d0s0 is currently mounted on /. Please see umount(1M).
/dev/dsk/c1t0d0s1 is currently mounted on swap. Please see swap(1M).
/dev/dsk/c1t1d0s0 is part of active ZFS pool zeepool. Please see zpool(1M).
```

일부 오류는 `-f` 옵션을 대체할 수 있지만, 대부분의 오류는 그럴 수 없습니다. 다음 조건은 `-f` 옵션을 사용하여 대체할 수 없으며 수동으로 해결해야 합니다.

마운트된 파일 시스템 디스크 또는 해당 슬라이스 중 하나가 현재 마운트된 파일 시스템을 포함하고 있습니다. 이 오류를 해결하려면 `umount` 명령을 사용합니다.

/etc/vfstab의 파일 시스템 디스크가 /etc/vfstab 파일에 나열된 파일 시스템을 포함하지만, 파일 시스템이 현재 마운트되어 있지 않습니다. 이 오류를 해결하려면 /etc/vfstab 파일에서 라인을 제거하거나 주석 처리합니다.

전용 덤프 장치 디스크가 시스템에 대한 전용 덤프 장치로 사용 중입니다. 이 오류를 해결하려면 dumpadm 명령을 사용합니다.

ZFS 풀의 일부 디스크 또는 파일이 활성 ZFS 저장소 풀의 일부입니다. 이 오류를 해결하려면 zpool destroy 명령을 사용하여 다른 풀을 삭제합니다(더 이상 필요하지 않은 경우). 또는 zpool detach 명령을 사용하여 다른 풀에서 디스크를 분리합니다. 미러된 저장소 풀에서만 디스크를 분리할 수 있습니다.

다음 사용 중 여부 확인은 유용한 경고로 사용되며 -f 옵션을 사용하여 대체하고 풀을 만들 수 있습니다.

파일 시스템 포함 마운트되어 있지 않고 사용 중이 아닌 것으로 보이지만 디스크가 알려진 파일 시스템을 포함하고 있습니다.

볼륨의 일부 디스크가 Solaris Volume Manager 볼륨의 일부입니다.

Live Upgrade 디스크가 Oracle Solaris Live Upgrade에 대한 대체 부트 환경으로 사용 중입니다.

내보낸 ZFS 풀의 일부 디스크가 시스템에서 내보내졌거나 수동으로 제거된 저장소 풀의 일부입니다. 후자의 경우 디스크가 다른 시스템에서 사용 중인 네트워크 연결 드라이브이거나 아닐 수 있으므로 풀은 **잠재적으로 활성** 상태로 보고됩니다. 잠재적으로 활성 상태인 풀을 대체할 때 주의하십시오.

다음 예는 -f 옵션 사용 방법을 보여줍니다.

```
# zpool create tank c1t0d0
invalid vdev specification
use '-f' to override the following errors:
/dev/dsk/c1t0d0s0 contains a ufs filesystem.
# zpool create -f tank c1t0d0
```

이상적으로는 -f 옵션을 사용하여 대체하는 대신 오류를 해결해야 합니다.

일치하지 않는 복제 레벨

복제 레벨이 서로 다른 가상 장치로 풀을 만드는 것은 권장되지 않습니다. zpool 명령은 일치하지 않는 레벨의 중복성으로 풀을 만들지 못하도록 시도합니다. 이러한 구성으로 풀을 만들려고 시도할 경우 다음과 유사한 오류가 표시됩니다.

```
# zpool create tank c1t0d0 mirror c2t0d0 c3t0d0
invalid vdev specification
```

```
use '-f' to override the following errors:
mismatched replication level: both disk and mirror vdevs are present
# zpool create tank mirror c1t0d0 c2t0d0 mirror c3t0d0 c4t0d0 c5t0d0
invalid vdev specification
use '-f' to override the following errors:
mismatched replication level: 2-way mirror and 3-way mirror vdevs are present
```

이러한 오류는 `-f` 옵션으로 대체할 수 있지만, 이 방식은 피해야 합니다. 명령은 크기가 다른 장치를 사용하여 미러된 풀 또는 RAID-Z 풀을 만드는 것에 대해 경고하기도 합니다. 이 구성이 허용되기는 하지만, 중복성 레벨이 일치하지 않으면 큰 용량의 장치에서 사용되지 않는 디스크 공간이 생기게 됩니다. 경고를 대체하려면 `-f` 옵션이 필요합니다.

저장소 풀 만들기의 Dry Run 수행

풀을 만들려는 시도는 여러 가지 방식으로 예상치 않게 실패할 수 있으며, 디스크 포맷은 위험한 작업이 될 수 있습니다. 이러한 이유로 `zpool create` 명령에는 실제로 장치에 쓰지 않고 풀 만들기를 시뮬레이션하는 추가 옵션인 `-n`이 있습니다. 이 *dry run* 옵션은 장치 사용 중 여부 확인 및 복제 레벨 검증을 수행하고 이 과정에서 발생하는 모든 오류를 보고합니다. 오류가 발견되지 않으면 다음과 유사한 출력 결과가 표시됩니다.

```
# zpool create -n tank mirror c1t0d0 c1t1d0
would create 'tank' with the following layout:

        tank
          mirror
            c1t0d0
            c1t1d0
```

일부 오류는 실제로 풀을 만들지 않으면 감지할 수 없습니다. 가장 일반적인 예는 동일 구성에서 동일한 장치를 두 번 지정하는 것입니다. 이 오류는 실제로 데이터를 쓰지 않으면 사실상 감지할 수 없으므로 `zpool create -n` 명령에서 성공으로 보고할 수 있지만, 이 옵션 없이 명령을 실행하면 풀 만들기를 실패합니다.

저장소 풀에 대한 기본 마운트 지점

풀을 만들 때 최상위 레벨 파일 시스템의 기본 마운트 지점은 */pool-name*입니다. 이 디렉토리는 존재하지 않거나 비어 있어야 합니다. 디렉토리가 존재하지 않을 경우 자동으로 생성됩니다. 디렉토리가 비어 있을 경우 루트 파일 시스템이 기존 디렉토리 위에 마운트됩니다. 다른 기본 마운트 지점으로 풀을 만들려면 `zpool create` 명령의 `-m` 옵션을 사용합니다. 예를 들면 다음과 같습니다.

```
# zpool create home c1t0d0
default mountpoint '/home' exists and is not empty
use '-m' option to provide a different default
# zpool create -m /export/zfs home c1t0d0
```

이 명령은 마운트 지점이 `/export/zfs`인 `home` 파일 시스템 및 새로운 풀 `home`을 만듭니다.

마운트 지점에 대한 자세한 내용은 189 페이지 "ZFS 마운트 지점 관리"를 참조하십시오.

ZFS 저장소 풀 삭제

풀은 zpool destroy 명령을 사용하여 삭제됩니다. 이 명령은 마운트된 데이터 세트가 포함되어 있더라도 풀을 삭제합니다.

```
# zpool destroy tank
```

주의 – 풀을 삭제할 때 주의하십시오. 올바른 풀을 삭제하고 있는지 및 항상 데이터의 복사본을 가지고 있는지 확인하십시오. 실수로 잘못된 풀을 삭제할 경우 풀 복구를 시도할 수 있습니다. 자세한 내용은 99 페이지 "삭제된 ZFS 저장소 풀 복구"를 참조하십시오.

zpool destroy 명령으로 풀을 삭제할 경우 99 페이지 "삭제된 ZFS 저장소 풀 복구"에 설명된 대로 풀을 계속 가져올 수 있습니다. 따라서 풀에 속한 디스크에서 기밀 데이터를 계속 사용할 수 있습니다. 삭제된 풀의 디스크에서 데이터를 삭제하려면 삭제된 풀의 모든 디스크에서 format 유틸리티의 analyze->purge 옵션과 같은 기능을 사용해야 합니다.

사용할 수 없는 장치가 있는 풀 삭제

풀을 삭제하려면 풀이 더 이상 유효하지 않음을 나타내는 데이터를 디스크에 기록해야 합니다. 이 상태 정보는 가져오기를 수행할 때 해당 장치가 잠재적인 풀로 표시되지 않도록 합니다. 하나 이상의 장치를 사용할 수 없어도 풀을 삭제할 수 있습니다. 하지만 필요한 상태 정보는 이러한 사용할 수 없는 장치에 기록되지 않습니다.

이러한 장치가 제대로 복구되면 새 풀을 만들 때 **잠재적으로 활성** 상태로 보고됩니다. 가져올 풀을 검색할 때 유효한 장치로 나타납니다. 풀에 포함된 UNAVAIL 상태의 장치가 많아져서 풀 자체가 UNAVAIL 상태가 될 경우(최상위 레벨의 가상 장치가 UNAVAIL 상태인 경우), 이 명령으로 경고가 출력되며, -f 옵션을 사용하지 않으면 명령을 완료할 수 없습니다. 이 옵션은 풀을 열 수 없어 거기에 저장된 데이터를 알 수 없으므로 필요합니다. 예를 들면 다음과 같습니다.

```
# zpool destroy tank
cannot destroy 'tank': pool is faulted
use '-f' to force destruction anyway
# zpool destroy -f tank
```

풀 및 장치 건전성에 대한 자세한 내용은 87 페이지 "ZFS 저장소 풀의 건전성 상태 확인"을 참조하십시오.

풀 가져오기에 대한 자세한 내용은 96 페이지 "ZFS 저장소 풀 가져오기"를 참조하십시오.

ZFS 저장소 풀의 장치 관리

장치와 관련된 대부분의 기본 정보는 43 페이지 "ZFS 저장소 풀의 구성 요소"에서 다룹니다. 풀이 만들어진 후 풀 내의 물리적인 장치를 관리하기 위한 여러 가지 작업을 수행할 수 있습니다.

저장소 풀에 장치 추가

새 최상위 레벨 가상 장치를 추가하여 디스크 공간을 동적으로 추가할 수 있습니다. 이 디스크 공간은 풀의 모든 데이터 세트에서 즉시 사용할 수 있습니다. 풀에 새 가상 장치를 추가하려면 `zpool add` 명령을 사용합니다. 예를 들면 다음과 같습니다.

```
# zpool add zeepool mirror c2t1d0 c2t2d0
```

가상 장치를 지정하기 위한 형식은 `zpool create` 명령의 경우와 같습니다. 장치가 사용 중인지 여부가 확인되고, `-f` 옵션 없이는 명령에서 중복성 레벨을 변경할 수 없습니다. 명령은 dry run을 수행할 수 있도록 `-n` 옵션도 지원합니다. 예를 들면 다음과 같습니다.

```
# zpool add -n zeepool mirror c3t1d0 c3t2d0
would update 'zeepool' to the following configuration:
      zeepool
        mirror
            c1t0d0
            c1t1d0
        mirror
            c2t1d0
            c2t2d0
        mirror
            c3t1d0
            c3t2d0
```

이 명령 구문은 미러된 장치 `c3t1d0` 및 `c3t2d0`을 `zeepool` 풀의 기존 구성에 추가합니다.

가상 장치 검증 수행 방식에 대한 자세한 내용은 56 페이지 "사용 중인 장치 감지"를 참조하십시오.

예 3–1 미러된 ZFS 구성에 디스크 추가

다음 예에서는 미러링된 기존 ZFS 구성에 다른 미러가 추가됩니다.

예 3–1 미러된 ZFS 구성에 디스크 추가 *(계속)*

```
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
          mirror-1  ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0

errors: No known data errors
# zpool add tank mirror c0t3d0 c1t3d0
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
          mirror-1  ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
          mirror-2  ONLINE       0     0     0
            c0t3d0  ONLINE       0     0     0
            c1t3d0  ONLINE       0     0     0

errors: No known data errors
```

예 3–2 RAID-Z 구성에 디스크 추가

마찬가지로 추가 디스크를 RAID-Z 구성에 추가할 수 있습니다. 다음 예는 3개의 디스크를 포함하는 하나의 RAID-Z 장치를 갖춘 저장소 풀을 각각 3개의 디스크를 포함하는 두 개의 RAID-Z 장치를 갖춘 저장소 풀로 변환하는 방법을 보여줍니다.

```
# zpool status rzpool
  pool: rzpool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        rzpool      ONLINE       0     0     0
          raidz1-0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
```

예 3–2 RAID-Z 구성에 디스크 추가 *(계속)*

```
            c1t3d0  ONLINE       0     0     0
            c1t4d0  ONLINE       0     0     0

errors: No known data errors
# zpool add rzpool raidz c2t2d0 c2t3d0 c2t4d0
# zpool status rzpool
  pool: rzpool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        rzpool      ONLINE       0     0     0
          raidz1-0  ONLINE       0     0     0
            c1t0d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
            c1t3d0  ONLINE       0     0     0
          raidz1-1  ONLINE       0     0     0
            c2t2d0  ONLINE       0     0     0
            c2t3d0  ONLINE       0     0     0
            c2t4d0  ONLINE       0     0     0

errors: No known data errors
```

예 3–3 미러된 로그 장치 추가 및 제거

다음 예에서는 미러링된 저장소 풀에 미러링된 로그 장치를 추가하는 방법을 보여줍니다.

```
# zpool status newpool
  pool: newpool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        newpool     ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t4d0  ONLINE       0     0     0
            c0t5d0  ONLINE       0     0     0

errors: No known data errors
# zpool add newpool log mirror c0t6d0 c0t7d0
# zpool status newpool
  pool: newpool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        newpool     ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t4d0  ONLINE       0     0     0
            c0t5d0  ONLINE       0     0     0
```

예 3–3 미러된 로그 장치 추가 및 제거 *(**계속**)*

```
        logs
          mirror-1  ONLINE       0     0     0
            c0t6d0  ONLINE       0     0     0
            c0t7d0  ONLINE       0     0     0

errors: No known data errors
```

로그 장치를 기존 로그 장치에 연결하여 미러된 로그 장치를 만들 수 있습니다. 이 작업은 미러되지 않은 저장소 풀에서 장치를 연결하는 것과 동일합니다.

zpool remove 명령을 사용하여 로그 장치를 제거할 수 있습니다. 이전 예에서 미러된 로그 장치는 mirror-1 인수를 지정하여 제거할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool remove newpool mirror-1
# zpool status newpool
  pool: newpool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        newpool     ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t4d0  ONLINE       0     0     0
            c0t5d0  ONLINE       0     0     0

errors: No known data errors
```

풀 구성에 하나의 로그 장치만 포함되어 있을 경우 장치 이름을 지정하여 로그 장치를 제거할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool status pool
  pool: pool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        pool        ONLINE       0     0     0
          raidz1-0  ONLINE       0     0     0
            c0t8d0  ONLINE       0     0     0
            c0t9d0  ONLINE       0     0     0
        logs
          c0t10d0   ONLINE       0     0     0

errors: No known data errors
# zpool remove pool c0t10d0
```

예 3–4 캐시 장치 추가 및 제거

캐시 장치를 ZFS 저장소 풀에 추가하고 더 이상 필요하지 않을 경우 제거할 수 있습니다.

`zpool add` 명령을 사용하여 캐시 장치를 추가합니다. 예를 들면 다음과 같습니다.

```
# zpool add tank cache c2t5d0 c2t8d0
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c2t0d0  ONLINE       0     0     0
            c2t1d0  ONLINE       0     0     0
            c2t3d0  ONLINE       0     0     0
        cache
          c2t5d0    ONLINE       0     0     0
          c2t8d0    ONLINE       0     0     0

errors: No known data errors
```

캐시 장치는 미러링하거나 RAID-Z 구성의 일부가 될 수 없습니다.

`zpool remove` 명령을 사용하여 캐시 장치를 제거합니다. 예를 들면 다음과 같습니다.

```
# zpool remove tank c2t5d0 c2t8d0
# zpool status tank
 pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c2t0d0  ONLINE       0     0     0
            c2t1d0  ONLINE       0     0     0
            c2t3d0  ONLINE       0     0     0

errors: No known data errors
```

현재 `zpool remove` 명령만 핫 스페어, 로그 장치 및 캐시 장치 제거를 지원합니다. 기본 미러된 풀 구성의 일부인 장치는 `zpool detach` 명령을 사용하여 제거할 수 있습니다. 중복되지 않은 장치 및 RAID-Z 장치는 풀에서 제거할 수 없습니다.

ZFS 저장소 풀에서 캐시 장치 사용에 대한 자세한 내용은 54 페이지 “캐시 장치를 사용하여 ZFS 저장소 풀 만들기”를 참조하십시오.

저장소 풀에서 장치 연결 및 분리

zpool add 명령 이외에 zpool attach 명령을 사용하여 새 장치를 기존 미러된 장치 또는 미러되지 않은 장치에 추가할 수 있습니다.

디스크를 연결하여 미러된 루트 풀을 만드는 경우 113 페이지 "미러링된 ZFS 루트 풀을 만드는 방법(사후 설치)".

ZFS 루트 풀에서 디스크를 교체하는 경우 157 페이지 "ZFS 루트 풀 또는 루트 풀 스냅샷 복구"를 참조하십시오.

예 3–5 이중 미러된 저장소 풀을 삼중 미러된 저장소 풀로 변환

이 예에서는 zeepool이 기존 이중 미러이고, 새 장치 c2t1d0을 기존 장치 c1t1d0에 연결하여 삼중 미러로 변환합니다.

```
# zpool status zeepool
  pool: zeepool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        zeepool     ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0

errors: No known data errors
# zpool attach zeepool c1t1d0 c2t1d0
# zpool status zeepool
  pool: zeepool
 state: ONLINE
 scrub: resilver completed after 0h0m with 0 errors on Fri Jan  8 12:59:20 2010
config:

        NAME        STATE     READ WRITE CKSUM
        zeepool     ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
            c2t1d0  ONLINE       0     0     0  592K resilvered

errors: No known data errors
```

기존 장치가 삼중 미러의 일부인 경우 새 장치를 연결하면 사중 미러가 만들어지고, 이런 방식으로 계속 이어집니다. 어떠한 경우든지 새 장치는 즉시 재구성을 시작합니다.

예 3–6 중복되지 않은 ZFS 저장소 풀을 미러된 ZFS 저장소 풀로 변환

또한 zpool attach 명령을 사용하여 중복되지 않은 저장소 풀을 중복된 저장소 풀로 변환할 수 있습니다. 예를 들면 다음과 같습니다.

예 3–6 중복되지 않은 ZFS 저장소 풀을 미러된 ZFS 저장소 풀로 변환 *(계속)*

```
# zpool create tank c0t1d0
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:
        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          c0t1d0    ONLINE       0     0     0

errors: No known data errors
# zpool attach tank c0t1d0 c1t1d0
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: resilver completed after 0h0m with 0 errors on Fri Jan  8 14:28:23 2010
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0  73.5K resilvered

errors: No known data errors
```

zpool detach 명령을 사용하여 미러된 저장소 풀에서 장치를 분리할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool detach zeepool c2t1d0
```

하지만 이 작업은 데이터의 다른 유효한 복제본이 존재할 경우 실패합니다. 예를 들면 다음과 같습니다.

```
# zpool detach newpool c1t2d0
cannot detach c1t2d0: only applicable to mirror and replacing vdevs
```

미러링된 ZFS 저장소 풀을 분할하여 새로운 풀 만들기

zpool split 명령을 사용하여 미러링된 ZFS 저장소 풀을 백업 풀로 신속하게 복제할 수 있습니다. 이 기능을 사용하여 미러링된 루트 풀을 분할할 수 있지만 일부 추가 단계를 수행하기 전까지는 분할된 풀로 부트할 수 없습니다.

zpool split 명령을 사용하면 미러된 ZFS 저장소 풀에서 하나 이상의 디스크를 분리하여 분리된 디스크로 새 풀을 만들 수 있습니다. 새 풀은 원래 미러된 ZFS 저장소 풀과 동일한 컨텐츠를 가집니다.

기본적으로 미러된 풀에서 zpool split 작업은 새로 만들어진 풀에 대한 마지막 디스크를 분리합니다. 분할 작업 이후에는 새 풀을 가져올 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c1t0d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0

errors: No known data errors
# zpool split tank tank2
# zpool import tank2
# zpool status tank tank2
  pool: tank
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          c1t0d0    ONLINE       0     0     0

errors: No known data errors

  pool: tank2
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank2       ONLINE       0     0     0
          c1t2d0    ONLINE       0     0     0

errors: No known data errors
```

zpool split 명령에서 새로 만들어진 풀에 대해 사용해야 하는 디스크를 지정할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool split tank tank2 c1t0d0
```

실제 분할 작업이 이루어지기 전에 메모리의 데이터는 미러된 디스크로 비워집니다. 데이터가 비워진 후 디스크는 풀에서 분리되고 새 풀 GUID가 부여됩니다. 새 풀 GUID는 분할된 동일 시스템에서 풀을 가져올 수 있도록 하기 위해 생성됩니다.

분할할 풀에 비기본 파일 시스템 마운트 지점이 있고 새 풀이 동일 시스템에서 생성되는 경우 기존 마운트 지점이 서로 충돌하지 않도록 zpool split -R 옵션을 사용하여 새 풀에 대한 대체 루트 디렉토리를 지정해야 합니다. 예를 들면 다음과 같습니다.

```
# zpool split -R /tank2 tank tank2
```

zpool split -R 옵션을 사용하지 않고 새 풀을 가져오려고 할 때 마운트 지점 충돌을 알 수 있는 경우 -R 옵션으로 새 풀을 가져오십시오. 새 풀이 다른 시스템에서 만들어질 경우 마운트 지점 충돌이 발생하지 않는다면 대체 루트 디렉토리 지정은 필요하지 않습니다.

zpool split 기능을 사용하기 전에 다음 고려 사항을 검토하십시오.

- 이 기능은 여러 디스크의 RAID-Z 구성 또는 중복되지 않은 풀에 대해 사용할 수 없습니다.
- 데이터 및 응용 프로그램 작업은 zpool split 작업을 시도하기 전에 끝내야 합니다.
- 리실버링이 진행 중인 경우 풀을 분할할 수 없습니다.
- 미러된 풀 분할은 풀에 2 ~ 3개의 디스크가 포함되어 있을 때 가장 좋습니다. 이 경우 원래 풀의 마지막 디스크가 새로 만들어진 풀에 사용됩니다. 그런 다음 zpool attach 명령을 사용하여 원래 미러된 저장소 풀을 다시 만들거나 새로 만들어진 풀을 미러된 저장소 풀로 변환할 수 있습니다. 새로운(분할된) 풀은 중복성이 없기 때문에 한 번의 zpool split 작업으로 **기존**의 미러링된 풀에서 **새로운** 미러링된 풀을 만드는 방법은 현재까지 없습니다.
- 기존 풀이 삼중 미러인 경우 새 풀은 분할 작업 후 하나의 디스크를 포함합니다. 기존 풀이 2개의 디스크로 이루어진 이중 미러인 경우 결과는 2개의 디스크로 이루어진 2개의 중복되지 않은 풀입니다. 중복되지 않은 풀을 미러된 풀로 변환하려면 2개의 추가 디스크를 연결해야 합니다.
- 분할 작업 중 데이터를 중복으로 유지하기 위한 좋은 방법은 3개의 디스크를 포함하는 미러된 저장소 풀을 분할하여 분할 작업 후 원래 풀이 2개의 미러된 디스크를 포함하도록 하는 것입니다.
- 미러링된 풀을 분할하기 전에 하드웨어가 올바르게 구성되었는지 확인하십시오. 하드웨어의 캐시 비우기 설정 확인과 관련된 자세한 내용은 303 페이지 “일반 시스템 방법”을 참조하십시오.

예 3–7 미러된 ZFS 풀 분할

다음 예에서는 세 개의 디스크가 포함된 mothership이라는 미러링된 저장소 풀이 분할됩니다. 그 결과로 생성되는 두 개의 풀은 두 개의 디스크를 포함하는 mothership이라는 미러링된 풀과 한 개의 디스크를 포함하는 luna라는 새 풀입니다. 각 풀은 동일한 컨텐츠를 가집니다.

luna 풀은 백업 목적으로 다른 시스템으로 가져올 수 있습니다. 백업이 완료된 후에는 luna 풀을 삭제할 수 있으며 디스크가 mothership에 다시 연결됩니다. 그런 후 프로세스를 반복할 수 있습니다.

```
# zpool status mothership
  pool: mothership
 state: ONLINE
  scan: none requested
config:
```

예 3–7 미러된 ZFS 풀 분할 **(계속)**

```
        NAME                        STATE     READ WRITE CKSUM
        mothership                  ONLINE       0     0     0
          mirror-0                  ONLINE       0     0     0
            c0t5000C500335F95E3d0   ONLINE       0     0     0
            c0t5000C500335BD117d0   ONLINE       0     0     0
            c0t5000C500335F907Fd0   ONLINE       0     0     0

errors: No known data errors
# zpool split mothership luna
# zpool import luna
# zpool status mothership luna
  pool: luna
 state: ONLINE
  scan: none requested
config:

        NAME                      STATE     READ WRITE CKSUM
        luna                      ONLINE       0     0     0
          c0t5000C500335F907Fd0   ONLINE       0     0     0

errors: No known data errors

  pool: mothership
 state: ONLINE
  scan: none requested
config:

        NAME                        STATE     READ WRITE CKSUM
        mothership                  ONLINE       0     0     0
          mirror-0                  ONLINE       0     0     0
            c0t5000C500335F95E3d0   ONLINE       0     0     0
            c0t5000C500335BD117d0   ONLINE       0     0     0

errors: No known data errors
```

저장소 풀에서 장치 온라인 및 오프라인 전환

ZFS에서는 개별 장치를 오프라인이나 온라인으로 전환할 수 있습니다. 하드웨어가 불안정하거나 제대로 작동하지 않을 경우 이러한 조건이 일시적이라면 ZFS는 데이터 읽기나 데이터 쓰기를 계속합니다. 조건이 일시적이 아니라면 장치를 오프라인으로 전환하여 장치를 무시하도록 ZFS에 지시할 수 있습니다. ZFS는 오프라인 장치에 요청을 보내지 않습니다.

주 – 장치를 교체하기 위해 오프라인으로 전환할 필요는 없습니다.

장치 오프라인 전환

zpool offline 명령을 사용하여 장치를 오프라인으로 전환할 수 있습니다. 장치는 경로 또는 단축 이름으로 지정할 수 있습니다(장치가 디스크인 경우). 예를 들면 다음과 같습니다.

```
# zpool offline tank c0t5000C500335F95E3d0
```

장치를 오프라인으로 전환할 때 다음 사항을 고려하십시오.

- 풀이 UNAVAIL이 되는 지점으로 풀을 오프라인으로 전환할 수 없습니다. 예를 들어, raidz1 구성에서 2개의 장치를 오프라인으로 전환할 수 없으며, 최상위 레벨 가상 장치를 오프라인으로 전환할 수 없습니다.

  ```
  # zpool offline tank c0t5000C500335F95E3d0
  cannot offline c0t5000C500335F95E3d0: no valid replicas
  ```

- 기본적으로 OFFLINE 상태가 지속됩니다. 시스템이 재부트되어도 장치는 오프라인을 유지합니다.

  장치를 일시적으로 오프라인으로 전환하려면 zpool offline -t 옵션을 사용하십시오. 예를 들면 다음과 같습니다.

  ```
  # zpool offline -t tank c1t0d0
  ```

  시스템이 재부트되면 이 장치는 자동으로 ONLINE 상태로 돌아갑니다.

- 장치가 오프라인으로 전환되었을 때 저장소 풀에서 분리된 것이 아닙니다. 다른 풀에서 오프라인 장치를 사용하려고 시도하면 원래 풀이 삭제된 이후라도 다음과 유사한 메시지가 나타납니다.

  device is part of exported or potentially active ZFS *pool*. Please see zpool(1M)

  원래 저장소 풀을 삭제한 후 다른 저장소 풀에서 오프라인 장치를 사용하려는 경우에는 먼저 장치를 온라인으로 전환한 다음 원래 저장소 풀을 삭제하십시오.

  원래 저장소 풀을 유지하면서 다른 저장소 풀에서 장치를 사용하는 다른 방법은 원래 저장소 풀의 기존 장치를 다른 호환 장치로 교체하는 것입니다. 장치 교체에 대한 자세한 내용은 72 페이지 “저장소 풀의 장치 교체”를 참조하십시오.

오프라인 장치는 풀 상태를 질의할 때 OFFLINE 상태에 있습니다. 풀 상태 질의에 대한 자세한 내용은 82 페이지 “ZFS 저장소 풀 상태 질의”를 참조하십시오.

장치 건전성에 대한 자세한 내용은 87 페이지 “ZFS 저장소 풀의 건전성 상태 확인”을 참조하십시오.

온라인으로 장치 설정

장치가 오프라인으로 전환된 후 zpool online 명령을 사용하여 다시 온라인으로 전환할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool online tank c0t5000C500335F95E3d0
```

장치가 온라인으로 전환되었을 때 풀에 쓰여진 모든 데이터는 새로 사용 가능한 장치와 재동기화됩니다. 디스크를 교체하기 위해 장치를 온라인으로 전환할 수 없습니다. 장치를 오프라인으로 전환하고 장치를 교체한 다음 온라인으로 전환하려고 하는 경우 UNAVAIL 상태가 지속됩니다.

UNAVAIL 장치를 온라인으로 전환하려고 시도하면 다음과 비슷한 메시지가 표시됩니다.

```
# zpool online tank c1t0d0
warning: device 'c1t0d0' onlined, but remains in faulted state
use 'zpool replace' to replace devices that are no longer present
```

결함이 있는 디스크 메시지는 콘솔에 표시되거나 /var/adm/messages 파일에 기록될 수도 있습니다. 예를 들면 다음과 같습니다.

```
SUNW-MSG-ID: ZFS-8000-D3, TYPE: Fault, VER: 1, SEVERITY: Major
EVENT-TIME: Wed Jun 20 11:35:26 MDT 2012
PLATFORM: SUNW,Sun-Fire-880, CSN: -, HOSTNAME: neo
SOURCE: zfs-diagnosis, REV: 1.0
EVENT-ID: 504a1188-b270-4ab0-af4e-8a77680576b8
DESC: A ZFS device failed.  Refer to http://sun.com/msg/ZFS-8000-D3 for more information.
AUTO-RESPONSE: No automated response will occur.
IMPACT: Fault tolerance of the pool may be compromised.
REC-ACTION: Run 'zpool status -x' and replace the bad device.
```

결함이 있는 장치 교체에 대한 자세한 내용은 278 페이지 "누락되었거나 제거된 장치 해결"을 참조하십시오.

큰 용량의 디스크가 풀에 연결되었거나 작은 용량의 디스크가 큰 용량의 디스크로 교체된 경우 zpool online -e 명령을 사용하여 풀 크기를 확장할 수 있습니다. 기본적으로 풀에 추가된 디스크는 autoexpand 풀 등록 정보가 사용으로 설정되지 않으면 전체 크기로 확장되지 않습니다. 교체 디스크가 이미 온라인 상태이거나 디스크가 현재 오프라인 상태인 경우에도 zpool online -e 명령을 사용하여 풀을 자동으로 확장할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool online -e tank c0t5000C500335F95E3d0
```

저장소 풀 장치 오류 지우기

실패로 인해 장치가 오프라인으로 전환되어 zpool status 출력 결과에 오류가 나열될 경우 zpool clear 명령을 사용하여 오류 수를 지울 수 있습니다.

인수 없이 지정되면 이 명령은 풀 내의 모든 장치 오류를 지웁니다. 예를 들면 다음과 같습니다.

```
# zpool clear tank
```

하나 이상의 장치가 지정되면 이 명령은 지정된 장치와 연관된 오류만 지웁니다. 예를 들면 다음과 같습니다.

```
# zpool clear tank c0t5000C500335F95E3d0
```

zpool 오류 지우기에 대한 자세한 내용은 283 페이지 "일시적인 데이터 오류 지우기"를 참조하십시오.

저장소 풀의 장치 교체

zpool replace 명령을 사용하여 저장소 풀의 장치를 교체할 수 있습니다.

중복된 풀의 동일 위치에서 다른 장치로 장치를 물리적으로 교체하는 경우 교체되는 장치만 식별하면 됩니다. 일부 하드웨어에서 ZFS는 장치가 동일 위치의 다른 디스크에 있다고 인식합니다. 예를 들어, 디스크를 제거하고 동일 위치에서 교체하여 실패한 디스크(c1t1d0)를 교체하려면 다음 구문을 사용합니다.

```
# zpool replace tank c1t1d0
```

다른 물리적 위치에 있는 디스크로 저장소 풀의 디스크를 교체하는 경우 두 장치를 모두 지정해야 합니다. 예를 들면 다음과 같습니다.

```
# zpool replace tank c1t1d0 c1t2d0
```

ZFS 루트 풀에서 디스크를 교체하는 경우 157 페이지 "ZFS 루트 풀 또는 루트 풀 스냅샷 복구"를 참조하십시오.

다음은 디스크 교체를 위한 기본 단계입니다.

1. 필요한 경우 zpool offline 명령을 사용하여 디스크를 오프라인으로 전환합니다.
2. 교체할 디스크를 제거합니다.
3. 교체 디스크를 삽입합니다.
4. 교체 디스크가 표시되는지 여부를 확인하려면 format 출력 결과를 검토합니다.

   또한 장치 ID가 변경되었는지 여부를 확인합니다. 교체 디스크에 WWN이 포함된 경우 결함이 있는 디스크에 대한 장치 ID가 변경된 것입니다.
5. 디스크가 교체되었음을 ZFS에 알립니다. 예를 들면 다음과 같습니다.

   ```
   # zpool replace tank c1t1d0
   ```

   교체 디스크에 위에서 식별한 것과 다른 장치 ID가 포함된 경우 새 장치 ID를 포함시킵니다.

   ```
   # zpool replace tank c0t5000C500335FC3E7d0 c0t5000C500335BA8C3d0
   ```
6. 필요에 따라 zpool online 명령을 사용하여 디스크를 온라인으로 전환합니다.

7. 장치가 교체되었다고 FMA에 알립니다.

   `fmadm faulty` 출력 결과의 `Affects:` 절에서 `zfs://pool=name/vdev=guid` 문자열을 식별하고 해당 문자열을 `fmadm repaired` 명령의 인수로 제공합니다.

   ```
   # fmadm faulty
   # fmadm repaired zfs://pool=name/vdev=guid
   ```

SATA 디스크가 있는 일부 시스템에서는 오프라인 상태로 전환하기 전에 디스크의 구성을 해제해야 합니다. 이 시스템의 동일 슬롯 위치에서 디스크를 교체하는 경우 이 절의 첫번째 예에 설명된 대로 `zpool replace` 명령만 실행하면 됩니다.

SATA 디스크 교체 예는 예 10–1을 참조하십시오.

ZFS 저장소 풀에서 장치를 교체할 때 다음을 고려하십시오.

- `autoreplace` 풀 등록 정보를 `on`으로 설정한 경우 이전에 풀에 속한 장치와 동일한 물리적 위치에서 발견된 모든 새 장치가 자동으로 포맷되고 교체됩니다. 이 등록 정보가 사용으로 설정되면 `zpool replace` 명령을 사용할 필요가 없습니다. 이 기능은 일부 하드웨어 유형에서는 사용할 수 없습니다.
- 시스템을 실행하는 동안 장치 또는 핫 스패어가 실제로 제거되면 저장소 풀 상태 `REMOVED`가 제공됩니다. 가능한 경우, 제거된 장치 대신 핫 스패어 장치가 대체됩니다.
- 장치를 제거한 후 다시 삽입하면 장치가 온라인으로 배치됩니다. 장치를 다시 삽입할 때 핫 스패어가 활성화된 경우, 온라인 작업이 완료되면 핫 스패어가 제거됩니다.
- 장치 제거 또는 삽입 자동 감지는 하드웨어에 따라 다르며 일부 플랫폼에서는 지원되지 않을 수 있습니다. 예를 들어, USB 장치는 삽입 즉시 자동으로 구성됩니다. 그러나 `cfgadm -c configure` 명령을 사용하여 SATA 드라이브를 구성해야 할 수 있습니다.
- 핫 스패어는 온라인 상태이고 사용 가능한지 정기적으로 점검됩니다.
- 교체 장치의 크기는 미러된 구성 또는 RAID-Z 구성에서 가장 작은 용량의 디스크보다 크거나 같아야 합니다.
- 교체하는 장치보다 용량이 큰 교체 장치가 풀에 추가될 경우 자동으로 전체 크기로 확장되지 않습니다. 큰 용량의 디스크가 풀에 추가될 경우 `autoexpand` 풀 등록 정보 값이 풀의 확장 여부를 결정합니다. 기본적으로 `autoexpand` 등록 정보는 사용 안함으로 설정됩니다. 큰 용량이 디스크가 풀에 추가되기 전이나 후에 이 등록 정보를 사용으로 설정하여 풀 크기를 확장할 수 있습니다.

  다음 예에서는 미러된 풀에서 두 개의 16GB 디스크가 두 개의 72GB 디스크로 교체되었습니다. 두번째 장치 교체를 시도하기 전에 첫번째 장치가 완전히 리실버링되었는지 확인하십시오. 디스크 교체 후 전체 디스크 크기를 확장하기 위해 `autoexpand` 등록 정보가 사용으로 설정됩니다.

  ```
  # zpool create pool mirror c1t16d0 c1t17d0
  # zpool status
    pool: pool
   state: ONLINE
   scrub: none requested
  ```

```
config:

        NAME         STATE     READ WRITE CKSUM
        pool         ONLINE       0     0     0
          mirror     ONLINE       0     0     0
            c1t16d0  ONLINE       0     0     0
            c1t17d0  ONLINE       0     0     0

zpool list pool
NAME   SIZE   ALLOC  FREE    CAP  HEALTH  ALTROOT
pool  16.8G  76.5K  16.7G     0%  ONLINE  -
# zpool replace pool c1t16d0 c1t1d0
# zpool replace pool c1t17d0 c1t2d0
# zpool list pool
NAME   SIZE   ALLOC  FREE    CAP  HEALTH  ALTROOT
pool  16.8G  88.5K  16.7G     0%  ONLINE  -
# zpool set autoexpand=on pool
# zpool list pool
NAME   SIZE   ALLOC  FREE    CAP  HEALTH  ALTROOT
pool  68.2G   117K  68.2G     0%  ONLINE  -
```

- 대형 풀에서 많은 디스크를 교체하는 작업은 새 디스크로 데이터 리실버링으로 인해 시간이 오래 걸립니다. 또한 디스크 교체 사이에 `zpool scrub` 명령을 실행하여 교체 장치가 정상 작동하고 데이터가 올바르게 쓰여지는지 확인해야 할 수 있습니다.
- 실패한 디스크가 핫 스패어로 자동 교체된 경우에는 실패한 디스크가 교체된 후 스페어를 분리해야 합니다. `zpool detach` 명령을 사용하여 미러된 풀 또는 RAID-Z 풀에서 스페어를 분리할 수 있습니다. 핫 스패어 분리에 대한 자세한 내용은 76 페이지 "저장소 풀에서 핫 스패어 활성화 및 비활성화"를 참조하십시오.

장치 교체에 대한 자세한 내용은 278 페이지 "누락되었거나 제거된 장치 해결" 및 281 페이지 "손상된 장치 교체 또는 복구"를 참조하십시오.

저장소 풀에서 핫 스패어 지정

핫 스패어 기능을 사용하여 저장소 풀에서 장애 또는 결함이 있는 장치를 교체하는 데 사용할 수 있는 디스크를 식별할 수 있습니다. 장치를 **핫 스패어**로 지정하면 해당 장치는 풀에서 활성 장치가 아니지만, 풀의 활성 장치가 실패할 경우 핫 스패어가 자동으로 실패한 장치를 교체하게 됩니다.

다음 방법으로 장치를 핫 스패어로 지정할 수 있습니다.

- 풀이 `zpool create` 명령을 사용하여 만들어진 경우
- 풀이 `zpool add` 명령을 사용하여 만들어진 후

다음 예는 풀이 만들어진 경우 장치를 핫 스패어로 지정하는 방법을 보여줍니다.

```
# zpool create zeepool mirror c0t5000C500335F95E3d0 c0t5000C500335F907Fd0
mirror c0t5000C500335BD117d0 c0t5000C500335DC60Fd0 spare c0t5000C500335E106Bd0 c0t5000C500335FC3E7d0
# zpool status zeepool
  pool: zeepool
```

```
 state: ONLINE
  scan: none requested
config:

        NAME                      STATE     READ WRITE CKSUM
        zeepool                   ONLINE       0     0     0
          mirror-0                ONLINE       0     0     0
            c0t5000C500335F95E3d0 ONLINE       0     0     0
            c0t5000C500335F907Fd0 ONLINE       0     0     0
          mirror-1                ONLINE       0     0     0
            c0t5000C500335BD117d0 ONLINE       0     0     0
            c0t5000C500335DC60Fd0 ONLINE       0     0     0
        spares
          c0t5000C500335E106Bd0   AVAIL
          c0t5000C500335FC3E7d0   AVAIL

errors: No known data errors
```

다음 예는 풀이 만들어진 후 풀에 장치를 추가하여 핫 스패어를 지정하는 방법을 보여줍니다.

```
# zpool add zeepool spare c0t5000C500335E106Bd0 c0t5000C500335FC3E7d0
# zpool status zeepool
  pool: zeepool
 state: ONLINE
  scan: none requested
config:

        NAME                      STATE     READ WRITE CKSUM
        zeepool                   ONLINE       0     0     0
          mirror-0                ONLINE       0     0     0
            c0t5000C500335F95E3d0 ONLINE       0     0     0
            c0t5000C500335F907Fd0 ONLINE       0     0     0
          mirror-1                ONLINE       0     0     0
            c0t5000C500335BD117d0 ONLINE       0     0     0
            c0t5000C500335DC60Fd0 ONLINE       0     0     0
        spares
          c0t5000C500335E106Bd0   AVAIL
          c0t5000C500335FC3E7d0   AVAIL

errors: No known data errors
```

핫 스패어는 `zpool remove` 명령을 사용하여 저장소 풀에서 제거할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool remove zeepool c0t5000C500335FC3E7d0
# zpool status zeepool
  pool: zeepool
 state: ONLINE
  scan: none requested
config:

        NAME                      STATE     READ WRITE CKSUM
        zeepool                   ONLINE       0     0     0
          mirror-0                ONLINE       0     0     0
            c0t5000C500335F95E3d0 ONLINE       0     0     0
```

```
            c0t5000C500335F907Fd0  ONLINE       0     0     0
          mirror-1                 ONLINE       0     0     0
            c0t5000C500335BD117d0  ONLINE       0     0     0
            c0t5000C500335DC60Fd0  ONLINE       0     0     0
        spares
          c0t5000C500335E106Bd0    AVAIL

errors: No known data errors
```

핫 스페어는 저장소 풀에서 현재 사용되는 경우 제거할 수 없습니다.

ZFS 핫 스페어를 사용할 때 다음을 고려하십시오.

- 현재 `zpool remove` 명령은 핫 스페어, 캐시 장치 및 로그 장치를 제거하는 데만 사용할 수 있습니다.
- 디스크를 핫 스페어로 추가하려면 핫 스페어가 풀에서 가장 큰 용량의 디스크보다 크거나 같아야 합니다. 풀에서 더 작은 용량의 디스크를 스페어로 추가하는 것도 가능합니다. 하지만 자동으로 또는 `zpool replace` 명령으로 더 작은 용량의 스페어가 활성화되면 다음과 유사한 오류와 함께 작업을 실패합니다.

  ```
  cannot replace disk3 with disk4: device is too small
  ```

저장소 풀에서 핫 스페어 활성화 및 비활성화

핫 스페어는 다음 방법으로 활성화됩니다.

- 수동 교체 – `zpool replace` 명령을 사용하여 저장소 풀에서 실패한 장치를 핫 스페어로 교체합니다.
- 자동 교체 – 결함이 감지되면 FMA 에이전트가 풀을 조사하여 사용 가능한 핫 스페어가 있는지 확인합니다. 있을 경우 사용 가능한 스페어로 결함이 있는 장치를 교체합니다.

  현재 사용 중인 핫 스페어가 실패할 경우 FMA 에이전트는 스페어를 분리하고 교체를 취소합니다. 그런 다음 사용 가능한 다른 핫 스페어로 장치 교체를 시도합니다. 장치가 시스템에서 사라질 때만 ZFS 진단 엔진에서 결함을 생성하므로 이 기능은 현재 제한적입니다.

  활성 스페어로 실패한 장치를 물리적으로 교체하는 경우 `zpool detach` 명령을 사용하여 스페어를 분리하고 원래 장치를 다시 활성화할 수 있습니다. `autoreplace` 풀 등록 정보를 `on`으로 설정한 경우 새 장치가 삽입되고 온라인 작업이 완료되면 스페어가 자동으로 분리되고 스페어 풀로 돌아갑니다.

`UNAVAIL` 장치는 핫 스페어를 사용할 수 있는 경우 자동으로 교체됩니다. 예를 들면 다음과 같습니다.

```
# zpool status -x
  pool: zeepool
 state: DEGRADED
status: One or more devices are unavailable in response to persistent errors.
        Sufficient replicas exist for the pool to continue functioning in a
```

```
        degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
  scan: resilvered 3.15G in 0h0m with 0 errors on Mon Nov 12 15:53:42 2012
config:

        NAME                        STATE     READ WRITE CKSUM
        zeepool                     DEGRADED     0     0     0
          mirror-0                  ONLINE       0     0     0
            c0t5000C500335F95E3d0   ONLINE       0     0     0
            c0t5000C500335F907Fd0   ONLINE       0     0     0
          mirror-1                  DEGRADED     0     0     0
            c0t5000C500335BD117d0   ONLINE       0     0     0
            spare-1                 DEGRADED   449     0     0
              c0t5000C500335DC60Fd0 UNAVAIL      0     0     0
              c0t5000C500335E106Bd0 ONLINE       0     0     0
        spares
          c0t5000C500335E106Bd0     INUSE

errors: No known data errors
```

현재 다음 방법으로 핫 스페어를 비활성화할 수 있습니다.

- 저장소 풀에서 핫 스페어를 제거합니다.
- 실패한 디스크가 물리적으로 교체된 후 핫 스페어를 분리합니다. 예 3–8을 참조하십시오.
- 다른 핫 스페어를 일시적으로 또는 영구적으로 교체합니다. 예 3–9를 참조하십시오.

예 3–8 실패한 디스크가 교체된 후 핫 스페어 분리

이 예에서 결함이 있는 디스크(c0t5000C500335DC60Fd0)는 물리적으로 교체되고 zpool replace 명령을 사용하여 ZFS에 통지됩니다.

```
# zpool replace zeepool c0t5000C500335DC60Fd0
# zpool status zeepool
  pool: zeepool
 state: ONLINE
  scan: resilvered 3.15G in 0h0m with 0 errors on Thu Jun 21 16:53:43 2012
config:

        NAME                        STATE     READ WRITE CKSUM
        zeepool                     ONLINE       0     0     0
          mirror-0                  ONLINE       0     0     0
            c0t5000C500335F95E3d0   ONLINE       0     0     0
            c0t5000C500335F907Fd0   ONLINE       0     0     0
          mirror-1                  ONLINE       0     0     0
            c0t5000C500335BD117d0   ONLINE       0     0     0
            c0t5000C500335DC60Fd0   ONLINE       0     0     0
        spares
          c0t5000C500335E106Bd0     AVAIL
```

필요에 따라 zpool detach 명령을 사용하여 핫 스페어를 스페어 풀로 복귀시킬 수 있습니다. 예를 들면 다음과 같습니다.

예 3–8 실패한 디스크가 교체된 후 핫 스페어 분리 *(계속)*

```
# zpool detach zeepool c0t5000C500335E106Bd0
```

예 3–9 실패한 디스크 분리 및 핫 스페어 사용

현재 교체 중인 핫 스페어를 일시적으로 또는 영구적으로 교체하여 장애가 발생한 디스크를 교체하려는 경우 장애가 발생한 원래 디스크를 분리합니다. 실패한 디스크가 교체되면 저장소 풀에 스페어로 다시 추가할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool status zeepool
  pool: zeepool
 state: DEGRADED
status: One or more devices are unavailable in response to persistent errors.
        Sufficient replicas exist for the pool to continue functioning in a
        degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
  scan: resilver in progress since Mon Nov 12 16:04:12 2012
     4.80G scanned out of 12.0G at 55.8M/s, 0h2m to go
     4.80G scanned out of 12.0G at 55.8M/s, 0h2m to go
     4.77G resilvered, 39.97% done
config:

        NAME                       STATE     READ WRITE CKSUM
        zeepool                    DEGRADED     0     0     0
          mirror-0                 ONLINE       0     0     0
            c0t5000C500335F95E3d0  ONLINE       0     0     0
            c0t5000C500335F907Fd0  ONLINE       0     0     0
          mirror-1                 DEGRADED     0     0     0
            c0t5000C500335BD117d0  ONLINE       0     0     0
            c0t5000C500335DC60Fd0  UNAVAIL      0     0     0
        spares
          c0t5000C500335E106Bd0    AVAIL

errors: No known data errors
# zpool detach zeepool c0t5000C500335DC60Fd0
# zpool status zeepool
  pool: zeepool
 state: ONLINE
  scan: resilvered 11.3G in 0h3m with 0 errors on Mon Nov 12 16:07:12 2012
config:

        NAME                       STATE     READ WRITE CKSUM
        zeepool                    ONLINE       0     0     0
          mirror-0                 ONLINE       0     0     0
            c0t5000C500335F95E3d0  ONLINE       0     0     0
            c0t5000C500335F907Fd0  ONLINE       0     0     0
          mirror-1                 ONLINE       0     0     0
            c0t5000C500335BD117d0  ONLINE       0     0     0
            c0t5000C500335E106Bd0  ONLINE       0     0     0

errors: No known data errors
(Original failed disk c0t5000C500335DC60Fd0 is physically replaced)
# zpool add zeepool spare c0t5000C500335DC60Fd0
```

예 3–9 실패한 디스크 분리 및 핫 스페어 사용 *(계속)*

```
# zpool status zeepool
  pool: zeepool
 state: ONLINE
  scan: resilvered 11.2G in 0h3m with 0 errors on Mon Nov 12 16:07:12 2012

config:

        NAME                       STATE     READ WRITE CKSUM
        zeepool                    ONLINE       0     0     0
          mirror-0                 ONLINE       0     0     0
            c0t5000C500335F95E3d0  ONLINE       0     0     0
            c0t5000C500335F907Fd0  ONLINE       0     0     0
          mirror-1                 ONLINE       0     0     0
            c0t5000C500335BD117d0  ONLINE       0     0     0
            c0t5000C500335E106Bd0  ONLINE       0     0     0
        spares
          c0t5000C500335DC60Fd0    AVAIL

errors: No known data errors
```

디스크를 교체하고 스페어를 분리한 후에는 FMA에 디스크가 교체되었음을 알립니다.

```
# fmadm faulty
# fmadm repaired zfs://pool=name/vdev=guid
```

ZFS 저장소 풀 등록 정보 관리

zpool get 명령을 사용하여 풀 등록 정보를 표시할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool get all tank
tank  size           68G                  -
tank  capacity       0%                   -
tank  altroot        -                    default
tank  health         ONLINE               -
tank  guid           15560293364730146756  -
tank  version        32                   default
tank  bootfs         -                    default
tank  delegation     on                   default
tank  autoreplace    off                  default
tank  cachefile      -                    default
tank  failmode       wait                 default
tank  listsnapshots  on                   default
tank  autoexpand     off                  default
tank  free           68.0G                -
tank  allocated      124K                 -
tank  readonly       off                  -
```

저장소 풀 등록 정보는 zpool set 명령으로 설정할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool set autoreplace=on zeepool
# zpool get autoreplace zeepool
NAME     PROPERTY     VALUE    SOURCE
zeepool  autoreplace  on       local
```

완전히 가득 찬 풀에서 풀 등록 정보를 설정하려고 시도하면 다음과 유사한 메시지가 표시됩니다.

```
# zpool set autoreplace=on tank
cannot set property for 'tank': out of space
```

풀 공간 용량 문제가 발생하지 않도록 하는 방법은 11 장, “Oracle Solaris ZFS 권장 방법”을 참조하십시오.

표 3–1 ZFS 풀 등록 정보 설명

등록 정보 이름	유형	기본값	설명
allocated	문자열	해당 없음	물리적으로 할당된 풀 내에서 저장 공간의 양을 식별하는 읽기 전용 값입니다.
altroot	문자열	off	대체 루트 디렉토리를 식별합니다. 설정되면 이 디렉토리가 풀 내의 모든 마운트 지점 앞에 추가됩니다. 이 등록 정보는 마운트 지점을 신뢰할 수 없거나 일반적인 경로가 유효하지 않은 대체 부트 환경에서 알 수 없는 풀을 조사할 때 사용할 수 있습니다.
autoreplace	부울	off	자동 장치 교체를 제어합니다. off로 설정되면 장치 교체가 zpool replace 명령을 사용하여 시작되어야 합니다. on으로 설정되면 이전에 풀에 속해 있던 장치와 동일한 물리적 위치에서 발견된 모든 새 장치가 자동으로 포맷되고 교체됩니다. 이 등록 정보의 약어는 replace입니다.
bootfs	부울	해당 없음	루트 풀의 기본 부트 가능 파일 시스템을 식별합니다. 이 등록 정보는 일반적으로 설치 프로그램에서 설정됩니다.
cachefile	문자열	해당 없음	풀 구성 정보가 캐시에 저장되는 위치를 제어합니다. 시스템이 부트되면 캐시에 있는 모든 풀을 자동으로 가져옵니다. 그러나 설치 및 클러스터링 환경에서는 풀을 자동으로 가져오지 않도록 이 정보를 다른 위치에 캐시해야 할 수 있습니다. 다른 위치에 풀 구성 정보를 캐시에 저장하도록 이 등록 정보를 설정할 수 있습니다. 이 정보는 나중에 zpool import -c 명령을 사용하여 가져올 수 있습니다. 대부분의 ZFS 구성에서 이 등록 정보는 사용되지 않습니다.
capacity	숫자	해당 없음	사용된 풀 공간의 백분율을 식별하는 읽기 전용 값입니다. 이 등록 정보의 약어는 cap입니다.

표 3–1 ZFS 풀 등록 정보 설명 (계속)

등록 정보 이름	유형	기본값	설명
delegation	부울	on	권한이 부여되지 않은 사용자에게 파일 시스템에 대해 정의된 액세스 권한을 부여할 수 있는지 여부를 제어합니다. 자세한 내용은 8 장, “Oracle Solaris ZFS 위임 관리”를 참조하십시오.
failmode	문자열	wait	대대적인 풀 실패가 발생할 경우 시스템 동작을 제어합니다. 이 조건은 대개 기본 저장 장치에 대한 연결 끊김 또는 풀 내의 모든 장치 실패의 결과입니다. 이러한 이벤트의 동작은 다음 값 중 하나로 결정됩니다. ■ wait – 장치 연결이 복원되고 zpool clear 명령을 사용하여 오류가 지워질 때까지 풀에 대한 모든 I/O 요청을 차단합니다. 이 상태에서 풀에 대한 I/O 작업은 차단되지만 읽기 작업은 성공할 수 있습니다. 풀은 장치 문제가 해결될 때까지 wait 상태를 유지합니다. ■ continue – 모든 새 쓰기 I/O 요청에 대해 EIO 오류를 반환하지만 나머지 양호한 장치에 대한 읽기는 허용합니다. 디스크에 커밋되어야 하는 모든 쓰기 요청은 차단됩니다. 장치가 다시 연결되거나 교체된 후 zpool clear 명령을 사용하여 오류를 지워야 합니다. ■ panic – 콘솔에 메시지를 출력하고 시스템 충돌 덤프를 생성합니다.
free	문자열	해당 없음	할당되지 않은 풀 내의 블록 수를 식별하는 읽기 전용 값입니다.
guid	문자열	해당 없음	풀에 대한 고유 식별자를 식별하는 읽기 전용 등록 정보입니다.
health	문자열	해당 없음	풀의 현재 건전성을 ONLINE, DEGRADED, SUSPENDED, REMOVED 또는 UNAVAIL로 식별하는 읽기 전용 등록 정보입니다.
listshares	문자열	Off	이 풀의 공유 정보를 zfs list 명령으로 표시할지를 제어합니다. 기본값은 off입니다.
listsnapshots	문자열	on	이 풀과 연관된 스냅샷 정보가 zfs list 명령으로 표시되는지 여부를 제어합니다. 이 등록 정보가 사용 안함으로 설정되면 zfs list - t snapshot 명령으로 스냅샷 정보를 표시할 수 있습니다.
size	숫자	해당 없음	저장소 풀의 총 크기를 식별하는 읽기 전용 등록 정보입니다.

표 3–1 ZFS 풀 등록 정보 설명 (계속)

등록 정보 이름	유형	기본값	설명
version	숫자	해당 없음	풀의 현재 디스크 버전을 식별합니다. 이 등록 정보는 역호환성을 위해 특정 버전이 필요할 때 사용할 수 있지만 풀 업데이트를 위해 선호되는 방법은 zpool upgrade 명령을 사용하는 것입니다. 이 등록 정보는 1과 zpool upgrade -v 명령으로 보고된 현재 버전 사이의 숫자로 설정할 수 있습니다.

ZFS 저장소 풀 상태 질의

zpool list 명령은 풀 상태에 관한 정보를 요청할 수 있는 여러 가지 방법을 제공합니다. 일반적으로 사용 가능한 정보는 기본 사용 정보, I/O 통계 및 건전성 상태의 세 범주에 속합니다. 이 절에서는 이러한 세 가지 유형의 저장소 풀 정보를 다룹니다.

- 82 페이지 “ZFS 저장소 풀에 대한 정보 표시”
- 85 페이지 “ZFS 저장소 풀에 대한 I/O 통계 보기 ”
- 87 페이지 “ZFS 저장소 풀의 건전성 상태 확인”

ZFS 저장소 풀에 대한 정보 표시

zpool list 명령을 사용하여 풀에 대한 기본 정보를 표시할 수 있습니다.

모든 저장소 풀 또는 특정 풀에 대한 정보 표시

인수 없는 zpool list 명령은 시스템의 모든 풀에 대한 다음 정보를 표시합니다.

```
# zpool list
NAME                    SIZE    ALLOC   FREE    CAP  HEALTH     ALTROOT
tank                   80.0G   22.3G   47.7G    28%  ONLINE     -
dozer                   1.2T    384G    816G    32%  ONLINE     -
```

이 명령 출력 결과에는 다음 정보가 표시됩니다.

NAME 풀의 이름입니다.

SIZE 모든 최상위 레벨 가상 장치의 합계와 같은 풀의 총 크기입니다.

ALLOC 모든 데이터 세트 및 내부 메타 데이터에 할당된 물리적 공간의 양입니다. 이 양은 파일 시스템 레벨에서 보고되는 디스크 공간의 양과 다를 수 있습니다.

사용 가능한 파일 시스템 공간에 대한 자세한 내용은 30 페이지 “ZFS 디스크 공간 계산”을 참조하십시오.

FREE 풀에서 할당되지 않은 공간의 양입니다.

CAP (CAPACITY) 사용된 디스크 공간의 양으로 총 디스크 공간의 백분율로 표시됩니다.

HEALTH 풀의 현재 건전성 상태입니다.

풀 건전성에 대한 자세한 내용은 87 페이지 "ZFS 저장소 풀의 건전성 상태 확인"을 참조하십시오.

ALTROOT 풀의 대체 루트입니다(존재하는 경우).

대체 루트 풀에 대한 자세한 내용은 269 페이지 "ZFS 대체 루트 풀 사용"을 참조하십시오.

풀 이름을 지정하면 특정 풀에 대한 통계를 수집할 수도 있습니다. 예를 들면 다음과 같습니다.

```
# zpool list tank
NAME                    SIZE    ALLOC   FREE    CAP   HEALTH     ALTROOT
tank                   80.0G    22.3G   47.7G    28%  ONLINE     -
```

zpool list 간격 및 수 옵션을 사용하여 기간에 따른 통계를 수집할 수 있습니다. 또한 -T 옵션을 사용하면 시간 기록을 표시할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool list -T d 3 2
Tue Nov  2 10:36:11 MDT 2010
NAME    SIZE  ALLOC   FREE    CAP  DEDUP  HEALTH  ALTROOT
pool   33.8G  83.5K  33.7G     0%  1.00x  ONLINE  -
rpool  33.8G  12.2G  21.5G    36%  1.00x  ONLINE  -
Tue Nov  2 10:36:14 MDT 2010
pool   33.8G  83.5K  33.7G     0%  1.00x  ONLINE  -
rpool  33.8G  12.2G  21.5G    36%  1.00x  ONLINE  -
```

특정 저장소 풀 통계 표시

-o 옵션을 사용하여 특정 통계를 요청할 수 있습니다. 이 옵션은 사용자 정의 보고서 또는 관련 정보를 나열할 수 있는 빠른 방법을 제공합니다. 예를 들어, 각 풀의 이름과 크기만 나열하려면 다음 구문을 사용합니다.

```
# zpool list -o name,size
NAME                    SIZE
tank                   80.0G
dozer                   1.2T
```

열 이름은 82 페이지 "모든 저장소 풀 또는 특정 풀에 대한 정보 표시"에 나열된 등록 정보와 일치합니다.

ZFS 저장소 풀 출력 결과 스크립팅

zpool list 명령에 대한 기본 출력은 읽기 편의성을 위주로 디자인되었으며 셸 스크립트의 일부로 사용하기는 쉽지 않습니다. 명령의 프로그래밍 사용 목적을

위해서는 -H 옵션을 사용하여 열 머리글을 숨기고 공백 대신 탭으로 필드를 구분할 수 있습니다. 예를 들어, 시스템의 모든 풀 이름 목록을 요청하려면 다음 구문을 사용합니다.

```
# zpool list -Ho name
tank
dozer
```

다음은 다른 예입니다.

```
# zpool list -H -o name,size
tank   80.0G
dozer  1.2T
```

ZFS 저장소 풀 명령 내역 표시

ZFS는 풀 상태 정보를 수정하는 데 성공한 zfs 및 zpool 명령을 자동으로 기록합니다. 이 정보는 zpool history 명령을 사용하여 표시할 수 있습니다.

예를 들어, 다음 구문은 루트 풀에 대한 명령 출력 결과입니다.

```
# zpool history
History for 'rpool':
2010-05-11.10:18:54 zpool create -f -o failmode=continue -R /a -m legacy -o
cachefile=/tmp/root/etc/zfs/zpool.cache rpool mirror c1t0d0s0 c1t1d0s0
2010-05-11.10:18:55 zfs set canmount=noauto rpool
2010-05-11.10:18:55 zfs set mountpoint=/rpool rpool
2010-05-11.10:18:56 zfs create -o mountpoint=legacy rpool/ROOT
2010-05-11.10:18:57 zfs create -b 8192 -V 2048m rpool/swap
2010-05-11.10:18:58 zfs create -b 131072 -V 1536m rpool/dump
2010-05-11.10:19:01 zfs create -o canmount=noauto rpool/ROOT/zfsBE
2010-05-11.10:19:02 zpool set bootfs=rpool/ROOT/zfsBE rpool
2010-05-11.10:19:02 zfs set mountpoint=/ rpool/ROOT/zfsBE
2010-05-11.10:19:03 zfs set canmount=on rpool
2010-05-11.10:19:04 zfs create -o mountpoint=/export rpool/export
2010-05-11.10:19:05 zfs create rpool/export/home
2010-05-11.11:11:10 zpool set bootfs=rpool rpool
2010-05-11.11:11:10 zpool set bootfs=rpool/ROOT/zfsBE rpool
```

시스템에서 유사한 출력을 사용하여 오류 조건을 해결하기 위해 실행된 **실제** ZFS 명령을 식별할 수 있습니다.

내역 로그의 특징은 다음과 같습니다.

- 로그를 사용 안함으로 설정할 수 없습니다.
- 로그는 디스크에 지속적으로 저장됩니다. 즉, 시스템을 재부트해도 로그가 손실되지 않습니다.
- 링 버퍼로 구현됩니다. 최소 크기는 128KB입니다. 최대 크기는 32MB입니다.
- 소형 풀의 경우 최대 크기가 풀 크기의 1%로 제한됩니다. 이 **크기**는 풀 생성 시점에 결정됩니다.

- 로그는 관리가 필요하지 않습니다. 즉, 로그 크기 조정이나 로그 위치 변경이 불필요합니다.

특정 저장소 풀의 명령 내역을 확인하려면 다음과 유사한 구문을 사용합니다.

```
# zpool history tank
2012-01-25.16:35:32 zpool create -f tank mirror c3t1d0 c3t2d0 spare c3t3d0
2012-02-17.13:04:10 zfs create tank/test
2012-02-17.13:05:01 zfs snapshot -r tank/test@snap1
```

`-l` 옵션을 사용하여 작업이 수행된 사용자 이름, 호스트 이름 및 영역이 포함된 긴 형식을 표시합니다. 예를 들면 다음과 같습니다.

```
# zpool history -l tank
History for 'tank':
2012-01-25.16:35:32 zpool create -f tank mirror c3t1d0 c3t2d0 spare c3t3d0
[user root on tardis:global]
2012-02-17.13:04:10 zfs create tank/test [user root on tardis:global]
2012-02-17.13:05:01 zfs snapshot -r tank/test@snap1 [user root on tardis:global]
```

`-i` 옵션을 사용하여 진단용으로 사용할 수 있는 내부 이벤트 정보를 표시합니다. 예를 들면 다음과 같습니다.

```
# zpool history -i tank
History for 'tank':
2012-01-25.16:35:32 zpool create -f tank mirror c3t1d0 c3t2d0 spare c3t3d0
2012-01-25.16:35:32 [internal pool create txg:5] pool spa 33; zfs spa 33; zpl 5;
uts tardis 5.11 11.1 sun4v
2012-02-17.13:04:10 zfs create tank/test
2012-02-17.13:04:10 [internal property set txg:66094] $share2=2 dataset = 34
2012-02-17.13:04:31 [internal snapshot txg:66095] dataset = 56
2012-02-17.13:05:01 zfs snapshot -r tank/test@snap1
2012-02-17.13:08:00 [internal user hold txg:66102] <.send-4736-1> temp = 1 ...
```

ZFS 저장소 풀에 대한 I/O 통계 보기

풀 또는 특정 가상 장치에 대한 I/O 통계를 요청하려면 `zpool iostat` 명령을 사용합니다. `iostat` 명령과 마찬가지로 이 명령은 모든 지정된 간격 동안의 업데이트된 통계는 물론 모든 I/O 작업의 정적 스냅샷을 표시할 수 있습니다. 다음 통계가 보고됩니다.

`alloc capacity`
: 풀 또는 장치에 현재 저장된 데이터의 양입니다. 이 양은 내부 구현 세부 사항으로 인해 실제 파일 시스템에서 사용할 수 있는 디스크 공간의 양과 약간 차이가 납니다.

  풀 공간과 데이터 세트 공간 간의 차이에 대한 자세한 내용은 30 페이지 "ZFS 디스크 공간 계산"을 참조하십시오.

`free capacity`	풀 또는 장치에서 사용할 수 있는 디스크 공간의 양입니다. `used` 통계와 마찬가지로 이 양은 데이터 세트에서 사용할 수 있는 디스크 공간의 양과 약간 차이가 납니다.
`read operations`	메타 데이터 요청을 포함하여 풀 또는 디스크로 보낸 읽기 I/O 작업의 수입니다.
`write operations`	풀 또는 장치로 보낸 쓰기 I/O 작업의 수입니다.
`read bandwidth`	모든 읽기 작업(메타 데이터 포함)의 대역폭으로 초당 단위로 표시됩니다.
`write bandwidth`	모든 쓰기 작업의 대역폭으로 초당 단위로 표시됩니다.

풀 전역 I/O 통계 나열

옵션 없는 `zpool iostat` 명령은 시스템의 모든 풀에 대한 부트 이후 누적 통계를 표시합니다. 예를 들면 다음과 같습니다.

```
# zpool iostat
               capacity     operations    bandwidth
pool        alloc   free   read  write   read  write
----------  -----  -----  -----  -----  -----  -----
rpool       6.05G  61.9G      0      0    786    107
tank        31.3G  36.7G      4      1   296K  86.1K
----------  -----  -----  -----  -----  -----  -----
```

이러한 통계는 부트 이후 누적되므로 풀이 상대적으로 유휴 상태인 경우 대역폭이 낮게 나타날 수 있습니다. 간격을 지정하면 현재 대역폭 사용에 대한 좀더 정확한 보기를 요청할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool iostat tank 2
               capacity     operations    bandwidth
pool        alloc   free   read  write   read  write
----------  -----  -----  -----  -----  -----  -----
tank        18.5G  49.5G      0    187      0  23.3M
tank        18.5G  49.5G      0    464      0  57.7M
tank        18.5G  49.5G      0    457      0  56.6M
tank        18.8G  49.2G      0    435      0  51.3M
```

위 예에서 명령은 Ctrl-C를 입력할 때까지 2초마다 `tank` 풀에 대한 사용량 통계를 표시합니다. 또는 추가 `count` 인수를 지정하여 지정된 반복 수가 경과하면 명령이 종료되도록 할 수 있습니다.

예를 들어, `zpool iostat 2 3`은 2초마다 3회 반복으로 총 6초 동안의 요약을 출력합니다. 단일 풀만 있을 경우에는 통계가 연속 라인에 표시됩니다. 둘 이상의 풀이 존재할 경우에는 추가 대시 라인이 각 반복을 나타내어 시각적 구분을 제공합니다.

가상 장치 I/O 통계 사용

풀 전역 I/O 통계와 함께 `zpool iostat` 명령은 가상 장치에 대한 I/O 통계를 표시할 수 있습니다. 이 명령은 비정상적으로 느린 장치를 식별하거나 ZFS에서 생성된 I/O의 분포를 관찰하는 데 사용할 수 있습니다. 모든 I/O 통계와 함께 전체 가상 장치 레이아웃을 요청하려면 `zpool iostat -v` 명령을 사용합니다. 예를 들면 다음과 같습니다.

```
# zpool iostat -v
                capacity     operations    bandwidth
pool         alloc   free   read  write   read  write
----------  -----  -----  -----  -----  -----  -----
rpool       6.05G  61.9G      0      0    785    107
  mirror    6.05G  61.9G      0      0    785    107
    c1t0d0s0    -      -      0      0    578    109
    c1t1d0s0    -      -      0      0    595    109
----------  -----  -----  -----  -----  -----  -----
tank        36.5G  31.5G      4      1   295K   146K
  mirror    36.5G  31.5G    126     45  8.13M  4.01M
    c1t2d0      -      -      0      3   100K   386K
    c1t3d0      -      -      0      3   104K   386K
----------  -----  -----  -----  -----  -----  -----
```

가상 장치에 대한 I/O 통계를 볼 때 두 가지 중요한 사항이 있습니다.

- 첫째, 디스크 공간 사용 통계는 최상위 레벨 가상 장치에 대해서만 사용할 수 있습니다. 디스크 공간이 미러 및 RAID-Z 가상 장치에서 할당되는 방식은 구현에 따라 고유하고 단일 숫자로 쉽게 표현되지 않습니다.
- 둘째, 숫자가 예상한 대로 정확하게 증가하지 않을 수 있습니다. 특히, RAID-Z 및 미러된 장치에 걸친 작업은 정확하게 같지 않습니다. 이 차이는 풀이 생성된 직후 많은 양의 I/O가 풀 생성의 일부로 디스크에 직접 보내지고 미러 레벨에서 계산되지 않을 때 쉽게 확인할 수 있습니다. 시간이 지남에 따라 이러한 숫자는 점차 같아집니다. 하지만 손상되거나 응답하지 않거나 오프라인 상태의 장치도 이 대칭에 영향을 줄 수 있습니다.

가상 장치 통계를 조사할 때 동일한 옵션 세트(간격 및 수)를 사용할 수 있습니다.

ZFS 저장소 풀의 건전성 상태 확인

ZFS는 풀 및 장치 건전성을 조사하는 통합된 방법을 제공합니다. 풀의 건전성은 모든 장치의 상태에서 결정됩니다. 이 상태 정보는 `zpool status` 명령을 사용하여 표시됩니다. 또한 잠재적인 풀 및 장치 실패가 `fmd`에 의해 보고되고, 시스템 콘솔에 표시되며, `/var/adm/messages` 파일에 기록됩니다.

이 절에서는 풀 및 장치 건전성을 확인하는 방법을 설명합니다. 이 장에서는 건전하지 않은 풀에서 복구하는 방법을 다루지 않습니다. 문제 해결 및 데이터 복구에 대한 자세한 내용은 10 장, “Oracle Solaris ZFS 문제 해결 및 풀 복구”를 참조하십시오.

풀의 건전성 상태는 다음 4개 상태 중 하나로 설명됩니다.

DEGRADED
: 결함이 있는 장치가 한 개 이상이지만 중복 구성으로 인해 데이터는 계속 사용할 수 있는 풀입니다.

ONLINE
: 모든 장치가 정상적으로 작동 중인 풀입니다.

SUSPENDED
: 장치 연결 복원을 기다리는 중인 풀입니다. 풀은 장치 문제가 해결될 때까지 SUSPENDED 상태를 유지합니다.

UNAVAIL
: 메타 데이터가 손상되었거나 하나 이상의 장치가 사용할 수 없는 상태이고 작동을 계속하기 위해 필요한 복제본이 부족한 상태의 풀입니다.

각 풀 장치는 다음 상태 중 하나에 속할 수 있습니다.

DEGRADED
: 가상 장치에서 실패가 발생하지만 여전히 작동 가능합니다. 이 상태는 미러 또는 RAID-Z 장치가 하나 이상의 구성 장치를 잃을 때 가장 일반적으로 나타납니다. 다른 장치에서 다음에 발생하는 결함을 복구할 수 없는 경우 풀의 내결함성이 침해될 수 있습니다.

OFFLINE
: 장치가 관리자에 의해 명시적으로 오프라인으로 전환되었습니다.

ONLINE
: 장치 또는 가상 장치가 정상적으로 작동하는 상태입니다. 일부 일시적인 오류가 계속 발생할 수 있지만 장치가 정상적으로 작동하는 중입니다.

REMOVED
: 시스템이 실행되는 동안 장치가 물리적으로 제거되었습니다. 장치 제거 감지는 하드웨어에 따라 다르며 일부 플랫폼에서 지원되지 않을 수 있습니다.

UNAVAIL
: 장치 또는 가상 장치를 열 수 없습니다. 경우에 따라 UNAVAIL 장치가 있는 풀이 DEGRADED 모드로 나타날 수 있습니다. 최상위 레벨 가상 장치가 UNAVAIL 상태이면 풀에서 아무것도 액세스할 수 없습니다.

풀의 건전성은 모든 최상위 레벨 가상 장치의 건전성에서 결정됩니다. 모든 가상 장치가 ONLINE이면 풀도 ONLINE입니다. 가상 장치 중 하나라도 DEGRADED 또는 UNAVAIL이면 풀도 DEGRADED입니다. 최상위 가상 장치가 UNAVAIL 또는 OFFLINE 상태이면 이 풀도 UNAVAIL 또는 SUSPENDED 상태입니다. UNAVAIL 또는 SUSPENDED 상태의 풀은 완전히 액세스할 수 없습니다. 필요한 장치가 연결되거나 복구될 때까지 데이터를 복구할 수 없습니다. DEGRADED 상태의 풀은 계속해서 실행되지만, 풀이 온라인일 때만큼 동일한 레벨의 데이터 중복성 또는 데이터 처리량을 기대할 수 없습니다.

zpool status 명령은 리실버링 및 스크러빙 작업에 대한 세부 정보도 제공합니다.

- 리실버링 진행 중 보고입니다. 예를 들면 다음과 같습니다.

```
scan: resilver in progress since Wed Jun 20 14:19:38 2012
    7.43G scanned out of 71.8G at 36.4M/s, 0h30m to go
    7.43G resilvered, 10.35% done
```

- 스크러빙 진행 중 보고입니다. 예를 들면 다음과 같습니다.

```
scan: scrub in progress since Wed Jun 20 14:56:52 2012
    529M scanned out of 71.8G at 48.1M/s, 0h25m to go
    0 repaired, 0.72% done
```

- 리실버링 완료 메시지입니다. 예를 들면 다음과 같습니다.

```
scan: resilvered 71.8G in 0h14m with 0 errors on Wed Jun 20 14:33:42 2012
```

- 스크러빙 완료 메시지입니다. 예를 들면 다음과 같습니다.

```
scan: scrub repaired 0 in 0h11m with 0 errors on Wed Jun 20 15:08:23 2012
```

- 진행 중인 스크러빙 취소 메시지입니다. 예를 들면 다음과 같습니다.

```
scan: scrub canceled on Wed Jun 20 16:04:40 2012
```

- 스크러빙 및 리실버링 완료 메시지는 시스템 재부트 시에도 지속됩니다.

기본 저장소 풀 건전성 상태

다음과 같이 zpool status 명령을 사용하여 풀 건전성 상태를 빠르게 검토할 수 있습니다.

```
# zpool status -x
all pools are healthy
```

명령 구문에 풀 이름을 지정하면 특정 풀을 조사할 수 있습니다. ONLINE 상태에 있지 않은 모든 풀은 다음 절에 설명된 대로 잠재적인 문제를 조사해야 합니다.

자세한 건전성 상태

- v 옵션을 사용하여 좀더 자세한 건전성 요약 상태를 요청할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool status -v tank
  pool: tank
 state: DEGRADED
status: One or more devices could not be opened.  Sufficient replicas exist for
        the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
 scrub: scrub completed after 0h0m with 0 errors on Wed Jan 20 15:13:59 2010
config:

        NAME        STATE     READ WRITE CKSUM
        tank        DEGRADED     0     0     0
          mirror-0  DEGRADED     0     0     0
            c1t0d0  ONLINE       0     0     0
            c1t1d0  UNAVAIL      0     0     0  cannot open
```

```
errors: No known data errors
```

이 출력 결과는 풀이 현재 상태인 이유에 대해 자세히 설명합니다. 여기에는 이해하기 쉬운 설명과 추가 정보를 얻을 수 있는 기술 자료 문서 링크가 포함됩니다. 기술 자료 문서는 현재 문제로부터 복구할 수 있는 가장 좋은 방법에 대한 최신 정보를 제공합니다. 자세한 구성 정보를 사용하면 어떤 장치가 손상되고 어떻게 풀을 복구할 수 있는지 확인할 수 있습니다.

위 예에서 UNAVAIL 장치는 교체해야 합니다. 장치가 교체된 후 필요에 따라 zpool online 명령을 사용하여 장치를 온라인으로 전환합니다. 예를 들면 다음과 같습니다.

```
# zpool online tank c1t0d0
Bringing device c1t0d0 online
# zpool status -x
all pools are healthy

# zpool online pond c0t5000C500335F907Fd0
warning: device 'c0t5000C500335DC60Fd0' onlined, but remains in degraded state
# zpool status -x
all pools are healthy
```

위 출력 결과에서는 리실버링이 완료될 때까지 장치가 성능 저하 상태로 유지됨을 알 수 있습니다.

autoreplace 등록 정보가 on인 경우에는 교체된 장치를 온라인으로 전환하지 않아도 될 수 있습니다.

풀에 오프라인 장치가 있을 경우 명령 출력 결과는 문제가 있는 풀을 나타냅니다. 예를 들면 다음과 같습니다.

```
# zpool status -x
  pool: tank
 state: DEGRADED
status: One or more devices has been taken offline by the administrator.
        Sufficient replicas exist for the pool to continue functioning in a
        degraded state.
action: Online the device using 'zpool online' or replace the device with
        'zpool replace'.
 scrub: resilver completed after 0h0m with 0 errors on Wed Jan 20 15:15:09 2010
config:

        NAME        STATE     READ WRITE CKSUM
        tank        DEGRADED     0     0     0
          mirror-0  DEGRADED     0     0     0
            c1t0d0  ONLINE       0     0     0
            c1t1d0  OFFLINE      0     0     0  48K resilvered

errors: No known data errors

# zpool status -x
 pool: pond
```

```
 state: DEGRADED
status: One or more devices has been taken offline by the administrator.
        Sufficient replicas exist for the pool to continue functioning in a
        degraded state.
action: Online the device using 'zpool online' or replace the device with
        'zpool replace'.
config:

        NAME                      STATE     READ WRITE CKSUM
        pond                      DEGRADED     0     0     0
          mirror-0                DEGRADED     0     0     0
            c0t5000C500335F95E3d0 ONLINE       0     0     0
            c0t5000C500335F907Fd0 OFFLINE      0     0     0
          mirror-1                ONLINE       0     0     0
            c0t5000C500335BD117d0 ONLINE       0     0     0
            c0t5000C500335DC60Fd0 ONLINE       0     0     0

errors: No known data errors
```

READ 및 WRITE 열은 장치에서 발생한 I/O 오류 수를 제공하고, CKSUM 열은 장치에서 발생한 수정할 수 없는 체크섬 오류 수를 제공합니다. 두 오류 수는 모두 잠재적인 장치 실패를 나타내며, 일부는 수정 조치가 필요합니다. 최상위 레벨 가상 장치에 대해 0이 아닌 오류가 보고될 경우 데이터 중 일부에 액세스하지 못할 수 있습니다.

errors: 필드는 알려진 데이터 오류를 나타냅니다.

위의 예에 나온 출력 결과에서 오프라인 장치는 데이터 오류를 유발하지 않습니다.

UNAVAIL 상태의 풀 및 데이터 진단과 복구에 대한 자세한 내용은 10 장, “Oracle Solaris ZFS 문제 해결 및 풀 복구”를 참조하십시오.

ZFS 저장소 풀 상태 정보 수집

zpool status 간격 및 수 옵션을 사용하여 기간에 따른 통계를 수집할 수 있습니다. 또한 -T 옵션을 사용하면 시간 기록을 표시할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool status -T d 3 2
Wed Jun 20 16:10:09 MDT 2012
  pool: pond
 state: ONLINE
  scan: resilvered 9.50K in 0h0m with 0 errors on Wed Jun 20 16:07:34 2012
config:

        NAME                      STATE     READ WRITE CKSUM
        pond                      ONLINE       0     0     0
          mirror-0                ONLINE       0     0     0
            c0t5000C500335F95E3d0 ONLINE       0     0     0
            c0t5000C500335F907Fd0 ONLINE       0     0     0
          mirror-1                ONLINE       0     0     0
            c0t5000C500335BD117d0 ONLINE       0     0     0
            c0t5000C500335DC60Fd0 ONLINE       0     0     0

errors: No known data errors
```

```
  pool: rpool
 state: ONLINE
  scan: scrub repaired 0 in 0h11m with 0 errors on Wed Jun 20 15:08:23 2012
config:

        NAME                          STATE     READ WRITE CKSUM
        rpool                         ONLINE       0     0     0
          mirror-0                    ONLINE       0     0     0
            c0t5000C500335BA8C3d0s0   ONLINE       0     0     0
            c0t5000C500335FC3E7d0s0   ONLINE       0     0     0

errors: No known data errors
Wed Jun 20 16:10:12 MDT 2012

  pool: pond
 state: ONLINE
  scan: resilvered 9.50K in 0h0m with 0 errors on Wed Jun 20 16:07:34 2012
config:

        NAME                        STATE     READ WRITE CKSUM
        pond                        ONLINE       0     0     0
          mirror-0                  ONLINE       0     0     0
            c0t5000C500335F95E3d0   ONLINE       0     0     0
            c0t5000C500335F907Fd0   ONLINE       0     0     0
          mirror-1                  ONLINE       0     0     0
            c0t5000C500335BD117d0   ONLINE       0     0     0
            c0t5000C500335DC60Fd0   ONLINE       0     0     0

errors: No known data errors

  pool: rpool
 state: ONLINE
  scan: scrub repaired 0 in 0h11m with 0 errors on Wed Jun 20 15:08:23 2012
config:

        NAME                          STATE     READ WRITE CKSUM
        rpool                         ONLINE       0     0     0
          mirror-0                    ONLINE       0     0     0
            c0t5000C500335BA8C3d0s0   ONLINE       0     0     0
            c0t5000C500335FC3E7d0s0   ONLINE       0     0     0

errors: No known data errors
```

ZFS 저장소 풀 마이그레이션

때때로 시스템 간에 저장소 풀을 이동해야 할 수 있습니다. 이를 위해서는 저장 장치를 원래 시스템에서 분리하고 대상 시스템에 다시 연결해야 합니다. 이 작업은 장치 케이블을 물리적으로 연결하거나 SAN의 장치와 같은 다중 포트 장치를 사용하여 수행할 수 있습니다. 시스템의 아키텍처 엔디안이 서로 다르더라도 ZFS를 통해 한 시스템에서 풀을 내보내고 대상 시스템에서 가져올 수 있습니다. 서로 다른 시스템에 상주할 수 있는 서로 다른 저장소 풀 간의 파일 시스템 복제 또는 마이그레이션에 대한 자세한 내용은 212 페이지 “ZFS 데이터 전송 및 수신”을 참조하십시오.

ZFS 저장소 풀 마이그레이션 준비

마이그레이션 준비가 되었음을 나타내기 위해서는 저장소 풀을 명시적으로 내보내기되어 있어야 합니다. 이 작업은 쓰여지지 않은 데이터를 디스크에 비우고 데이터를 디스크에 기록하여 내보내기가 수행되었음을 나타내고, 시스템에서 풀에 대한 모든 정보를 제거합니다.

풀을 명시적으로 내보내지 않고 대신, 디스크를 수동으로 제거할 경우에도 다른 시스템에서 결과 풀을 가져올 수 있습니다. 하지만 몇 초 정도의 데이터 트랜잭션을 잃을 수 있고, 장치가 더 이상 존재하지 않으므로 원래 시스템에는 풀이 UNAVAIL 상태인 것으로 나타납니다. 기본적으로 대상 시스템은 명시적으로 내보내기되지 않은 풀은 가져올 수 없습니다. 이 조건은 아직 다른 시스템에서 사용 중인 네트워크 연결 저장소를 구성하는 활성 풀을 실수로 가져오지 못하도록 하기 위해 필요합니다.

ZFS 저장소 풀 내보내기

풀을 내보내려면 zpool export 명령을 사용합니다. 예를 들면 다음과 같습니다.

```
# zpool export tank
```

명령은 계속하기 전에 풀 내에서 마운트된 모든 파일 시스템을 마운트 해제하려고 시도합니다. 파일 시스템 마운트 해제를 실패할 경우 -f 옵션을 사용하여 강제로 마운트 해제할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool export tank
cannot unmount '/export/home/eric': Device busy
# zpool export -f tank
```

이 명령이 실행된 후 tank 풀은 시스템에서 더 이상 보이지 않습니다.

내보내기 시점에 장치를 사용할 수 없을 경우에는 장치가 분명히 내보내졌는지 확인할 수 없습니다. 나중에 이러한 장치 중 하나를 작동하는 장치가 없는 시스템에 연결할 경우 “잠재적으로 활성” 상태로 나타납니다.

ZFS 볼륨이 풀에서 사용 중인 경우에는 -f 옵션으로도 풀을 내보낼 수 없습니다. ZFS 볼륨이 있는 풀을 내보내려면 먼저 볼륨의 모든 소비자가 더 이상 활성 상태가 아닌지 확인합니다.

ZFS 볼륨에 대한 자세한 내용은 261 페이지 “ZFS 볼륨”을 참조하십시오.

가져올 수 있는 저장소 풀 결정

풀이 시스템에서 제거되었으면(명시적인 내보내기를 통해 또는 강제로 장치를 제거하여) 장치를 대상 시스템에 연결할 수 있습니다. ZFS는 장치 중 일부만 사용할 수 있는 몇 가지 상황을 처리할 수 있지만, 성공적인 풀 마이그레이션은 장치의 전체적인 건전성에 좌우됩니다. 또한 장치를 반드시 동일한 장치 이름으로 연결할 필요는 없습니다. ZFS는 이동되거나 이름이 바뀐 장치를 감지하고 그에 따라 구성을 조정합니다. 사용 가능한 풀을 검색하려면 `zpool import` 명령을 옵션 없이 실행합니다. 예를 들면 다음과 같습니다.

```
# zpool import
 pool: tank
    id: 11809215114195894163
 state: ONLINE
action: The pool can be imported using its name or numeric identifier.
config:

        tank        ONLINE
          mirror-0  ONLINE
            c1t0d0  ONLINE
            c1t1d0  ONLINE
```

이 예에서 `tank` 풀을 대상 시스템에서 가져올 수 있습니다. 각 풀은 이름 및 고유한 숫자 식별자로 식별됩니다. 동일한 이름의 여러 풀을 가져올 수 있을 경우 숫자 식별자를 사용하여 구분할 수 있습니다.

`zpool status` 명령과 마찬가지로 `zpool import` 출력 결과에는 풀을 가져오지 못하게 하는 문제에 대한 복구 절차와 관련된 최신 정보를 제공하는 기술 자료 문서 링크가 포함되어 있습니다. 이 상황에서 사용자는 풀을 강제로 가져올 수 있습니다. 하지만 현재 다른 시스템에서 저장소 네트워크를 통해 사용 중인 풀을 가져오면 두 시스템에서 동일 저장소에 쓰기를 시도할 때 데이터 손상 및 패닉이 발생할 수 있습니다. 풀에서 일부 장치를 사용할 수 없지만 충분한 중복 데이터가 존재하여 풀이 사용 가능한 경우 풀은 `DEGRADED` 상태로 나타납니다. 예를 들면 다음과 같습니다.

```
# zpool import
  pool: tank
    id: 11809215114195894163
 state: DEGRADED
status: One or more devices are missing from the system.
action: The pool can be imported despite missing or damaged devices.  The
        fault tolerance of the pool may be compromised if imported.
   see: http://www.sun.com/msg/ZFS-8000-2Q
config:

        NAME        STATE     READ WRITE CKSUM
        tank        DEGRADED     0     0     0
          mirror-0  DEGRADED     0     0     0
```

```
            c1t0d0  UNAVAIL      0     0     0  cannot open
            c1t3d0  ONLINE       0     0     0
```

이 예에서 미러된 데이터에 여전히 액세스할 수 있으므로 풀을 가져올 수 있더라도 첫번째 디스크는 손상되거나 누락되었습니다. 사용할 수 없는 장치가 너무 많은 경우 풀을 가져올 수 없습니다.

이 예에서는 2개의 디스크가 RAID-Z 가상 장치에서 누락되었으므로 풀을 재구성할 수 있을 만큼 충분한 중복 데이터를 사용할 수 없습니다. 전체 구성을 결정할 수 있을 만큼 충분한 장치가 존재하지 않을 수 있습니다. ZFS는 상황에 대해 가능한 많은 정보를 보고하지만 이 경우 ZFS는 어떤 다른 장치가 풀의 일부였는지 확인할 수 없습니다. 예를 들면 다음과 같습니다.

```
# zpool import
pool: dozer
    id: 9784486589352144634
 state: FAULTED
status: One or more devices are missing from the system.
action: The pool cannot be imported. Attach the missing
        devices and try again.
   see: http://www.sun.com/msg/ZFS-8000-6X
config:
        dozer          FAULTED   missing device
          raidz1-0     ONLINE
            c1t0d0     ONLINE
            c1t1d0     ONLINE
            c1t2d0     ONLINE
            c1t3d0     ONLINE
Additional devices are known to be part of this pool, though their
exact configuration cannot be determined.
```

대체 디렉토리에서 ZFS 저장소 풀 가져오기

기본적으로 `zpool import` 명령은 `/dev/dsk` 디렉토리 내의 장치만 검색합니다. 장치가 다른 디렉토리에 존재하거나 파일로 백업되는 풀을 사용 중인 경우 `-d` 옵션을 사용하여 대체 디렉토리를 검색해야 합니다. 예를 들면 다음과 같습니다.

```
# zpool create dozer mirror /file/a /file/b
# zpool export dozer
# zpool import -d /file
  pool: dozer
    id: 7318163511366751416
 state: ONLINE
action: The pool can be imported using its name or numeric identifier.
config:

        dozer        ONLINE
          mirror-0   ONLINE
            /file/a  ONLINE
            /file/b  ONLINE
# zpool import -d /file dozer
```

장치가 여러 디렉토리에 존재할 경우 여러 -d 옵션을 지정할 수 있습니다.

ZFS 저장소 풀 가져오기

가져올 풀이 확인되었으면 풀의 이름이나 숫자 식별자를 zpool import 명령의 인수로 지정하여 가져올 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool import tank
```

여러 사용 가능한 풀의 이름이 동일한 경우 숫자 식별자를 사용하여 가져올 풀을 지정해야 합니다. 예를 들면 다음과 같습니다.

```
# zpool import
  pool: dozer
    id: 2704475622193776801
 state: ONLINE
action: The pool can be imported using its name or numeric identifier.
config:

        dozer       ONLINE
          c1t9d0    ONLINE

  pool: dozer
    id: 6223921996155991199
 state: ONLINE
action: The pool can be imported using its name or numeric identifier.
config:

        dozer       ONLINE
          c1t8d0    ONLINE
# zpool import dozer
cannot import 'dozer': more than one matching pool
import by numeric ID instead
# zpool import 6223921996155991199
```

풀 이름이 기존 풀 이름과 충돌하는 경우에는 다른 이름으로 풀을 가져올 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool import dozer zeepool
```

이 명령은 새 이름 zeepool을 사용하여 내보낸 풀 dozer를 가져옵니다. 새 풀 이름은 지속됩니다.

풀이 명시적으로 내보내기되지 않은 경우 ZFS는 -f 플래그를 통해 아직 다른 시스템에서 사용 중인 풀을 사용자가 실수로 가져오지 못하도록 해야 합니다. 예를 들면 다음과 같습니다.

```
# zpool import dozer
cannot import 'dozer': pool may be in use on another system
```

```
use '-f' to import anyway
# zpool import -f dozer
```

주 – 한 시스템에서 활성 상태인 풀을 다른 시스템으로 가져오려고 시도하지 마십시오. ZFS는 고유 클러스터, 분산 또는 병렬 파일 시스템이 아니며 서로 다른 여러 호스트에서 동시 액세스를 제공하지 못합니다.

-R 옵션을 사용하면 대체 루트로 풀을 가져올 수도 있습니다. 대체 루트 풀에 대한 자세한 내용은 269 페이지 "ZFS 대체 루트 풀 사용"을 참조하십시오.

누락된 로그 장치가 있는 풀 가져오기

기본적으로 로그 장치가 누락된 풀은 가져올 수 없습니다. zpool import -m 명령을 사용하면 로그 장치가 누락된 풀을 강제로 가져올 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool import dozer
The devices below are missing, use '-m' to import the pool anyway:
            c3t3d0 [log]

cannot import 'dozer': one or more devices is currently unavailable
```

로그 장치가 누락된 풀을 가져옵니다. 예를 들면 다음과 같습니다.

```
# zpool import -m dozer
# zpool status dozer
  pool: dozer
 state: DEGRADED
status: One or more devices could not be opened.  Sufficient replicas exist for
        the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
  scan: none requested
config:

        NAME                     STATE     READ WRITE CKSUM
        dozer                    DEGRADED     0     0     0
          mirror-0               ONLINE       0     0     0
            c8t0d0               ONLINE       0     0     0
            c8t1d0               ONLINE       0     0     0
        logs
          2189413556875979854    UNAVAIL      0     0     0

errors: No known data errors
```

누락된 로그 장치를 연결한 후 zpool clear 명령을 실행하여 풀 오류를 치웁니다.

미러된 로그 장치가 누락된 경우에도 유사한 복구를 시도할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool import dozer
The devices below are missing, use '-m' to import the pool anyway:
            mirror-1 [log]
              c3t3d0
              c3t4d0

cannot import 'dozer': one or more devices is currently unavailable
# zpool import -m dozer
# zpool status dozer
  pool: dozer
 state: DEGRADED
status: One or more devices could not be opened.  Sufficient replicas exist for
        the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
 scan: scrub repaired 0 in 0h0m with 0 errors on Fri Oct 15 16:51:39 2010
config:

        NAME                      STATE     READ WRITE CKSUM
        dozer                     DEGRADED     0     0     0
          mirror-0                ONLINE       0     0     0
            c3t1d0                ONLINE       0     0     0
            c3t2d0                ONLINE       0     0     0
        logs
          mirror-1                UNAVAIL      0     0     0  insufficient replicas
            13514061426445294202  UNAVAIL      0     0     0  was c3t3d0
            16839344638582008929  UNAVAIL      0     0     0  was c3t4d0
```

누락된 로그 장치를 연결한 후 `zpool clear` 명령을 실행하여 풀 오류를 치웁니다.

읽기 전용 모드로 풀 가져오기

읽기 전용 모드로 풀을 가져올 수 있습니다. 풀이 많이 손상되어 액세스할 수 없을 경우 이 기능을 통해 풀의 데이터를 복구할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool import -o readonly=on tank
# zpool scrub tank
cannot scrub tank: pool is read-only
```

풀을 읽기 전용 모드로 가져올 때 다음 조건이 적용됩니다.

- 모든 파일 시스템 및 볼륨이 읽기 전용 모드로 마운트됩니다.
- 풀 트랜잭션 처리가 사용 안함으로 설정됩니다. 또한 의도 로그에서 보류 중인 동기화 쓰기는 풀이 읽기-쓰기로 가져올 때까지 실행되지 않습니다.
- 읽기 전용 가져오기 중 풀 등록 정보를 설정하려는 시도는 무시됩니다.

읽기 전용 풀은 풀을 내보내기 및 가져오기하여 읽기-쓰기 모드로 다시 설정할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool export tank
# zpool import tank
# zpool scrub tank
```

특정 장치 경로로 풀 가져오기

이 예에서 다음 명령은 풀의 특정 장치 중 하나인 /dev/dsk/c2t3d0을 식별하여 dpool 풀을 가져옵니다.

```
# zpool import -d /dev/dsk/c2t3d0s0 dpool
# zpool status dpool
  pool: dpool
 state: ONLINE
 scan: resilvered 952K in 0h0m with 0 errors on Fri Jun 29 16:22:06 2012
config:

        NAME        STATE     READ WRITE CKSUM
        dpool       ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c2t3d0  ONLINE       0     0     0
            c2t1d0  ONLINE       0     0     0
```

이 풀이 전체 디스크로 구성되어 있더라도 명령에는 특정 장치의 슬라이스 식별자가 포함되어야 합니다.

삭제된 ZFS 저장소 풀 복구

zpool import -D 명령을 사용하여 삭제된 저장소 풀을 복구할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool destroy tank
# zpool import -D
  pool: tank
    id: 5154272182900538157
 state: ONLINE (DESTROYED)
action: The pool can be imported using its name or numeric identifier.
config:

        tank        ONLINE
          mirror-0  ONLINE
            c1t0d0  ONLINE
            c1t1d0  ONLINE
```

이 zpool import 출력 결과에서는 다음 상태 정보 덕분에 tank 풀을 삭제된 풀로 식별할 수 있습니다.

```
state: ONLINE (DESTROYED)
```

삭제된 풀을 복구하려면 복구할 풀과 함께 zpool import - D 명령을 다시 실행합니다. 예를 들면 다음과 같습니다.

```
# zpool import -D tank
# zpool status tank
  pool: tank
 state: ONLINE
```

```
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE
          mirror-0  ONLINE
            c1t0d0  ONLINE
            c1t1d0  ONLINE

errors: No known data errors
```

삭제된 풀에서 장치 중 하나를 사용할 수 없는 경우 `-f` 옵션을 포함시키면 삭제된 풀을 강제로 복구할 수 있습니다. 이 시나리오에서는 결함이 있는 풀을 가져온 다음 장치 실패 수정을 시도합니다. 예를 들면 다음과 같습니다.

```
# zpool destroy dozer
# zpool import -D
  pool: dozer
    id: 4107023015970708695
 state: DEGRADED (DESTROYED)
status: One or more devices could not be opened.  Sufficient replicas exist for
        the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
config:
        dozer              DEGRADED
          raidz2-0         DEGRADED
            c8t0d0         ONLINE
            c8t1d0         ONLINE
            c8t2d0         ONLINE
            c8t3d0         UNAVAIL  cannot open
            c8t4d0         ONLINE
errors: No known data errors
# zpool import -Df dozer
# zpool status -x
  pool: dozer
 state: DEGRADED
status: One or more devices could not be opened.  Sufficient replicas exist for
        the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
   scan: none requested
config:

        NAME                     STATE     READ WRITE CKSUM
        dozer                    DEGRADED     0     0     0
          raidz2-0               DEGRADED     0     0     0
            c8t0d0               ONLINE       0     0     0
            c8t1d0               ONLINE       0     0     0
            c8t2d0               ONLINE       0     0     0
            4881130428504041127  UNAVAIL      0     0     0
            c8t4d0        ONLINE       0     0     0

errors: No known data errors
# zpool online dozer c8t4d0
# zpool status -x
all pools are healthy
```

ZFS 저장소 풀 업그레이드

이전 Solaris 릴리스의 ZFS 저장소 풀이 있는 경우 `zpool upgrade` 명령으로 풀을 업그레이드하여 현재 릴리스의 풀 기능을 활용할 수 있습니다. 또한 `zpool status` 명령은 풀에서 이전 버전을 실행할 경우 이를 사용자에게 알립니다. 예를 들면 다음과 같습니다.

```
# zpool status
  pool: tank
 state: ONLINE
status: The pool is formatted using an older on-disk format.  The pool can
        still be used, but some features are unavailable.
action: Upgrade the pool using 'zpool upgrade'.  Once this is done, the
        pool will no longer be accessible on older software versions.
 scrub: none requested
config:
        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c1t0d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
errors: No known data errors
```

다음 구문을 사용하면 특정 버전 및 지원되는 릴리스에 대한 추가 정보를 확인할 수 있습니다.

```
# zpool upgrade -v
This system is currently running ZFS pool version 22.

The following versions are supported:

VER  DESCRIPTION
---  --------------------------------------------------------
 1   Initial ZFS version
 2   Ditto blocks (replicated metadata)
 3   Hot spares and double parity RAID-Z
 4   zpool history
 5   Compression using the gzip algorithm
 6   bootfs pool property
 7   Separate intent log devices
 8   Delegated administration
 9   refquota and refreservation properties
 10  Cache devices
 11  Improved scrub performance
 12  Snapshot properties
 13  snapused property
 14  passthrough-x aclinherit
 15  user/group space accounting
 16  stmf property support
 17  Triple-parity RAID-Z
 18  Snapshot user holds
 19  Log device removal
 20  Compression using zle (zero-length encoding)
 21  Reserved
 22  Received properties
```

```
For more information on a particular version, including supported releases,
see the ZFS Administration Guide.
```

그런 다음 `zpool upgrade` 명령을 실행하여 모든 풀을 업그레이드할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool upgrade -a
```

주 – 풀을 최신 ZFS 버전으로 업그레이드하면 이전 ZFS 버전을 실행하는 시스템에서 풀에 액세스하지 못하게 됩니다.

이전 Solaris 릴리스의 풀을 사용하는 시스템에서 ZFS 관리 콘솔을 사용하려면 콘솔을 사용하기 전에 풀을 업그레이드해야 합니다. 풀을 업그레이드해야 하는지 확인하려면 `zpool status` 명령을 사용합니다.

◆ ◆ ◆ 4 장

4 Oracle Solaris ZFS 루트 파일 시스템 설치 및 부트

이 장에서는 Oracle Solaris ZFS 루트 파일 시스템 설치 및 부트 방법에 대해 설명합니다. Oracle Solaris Live Upgrade 기능을 사용하여 UFS 루트 파일 시스템을 ZFS 파일 시스템으로 마이그레이션하는 내용도 다룹니다.

이 장에서는 다음과 같은 내용을 다룹니다.

이 릴리스의 알려진 문제 목록은 **Oracle Solaris 10 1/13 릴리스 노트**를 참조하십시오.

Oracle Solaris ZFS 루트 파일 시스템 설치 및 부트(개요)

다음 방법으로 ZFS 루트 파일 시스템을 설치하고 부트할 수 있습니다.

- **Oracle Solaris 초기 설치(대화식 텍스트 모드 설치 방법)**
  - ZFS를 루트 파일 시스템으로 선택하여 설치합니다.
  - ZFS Flash 아카이브를 설치합니다.
- **Oracle Solaris Live Upgrade 기능**
  - UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션합니다.
  - 새 ZFS 루트 풀에서 부트 환경을 새로 만듭니다.

  - 기존 ZFS 루트 풀에서 부트 환경을 만들거나 업데이트합니다.
  - ZFS Flash 아카이브를 사용하여 대체 BE(부트 환경)를 업그레이드합니다.
- **Oracle Solaris JumpStart 기능**
  - 프로파일을 만들어 ZFS 루트 파일 시스템이 있는 시스템을 자동으로 설치합니다.
  - 프로파일을 만들어 ZFS Flash 아카이브를 통해 시스템을 자동으로 설치합니다.

SPARC 기반 또는 x86 기반 시스템이 설치되거나 ZFS 루트 파일 시스템으로 마이그레이션되면 ZFS 루트 파일 시스템에서 시스템이 자동으로 부트됩니다. 부트 변경 사항에 대한 자세한 내용은 150 페이지 “ZFS 루트 파일 시스템에서 부트”를 참조하십시오.

ZFS 설치 기능

이 Oracle Solaris 릴리스에서는 다음과 같은 ZFS 설치 기능이 제공됩니다.

- 대화식 텍스트 설치 프로그램 기능을 사용하여 UFS 또는 ZFS 루트 파일 시스템을 설치할 수 있습니다. 이 릴리스의 기본 파일 시스템은 이전과 마찬가지로 UFS입니다. 다음 방법으로 대화식 텍스트 설치 프로그램에 액세스할 수 있습니다.
  - SPARC: Oracle Solaris 설치 DVD에 다음 구문을 사용합니다.

    ```
    ok boot cdrom - text
    ```
  - SPARC: 네트워크에서 부트할 때 다음 구문을 사용합니다.

    ```
    ok boot net - text
    ```
  - x86: 텍스트 모드 설치 방법을 선택합니다.
- 사용자 정의 JumpStart 프로파일은 다음 기능을 제공합니다.
  - 프로파일을 설정하여 ZFS 저장소 풀을 만들고 부트 가능 ZFS 파일 시스템을 지정할 수 있습니다.
  - 프로파일을 설정하여 ZFS 루트 풀의 Flash 아카이브를 설치할 수 있습니다.
- Live Upgrade를 사용하여 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션할 수 있습니다.
- 설치 중 두 개의 디스크를 선택하여 미러링된 ZFS 루트 풀을 설정할 수 있습니다. 또는 설치 후 추가 디스크를 연결하여 미러링된 ZFS 루트 풀을 만들 수 있습니다.
- ZFS 루트 풀의 ZFS 볼륨에 스왑 및 덤프 장치가 자동으로 만들어집니다.

이 릴리스에서는 다음과 같은 설치 기능이 제공되지 않습니다.

- 현재 ZFS 루트 파일 시스템 설치에 GUI 설치 기능을 사용할 수 없습니다. ZFS 루트 파일 시스템을 설치하려면 텍스트 모드 설치 방법을 선택해야 합니다.
- 표준 업그레이드 프로그램을 사용하여 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 업그레이드할 수 없습니다.

ZFS 지원을 위한 Oracle Solaris 설치 및 Live Upgrade 요구 사항

ZFS 루트 파일 시스템과 함께 시스템을 설치하거나 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션하기 전에 다음 요구 사항이 충족되는지 확인하십시오.

Oracle Solaris 릴리스 요구 사항

다음 방법으로 ZFS 루트 파일 시스템을 설치 및 부트하거나 ZFS 루트 파일 시스템으로 마이그레이션할 수 있습니다.

- ZFS 루트 파일 시스템 설치 – Solaris 10 10/08 릴리스부터 제공됩니다.
- Live Upgrade를 통해 UFS 루트 파일 시스템에서 ZFS 루트 파일 시스템으로 마이그레이션 – Solaris 10 10/08 이상 릴리스가 설치되어 있거나 Solaris 10 10/08 이상 릴리스로 업그레이드되어 있어야 합니다.

일반 ZFS 루트 풀 요구 사항

다음 절에서는 ZFS 루트 풀 공간 및 구성 요구 사항에 대해 설명합니다.

ZFS 루트 풀의 디스크 공간 요구 사항

ZFS 루트 파일 시스템의 경우 UFS 루트 파일 시스템에 비해 사용 가능한 최소 풀 공간이 많이 필요합니다. ZFS 루트 환경에서는 스왑 장치와 덤프 장치가 별도의 장치여야 하기 때문입니다. 기본적으로 UFS 루트 파일 시스템에서는 스왑 장치와 덤프 장치가 동일한 장치입니다.

ZFS 루트 파일 시스템이 있는 시스템을 설치 또는 업그레이드하는 경우 스왑 영역 및 덤프 장치의 크기는 물리적 메모리 양에 따라 다릅니다. 부트 가능 ZFS 루트 파일 시스템에서 사용 가능한 최소 풀 공간은 물리적 메모리, 사용 가능한 디스크 공간 및 만들려는 BE(부트 환경) 수에 따라 다릅니다.

ZFS 저장소 풀에 대한 다음 디스크 공간 요구 사항을 검토하십시오.

- ZFS 루트 파일 시스템 설치에 필요한 최소 메모리는 1536MB입니다.
- 전반적인 ZFS 성능 향상을 위해 권장되는 메모리는 1536MB 이상입니다.
- 권장되는 최소 디스크 공간은 16GB입니다. 디스크 공간은 다음과 같이 사용됩니다.
  - **스왑 영역 및 덤프 장치** – Oracle Solaris 설치 프로그램에서 만드는 기본 스왑 및 덤프 볼륨 크기는 다음과 같습니다.
    - **초기 설치** – 새 ZFS 부트 환경에서 기본 스왑 크기는 물리적 메모리 크기의 절반(일반적으로 512MB-2GB 범위)으로 계산됩니다. 초기 설치 중 스왑 크기를 조정할 수 있습니다.
    - 기본 덤프 크기는 `dumpadm` 정보 및 물리적 메모리 크기를 기반으로 커널을 통해 계산됩니다. 초기 설치 중 덤프 크기를 조정할 수 있습니다.

- **Live Upgrade** – UFS 루트 파일 시스템이 ZFS 루트 파일 시스템으로 마이그레이션되면 UFS BE 스왑 장치의 크기로 ZFS BE의 기본 스왑 크기가 계산됩니다. 기본 스왑 크기 계산을 통해 UFS BE에 있는 모든 스왑 장치의 크기가 합계되며 ZFS BE에 해당 크기의 ZFS 볼륨이 만들어집니다. UFS BE에 정의된 스왑 장치가 없을 경우 기본 스왑 크기는 512MB로 설정됩니다.
- ZFS BE에서 기본 덤프 크기는 물리적 메모리 크기의 절반(512MB-2GB)으로 설정됩니다.

스왑 및 덤프 볼륨의 크기를 선택한 크기로 조정할 수 있습니다. 단, 새 크기가 시스템 작동을 지원해야 합니다. 자세한 내용은 147 페이지 "ZFS 스왑 장치 및 덤프 장치의 크기 조정"을 참조하십시오.

- **BE(부트 환경)** – UFS BE에서 마이그레이션된 ZFS BE에는 새 스왑 및 덤프 공간 요구 사항 또는 조정된 스왑 및 덤프 장치 크기 외에 추가로 약 6GB가 필요합니다. 다른 ZFS BE에서 복제된 각 ZFS BE에는 추가 디스크 공간이 필요하지 않지만 패치가 적용될 때 BE 크기가 증가한다는 점을 고려하십시오. 동일한 루트 풀에 있는 모든 ZFS BE는 동일한 스왑 및 덤프 장치를 사용합니다.
- **Oracle Solaris OS 구성 요소** – `/var`을 제외하고 OS 이미지에 속한 루트 파일 시스템의 모든 하위 디렉토리는 루트 파일 시스템과 동일한 데이터 세트에 있어야 합니다. 스왑 및 덤프 장치를 제외한 모든 OS 구성 요소도 루트 풀에 상주해야 합니다.

  기본 스왑 및 덤프 장치를 Live Upgrade로 변경에 대한 자세한 내용은 148 페이지 "ZFS 스왑 및 덤프 볼륨 사용자 정의"를 참조하십시오.

  또한 `/var` 디렉토리 또는 데이터 세트는 단일 데이터 세트가어야 합니다. 예를 들어, Live Upgrade를 사용하여 ZFS BE를 마이그레이션하거나 패치를 적용하거나 이 풀의 ZFS Flash 아카이브를 만들려는 경우 종속 `/var` 데이터 세트(예: `/var/tmp`)를 만들 수 없습니다.

예를 들어, 디스크 공간이 12GB인 시스템은 부트 가능 ZFS 환경에 너무 작은 것일 수 있습니다. 스왑 및 덤프 장치마다 2GB의 디스크 공간이 필요하며 UFS BE에서 마이그레이션된 ZFS BE에 약 6GB의 디스크 공간이 필요하기 때문입니다.

ZFS 루트 풀 구성 요구 사항

다음 ZFS 루트 풀 구성 요구 사항을 검토하십시오.

- 루트 풀로 사용할 풀의 레이블은 SMI여야 합니다. 풀이 디스크 슬라이스로 만들어진 경우 일반적으로 이 요구 사항이 충족됩니다.
- 풀은 디스크 슬라이스 또는 미러링된 디스크 슬라이스에 존재해야 합니다. Live Upgrade 마이그레이션 중 지원되지 않는 풀 구성을 사용하려고 시도하면 다음과 유사한 메시지가 표시됩니다.

  ```
  ERROR: ZFS pool name does not support boot environments
  ```

  지원되는 ZFS 루트 풀 구성에 대한 자세한 내용은 50 페이지 "ZFS 루트 풀 만들기"를 참조하십시오.

- x86: 디스크에 Oracle Solaris `fdisk` 분할 영역이 포함되어야 합니다. 이 `fdisk` 분할 영역은 x86 기반 시스템이 설치될 때 자동으로 만들어집니다. Solaris `fdisk` 분할 영역에 대한 자세한 내용은 **System Administration Guide: Devices and File Systems**의 "Guidelines for Creating an fdisk Partition"을 참조하십시오.
- ZFS 루트 풀 부트용으로 지정된 디스크는 SPARC 기반 시스템과 x86 기반 시스템에서 모두 2TB 미만이어야 합니다.
- 루트 풀에서 압축을 사용할 수 있지만 이는 루트 풀이 설치된 후에만 가능합니다. 설치 중에는 루트 풀을 압축할 수 있는 방법이 없습니다. 루트 풀에서는 `gzip` 압축 알고리즘이 지원되지 않습니다.
- 초기 설치로 만들어진 후 또는 Solaris Live Upgrade를 통해 ZFS 루트 파일 시스템으로 마이그레이션된 후 루트 풀의 이름을 바꾸지 마십시오. 루트 풀의 이름을 바꾸면 시스템이 부트되지 않을 수 있습니다.

  또한 Live Upgrade를 사용하려면 루트 풀 구성 요소의 기본 마운트 지점을 변경하지 마십시오.
- 스왑 및 덤프 장치를 변경하고 Live Upgrade를 사용하려면 148 페이지 "ZFS 스왑 및 덤프 볼륨 사용자 정의"를 참조하십시오.

ZFS 루트 파일 시스템 설치(Oracle Solaris 초기 설치)

이 Oracle Solaris 릴리스에서는 다음 방법으로 초기 설치를 수행할 수 있습니다.

- 대화식 텍스트 설치 프로그램을 사용하여 부트 가능 ZFS 루트 파일 시스템을 포함하는 ZFS 저장소 풀을 처음으로 설치합니다. ZFS 루트 파일 시스템에 사용할 기존 ZFS 저장소 풀이 있을 경우 Live Upgrade를 사용하여 기존 UFS 루트 파일 시스템을 기존 ZFS 저장소 풀의 ZFS 루트 파일 시스템으로 마이그레이션해야 합니다. 자세한 내용은 122 페이지 "ZFS 루트 파일 시스템으로 마이그레이션 또는 ZFS 루트 파일 시스템 업데이트(Live Upgrade)"를 참조하십시오.
- 대화식 텍스트 설치 프로그램을 사용하여 ZFS Flash 아카이브의 부트 가능 ZFS 루트 파일 시스템을 포함하는 ZFS 저장소 풀을 처음으로 설치합니다.

ZFS 저장소 풀을 만드는 초기 설치를 시작하기 전에 105 페이지 "ZFS 지원을 위한 Oracle Solaris 설치 및 Live Upgrade 요구 사항"을 참조하십시오.

ZFS 루트 파일 시스템 초기 설치 후 영역을 구성하고 시스템에 패치를 적용하거나 시스템을 업그레이드하려면 131 페이지 "Live Upgrade를 사용하여 영역이 있는 시스템 마이그레이션 또는 업그레이드(Solaris 10 10/08)" 또는 136 페이지 "Oracle Solaris Live Upgrade를 사용하여 영역이 있는 시스템 마이그레이션 또는 업그레이드(Solaris 10 5/09 이상)"를 참조하십시오.

시스템에 ZFS 저장소 풀이 이미 있을 경우 다음 메시지가 표시됩니다. 하지만 새 저장소 풀을 만들기 위해 기존 풀의 디스크를 선택하지 않을 경우 해당 풀은 그대로 유지됩니다.

```
There are existing ZFS pools available on this system.  However, they can only be upgraded
using the Live Upgrade tools.  The following screens will only allow you to install a ZFS root system,
not upgrade one.
```

주의 – 새 풀에 대한 디스크가 선택되면 기존 풀이 삭제됩니다.

예 4–1 부트 가능 ZFS 루트 파일 시스템 초기 설치

일반적으로 대화식 텍스트 설치 프로세스는 UFS 루트 파일 시스템을 만들지 아니면 ZFS 루트 파일 시스템을 만들지 묻는 프롬프트가 표시되는 것만 다르고 이전 Oracle Solaris 릴리스와 동일합니다. 이 릴리스에서도 UFS가 기본 파일 시스템입니다. ZFS 루트 파일 시스템을 선택하면 ZFS 저장소 풀을 만들지 묻는 프롬프트가 표시됩니다. ZFS 루트 파일 시스템 설치 단계는 다음과 같습니다.

1. Oracle Solaris 설치 매체를 넣거나 설치 서버에서 시스템을 부트합니다. 그런 다음 부트 가능 ZFS 루트 파일 시스템을 만들 대화식 텍스트 설치 방법을 선택합니다.
   - SPARC: Oracle Solaris 설치 DVD에 다음 구문을 사용합니다.

     ```
     ok boot cdrom - text
     ```
   - SPARC: 네트워크에서 부트할 때 다음 구문을 사용합니다.

     ```
     ok boot net - text
     ```
   - x86: 텍스트 모드 설치 방법을 선택합니다.

   다음 방법으로 설치할 ZFS Flash 아카이브를 만들 수도 있습니다.
   - JumpStart 설치. 자세한 내용은 예 4–2를 참조하십시오.
   - 초기 설치. 자세한 내용은 예 4–3을 참조하십시오.

   기존의 부트 가능 ZFS 파일 시스템을 업그레이드할 때는 표준 업그레이드를 수행할 수 있지만 부트 가능 ZFS 파일 시스템을 새로 만들 때는 이 옵션을 사용할 수 없습니다. Solaris 10 10/08 릴리스부터는 Solaris 10 10/08 이상 릴리스가 이미 설치된 경우 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션할 수 있습니다. ZFS 루트 파일 시스템으로 마이그레이션하는 방법은 122 페이지 “ZFS 루트 파일 시스템으로 마이그레이션 또는 ZFS 루트 파일 시스템 업데이트(Live Upgrade)”를 참조하십시오.
2. ZFS 루트 파일 시스템을 만들려면 ZFS 옵션을 선택합니다. 예를 들면 다음과 같습니다.

   ```
   Choose Filesystem Type

     Select the filesystem to use for your Solaris installation


                 [ ] UFS
                 [X] ZFS
   ```
3. 설치할 소프트웨어를 설치한 후 ZFS 저장소 풀을 만들려면 디스크를 선택하라는 메시지가 표시됩니다. 이 화면은 이전 릴리스와 유사합니다.

예 4–1 부트 가능 ZFS 루트 파일 시스템 초기 설치 *(계속)*

```
Select Disks
  On this screen you must select the disks for installing Solaris software.
  Start by looking at the Suggested Minimum field; this value is the
  approximate space needed to install the software you've selected. For ZFS,
  multiple disks will be configured as mirrors, so the disk you choose, or the
  slice within the disk must exceed the Suggested Minimum value.
  NOTE: ** denotes current boot disk

  Disk Device                                                Available Space
  ============================================================================
  [X] ** c1t0d0                                            139989 MB  (F4 to edit
) [ ]    c1t1d0                                            139989 MB
  [ ]    c1t2d0                                            139989 MB
  [ ]    c1t3d0                                            139989 MB
  [ ]    c2t0d0                                            139989 MB
  [ ]    c2t1d0                                            139989 MB
  [ ]    c2t2d0                                            139989 MB
  [ ]    c2t3d0                                            139989 MB
                                  Maximum Root Size: 139989 MB
                                  Suggested Minimum:  11102 MB
```

ZFS 루트 풀에 사용할 디스크를 하나 이상 선택할 수 있습니다. 디스크를 두 개 선택하는 경우 미러된 2-디스크 구성이 루트 풀에 대해 설정됩니다. 2-디스크 또는 3-디스크로 미러링된 풀이 최적입니다. 디스크가 여덟 개 있으며 여덟 개를 모두 선택하는 경우 이러한 여덟 개의 디스크가 하나의 큰 미러로 루트 풀에 사용됩니다. 이 구성은 최적이 아닙니다. 초기 설치 완료 후 미러링된 루트 풀을 만들 수도 있습니다. 루트 풀에 대한 RAID-Z 풀 구성은 지원되지 않습니다.

ZFS 저장소 풀 구성에 대한 자세한 내용은 47 페이지 "ZFS 저장소 풀의 복제 기능"을 참조하십시오.

4. 미러링된 루트 풀을 만들기 위해 두 개의 디스크를 선택하려면 커서 제어 키를 사용하여 두번째 디스크를 선택합니다.

   다음 예에서는 `c1t0d0`과 `c1t1d0`이 루트 풀 디스크에 대해 선택됩니다. 두 디스크의 레이블은 SMI, 슬라이스는 0이어야 합니다. 디스크의 레이블이 SMI로 지정되지 않거나 디스크에 슬라이스가 포함되지 않은 경우 설치 프로그램을 종료하고 `format` 유틸리티를 사용하여 레이블을 재지정한 후 설치 프로그램을 다시 시작해야 합니다.

```
Select Disks
  On this screen you must select the disks for installing Solaris software.
  Start by looking at the Suggested Minimum field; this value is the
  approximate space needed to install the software you've selected. For ZFS,
  multiple disks will be configured as mirrors, so the disk you choose, or the
  slice within the disk must exceed the Suggested Minimum value.
  NOTE: ** denotes current boot disk

  Disk Device                                                Available Space
  ============================================================================
  [X] ** c1t0d0                                            139989 MB  (F4 to edit
) [X]    c1t1d0                                            139989 MB
  [ ]    c1t2d0                                            139989 MB
  [ ]    c1t3d0                                            139989 MB
```

예 4–1 부트 가능 ZFS 루트 파일 시스템 초기 설치 *(계속)*

```
[ ]    c2t0d0                                          139989 MB
[ ]    c2t1d0                                          139989 MB
[ ]    c2t2d0                                          139989 MB
[ ]    c2t3d0                                          139989 MB

                                  Maximum Root Size: 139989 MB
                                  Suggested Minimum:  11102 MB
```

Available Space(사용 가능한 공간) 열에 0MB가 표시되면 디스크의 레이블이 EFI인 것일 수 있습니다. 레이블이 EFI인 디스크를 사용하려면 설치 프로그램을 종료하고 `format -e` 명령을 사용하여 디스크의 레이블을 SMI로 재지정한 후 설치 프로그램을 다시 시작해야 합니다.

설치 중 미러링된 루트 풀을 만들지 않는 경우에도 설치 후 간편하게 만들 수 있습니다. 자세한 내용은 113 페이지 “미러링된 ZFS 루트 풀을 만드는 방법(사후 설치)”을 참조하십시오.

ZFS 저장소 풀에 대한 디스크를 하나 이상 선택한 후에는 다음과 유사한 화면이 표시됩니다.

```
Configure ZFS Settings
  Specify the name of the pool to be created from the disk(s) you have chosen.
  Also specify the name of the dataset to be created within the pool that is
  to be used as the root directory for the filesystem.


              ZFS Pool Name: rpool
      ZFS Root Dataset Name: s10nameBE
      ZFS Pool Size (in MB): 139990
  Size of Swap Area (in MB): 4096
  Size of Dump Area (in MB): 1024
        (Pool size must be between 7006 MB and 139990 MB)

                          [X] Keep / and /var combined
                          [ ] Put /var on a separate dataset
```

5. 이 화면에서 커서 제어 키로 항목을 이동하면서 기본값을 새 값으로 바꾸는 방식으로 필요에 따라 ZFS 풀 이름, 데이터 세트 이름, 풀 크기, 스왑 및 덤프 장치 크기를 변경할 수 있습니다. 또는 기본값을 그대로 적용할 수도 있습니다. 또한 `/var` 파일 시스템을 만들고 마운트하는 방식을 수정할 수 있습니다.

   이 예에서는 루트 데이터 세트 이름이 `zfsBE`로 변경됩니다.

```
              ZFS Pool Name: rpool
      ZFS Root Dataset Name: zfsBE
      ZFS Pool Size (in MB): 139990
  Size of Swap Area (in MB): 4096
  Size of Dump Area (in MB): 1024
        (Pool size must be between 7006 MB and 139990 MB)
```

6. 이 마지막 설치 화면에서 필요에 따라 설치 프로파일을 변경할 수 있습니다. 예를 들면 다음과 같습니다.

예 4–1 부트 가능 ZFS 루트 파일 시스템 초기 설치 *(계속)*

```
Profile

  The information shown below is your profile for installing Solaris software.
  It reflects the choices you've made on previous screens.

  ============================================================================

              Installation Option: Initial
                        Boot Device: c1t0d0
                  Root File System Type: ZFS
                       Client Services: None

                              Regions: North America
                         System Locale: C ( C )

                             Software: Solaris 10, Entire Distribution
                            Pool Name: rpool
                  Boot Environment Name: zfsBE
                            Pool Size: 139990 MB
                       Devices in Pool: c1t0d0
                                        c1t1d0
```

7. 설치가 완료되면 결과로 반환된 ZFS 저장소 풀 및 파일 시스템 정보를 검토합니다. 예를 들면 다음과 같습니다.

```
# zpool status
  pool: rpool
 state: ONLINE
 scrub: none requested
config:

        NAME          STATE     READ WRITE CKSUM
        rpool         ONLINE       0     0     0
          mirror-0    ONLINE       0     0     0
            c1t0d0s0  ONLINE       0     0     0
            c1t1d0s0  ONLINE       0     0     0

errors: No known data errors
# zfs list
NAME                 USED  AVAIL  REFER  MOUNTPOINT
rpool               10.1G   124G   106K  /rpool
rpool/ROOT          5.01G   124G    31K  legacy
rpool/ROOT/zfsBE    5.01G   124G  5.01G  /
rpool/dump          1.00G   124G  1.00G  -
rpool/export          63K   124G    32K  /export
rpool/export/home     31K   124G    31K  /export/home
rpool/swap          4.13G   124G  4.00G  -
```

샘플 `zfs list` 출력 결과에서는 기본적으로 액세스할 수 없는 루트 풀 구성 요소(예: `rpool/ROOT` 디렉토리)를 식별합니다.

8. 동일한 저장소 풀에 다른 ZFS BE(부트 환경)를 만들려면 `lucreate` 명령을 사용합니다.

   다음 예에서는 `zfs2BE`라는 이름의 새 BE가 만들어집니다. 현재 BE의 이름은 `zfs list` 출력과 같이 `zfsBE`입니다. 하지만 현재 BE는 새 BE가 만들어지기 전까지 `lustatus` 출력 결과에서 확인되지 않습니다.

예 4–1 부트 가능 ZFS 루트 파일 시스템 초기 설치 *(계속)*

```
# lustatus
ERROR: No boot environments are configured on this system
ERROR: cannot determine list of all boot environment names
```

동일한 풀에 새 ZFS BE를 만드는 경우 다음과 유사한 구문을 사용합니다.

```
# lucreate -n zfs2BE
INFORMATION: The current boot environment is not named - assigning name <zfsBE>.
Current boot environment is named <zfsBE>.
Creating initial configuration for primary boot environment <zfsBE>.
The device </dev/dsk/c1t0d0s0> is not a root device for any boot environment; cannot get BE ID.
PBE configuration successful: PBE name <zfsBE> PBE Boot Device </dev/dsk/c1t0d0s0>.
Comparing source boot environment <zfsBE> file systems with the file
system(s) you specified for the new boot environment. Determining which
file systems should be in the new boot environment.
Updating boot environment description database on all BEs.
Updating system configuration files.
Creating configuration for boot environment <zfs2BE>.
Source boot environment is <zfsBE>.
Creating boot environment <zfs2BE>.
Cloning file systems from boot environment <zfsBE> to create boot environment <zfs2BE>.
Creating snapshot for <rpool/ROOT/zfsBE> on <rpool/ROOT/zfsBE@zfs2BE>.
Creating clone for <rpool/ROOT/zfsBE@zfs2BE> on <rpool/ROOT/zfs2BE>.
Setting canmount=noauto for </> in zone <global> on <rpool/ROOT/zfs2BE>.
Population of boot environment <zfs2BE> successful.
Creation of boot environment <zfs2BE> successful.
```

동일한 풀에 ZFS BE를 만들 때는 ZFS 복제 및 스냅샷 기능을 통해 BE가 즉시 만들어집니다. ZFS 루트 마이그레이션에 Live Upgrade를 사용하는 방법은 122 페이지 “ZFS 루트 파일 시스템으로 마이그레이션 또는 ZFS 루트 파일 시스템 업데이트(Live Upgrade)”를 참조하십시오.

9. 다음으로 새 부트 환경을 확인합니다. 예를 들면 다음과 같습니다.

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
zfsBE                      yes      yes    yes       no     -
zfs2BE                     yes      no     no        yes    -
# zfs list
NAME                      USED  AVAIL  REFER  MOUNTPOINT
rpool                    10.1G   124G   106K  /rpool
rpool/ROOT               5.00G   124G    31K  legacy
rpool/ROOT/zfs2BE         218K   124G  5.00G  /
rpool/ROOT/zfsBE         5.00G   124G  5.00G  /
rpool/ROOT/zfsBE@zfs2BE   104K      -  5.00G  -
rpool/dump               1.00G   124G  1.00G  -
rpool/export               63K   124G    32K  /export
rpool/export/home          31K   124G    31K  /export/home
rpool/swap               4.13G   124G  4.00G  -
```

10. 대체 BE에서 부트하려면 `luactivate` 명령을 사용합니다.
    - SPARC - 부트 장치에 ZFS 저장소 풀이 포함된 경우 `boot -L` 명령을 사용하여 사용 가능한 BE를 식별합니다.

예 4–1 부트 가능 ZFS 루트 파일 시스템 초기 설치 *(계속)*

에를 들어, SPARC 기반 시스템에서는 `boot -L` 명령으로 사용 가능한 BE 목록을 표시합니다. 새 BE(`zfs2BE`)에서 부트하려면 옵션 2를 선택합니다. 그런 다음 표시된 `boot -Z` 명령을 입력합니다.

```
ok boot -L
Executing last command: boot -L
Boot device: /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0  File and args: -L
1 zfsBE
2 zfs2BE
Select environment to boot: [ 1 - 2 ]: 2

To boot the selected entry, invoke:
boot [<root-device>] -Z rpool/ROOT/zfs2BE
ok boot -Z rpool/ROOT/zfs2BE
```

- x86 – GRUB 메뉴에서 부트할 BE를 식별합니다.

ZFS 파일 시스템 부트에 대한 자세한 내용은 150 페이지 “ZFS 루트 파일 시스템에서 부트”를 참조하십시오.

▼ 미러링된 ZFS 루트 풀을 만드는 방법(사후 설치)

설치 중 미러링된 ZFS 루트 풀을 만들지 않은 경우에도 설치 후 간편하게 만들 수 있습니다.

루트 풀의 디스크 교체 방법은 157 페이지 “ZFS 루트 풀의 디스크 교체 방법”을 참조하십시오.

1 현재 루트 풀 상태를 표시합니다.

```
# zpool status rpool
  pool: rpool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        rpool       ONLINE       0     0     0
          c1t0d0s0  ONLINE       0     0     0

errors: No known data errors
```

2 두번째 디스크를 연결하여 미러링된 루트 풀을 구성합니다.

```
# zpool attach rpool c1t0d0s0 c1t1d0s0
Make sure to wait until resilver is done before rebooting.
```

3 루트 풀 상태를 통해 리실버링이 완료되었는지 확인합니다.

```
# zpool status rpool
  pool: rpool
 state: ONLINE
```

```
status: One or more devices is currently being resilvered.  The pool will
        continue to function, possibly in a degraded state.
action: Wait for the resilver to complete.
 scrub: resilver in progress for 0h1m, 24.26% done, 0h3m to go
config:

        NAME          STATE     READ WRITE CKSUM
        rpool         ONLINE       0     0     0
          mirror-0    ONLINE       0     0     0
            c1t0d0s0  ONLINE       0     0     0
            c1t1d0s0  ONLINE       0     0     0  3.18G resilvered

errors: No known data errors
```

앞의 출력 결과는 리실버링 프로세스가 완료되지 않은 상태를 보여 줍니다. 다음과 유사한 메시지가 표시되면 리실버링이 완료된 것입니다.

```
resilvered 10.0G in 0h10m with 0 errors on Thu Nov 15 12:48:33 2012
```

4 **두번째 디스크에서 성공적으로 부트할 수 있는지 확인합니다.**

5 **필요한 경우 새 디스크에서 자동으로 부트하도록 시스템을 설정합니다.**

- SPARC - SPARC 부트 PROM에서 `eeprom` 명령 또는 `setenv` 명령을 사용하여 기본 부트 장치를 재설정합니다.
- x86 - 시스템 BIOS를 재구성합니다.

ZFS 루트 파일 시스템 설치(Oracle Solaris Flash 아카이브 설치)

Solaris 10 10/09 릴리스부터는 UFS 루트 파일 시스템 또는 ZFS 루트 파일 시스템이 있는 시스템에서 Flash 아카이브를 만들 수 있습니다. ZFS 루트 풀의 Flash 아카이브에는 스왑 볼륨, 덤프 볼륨 및 제외된 모든 데이터 세트를 제외한 전체 풀 계층이 포함됩니다. 스왑 및 덤프 볼륨은 Flash 아카이브가 설치될 때 만들어집니다. 다음과 같이 Flash 아카이브 설치 방법을 사용할 수 있습니다.

- ZFS 루트 파일 시스템이 있는 시스템을 설치 및 부트하는 데 사용할 수 있는 Flash 아카이브를 만듭니다.
- ZFS Flash 아카이브를 사용하여 복제 시스템의 JumpStart 설치 또는 초기 설치를 수행합니다. ZFS Flash 아카이브를 만들면 개별 부트 환경이 아닌 전체 루트 풀이 복제됩니다. `flarcreate` 및 `flar` 명령에 `-D` 옵션을 사용하여 풀의 데이터 세트를 개별적으로 제외시킬 수 있습니다.

ZFS Flash 아카이브를 사용하여 시스템을 설치하기 전에 다음 제한 사항을 검토하십시오.

- Oracle Solaris 10 8/11 릴리스부터는 대화식 설치의 Flash 아카이브 옵션을 사용하여 ZFS 루트 파일 시스템이 있는 시스템을 설치할 수 있습니다. 또한 `luupgrade` 명령을 사용하여 Flash 아카이브를 통해 대체 ZFS BE를 업데이트할 수 있습니다.

  - Solaris 10 9/10 릴리스를 실행 중인 시스템은 124630-51(SPARC) 패치 또는 124631-51(x86) 패치를 추가하여 ABE에 대한 플래시 아카이브를 설치해야 합니다.
  - 플래시 아카이브 및 복제 시스템을 만드는 중이고 플래시 아카이브를 설치 중인 마스터 시스템은 커널 패치 레벨과 동일해야 합니다. 예를 들어 Solaris 10 8/11 릴리스를 실행 중인 시스템에 ZFS 플래시 아카이브를 만드는 경우 복제 시스템도 동일한 Solaris 10 8/11 커널 패치 레벨을 실행 중인지 확인하십시오. 그렇지 않으면 플래시 아카이브 설치가 `zfs receive` 명령의 오류와 함께 실패할 수 있습니다.
  - 플래시 아카이브를 만들고 ABE에 적용하려면 종속 루트 파일 시스템(예: 별도 `/var` 파일 시스템)으로 Solaris 10 9/10 릴리스를 실행 중인 마스터 시스템을 Solaris 10 8/11 릴리스로 업그레이드해야 합니다. 그렇지 않으면 플래시 아카이브 설치가 실패합니다.
- ZFS Flash 아카이브를 만든 시스템과 구조가 동일한 시스템에만 Flash 아카이브를 설치할 수 있습니다. 예를 들어, `sun4v` 시스템에서 만들어진 아카이브는 `sun4u` 시스템에 설치할 수 없습니다.
- ZFS Flash 아카이브의 전체 초기 설치만 지원됩니다. ZFS 루트 파일 시스템의 차등 Flash 아카이브를 설치하거나 UFS/ZFS 혼성 아카이브를 설치할 수 없습니다.
- Solaris 10 8/11 릴리스부터는 UFS Flash 아카이브를 사용하여 ZFS 루트 파일 시스템을 설치할 수 있습니다. 예를 들면 다음과 같습니다.
  - JumpStart 프로파일에 `pool` 키워드를 사용하면 UFS Flash 아카이브가 ZFS 루트 풀에 설치됩니다.

    ```
    pool rpool auto auto auto mirror c0t0d0s0 c0t1d0s0
    ```
  - UFS Flash 아카이브의 대화식 설치 중 ZFS를 파일 시스템 유형으로 선택합니다.
- 명시적으로 제외된 데이터 세트를 제외한 전체 루트 풀이 아카이브되고 설치되지만 Flash 아카이브가 설치된 후에는 아카이브를 만들 때 부트된 ZFS BE만 사용할 수 있습니다. 하지만 `flarcreate` 또는 `flar` 명령의 `-R` *rootdir* 옵션을 사용하여 아카이브된 풀을 사용하면 현재 부트된 루트 풀 외에 다른 루트 풀을 아카이브할 수 있습니다.
- ZFS Flash 아카이브에서는 개별 파일 포함 및 제외에 사용되는 `flarcreate` 및 `flar` 명령 옵션이 지원되지 않습니다. ZFS Flash 아카이브에서는 전체 데이터 세트만 제외시킬 수 있습니다.
- ZFS Flash 아카이브에 대해서는 `flar info` 명령이 지원되지 않습니다. 예를 들면 다음과 같습니다.

  ```
  # flar info -l zfs10upflar
  ERROR: archive content listing not supported for zfs archives.
  ```

마스터 시스템이 설치되거나 Solaris 10 10/09 이상 릴리스로 업그레이드된 후에는 대상 시스템 설치에 사용할 ZFS Flash 아카이브를 만들 수 있습니다. 기본적인 프로세스는 다음과 같습니다.

- 마스터 시스템에서 `flarcreate` 명령을 사용하여 ZFS Flash 아카이브를 만듭니다. 스왑 및 덤프 볼륨을 제외한 루트 풀의 모든 데이터 세트가 ZFS Flash 아카이브에 포함됩니다.

- JumpStart 프로파일을 만들어 설치 서버의 Flash 아카이브 정보를 포함시킵니다.
- 대상 시스템에 ZFS Flash 아카이브를 설치합니다.

Flash 아카이브를 사용하여 ZFS 루트 풀을 설치할 때는 다음과 같은 아카이브 옵션이 지원됩니다.

- `flarcreate` 또는 `flar` 명령을 사용하여 지정한 ZFS 루트 풀의 Flash 아카이브를 만들 수 있습니다. 지정하지 않을 경우 기본 루트 풀의 Flash 아카이브가 만들어집니다.
- `flarcreate -D` *dataset*를 사용하여 Flash 아카이브에서 지정한 데이터 세트를 제외시킬 수 있습니다. 이 옵션은 여러 데이터 세트를 제외시킬 때 여러 번 사용할 수 있습니다.

ZFS Flash 아카이브가 설치되면 시스템이 다음과 같이 구성됩니다.

- 아카이브를 만들 때 특별히 제외된 데이터 세트를 제외하고 Flash 아카이브가 만들어진 시스템에 존재하는 전체 데이터 세트 계층이 대상 시스템에 다시 만들어집니다. 스왑 및 덤프 볼륨은 Flash 아카이브에 포함되지 않습니다.
- 루트 풀의 이름은 아카이브를 만들 때 사용된 풀의 이름과 동일합니다.
- Flash 아카이브를 만들 때 활성 상태였던 BE가 배포된 시스템에서 활성 기본 BE로 설정됩니다.

예 4–2 ZFS Flash 아카이브를 사용하여 시스템 설치(JumpStart 설치)

마스터 시스템이 설치되거나 Solaris 10 10/09 이상 릴리스로 업그레이드되면 ZFS 루트 풀의 Flash 아카이브를 만듭니다. 예를 들면 다음과 같습니다.

```
# flarcreate -n zfsBE zfs10upflar
Full Flash
Checking integrity...
Integrity OK.
Running precreation scripts...
Precreation scripts done.
Determining the size of the archive...
The archive will be approximately 6.77GB.
Creating the archive...
Archive creation complete.
Running postcreation scripts...
Postcreation scripts done.

Running pre-exit scripts...
Pre-exit scripts done.
```

그런 다음 설치 서버로 사용할 시스템에서 시스템을 설치할 JumpStart 프로파일을 만듭니다. 예를 들어, `zfs10upflar` 아카이브 설치에는 다음 프로파일이 사용됩니다.

```
install_type flash_install
archive_location nfs system:/export/jump/zfs10upflar
partitioning explicit
pool rpool auto auto auto mirror c0t1d0s0 c0t0d0s0
```

예 4–3 부트 가능 ZFS 루트 파일 시스템 초기 설치(Flash 아카이브 설치)

Flash 설치 옵션을 선택하여 ZFS 루트 파일 시스템을 설치할 수 있습니다. 이 옵션은 ZFS Flash 아카이브가 이미 만들어져서 사용 가능한 상태라고 가정합니다.

1. Solaris Interactive Installation(Solaris 대화식 설치) 화면에서 F4_Flash 옵션을 선택합니다.
2. Reboot After Installation(설치 후 재부트) 화면에서 Auto Reboot or Manual Reboot(자동 재부트 또는 수동 재부트) 옵션을 선택합니다.
3. Choose Filesystem Type(파일 시스템 유형 선택) 화면에서 ZFS를 선택합니다.
4. Flash Archive Retrieval Method(Flash 아카이브 검색 방법) 화면에서 HTTP, FTP, NFS, Local File(로컬 파일), Local Tape(로컬 테이프), Local Device(로컬 장치) 등의 검색 방법을 선택합니다.

   예를 들어, ZFS Flash 아카이브가 NFS 서버에서 공유되는 경우 NFS를 선택합니다.
5. Flash Archive Addition(Flash 아카이브 추가) 화면에서 ZFS Flash 아카이브 위치를 지정합니다.

   예를 들어, 위치가 NFS 서버일 경우 해당 IP 주소로 서버를 식별한 다음 ZFS Flash 아카이브 경로를 지정합니다.

   ```
   NFS Location: 12.34.567.890:/export/zfs10upflar
   ```
6. Flash Archive Selection(Flash 아카이브 선택) 화면에서 검색 방법 및 ZFS BE 이름을 확인합니다.

   ```
   Flash Archive Selection

     You selected the following Flash archives to use to install this system.  If
     you want to add another archive to install select "New".

           Retrieval Method      Name
           ========================================================================
           NFS                   zfsBE
   ```
7. 초기 설치와 유사한 일련의 다음 화면을 검토하고 구성에 맞는 옵션을 선택합니다.
   - 디스크 선택
   - 데이터를 보존하겠습니까?
   - ZFS 설정 구성

     요약 정보를 검토하고 Continue(계속) 옵션을 선택합니다.

     예를 들면 다음과 같습니다.

     ```
     Configure ZFS Settings
       Specify the name of the pool to be created from the disk(s) you have chosen.
       Also specify the name of the dataset to be created within the pool that is
       to be used as the root directory for the filesystem.


                 ZFS Pool Name: rpool
           ZFS Root Dataset Name: s10zfsBE
           ZFS Pool Size (in MB): 69995
     ```

예 4–3 부트 가능 ZFS 루트 파일 시스템 초기 설치(Flash 아카이브 설치) *(계속)*

```
  Size of Swap Area (in MB): 2048
  Size of Dump Area (in MB): 1024
        (Pool size must be between 7591 MB and 69995 MB)
```

Flash 아카이브가 ZFS 전송 스트림일 경우 결합된 또는 별도의 `/var` 파일 시스템 옵션이 표시되지 않습니다. 이 경우 `/var` 결합 여부는 마스터 시스템에서의 구성 방식에 따라 다릅니다.

- Mount Remote File Systems(원격 파일 시스템을 마운트하겠습니까)? 화면에서 Continue(계속)를 누릅니다.
- Profile(프로파일) 화면을 검토하고 변경 사항을 적용하려면 F4 키를 누릅니다. 변경 사항을 적용하지 않으려면 Begin_Installation(설치 시작)(F2)을 누릅니다.

  예를 들면 다음과 같습니다.

```
Profile

  The information shown below is your profile for installing Solaris software.
  It reflects the choices you've made on previous screens.

  ============================================================================

                Installation Option: Flash
                        Boot Device: c1t0d0
              Root File System Type: ZFS
                    Client Services: None

                           Software: 1 Flash Archive
                                       NFS: zfsBE
                          Pool Name: rpool
              Boot Environment Name: s10zfsBE
                          Pool Size: 69995 MB
                    Devices in Pool: c1t0d0
```

ZFS 루트 파일 시스템 설치(JumpStart 설치)

JumpStart 프로파일을 만들어 ZFS 루트 파일 시스템 또는 UFS 루트 파일 시스템을 설치할 수 있습니다.

ZFS 관련 JumpStart 프로파일에는 새 `pool` 키워드가 포함되어야 합니다. `pool` 키워드는 새 루트 풀을 설치하며 이 경우 기본적으로 새 BE(부트 환경)가 만들어집니다. BE의 이름을 제공하고 `bootenv installbe` 키워드와 `bename` 및 `dataset` 옵션을 포함하는 별도의 `/var` 데이터 세트를 만들 수 있습니다.

JumpStart 기능 사용에 대한 일반적인 정보는 **Oracle Solaris 10 1/13 설치 설명서: JumpStart 설치**를 참조하십시오.

JumpStart를 통한 ZFS 루트 파일 시스템 설치 후 영역을 구성하고 시스템에 패치를 적용하거나 시스템을 업그레이드하려면 131 페이지 “Live Upgrade를 사용하여 영역이

있는 시스템 마이그레이션 또는 업그레이드(Solaris 10 10/08)" 또는 136 페이지 "Oracle Solaris Live Upgrade를 사용하여 영역이 있는 시스템 마이그레이션 또는 업그레이드(Solaris 10 5/09 이상)"를 참조하십시오.

ZFS에 대한 JumpStart 키워드

ZFS 관련 JumpStart 프로파일에는 다음 키워드를 사용할 수 있습니다.

`auto` 풀, 스왑 볼륨 또는 덤프 볼륨에 대한 슬라이스의 크기를 자동으로 지정합니다. 디스크 크기 확인을 통해 최소 크기를 조정할 수 있는지가 확인됩니다. 최소 크기를 조정할 수 있을 경우 제약 조건(예: 디스크 크기, 보존된 슬라이스 등)에 따라 가능한 최대 풀 크기가 할당됩니다.

예를 들어, `all` 또는 `auto` 키워드를 지정할 경우 `c0t0d0s0`을 지정하면 가능한 최대 크기로 루트 풀 슬라이스가 만들어집니다. 슬라이스, 스왑 볼륨 또는 덤프 볼륨에 대한 특정 크기를 지정할 수도 있습니다.

`auto` 키워드는 ZFS 루트 풀과 함께 사용되는 경우 `all` 키워드와 유사하게 작동합니다. 풀에 사용되지 않는 디스크 공간이 없기 때문입니다.

`bootenv` 부트 환경 특성을 식별합니다.

다음과 같은 `bootenv` 키워드 구문을 사용하여 부트 가능 ZFS 루트 환경을 만들 수 있습니다.

`bootenv installbe bename` *BE-name* [*dataset mount-point*]

`installbe` `bename` 옵션 및 *BE-name* 항목으로 식별된 새 BE를 만들고 설치합니다.

`bename` *BE-name* 설치할 *BE-name*을 식별합니다.

`bename`이 `pool` 키워드와 함께 사용되지 않으면 기본 BE가 만들어집니다.

`dataset` *mount-point* 선택적 `dataset` 키워드를 사용하여 루트 데이터 세트와 별도로 `/var` 데이터 세트를 식별할 수 있습니다. 현재 *mount-point* 값은 `/var`로 제한됩니다. 예를 들어, 별도의 `/var` 데이터 세트에 대한 `bootenv` 구문 라인은 다음과 유사합니다.

```
bootenv installbe bename zfsroot dataset /var
```

`pool` 새로 만들 루트 풀을 정의합니다. 다음과 같은 키워드 구문을 제공해야 합니다.

`pool` *poolname poolsize swapsize dumpsize vdevlist*

poolname 만들 풀의 이름을 식별합니다. 풀은 지정된 풀 *poolsize* 및 하나 이상의 장치 *vdevlist*와 함께 지정된 물리적 장치로 만들어집니다. 기존 풀을 겹쳐 쓰므로 *poolname* 값은 기존 풀의 이름을 식별하지 않아야 합니다.

poolsize 만들 풀의 크기를 지정합니다. 값은 `auto` 또는 `existing`일 수 있습니다. `auto` 값은 제약 조건(예: 디스크 크기 등)에 따라 가능한 최대 풀 크기를 할당합니다. `g`(GB)로 지정되지 않는 한 크기는 MB로 간주됩니다.

swapsize 만들 스왑 볼륨의 크기를 지정합니다. `auto` 값을 지정하면 기본 스왑 크기가 사용됩니다. *size* 값과 함께 크기를 지정할 수 있습니다. `g`(GB)로 지정되지 않는 한 크기는 MB로 간주됩니다.

dumpsize 만들 덤프 볼륨의 크기를 지정합니다. `auto` 값을 지정하면 기본 덤프 크기가 사용됩니다. *size* 값과 함께 크기를 지정할 수 있습니다. `g`(GB)로 지정되지 않는 한 크기는 MB로 간주됩니다.

vdevlist 풀을 만드는 데 사용할 장치를 하나 이상 지정합니다. *vdevlist*의 형식은 `zpool create` 명령의 형식과 동일합니다. 이 때 여러 장치가 지정되어 있으면 미러된 구성만 지원됩니다. *vdevlist*의 장치는 루트 풀의 슬라이스여야 합니다. `any` 값을 지정하면 설치 소프트웨어가 적합한 장치를 선택합니다.

원하는 수만큼 디스크를 미러링할 수 있지만 만들어진 풀의 크기는 지정된 디스크의 가장 작은 크기로 결정됩니다. 미러링된 저장소 풀을 만드는 방법은 47 페이지 “미러링된 저장소 풀 구성”을 참조하십시오.

ZFS에 대한 JumpStart 프로파일의 예

이 절에서는 ZFS 특정 JumpStart 프로파일의 예를 제공합니다.

다음 프로파일은 `pool newpool`로 식별된 새 풀에서 `install_type initial_install`을 통해 지정된 초기 설치를 수행합니다. 여기서 크기는 `auto` 키워드를 통해 지정된 디스크의 크기로 자동 설정됩니다. 스왑 영역 및 덤프 장치의 크기는 디스크(`mirror` 키워드 및 `c0t0d0s0`과 `c0t1d0s0`으로 지정된 디스크)의 미러링된 구성에 있는 `auto` 키워드를 통해 자동으로 지정됩니다. `installbe` 키워드를 사용하여 새 BE를 설치하기 위해 `bootenv` 키워드를 통해 부트 환경 특성이 설정되며 `s10-xx`라는 이름의 BE가 만들어집니다.

```
install_type initial_install
pool newpool auto auto auto mirror c0t0d0s0 c0t1d0s0
bootenv installbe bename s10-xx
```

다음 프로파일은 크기가 80GB인 newpool이라는 새 풀에 있는 SUNWCall 메타 클러스터의 install_type initial_install 키워드를 사용하여 초기 설치를 수행합니다. 이 풀은 80GB 풀을 만들 수 있을 만큼 충분한 크기의 사용 가능한 두 장치에 대한 미러링된 구성에서 2GB의 스왑 볼륨과 2GB의 덤프 볼륨으로 만들어집니다. 두 장치를 사용할 수 없을 경우 설치가 실패합니다. installbe 키워드를 사용하여 새 BE를 설치하기 위해 bootenv 키워드를 통해 부트 환경 특성이 설정되며 *s10–xx*라는 이름의 bename이 만들어집니다.

```
install_type initial_install
cluster SUNWCall
pool newpool 80g 2g 2g mirror any any
bootenv installbe bename s10-xx
```

JumpStart 설치 구문을 사용하여 ZFS 루트 풀도 포함하는 디스크에 UFS 파일 시스템을 보존하거나 만들 수 있습니다. 운용 시스템에는 이 구성이 권장되지 않습니다. 하지만 소형 시스템(예: 랩탑)에서 전환 또는 마이그레이션이 필요한 경우 이 구성을 사용할 수 있습니다.

ZFS에 대한 JumpStart 문제

JumpStart를 통해 부트 가능 ZFS 루트 시스템 설치를 시작하기 전에 다음 문제를 고려하십시오.

- 부트 가능 ZFS 루트 파일 시스템을 만들 때 JumpStart 설치에는 기존 ZFS 저장소 풀을 사용할 수 없습니다. 다음과 유사한 구문을 사용하여 새 ZFS 저장소 풀을 만들어야 합니다.

  ```
  pool rpool 20G 4G 4G c0t0d0s0
  ```

- 105 페이지 “ZFS 지원을 위한 Oracle Solaris 설치 및 Live Upgrade 요구 사항”에 설명된 대로 전체 디스크가 아닌 디스크 슬라이스로 고유의 풀을 만들어야 합니다. 예를 들어, 다음 예에서 굵게 표시된 구문은 사용할 수 없습니다.

  ```
  install_type initial_install
  cluster SUNWCall
  pool rpool all auto auto mirror c0t0d0 c0t1d0
  bootenv installbe bename newBE
  ```

  다음 예에서 굵게 표시된 구문은 사용할 수 있습니다.

  ```
  install_type initial_install
  cluster SUNWCall
  pool rpool all auto auto mirror c0t0d0s0 c0t1d0s0
  bootenv installbe bename newBE
  ```

ZFS 루트 파일 시스템으로 마이그레이션 또는 ZFS 루트 파일 시스템 업데이트(Live Upgrade)

UFS 구성 요소와 관련된 Live Upgrade 기능은 계속 사용할 수 있지만 이전 릴리스에서와 동일한 방식으로 작동합니다.

다음 기능을 사용할 수 있습니다.

- **UFS BE를 ZFS BE로 마이그레이션**
  - UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션할 때는 `-p` 옵션을 사용하여 기존 ZFS 저장소 풀을 지정해야 합니다.
  - UFS 루트 파일 시스템의 다른 슬라이스에 구성 요소가 있을 경우 ZFS 루트 풀로 마이그레이션됩니다.
  - Oracle Solaris 10 8/11 릴리스부터는 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션할 때 별도의 `/var` 파일 시스템을 지정할 수 있습니다.
  - UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션하는 기본적인 프로세스는 다음과 같습니다.
    1. 필요한 경우 Live Upgrade 패치를 설치합니다.
    2. 현재 Oracle Solaris 10 릴리스(Solaris 10 10/08-Oracle Solaris 10 8/11)를 설치하거나 표준 업그레이드 프로그램을 사용하여 지원되는 SPARC 기반 또는 x86 기반 시스템에서 이전 Oracle Solaris 10 릴리스를 업그레이드합니다.
    3. Solaris 10 10/08 이상 릴리스를 실행 중인 경우 ZFS 루트 파일 시스템용 ZFS 저장소 풀을 만듭니다.
    4. Live Upgrade를 사용하여 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션합니다.
    5. `luactivate` 명령을 사용하여 ZFS BE를 활성화합니다.
- **ZFS BE 패치 적용 또는 업그레이드**
  - `luupgrade` 명령을 사용하여 기존 ZFS BE를 업그레이드하거나 패치를 적용할 수 있습니다. 또한 `luupgrade`를 사용하여 ZFS Flash 아카이브를 통해 대체 ZFS BE를 업그레이드할 수 있습니다. 자세한 내용은 예 4–8을 참조하십시오.
  - 동일한 풀에 새 ZFS BE를 만들 때 Live Upgrade는 ZFS 스냅샷 및 복제 기능을 사용할 수 있습니다. 따라서 이전 릴리스에 비해 BE가 빨리 만들어집니다.
- **영역 마이그레이션 지원** – 영역이 있는 시스템을 마이그레이션할 수 있지만 Solaris 10 10/08 릴리스에서는 지원되는 구성이 제한됩니다. Solaris 10 5/09 릴리스부터는 보다 많은 영역 구성이 지원됩니다. 자세한 내용은 다음 절을 참조하십시오.
  - 131 페이지 “Live Upgrade를 사용하여 영역이 있는 시스템 마이그레이션 또는 업그레이드(Solaris 10 10/08)”
  - 136 페이지 “Oracle Solaris Live Upgrade를 사용하여 영역이 있는 시스템 마이그레이션 또는 업그레이드(Solaris 10 5/09 이상)”

영역이 없는 시스템을 마이그레이션하는 경우 124 페이지 “Live Upgrade를 사용하여 영역이 없는 ZFS 루트 파일 시스템 마이그레이션 또는 업데이트”를 참조하십시오.

Oracle Solaris 설치 및 Live Upgrade 기능에 대한 자세한 내용은 **Oracle Solaris 10 1/13 설치 설명서: Live Upgrade 및 업그레이드 계획**을 참조하십시오.

ZFS 및 Live Upgrade 요구 사항에 대한 자세한 내용은 105 페이지 “ZFS 지원을 위한 Oracle Solaris 설치 및 Live Upgrade 요구 사항”을 참조하십시오.

Live Upgrade를 통한 ZFS 마이그레이션 문제

Live Upgrade를 사용하여 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션하기 전에 다음 문제를 검토하십시오.

- UFS 루트 파일 시스템에서 ZFS 루트 파일 시스템으로 마이그레이션할 때는 Oracle Solaris 설치 GUI의 표준 업그레이드 옵션을 사용할 수 없습니다. UFS 파일 시스템에서 마이그레이션하려면 Live Upgrade를 사용해야 합니다.
- Live Upgrade를 작동하기 전에 부트에 사용할 ZFS 저장소 풀을 만들어야 합니다. 현재 부트 제한 사항으로 인해 전체 디스크 대신 슬라이스로 ZFS 루트 풀을 만들어야 합니다. 예를 들면 다음과 같습니다.

  ```
  # zpool create rpool mirror c1t0d0s0 c1t1d0s0
  ```

  새 풀을 만들기 전에 풀에서 사용할 디스크의 레이블이 EFI가 아닌 SMI(VTOC)인지 확인하십시오. 디스크의 레이블이 SMI로 재지정되는 경우 레이블 지정 프로세스를 통해 분할 체계가 변경되지 않습니다. 대부분의 경우 전체 디스크 용량은 루트 풀에 사용되는 슬라이스에 할당되어야 합니다.
- Oracle Solaris Live Upgrade를 통해서는 ZFS BE에서 UFS BE를 만들 수 없습니다. UFS BE를 ZFS BE로 마이그레이션하고 UFS BE를 보존하는 경우 UFS BE 또는 ZFS BE에서 부트할 수 있습니다.
- Live Upgrade는 이름 변경을 감지할 수 없으므로 `zfs rename` 명령을 사용하여 ZFS BE의 이름을 바꾸지 마십시오. 후속 명령(예: `ludelete`)이 실패합니다. 실제로 계속 사용하려는 기존 BE가 있는 경우에는 ZFS 풀 또는 파일 시스템의 이름을 바꾸지 마십시오.
- 기본 BE의 복제본인 대체 BE를 만들 때는 `-f`, `-x`, `-y`, `-Y` 및 `-z` 옵션을 사용하여 기본 BE에서 파일을 포함시키거나 제외시킬 수 없습니다. 다음과 같은 경우 설정된 포함 및 제외 옵션을 계속 사용할 수 있습니다.

  ```
  UFS -> UFS
  UFS -> ZFS
  ZFS -> ZFS (different pool)
  ```
- Live Upgrade를 사용하여 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 업그레이드할 수 있기는 하지만 Live Upgrade를 사용하여 비루트 또는 공유 파일 시스템을 업그레이드할 수는 없습니다.

- lu 명령을 사용하여 ZFS 루트 파일 시스템을 만들거나 마이그레이션할 수 없습니다.
- 시스템의 스왑 및 덤프 장치를 비루트 풀에 두려면 148 페이지 "ZFS 스왑 및 덤프 볼륨 사용자 정의"를 참조하십시오.

Live Upgrade를 사용하여 영역이 없는 ZFS 루트 파일 시스템 마이그레이션 또는 업데이트

다음 예에서는 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션하는 방법과 ZFS 루트 파일 시스템을 업데이트하는 방법을 보여 줍니다.

영역이 있는 시스템을 마이그레이션하거나 업데이트하는 경우 다음 절을 참조하십시오.

- 131 페이지 "Live Upgrade를 사용하여 영역이 있는 시스템 마이그레이션 또는 업그레이드(Solaris 10 10/08)"
- 136 페이지 "Oracle Solaris Live Upgrade를 사용하여 영역이 있는 시스템 마이그레이션 또는 업그레이드(Solaris 10 5/09 이상)"

예 4–4 Live Upgrade를 사용하여 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션

다음 예에서는 UFS 루트 파일 시스템에서 ZFS 루트 파일 시스템을 마이그레이션하는 방법을 보여 줍니다. UFS 루트 파일 시스템을 포함하는 현재 BE인 ufsBE는 -c 옵션으로 식별됩니다. 선택적 -c 옵션을 포함시키지 않을 경우 기본적으로 장치 이름이 현재 BE 이름으로 설정됩니다. 새 BE인 zfsBE는 -n 옵션으로 식별됩니다. lucreate 작업이 수행되기 전에 ZFS 저장소 풀이 존재해야 합니다.

업그레이드 및 부트가 가능하도록 하려면 전체 디스크가 아닌 슬라이스로 ZFS 저장소 풀을 만들어야 합니다. 새 풀을 만들기 전에 풀에서 사용할 디스크의 레이블이 EFI가 아닌 SMI(VTOC)인지 확인하십시오. 디스크의 레이블이 SMI로 재지정되는 경우 레이블 지정 프로세스를 통해 분할 체계가 변경되지 않습니다. 대부분의 경우 전체 디스크 용량은 루트 풀에 사용되는 슬라이스에 할당되어야 합니다.

```
# zpool create rpool mirror c1t2d0s0 c2t1d0s0
# lucreate -c ufsBE -n zfsBE -p rpool
Analyzing system configuration.
No name for current boot environment.
Current boot environment is named <ufsBE>.
Creating initial configuration for primary boot environment <ufsBE>.
The device </dev/dsk/c1t0d0s0> is not a root device for any boot environment; cannot get BE ID.
PBE configuration successful: PBE name <ufsBE> PBE Boot Device </dev/dsk/c1t0d0s0>.
Comparing source boot environment <ufsBE> file systems with the file
system(s) you specified for the new boot environment. Determining which
file systems should be in the new boot environment.
Updating boot environment description database on all BEs.
Updating system configuration files.
The device </dev/dsk/c1t2d0s0> is not a root device for any boot environment; cannot get BE ID.
Creating configuration for boot environment <zfsBE>.
Source boot environment is <ufsBE>.
```

예 4–4 Live Upgrade를 사용하여 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션 *(계속)*

```
Creating boot environment <zfsBE>.
Creating file systems on boot environment <zfsBE>.
Creating <zfs> file system for </> in zone <global> on <rpool/ROOT/zfsBE>.
Populating file systems on boot environment <zfsBE>.
Checking selection integrity.
Integrity check OK.
Populating contents of mount point </>.
Copying.
Creating shared file system mount points.
Creating compare databases for boot environment <zfsBE>.
Creating compare database for file system </rpool/ROOT>.
Creating compare database for file system </>.
Updating compare databases on boot environment <zfsBE>.
Making boot environment <zfsBE> bootable.
Creating boot_archive for /.alt.tmp.b-qD.mnt
updating /.alt.tmp.b-qD.mnt/platform/sun4u/boot_archive
Population of boot environment <zfsBE> successful.
Creation of boot environment <zfsBE> successful.
```

lucreate 작업이 완료된 후 lustatus 명령을 사용하여 BE 상태를 확인합니다. 예를 들면 다음과 같습니다.

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
ufsBE                      yes      yes    yes       no     -
zfsBE                      yes      no     no        yes    -
```

그런 다음 ZFS 구성 요소 목록을 검토합니다. 예를 들면 다음과 같습니다.

```
# zfs list
NAME                   USED  AVAIL  REFER  MOUNTPOINT
rpool                 7.17G  59.8G  95.5K  /rpool
rpool/ROOT            4.66G  59.8G    21K  /rpool/ROOT
rpool/ROOT/zfsBE      4.66G  59.8G  4.66G  /
rpool/dump               2G  61.8G    16K  -
rpool/swap             517M  60.3G    16K  -
```

다음으로 luactivate 명령을 사용하여 새 ZFS BE를 활성화합니다. 예를 들면 다음과 같습니다.

```
# luactivate zfsBE
A Live Upgrade Sync operation will be performed on startup of boot environment <zfsBE>.

**********************************************************************

The target boot environment has been activated. It will be used when you
reboot. NOTE: You MUST NOT USE the reboot, halt, or uadmin commands. You
MUST USE either the init or the shutdown command when you reboot. If you
do not use either init or shutdown, the system will not boot using the
target BE.
```

예 4–4 Live Upgrade를 사용하여 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션 *(계속)*

```
**********************************************************************
.
.
.
Modifying boot archive service
Activation of boot environment <zfsBE> successful.
```

다음으로 ZFS BE로 시스템을 재부트합니다.

```
# init 6
```

ZFS BE가 활성화되었는지 확인합니다.

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
ufsBE                      yes      no     no        yes    -
zfsBE                      yes      yes    yes       no     -
```

UFS BE로 다시 전환하는 경우 ZFS BE가 부트된 동안 만들어진 모든 ZFS 저장소 풀은 UFS BE에서 자동으로 사용 가능한 상태로 설정되지 않으므로 해당 ZFS 저장소 풀을 다시 가져와야 합니다.

UFS BE가 더 이상 필요하지 않을 경우 `ludelete` 명령을 사용하여 제거할 수 있습니다.

예 4–5 Live Upgrade를 사용하여 UFS BE에서 ZFS BE 만들기(별도의 `/var` 포함)

Oracle Solaris 10 8/11 릴리스에서는 `lucreate -D` 옵션을 사용하여 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션할 때 별도의 `/var` 파일 시스템이 만들어지도록 지정할 수 있습니다. 다음 예에서는 기존 UFS BE가 ZFS BE로 마이그레이션될 때 별도의 `/var` 파일 시스템이 만들어집니다.

```
# lucreate -n zfsBE -p rpool -D /var
Determining types of file systems supported
Validating file system requests
Preparing logical storage devices
Preparing physical storage devices
Configuring physical storage devices
Configuring logical storage devices
Analyzing system configuration.
No name for current boot environment.
INFORMATION: The current boot environment is not named - assigning name <c0t0d0s0>.
Current boot environment is named <c0t0d0s0>.
Creating initial configuration for primary boot environment <c0t0d0s0>.
INFORMATION: No BEs are configured on this system.
The device </dev/dsk/c0t0d0s0> is not a root device for any boot environment; cannot get BE ID.
PBE configuration successful: PBE name <c0t0d0s0> PBE Boot Device </dev/dsk/c0t0d0s0>.
Updating boot environment description database on all BEs.
```

예 4–5 Live Upgrade를 사용하여 UFS BE에서 ZFS BE 만들기(별도의 /var 포함) *(계속)*

```
Updating system configuration files.
The device </dev/dsk/c0t1d0s0> is not a root device for any boot environment; cannot get BE ID.
Creating configuration for boot environment <zfsBE>.
Source boot environment is <c0t0d0s0>.
Creating file systems on boot environment <zfsBE>.
Creating <zfs> file system for </> in zone <global> on <rpool/ROOT/zfsBE>.
Creating <zfs> file system for </var> in zone <global> on <rpool/ROOT/zfsBE/var>.
Populating file systems on boot environment <zfsBE>.
Analyzing zones.
Mounting ABE <zfsBE>.
Generating file list.
Copying data from PBE <c0t0d0s0> to ABE <zfsBE>
100% of filenames transferred
Finalizing ABE.
Fixing zonepaths in ABE.
Unmounting ABE <zfsBE>.
Fixing properties on ZFS datasets in ABE.
Reverting state of zones in PBE <c0t0d0s0>.
Making boot environment <zfsBE> bootable.
Creating boot_archive for /.alt.tmp.b-iaf.mnt
updating /.alt.tmp.b-iaf.mnt/platform/sun4u/boot_archive
Population of boot environment <zfsBE> successful.
Creation of boot environment <zfsBE> successful.
# luactivate zfsBE
A Live Upgrade Sync operation will be performed on startup of boot environment <zfsBE>.
.
.
.
Modifying boot archive service
Activation of boot environment <zfsBE> successful.
# init 6
```

새로 만들어진 ZFS 파일 시스템을 검토합니다. 예를 들면 다음과 같습니다.

```
# zfs list
NAME                   USED  AVAIL  REFER  MOUNTPOINT
rpool                 6.29G  26.9G  32.5K  /rpool
rpool/ROOT            4.76G  26.9G    31K  legacy
rpool/ROOT/zfsBE      4.76G  26.9G  4.67G  /
rpool/ROOT/zfsBE/var  89.5M  26.9G  89.5M  /var
rpool/dump             512M  26.9G   512M  -
rpool/swap            1.03G  28.0G    16K  -
```

예 4–6 Live Upgrade를 사용하여 ZFS BE에서 ZFS BE 만들기

동일한 풀에서 ZFS BE를 기반으로 ZFS BE를 만들 때는 ZFS 스냅샷 및 복제 기능이 사용되므로 이 작업은 매우 빨리 수행할 수 있습니다. 현재 BE가 동일한 ZFS 풀에 상주하는 경우 -p 옵션을 생략합니다.

ZFS BE가 여러 개 있을 경우 다음 작업을 수행하여 부트할 BE를 선택합니다.

- SPARC: boot -L 명령으로 사용 가능한 BE를 식별할 수 있습니다. 그런 다음 boot -Z 명령을 사용하여 부트할 BE를 선택합니다.

예 4–6 Live Upgrade를 사용하여 ZFS BE에서 ZFS BE 만들기 *(계속)*

- x86: GRUB 메뉴에서 BE를 선택할 수 있습니다.

자세한 내용은 예 4–12를 참조하십시오.

```
# lucreate -n zfs2BE
Analyzing system configuration.
No name for current boot environment.
INFORMATION: The current boot environment is not named - assigning name <zfsBE>.
Current boot environment is named <zfsBE>.
Creating initial configuration for primary boot environment <zfsBE>.
The device </dev/dsk/c1t0d0s0> is not a root device for any boot environment; cannot get BE ID.
PBE configuration successful: PBE name <zfsBE> PBE Boot Device </dev/dsk/c1t0d0s0>.
Comparing source boot environment <zfsBE> file systems with the file
system(s) you specified for the new boot environment. Determining which
file systems should be in the new boot environment.
Updating boot environment description database on all BEs.
Updating system configuration files.
Creating configuration for boot environment <zfs2BE>.
Source boot environment is <zfsBE>.
Creating boot environment <zfs2BE>.
Cloning file systems from boot environment <zfsBE> to create boot environment <zfs2BE>.
Creating snapshot for <rpool/ROOT/zfsBE> on <rpool/ROOT/zfsBE@zfs2BE>.
Creating clone for <rpool/ROOT/zfsBE@zfs2BE> on <rpool/ROOT/zfs2BE>.
Setting canmount=noauto for </> in zone <global> on <rpool/ROOT/zfs2BE>.
Population of boot environment <zfs2BE> successful.
Creation of boot environment <zfs2BE> successful.
```

예 4–7 ZFS BE 업데이트(luupgrade)

추가 패키지 또는 패치를 사용하여 ZFS BE를 업데이트할 수 있습니다.

기본적인 프로세스는 다음과 같습니다.

- `lucreate` 명령을 사용하여 대체 BE를 만듭니다.
- 대체 BE를 활성화하고 이 BE에서 부트합니다.
- `luupgrade` 명령으로 기본 ZFS BE를 업데이트하여 패키지 또는 패치를 추가합니다.

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
zfsBE                      yes      no     no        yes    -
zfs2BE                     yes      yes    yes       no     -
# luupgrade -p -n zfsBE -s /net/system/export/s10up/Solaris_10/Product SUNWchxge
Validating the contents of the media </net/install/export/s10up/Solaris_10/Product>.
Mounting the BE <zfsBE>.
Adding packages to the BE <zfsBE>.

Processing package instance <SUNWchxge> from </net/install/export/s10up/Solaris_10/Product>

Chelsio N110 10GE NIC Driver(sparc) 11.10.0,REV=2006.02.15.20.41
Copyright (c) 2010, Oracle and/or its affiliates. All rights reserved.
```

예 4–7 ZFS BE 업데이트(luupgrade) *(계속)*

```
This appears to be an attempt to install the same architecture and
version of a package which is already installed.  This installation
will attempt to overwrite this package.

Using </a> as the package base directory.
## Processing package information.
## Processing system information.
   4 package pathnames are already properly installed.
## Verifying package dependencies.
## Verifying disk space requirements.
## Checking for conflicts with packages already installed.
## Checking for setuid/setgid programs.

This package contains scripts which will be executed with super-user
permission during the process of installing this package.

Do you want to continue with the installation of <SUNWchxge> [y,n,?] y
Installing Chelsio N110 10GE NIC Driver as <SUNWchxge>

## Installing part 1 of 1.
## Executing postinstall script.

Installation of <SUNWchxge> was successful.
Unmounting the BE <zfsBE>.
The package add to the BE <zfsBE> completed.
```

또는 새 BE를 만들어 최신 Oracle Solaris 릴리스로 업데이트할 수도 있습니다. 예를 들면 다음과 같습니다.

```
# luupgrade -u -n newBE -s /net/install/export/s10up/latest
```

여기서 `-s` 옵션은 Solaris 설치 매체의 위치를 지정합니다.

예 4–8 ZFS Flash 아카이브를 사용하여 ZFS BE 만들기(luupgrade)

Oracle Solaris 10 8/11 릴리스에서는 `luupgrade` 명령을 사용하여 기존 ZFS Flash 아카이브에서 ZFS BE를 만들 수 있습니다. 기본적인 프로세스는 다음과 같습니다.

1. ZFS BE를 사용하여 마스터 시스템의 Flash 아카이브를 만듭니다.

   예를 들면 다음과 같습니다.

   ```
   master-system# flarcreate -n s10zfsBE /tank/data/s10zfsflar
   Full Flash
   Checking integrity...
   Integrity OK.
   Running precreation scripts...
   Precreation scripts done.
   Determining the size of the archive...
   The archive will be approximately 4.67GB.
   Creating the archive...
   Archive creation complete.
   Running postcreation scripts...
   ```

예 4–8 ZFS Flash 아카이브를 사용하여 ZFS BE 만들기(luupgrade) *(계속)*

```
Postcreation scripts done.

Running pre-exit scripts...
Pre-exit scripts done.
```

2. 마스터 시스템에서 만들어진 ZFS Flash 아카이브를 복제 시스템에서 사용 가능한 상태로 설정합니다.

   가능한 Flash 아카이브 위치는 로컬 파일 시스템, HTTP, FTP, NFS 등입니다.

3. 복제 시스템에서 빈 대체 ZFS BE를 만듭니다.

   `-s -` 옵션을 사용하여 이 BE를 ZFS Flash 아카이브 컨텐츠로 채울 빈 BE로 지정합니다.

   예를 들면 다음과 같습니다.

```
clone-system# lucreate -n zfsflashBE -s - -p rpool
Determining types of file systems supported
Validating file system requests
Preparing logical storage devices
Preparing physical storage devices
Configuring physical storage devices
Configuring logical storage devices
Analyzing system configuration.
No name for current boot environment.
INFORMATION: The current boot environment is not named - assigning name <s10zfsBE>.
Current boot environment is named <s10zfsBE>.
Creating initial configuration for primary boot environment <s10zfsBE>.
INFORMATION: No BEs are configured on this system.
The device </dev/dsk/c0t0d0s0> is not a root device for any boot environment; cannot get BE ID.
PBE configuration successful: PBE name <s10zfsBE> PBE Boot Device </dev/dsk/c0t0d0s0>.
Updating boot environment description database on all BEs.
Updating system configuration files.
The device </dev/dsk/c0t1d0s0> is not a root device for any boot environment; cannot get BE ID.
Creating <zfs> file system for </> in zone <global> on <rpool/ROOT/zfsflashBE>.
Creation of boot environment <zfsflashBE> successful.
```

4. 대체 BE에 ZFS Flash 아카이브를 설치합니다.

   예를 들면 다음과 같습니다.

```
clone-system# luupgrade -f -s /net/server/export/s10/latest -n zfsflashBE -a /tank/data/zfs10up2flar
miniroot filesystem is <lofs>
Mounting miniroot at </net/server/s10up/latest/Solaris_10/Tools/Boot>
Validating the contents of the media </net/server/export/s10up/latest>.
The media is a standard Solaris media.
Validating the contents of the miniroot </net/server/export/s10up/latest/Solaris_10/Tools/Boot>.
Locating the flash install program.
Checking for existence of previously scheduled Live Upgrade requests.
Constructing flash profile to use.
Creating flash profile for BE <zfsflashBE>.
Performing the operating system flash install of the BE <zfsflashBE>.
CAUTION: Interrupting this process may leave the boot environment unstable or unbootable.
Extracting Flash Archive: 100% completed (of 5020.86 megabytes)
The operating system flash install completed.
updating /.alt.tmp.b-rgb.mnt/platform/sun4u/boot_archive
```

예 4–8 ZFS Flash 아카이브를 사용하여 ZFS BE 만들기(luupgrade) *(계속)*

```
The Live Flash Install of the boot environment <zfsflashBE> is complete.
```

5. 대체 BE를 활성화합니다.

```
clone-system# luactivate zfsflashBE
A Live Upgrade Sync operation will be performed on startup of boot environment <zfsflashBE>.
.
.
.
Modifying boot archive service
Activation of boot environment <zfsflashBE> successful.
```

6. 시스템을 재부트합니다.

```
clone-system# init 6
```

Live Upgrade를 사용하여 영역이 있는 시스템 마이그레이션 또는 업그레이드(Solaris 10 10/08)

Live Upgrade를 사용하여 영역이 있는 시스템을 마이그레이션할 수 있지만 Solaris 10 10/08 릴리스에서는 지원되는 구성이 제한됩니다. Solaris 10 5/09 이상 릴리스를 설치하거나 업그레이드하는 경우 보다 많은 영역 구성이 지원됩니다. 자세한 내용은 136 페이지 “Oracle Solaris Live Upgrade를 사용하여 영역이 있는 시스템 마이그레이션 또는 업그레이드(Solaris 10 5/09 이상)”를 참조하십시오.

이 절에서는 Live Upgrade를 사용하여 업그레이드하고 패치를 적용할 수 있도록 영역이 있는 시스템을 설치하고 구성하는 방법에 대해 설명합니다. 영역이 없는 ZFS 루트 파일 시스템으로 마이그레이션하는 경우 124 페이지 “Live Upgrade를 사용하여 영역이 없는 ZFS 루트 파일 시스템 마이그레이션 또는 업데이트”를 참조하십시오.

Solaris 10 10/08 릴리스에서 영역이 있는 시스템을 마이그레이션하거나 구성하는 경우 다음 절차를 검토하십시오.

- 132 페이지 “UFS에 영역 루트가 있는 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션하는 방법(Solaris 10 10/08)”
- 133 페이지 “ZFS에 영역 루트가 있는 ZFS 루트 파일 시스템을 구성하는 방법(Solaris 10 10/08)”
- 135 페이지 “ZFS에 영역 루트가 있는 ZFS 루트 파일 시스템을 업그레이드하거나 패치를 적용하는 방법(Solaris 10 10/08)”
- 154 페이지 “부트가 성공하지 못하도록 하는 ZFS 마운트 지점 문제 해결(Solaris 10 10/08)”

시스템에서 Live Upgrade를 사용할 수 있으려면 ZFS 루트 파일 시스템이 있는 시스템에서 영역을 설정하는 다음 권장 절차를 따르십시오.

▼ UFS에 영역 루트가 있는 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션하는 방법(Solaris 10 10/08)

이 절차에서는 영역이 설치된 UFS 루트 파일 시스템을 업그레이드 또는 패치 적용이 가능한 ZFS 루트 파일 시스템과 ZFS 영역 루트 구성으로 마이그레이션하는 방법에 대해 설명합니다.

다음에 나오는 단계의 예에서 풀 이름은 `rpool`이며 활성 BE(부트 환경) 이름은 `s10BE*`로 시작합니다.

1 시스템이 이전 Solaris 10 릴리스를 실행 중인 경우 시스템을 Solaris 10 10/08 릴리스로 업그레이드합니다.

Solaris 10 릴리스를 실행 중인 시스템 업그레이드에 대한 자세한 내용은 **Oracle Solaris 10 1/13 설치 설명서: Live Upgrade 및 업그레이드 계획**을 참조하십시오.

2 루트 풀을 만듭니다.

```
# zpool create rpool mirror c0t1d0 c1t1d0
```

루트 풀 요구 사항에 대한 자세한 내용은 105 페이지 "ZFS 지원을 위한 Oracle Solaris 설치 및 Live Upgrade 요구 사항"을 참조하십시오.

3 UFS 환경의 영역이 부트되었는지 확인합니다.

4 새 ZFS 부트 환경을 만듭니다.

```
# lucreate -n s10BE2 -p rpool
```

이 명령은 새 BE용 루트 풀에서 데이터 세트를 설정하며 현재 BE(영역 포함)를 해당 데이터 세트에 복사합니다.

5 새 ZFS 부트 환경을 활성화합니다.

```
# luactivate s10BE2
```

그러면 시스템이 ZFS 루트 파일 시스템을 실행하지만 UFS의 영역 루트는 UFS 루트 파일 시스템에 남아 있습니다. UFS 영역을 지원되는 ZFS 구성으로 완전히 마이그레이션하려면 다음 단계를 수행해야 합니다.

6 시스템을 재부트합니다.

```
# init 6
```

7 영역을 ZFS BE로 마이그레이션합니다.

a. 영역을 부트합니다.

b. 풀에 다른 ZFS BE를 만듭니다.

```
# lucreate s10BE3
```

c. 새 부트 환경을 활성화합니다.

```
# luactivate s10BE3
```

d. 시스템을 재부트합니다.

```
# init 6
```

이 단계에서는 ZFS BE 및 영역이 부트되었는지 확인합니다.

8 잠재적인 마운트 지점 문제를 해결합니다.

Live Upgrade의 버그로 인해 BE 내 ZFS 데이터 세트 또는 영역의 ZFS 데이터 세트에 잘못된 마운트 지점이 있을 수 있으므로 비활성 BE의 부트가 실패할 수 있습니다.

a. `zfs list` 출력 결과를 검토합니다.

잘못된 임시 마운트 지점을 검색합니다. 예를 들면 다음과 같습니다.

```
# zfs list -r -o name,mountpoint rpool/ROOT/s10up

NAME                                MOUNTPOINT
rpool/ROOT/s10up                    /.alt.tmp.b-VP.mnt/
rpool/ROOT/s10up/zones              /.alt.tmp.b-VP.mnt//zones
rpool/ROOT/s10up/zones/zonerootA    /.alt.tmp.b-VP.mnt/zones/zonerootA
```

루트 ZFS BE 마운트 지점(`rpool/ROOT/s10up`)은 `/`여야 합니다.

b. ZFS BE 및 해당 데이터 세트에 대한 마운트 지점을 재설정합니다.

예를 들면 다음과 같습니다.

```
# zfs inherit -r mountpoint rpool/ROOT/s10up
# zfs set mountpoint=/ rpool/ROOT/s10up
```

c. 시스템을 재부트합니다.

특정 BE를 부트할 수 있는 옵션이 표시되면 OpenBoot PROM 프롬프트 또는 GRUB 메뉴에서 마운트 지점이 수정된 BE를 선택합니다.

▼ ZFS에 영역 루트가 있는 ZFS 루트 파일 시스템을 구성하는 방법(Solaris 10 10/08)

이 절차에서는 업그레이드 또는 패치 적용이 가능한 ZFS 루트 파일 시스템 및 ZFS 영역 루트 구성을 설정하는 방법에 대해 설명합니다. 이 구성에서는 ZFS 영역 루트가 ZFS 데이터 세트로 만들어집니다.

다음에 나오는 단계의 예에서 풀 이름은 `rpool`이며 활성 부트 환경 이름은 `s10BE`입니다. 유효한 데이터 세트 이름을 영역 데이터 세트 이름으로 사용할 수 있습니다. 다음 예에서 영역 데이터 세트 이름은 `zones`입니다.

1 **대화식 텍스트 설치 프로그램 또는 JumpStart 설치 방법을 사용하여 ZFS 루트가 있는 시스템을 설치합니다.**

선택하는 설치 방법에 따라 107 페이지 "ZFS 루트 파일 시스템 설치(Oracle Solaris 초기 설치)" 또는 118 페이지 "ZFS 루트 파일 시스템 설치(JumpStart 설치)"를 참조하십시오.

2 **새로 만들어진 루트 풀에서 시스템을 부트합니다.**

3 **영역 루트를 그룹화할 데이터 세트를 만듭니다.**

예를 들면 다음과 같습니다.

```
# zfs create -o canmount=noauto rpool/ROOT/s10BE/zones
```

canmount 등록 정보에 대해 noauto 값을 설정하면 Live Upgrade의 명시적 작업 및 시스템 시작 코드 이외의 다른 방법으로는 데이터 세트가 마운트되지 않습니다.

4 **새로 만들어진 영역 데이터 세트를 마운트합니다.**

```
# zfs mount rpool/ROOT/s10BE/zones
```

데이터 세트가 /zones에 마운트됩니다.

5 **각 영역 루트에 대한 데이터 세트를 만들고 마운트합니다.**

```
# zfs create -o canmount=noauto rpool/ROOT/s10BE/zones/zonerootA
# zfs mount rpool/ROOT/s10BE/zones/zonerootA
```

6 **영역 루트 디렉토리에 대해 적합한 권한을 설정합니다.**

```
# chmod 700 /zones/zonerootA
```

7 **다음과 같이 영역 경로를 설정하여 영역을 구성합니다.**

```
# zonecfg -z zoneA
    zoneA: No such zone configured
    Use 'create' to begin configuring a new zone.
    zonecfg:zoneA> create
    zonecfg:zoneA> set zonepath=/zones/zonerootA
```

다음 구문을 사용하여 시스템을 부트할 때 영역이 자동으로 부트되도록 설정할 수 있습니다.

```
zonecfg:zoneA> set autoboot=true
```

8 **영역을 설치합니다.**

```
# zoneadm -z zoneA install
```

9 **영역을 부트합니다.**

```
# zoneadm -z zoneA boot
```

▼ ZFS에 영역 루트가 있는 ZFS 루트 파일 시스템을 업그레이드하거나 패치를 적용하는 방법(Solaris 10 10/08)

ZFS에 영역 루트가 있는 ZFS 루트 파일 시스템을 업그레이드하거나 패치를 적용해야 할 경우 이 절차를 사용하십시오. 해당 업데이트는 시스템 업그레이드 또는 패치 적용으로 구성될 수 있습니다.

다음에 나오는 단계의 예에서 newBE는 업그레이드되거나 패치가 적용되는 BE의 이름입니다.

1 업그레이드하거나 패치를 적용할 BE를 만듭니다.

```
# lucreate -n newBE
```

모든 영역을 비롯하여 기존 BE가 복제됩니다. 원래 BE의 각 데이터 세트에 대한 데이터 세트가 만들어집니다. 새 데이터 세트는 현재 루트 풀과 동일한 풀에 만들어집니다.

2 다음 중 하나를 선택하여 시스템을 업그레이드하거나 새 BE에 패치를 적용합니다.

- 시스템을 업그레이드합니다.

  ```
  # luupgrade -u -n newBE -s /net/install/export/s10up/latest
  ```

  여기서 -s 옵션은 Oracle Solaris 설치 매체의 위치를 지정합니다.

- 새 BE에 패치를 적용합니다.

  ```
  # luupgrade -t -n newBE -t -s /patchdir 139147-02 157347-14
  ```

3 새 BE를 활성화합니다.

```
# luactivate newBE
```

4 새로 활성화된 BE에서 부트합니다.

```
# init 6
```

5 잠재적인 마운트 지점 문제를 해결합니다.

Live Upgrade의 버그로 인해 BE 내 ZFS 데이터 세트 또는 영역의 ZFS 데이터 세트에 잘못된 마운트 지점이 있을 수 있으므로 비활성 BE의 부트가 실패할 수 있습니다.

a. zfs list 출력 결과를 검토합니다.

잘못된 임시 마운트 지점을 검색합니다. 예를 들면 다음과 같습니다.

```
# zfs list -r -o name,mountpoint rpool/ROOT/newBE

NAME                                MOUNTPOINT
rpool/ROOT/newBE                    /.alt.tmp.b-VP.mnt/
rpool/ROOT/newBE/zones              /.alt.tmp.b-VP.mnt/zones
rpool/ROOT/newBE/zones/zonerootA    /.alt.tmp.b-VP.mnt/zones/zonerootA
```

루트 ZFS BE 마운트 지점(rpool/ROOT/newBE)은 /여야 합니다.

b. ZFS BE 및 해당 데이터 세트에 대한 마운트 지점을 재설정합니다.

예를 들면 다음과 같습니다.

```
# zfs inherit -r mountpoint rpool/ROOT/newBE
# zfs set mountpoint=/ rpool/ROOT/newBE
```

c. 시스템을 재부트합니다.

특정 부트 환경을 부트할 수 있는 옵션이 표시되면 OpenBoot PROM 프롬프트 또는 GRUB 메뉴에서 마운트 지점이 수정된 부트 환경을 선택합니다.

Oracle Solaris Live Upgrade를 사용하여 영역이 있는 시스템 마이그레이션 또는 업그레이드(Solaris 10 5/09 이상)

Solaris 10 10/08 릴리스부터는 Oracle Solaris Live Upgrade 기능을 사용하여 영역이 있는 시스템을 마이그레이션하거나 업그레이드할 수 있습니다. Solaris 10 5/09 릴리스부터는 Live Upgrade가 추가 스파스(루트 및 전체) 영역 구성을 지원합니다.

이 절에서는 Solaris 10 5/09 릴리스부터 Live Upgrade를 사용하여 업그레이드 및 패치 적용이 가능하도록 영역이 있는 시스템을 구성하는 방법에 대해 설명합니다. 영역이 없는 ZFS 루트 파일 시스템으로 마이그레이션하는 경우 124 페이지 “Live Upgrade를 사용하여 영역이 없는 ZFS 루트 파일 시스템 마이그레이션 또는 업데이트”를 참조하십시오.

Solaris 10 5/09 이상 릴리스부터 ZFS 및 영역과 함께 Oracle Solaris Live Upgrade를 사용할 때는 다음 사항을 고려하십시오.

- Solaris 10 5/09 릴리스부터 지원되는 영역 구성과 함께 Live Upgrade를 사용하려면 먼저 표준 업그레이드 프로그램을 사용하여 시스템을 Solaris 10 5/09 이상 릴리스로 업그레이드해야 합니다.
- 그런 다음 Live Upgrade를 사용하여 영역 루트가 있는 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션하거나, ZFS 루트 파일 시스템 및 영역 루트를 업그레이드하거나 패치를 적용할 수 있습니다.
- 이전 Solaris 10 릴리스의 지원되지 않는 영역 구성은 Solaris 10 5/09 이상 릴리스로 바로 마이그레이션할 수 없습니다.

Solaris 10 5/09 릴리스부터 지원되는 영역이 있는 시스템을 마이그레이션하거나 구성하는 경우 다음 정보를 검토하십시오.

- 137 페이지 “지원되는 ZFS 영역 루트 구성 정보(Solaris 10 5/09 이상)”
- 138 페이지 “ZFS 루트 파일 시스템 및 영역 루트가 있는 ZFS BE를 만드는 방법(Solaris 10 5/09 이상)”

- 140 페이지 "영역 루트가 있는 ZFS 루트 파일 시스템을 업그레이드하거나 패치를 적용하는 방법(Solaris 10 5/09 이상)"
- 143 페이지 "영역 루트가 있는 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션하는 방법(Solaris 10 5/09 이상)"

지원되는 ZFS 영역 루트 구성 정보(Solaris 10 5/09 이상)

Oracle Solaris Live Upgrade를 사용하여 영역이 있는 시스템을 마이그레이션하거나 업그레이드하기 전에 지원되는 영역 구성을 검토하십시오.

- **UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션** – 다음과 같은 영역 루트 구성이 지원됩니다.
  - UFS 루트 파일 시스템 내 디렉토리
  - UFS 루트 파일 시스템 내 마운트 지점의 하위 디렉토리
  - UFS 루트 파일 시스템 디렉토리 또는 UFS 루트 파일 시스템 마운트 지점의 하위 디렉토리 내 영역 루트가 있는 UFS 루트 파일 시스템 및 영역 루트가 있는 ZFS 비루트 풀

  영역 루트를 마운트 지점으로 사용하는 UFS 루트 파일 시스템은 지원되지 않습니다.
- **ZFS 루트 파일 시스템 마이그레이션 또는 업그레이드** – 다음과 같은 영역 루트 구성이 지원됩니다.
  - ZFS 루트 또는 비루트 풀의 파일 시스템. 예를 들어, `/zonepool/zones`를 사용할 수 있습니다. Live Upgrade 작업이 수행되기 전에 영역 루트에 대한 파일 시스템이 제공되지 않을 경우 Live Upgrade가 영역 루트(`zoneds`)에 대한 파일 시스템을 만드는 경우도 있습니다.
  - ZFS 파일 시스템의 하위 디렉토리 또는 종속 파일 시스템(다른 영역 경로가 중첩되지 않은 경우). 예를 들어, `/zonepool/zones/zone1` 및 `/zonepool/zones/zone1_dir`을 사용할 수 있습니다.

    다음 예에서는 `zonepool/zones`가 영역 루트를 포함하는 파일 시스템이며 `rpool`가 ZFS BE를 포함합니다.

    ```
    zonepool
    zonepool/zones
    zonepool/zones/myzone
    rpool
    rpool/ROOT
    rpool/ROOT/myBE
    ```

    다음 구문을 사용하면 Live Upgrade가 스냅샷을 만들고 `zonepool` 및 `rpool` BE에 영역을 복제합니다.

    ```
    # lucreate -n newBE
    ```

    `rpool/ROOT/newBE`에 `newBE` BE가 만들어집니다. 활성화할 경우 `newBE`가 `zonepool` 구성 요소에 대한 액세스 권한을 제공합니다.

앞선 예에서 /zonepool/zones가 하위 디렉토리라고 가정하고 별도의 파일 시스템이 없을 경우 Live Upgrade는 이 영역을 루트 풀 rpool의 구성 요소로 마이그레이션합니다.

- **다음과 같은 ZFS 및 영역 구성은 지원되지 않습니다.**
  - 영역 경로가 최상위 풀 파일 시스템의 마운트 지점으로 설정된 비전역 영역이 소스 BE에 있을 경우 Live upgrade를 사용하여 대체 BE를 만들 수 없습니다. 예를 들어, zonepool 풀에 /zonepool로 마운트된 파일 시스템이 있을 경우 영역 경로가 /zonepool로 설정된 비전역 영역을 사용할 수 없습니다.
  - 전역 영역의 /etc/vfstab 파일에 비전역 영역에 대한 파일 시스템 항목을 추가하지 마십시오. 대신 zonecfg 의 add fs 기능을 사용하여 비전역 영역에 파일 시스템을 추가하십시오.
- **UFS 및 ZFS에 대한 영역과 관련된 영역 마이그레이션 또는 업그레이드 정보** – UFS 및 ZFS 환경의 마이그레이션 또는 업그레이드에 영향을 끼칠 수 있는 다음 고려 사항을 검토하십시오.
  - 131 페이지 “Live Upgrade를 사용하여 영역이 있는 시스템 마이그레이션 또는 업그레이드(Solaris 10 10/08)”에 설명된 대로 Solaris 10 10/08 릴리스에서 영역을 구성하고 Solaris 10 5/09 이상으로 업그레이드한 경우 ZFS 루트 파일 시스템으로 마이그레이션하거나 Live Upgrade를 사용하여 Solaris 10 5/09 이상 릴리스로 업그레이드할 수 있습니다.
  - 중첩된 디렉토리(예: zones/zone1 및 zones/zone1/zone2)에 영역 루트를 만들지 마십시오. 만들 경우 부트 시 마운트가 실패할 수 있습니다.

▼ ZFS 루트 파일 시스템 및 영역 루트가 있는 ZFS BE를 만드는 방법(Solaris 10 5/09 이상)

Solaris 10 5/09 이상 릴리스 초기 설치를 수행한 후 ZFS 루트 파일 시스템을 만들려면 이 절차를 사용하십시오. luupgrade 명령을 사용하여 ZFS 루트 파일 시스템을 Solaris 10 5/09 이상 릴리스로 업그레이드한 후에도 이 절차를 사용할 수 있습니다. 그런 다음 이 절차를 통해 만들어진 ZFS BE를 업그레이드하거나 패치를 적용할 수 있습니다.

다음에 나오는 단계의 예에서 Oracle Solaris 10 9/10 시스템의 /rpool/zones에는 ZFS 루트 파일 시스템과 영역 루트 데이터 세트가 있습니다. zfs2BE라는 이름의 ZFS BE가 만들어지며 이후 이 BE를 업그레이드하거나 패치를 적용할 수 있습니다.

1 기존 ZFS 파일 시스템을 검토합니다.

```
# zfs list
NAME                   USED  AVAIL  REFER  MOUNTPOINT
rpool                 7.26G  59.7G    98K  /rpool
rpool/ROOT            4.64G  59.7G    21K  legacy
rpool/ROOT/zfsBE      4.64G  59.7G  4.64G  /
rpool/dump            1.00G  59.7G  1.00G  -
rpool/export            44K  59.7G    23K  /export
rpool/export/home       21K  59.7G    21K  /export/home
```

```
rpool/swap               1G  60.7G    16K  -
rpool/zones            633M  59.7G   633M  /rpool/zones
```

2 영역이 설치되어 있고 부트되었는지 확인합니다.

```
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   2 zfszone          running    /rpool/zones                   native   shared
```

3 ZFS BE를 만듭니다.

```
# lucreate -n zfs2BE
Analyzing system configuration.
No name for current boot environment.
INFORMATION: The current boot environment is not named - assigning name <zfsBE>.
Current boot environment is named <zfsBE>.
Creating initial configuration for primary boot environment <zfsBE>.
The device </dev/dsk/c1t0d0s0> is not a root device for any boot environment; cannot get BE ID.
PBE configuration successful: PBE name <zfsBE> PBE Boot Device </dev/dsk/c1t0d0s0>.
Comparing source boot environment <zfsBE> file systems with the file
system(s) you specified for the new boot environment. Determining which
file systems should be in the new boot environment.
Updating boot environment description database on all BEs.
Updating system configuration files.
Creating configuration for boot environment <zfs2BE>.
Source boot environment is <zfsBE>.
Creating boot environment <zfs2BE>.
Cloning file systems from boot environment <zfsBE> to create boot environment <zfs2BE>.
Creating snapshot for <rpool/ROOT/zfsBE> on <rpool/ROOT/zfsBE@zfs2BE>.
Creating clone for <rpool/ROOT/zfsBE@zfs2BE> on <rpool/ROOT/zfs2BE>.
Setting canmount=noauto for </> in zone <global> on <rpool/ROOT/zfs2BE>.
Population of boot environment <zfs2BE> successful.
Creation of boot environment <zfs2BE> successful.
```

4 ZFS BE를 활성화합니다.

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
zfsBE                      yes      yes    yes       no     -
zfs2BE                     yes      no     no        yes    -
# luactivate zfs2BE
A Live Upgrade Sync operation will be performed on startup of boot environment <zfs2BE>.
.
.
.
```

5 ZFS BE를 부트합니다.

```
# init 6
```

6 ZFS 파일 시스템 및 영역이 새 BE에 만들어졌는지 확인합니다.

```
# zfs list
NAME                              USED  AVAIL  REFER  MOUNTPOINT
rpool                            7.38G  59.6G    98K  /rpool
```

```
rpool/ROOT                          4.72G  59.6G    21K  legacy
rpool/ROOT/zfs2BE                   4.72G  59.6G  4.64G  /
rpool/ROOT/zfs2BE@zfs2BE            74.0M      -  4.64G  -
rpool/ROOT/zfsBE                    5.45M  59.6G  4.64G  /.alt.zfsBE
rpool/dump                          1.00G  59.6G  1.00G  -
rpool/export                          44K  59.6G    23K  /export
rpool/export/home                     21K  59.6G    21K  /export/home
rpool/swap                             1G  60.6G    16K  -
rpool/zones                         17.2M  59.6G   633M  /rpool/zones
rpool/zones-zfsBE                    653M  59.6G   633M  /rpool/zones-zfsBE
rpool/zones-zfsBE@zfs2BE            19.9M      -   633M  -
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   - zfszone          installed  /rpool/zones                   native   shared
```

▼ 영역 루트가 있는 ZFS 루트 파일 시스템을 업그레이드하거나 패치를 적용하는 방법(Solaris 10 5/09 이상)

Solaris 10 5/09 이상 릴리스에서 영역 루트가 있는 ZFS 루트 파일 시스템을 업그레이드하거나 패치를 적용해야 할 경우 이 절차를 사용하십시오. 해당 업데이트는 시스템 업그레이드 또는 패치 적용으로 구성될 수 있습니다.

다음에 나오는 단계의 예에서 `zfs2BE`는 업그레이드되거나 패치가 적용되는 BE의 이름입니다.

1 기존 ZFS 파일 시스템을 검토합니다.

```
# zfs list
NAME                                 USED  AVAIL  REFER  MOUNTPOINT
rpool                               7.38G  59.6G   100K  /rpool
rpool/ROOT                          4.72G  59.6G    21K  legacy
rpool/ROOT/zfs2BE                   4.72G  59.6G  4.64G  /
rpool/ROOT/zfs2BE@zfs2BE            75.0M      -  4.64G  -
rpool/ROOT/zfsBE                    5.46M  59.6G  4.64G  /
rpool/dump                          1.00G  59.6G  1.00G  -
rpool/export                          44K  59.6G    23K  /export
rpool/export/home                     21K  59.6G    21K  /export/home
rpool/swap                             1G  60.6G    16K  -
rpool/zones                         22.9M  59.6G   637M  /rpool/zones
rpool/zones-zfsBE                    653M  59.6G   633M  /rpool/zones-zfsBE
rpool/zones-zfsBE@zfs2BE            20.0M      -   633M  -
```

2 영역이 설치되어 있고 부트되었는지 확인합니다.

```
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   5 zfszone          running    /rpool/zones                   native   shared
```

3 업그레이드하거나 패치를 적용할 ZFS BE를 만듭니다.

```
# lucreate -n zfs2BE
Analyzing system configuration.
Comparing source boot environment <zfsBE> file systems with the file
```

```
system(s) you specified for the new boot environment. Determining which
file systems should be in the new boot environment.
Updating boot environment description database on all BEs.
Updating system configuration files.
Creating configuration for boot environment <zfs2BE>.
Source boot environment is <zfsBE>.
Creating boot environment <zfs2BE>.
Cloning file systems from boot environment <zfsBE> to create boot environment <zfs2BE>.
Creating snapshot for <rpool/ROOT/zfsBE> on <rpool/ROOT/zfsBE@zfs2BE>.
Creating clone for <rpool/ROOT/zfsBE@zfs2BE> on <rpool/ROOT/zfs2BE>.
Setting canmount=noauto for </> in zone <global> on <rpool/ROOT/zfs2BE>.
Creating snapshot for <rpool/zones> on <rpool/zones@zfs10092BE>.
Creating clone for <rpool/zones@zfs2BE> on <rpool/zones-zfs2BE>.
Population of boot environment <zfs2BE> successful.
Creation of boot environment <zfs2BE> successful.
```

4 다음 중 하나를 선택하여 시스템을 업그레이드하거나 새 BE에 패치를 적용합니다.

- 시스템을 업그레이드합니다.

  ```
  # luupgrade -u -n zfs2BE -s /net/install/export/s10up/latest
  ```

  여기서 -s 옵션은 Oracle Solaris 설치 매체의 위치를 지정합니다.

  이 프로세스는 매우 오래 걸릴 수 있습니다.

  luupgrade 프로세스의 전체 예는 예 4–9를 참조하십시오.

- 새 BE에 패치를 적용합니다.

  ```
  # luupgrade -t -n zfs2BE -t -s /patchdir patch-id-02 patch-id-04
  ```

5 새 부트 환경을 활성화합니다.

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
zfsBE                      yes      yes    yes       no     -
zfs2BE                     yes      no     no        yes    -
# luactivate zfs2BE
A Live Upgrade Sync operation will be performed on startup of boot environment <zfs2BE>.
.
.
.
```

6 새로 활성화된 부트 환경에서 부트합니다.

```
# init 6
```

예 4–9 영역 루트가 있는 ZFS 루트 파일 시스템을 Oracle Solaris 10 9/10 ZFS 루트 파일 시스템으로 업그레이드

이 예에서는 비루트 풀에 ZFS 루트 파일 시스템 및 영역 루트가 있는 Solaris 10 10/09 시스템에서 만들어진 ZFS BE(zfsBE)가 Oracle Solaris 10 9/10 릴리스로 업그레이드됩니다.

이 프로세스는 오래 걸릴 수 있습니다. 그런 다음 업그레이드된 BE(zfs2BE)가 활성화됩니다. 업그레이드를 시도하기 전에 영역이 설치되어 있고 부트되었는지 확인하십시오.

이 예에서는 zonepool 풀, /zonepool/zones 데이터 세트 및 zfszone 영역이 다음과 같이 만들어집니다.

```
# zpool create zonepool mirror c2t1d0 c2t5d0
# zfs create zonepool/zones
# chmod 700 zonepool/zones
# zonecfg -z zfszone
zfszone: No such zone configured
Use 'create' to begin configuring a new zone.
zonecfg:zfszone> create
zonecfg:zfszone> set zonepath=/zonepool/zones
zonecfg:zfszone> verify
zonecfg:zfszone> exit
# zoneadm -z zfszone install
cannot create ZFS dataset zonepool/zones: dataset already exists
Preparing to install zone <zfszone>.
Creating list of files to copy from the global zone.
Copying <8960> files to the zone.
.
.
.

# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   2 zfszone          running    /zonepool/zones                native   shared

# lucreate -n zfsBE
.
.
.
# luupgrade -u -n zfsBE -s /net/install/export/s10up/latest
40410 blocks
miniroot filesystem is <lofs>
Mounting miniroot at </net/system/export/s10up/latest/Solaris_10/Tools/Boot>
Validating the contents of the media </net/system/export/s10up/latest>.
The media is a standard Solaris media.
The media contains an operating system upgrade image.
The media contains <Solaris> version <10>.
Constructing upgrade profile to use.
Locating the operating system upgrade program.
Checking for existence of previously scheduled Live Upgrade requests.
Creating upgrade profile for BE <zfsBE>.
Determining packages to install or upgrade for BE <zfsBE>.
Performing the operating system upgrade of the BE <zfsBE>.
CAUTION: Interrupting this process may leave the boot environment unstable
or unbootable.
Upgrading Solaris: 100% completed
Installation of the packages from this media is complete.
Updating package information on boot environment <zfsBE>.
Package information successfully updated on boot environment <zfsBE>.
Adding operating system patches to the BE <zfsBE>.
```

```
The operating system patch installation is complete.
INFORMATION: The file </var/sadm/system/logs/upgrade_log> on boot
environment <zfsBE> contains a log of the upgrade operation.
INFORMATION: The file </var/sadm/system/data/upgrade_cleanup> on boot
environment <zfsBE> contains a log of cleanup operations required.
INFORMATION: Review the files listed above. Remember that all of the files
are located on boot environment <zfsBE>. Before you activate boot
environment <zfsBE>, determine if any additional system maintenance is
required or if additional media of the software distribution must be
installed.
The Solaris upgrade of the boot environment <zfsBE> is complete.
Installing failsafe
Failsafe install is complete.
# luactivate zfs2BE
# init 6
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
zfsBE                      yes      no     no        yes    -
zfs2BE                     yes      yes    yes       no     -
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   - zfszone          installed  /zonepool/zones                native   shared
```

▼ 영역 루트가 있는 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션하는 방법(Solaris 10 5/09 이상)

UFS 루트 파일 시스템 및 영역 루트가 있는 시스템을 Solaris 10 5/09 이상 릴리스로 마이그레이션하려면 이 절차를 사용하십시오. 그런 다음 Live Upgrade를 사용하여 ZFS BE를 만드십시오.

다음에 나오는 단계의 예에서 UFS BE 이름은 `c1t1d0s0`, UFS 영역 루트는 `zonepool/zfszone`, ZFS 루트 BE는 `zfsBE`입니다.

1 시스템이 이전 Solaris 10 릴리스를 실행 중인 경우 시스템을 Solaris 10 5/09 이상 릴리스로 업그레이드합니다.

Solaris 10 릴리스를 실행 중인 시스템을 업그레이드하는 방법은 **Oracle Solaris 10 1/13 설치 설명서: Live Upgrade 및 업그레이드 계획**을 참조하십시오.

2 루트 풀을 만듭니다.

루트 풀 요구 사항에 대한 자세한 내용은 105 페이지 “ZFS 지원을 위한 Oracle Solaris 설치 및 Live Upgrade 요구 사항”을 참조하십시오.

3 UFS 환경의 영역이 부트되었는지 확인합니다.

```
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   2 zfszone          running    /zonepool/zones                native   shared
```

4 새 ZFS BE를 만듭니다.

```
# lucreate -c c1t1d0s0 -n zfsBE -p rpool
```

이 명령은 새 BE용 루트 풀에서 데이터 세트를 설정하며 현재 BE(영역 포함)를 해당 데이터 세트에 복사합니다.

5 새 ZFS BE를 활성화합니다.

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
c1t1d0s0                   yes      no     no        yes    -
zfsBE                      yes      yes    yes       no     -         #
luactivate zfsBE
A Live Upgrade Sync operation will be performed on startup of boot environment <zfsBE>.
.
.
.
```

6 시스템을 재부트합니다.

```
# init 6
```

7 ZFS 파일 시스템 및 영역이 새 BE에 만들어졌는지 확인합니다.

```
# zfs list
NAME                                 USED  AVAIL  REFER  MOUNTPOINT
rpool                               6.17G  60.8G    98K  /rpool
rpool/ROOT                          4.67G  60.8G    21K  /rpool/ROOT
rpool/ROOT/zfsBE                    4.67G  60.8G  4.67G  /
rpool/dump                          1.00G  60.8G  1.00G  -
rpool/swap                           517M  61.3G    16K  -
zonepool                             634M  7.62G    24K  /zonepool
zonepool/zones                       270K  7.62G   633M  /zonepool/zones
zonepool/zones-c1t1d0s0              634M  7.62G   633M  /zonepool/zones-c1t1d0s0
zonepool/zones-c1t1d0s0@zfsBE        262K      -   633M  -
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   - zfszone          installed  /zonepool/zones                native   shared
```

예 4–10 영역 루트가 있는 UFS 루트 파일 시스템을 ZFS 루트 파일 시스템으로 마이그레이션

이 예에서는 UFS 루트 파일 시스템 및 영역 루트(/uzone/ufszone)를 비롯하여 ZFS 비루트 풀(pool) 및 영역 루트(/pool/zfszone)가 있는 Oracle Solaris 10 9/10이 ZFS 루트 파일 시스템으로 마이그레이션됩니다. 마이그레이션을 시도하기 전에 ZFS 루트 풀이 만들어졌으며 영역이 설치되어 있고 부트되었는지 확인하십시오.

```
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
```

```
                    2 ufszone          running    /uzone/ufszone                 native   shared
                    3 zfszone          running    /pool/zones/zfszone            native   shared

# lucreate -c ufsBE -n zfsBE -p rpool
Analyzing system configuration.
No name for current boot environment.
Current boot environment is named <zfsBE>.
Creating initial configuration for primary boot environment <zfsBE>.
The device </dev/dsk/c1t0d0s0> is not a root device for any boot environment; cannot get BE ID.
PBE configuration successful: PBE name <ufsBE> PBE Boot Device </dev/dsk/c1t0d0s0>.
Comparing source boot environment <ufsBE> file systems with the file
system(s) you specified for the new boot environment. Determining which
file systems should be in the new boot environment.
Updating boot environment description database on all BEs.
Updating system configuration files.
The device </dev/dsk/c1t1d0s0> is not a root device for any boot environment; cannot get BE ID.
Creating configuration for boot environment <zfsBE>.
Source boot environment is <ufsBE>.
Creating boot environment <zfsBE>.
Creating file systems on boot environment <zfsBE>.
Creating <zfs> file system for </> in zone <global> on <rpool/ROOT/zfsBE>.
Populating file systems on boot environment <zfsBE>.
Checking selection integrity.
Integrity check OK.
Populating contents of mount point </>.
Copying.
Creating shared file system mount points.
Copying root of zone <ufszone> to </.alt.tmp.b-EYd.mnt/uzone/ufszone>.
Creating snapshot for <pool/zones/zfszone> on <pool/zones/zfszone@zfsBE>.
Creating clone for <pool/zones/zfszone@zfsBE> on <pool/zones/zfszone-zfsBE>.
Creating compare databases for boot environment <zfsBE>.
Creating compare database for file system </rpool/ROOT>.
Creating compare database for file system </>.
Updating compare databases on boot environment <zfsBE>.
Making boot environment <zfsBE> bootable.
Creating boot_archive for /.alt.tmp.b-DLd.mnt
updating /.alt.tmp.b-DLd.mnt/platform/sun4u/boot_archive
Population of boot environment <zfsBE> successful.
Creation of boot environment <zfsBE> successful.
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
ufsBE                      yes      yes    yes       no     -
zfsBE                      yes      no     no        yes    -
# luactivate zfsBE
.
.
.
# init 6
.
.
.
# zfs list
NAME                                USED  AVAIL  REFER  MOUNTPOINT
pool                                628M  66.3G    19K  /pool
pool/zones                          628M  66.3G    20K  /pool/zones
pool/zones/zfszone                 75.5K  66.3G   627M  /pool/zones/zfszone
```

```
pool/zones/zfszone-ufsBE                628M  66.3G   627M  /pool/zones/zfszone-ufsBE
pool/zones/zfszone-ufsBE@zfsBE           98K      -   627M  -
rpool                                   7.76G  59.2G    95K  /rpool
rpool/ROOT                              5.25G  59.2G    18K  /rpool/ROOT
rpool/ROOT/zfsBE                        5.25G  59.2G  5.25G  /
rpool/dump                              2.00G  59.2G  2.00G  -
rpool/swap                               517M  59.7G    16K  -
# zoneadm list -cv
  ID NAME             STATUS     PATH                           BRAND    IP
   0 global           running    /                              native   shared
   - ufszone          installed  /uzone/ufszone                 native   shared
   - zfszone          installed  /pool/zones/zfszone            native   shared
```

Managing Your ZFS Swap and Dump Devices

Oracle Solaris OS 초기 설치를 수행하는 도중이나 Live Upgrade를 사용하여 UFS 파일 시스템에서 마이그레이션을 수행한 후 ZFS 루트 풀의 ZFS 볼륨에는 스왑 영역이 만들어집니다. 예를 들면 다음과 같습니다.

```
# swap -l
swapfile                  dev  swaplo  blocks   free
/dev/zvol/dsk/rpool/swap 256,1      16 4194288 4194288
```

Oracle Solaris OS 초기 설치를 수행하거나 UFS 파일 시스템에서 Live Upgrade를 사용하는 도중 ZFS 루트 풀의 ZFS 볼륨에는 덤프 장치가 만들어집니다. 일반적으로 덤프 장치는 설치 시 자동으로 설정되므로 관리 작업이 필요하지 않습니다. 예를 들면 다음과 같습니다.

```
# dumpadm
      Dump content: kernel pages
       Dump device: /dev/zvol/dsk/rpool/dump (dedicated)
Savecore directory: /var/crash/t2000
  Savecore enabled: yes
   Save compressed: on
```

덤프 장치를 사용 안함으로 설정하고 제거할 경우 다시 만든 후에는 `dumpadm` 명령을 통해 사용으로 설정해야 합니다. 대부분의 경우 `zfs` 명령을 통해서만 덤프 장치의 크기를 조정해야 합니다.

설치 프로그램을 통해 만들어진 스왑 및 덤프 볼륨 크기에 대한 자세한 내용은 105 페이지 “ZFS 지원을 위한 Oracle Solaris 설치 및 Live Upgrade 요구 사항”을 참조하십시오.

설치 도중, 그리고 설치 후 스왑 볼륨 크기와 덤프 볼륨 크기를 모두 조정할 수 있습니다. 자세한 내용은 147 페이지 “ZFS 스왑 장치 및 덤프 장치의 크기 조정”을 참조하십시오.

ZFS 스왑 및 덤프 장치를 사용할 때는 다음 문제를 고려하십시오.

- 스왑 영역과 덤프 장치에 별도의 ZFS 볼륨을 사용해야 합니다.
- 현재 ZFS 파일 시스템에서는 스왑 파일을 사용할 수 없습니다.

- 시스템을 설치하거나 업그레이드한 후 스왑 영역 또는 덤프 장치를 변경해야 할 경우 이전 릴리스에서와 마찬가지로 `swap` 및 `dumpadm` 명령을 사용하십시오. 자세한 내용은 **System Administration Guide: Devices and File Systems**의 16 장, "Configuring Additional Swap Space (Tasks)" 및 **시스템 관리 설명서: 고급 관리**의 17 장, "시스템 충돌 정보 관리(작업)"를 참조하십시오.

자세한 내용은 다음 절을 참조하십시오.

- 147 페이지 "ZFS 스왑 장치 및 덤프 장치의 크기 조정"
- 149 페이지 "ZFS 덤프 장치 문제 해결"

ZFS 스왑 장치 및 덤프 장치의 크기 조정

설치 후 스왑 및 덤프 장치의 크기를 조정해야 하거나, 스왑 및 덤프 볼륨을 다시 만들어야 할 수 있습니다.

- 초기 설치 도중 스왑 및 덤프 볼륨의 크기를 조정할 수 있습니다. 자세한 내용은 예 4–1을 참조하십시오.
- Live Upgrade 작업을 수행하기 전에 스왑 및 덤프 볼륨을 만들고 크기를 지정할 수 있습니다. 예를 들면 다음과 같습니다.

  1. 저장소 풀을 만듭니다.

     ```
     # zpool create rpool mirror c0t0d0s0 c0t1d0s0
     ```

  2. 덤프 장치를 만듭니다.

     ```
     # zfs create -V 2G rpool/dump
     ```

  3. 덤프 장치를 사용으로 설정합니다.

     ```
     # dumpadm -d /dev/zvol/dsk/rpool/dump
           Dump content: kernel pages
            Dump device: /dev/zvol/dsk/rpool/dump (dedicated)
     Savecore directory: /var/crash/t2000
       Savecore enabled: yes
        Save compressed: on
     ```

  4. 스왑 볼륨 만들기:

     ```
     # zfs create -V 2G rpool/swap
     ```

  5. 새 스왑 장치를 추가하거나 변경하는 경우 스왑 영역을 사용으로 설정해야 합니다.

     ```
     # swap -a /dev/zvol/dsk/rpool/swap
     ```

  6. 스왑 볼륨에 대한 항목을 `/etc/vfstab` 파일에 추가합니다.

  Live Upgrade는 기존 스왑 및 덤프 볼륨의 크기를 조정하지 않습니다.

- 시스템이 설치된 후 덤프 장치의 `volsize` 등록 정보를 재설정할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs set volsize=2G rpool/dump
# zfs get volsize rpool/dump
NAME        PROPERTY  VALUE       SOURCE
rpool/dump  volsize   2G          -
```

- 현재 스왑 영역이 사용 중이 아니면 현재 스왑 볼륨의 크기를 조정할 수 있지만 늘어난 스왑 공간 크기를 확인하려면 시스템을 재부트해야 합니다.

```
# zfs get volsize rpool/swap
NAME        PROPERTY  VALUE    SOURCE
rpool/swap  volsize   4G       local
# zfs set volsize=8g rpool/swap
# zfs get volsize rpool/swap
NAME        PROPERTY  VALUE    SOURCE
rpool/swap  volsize   8G       local
# init 6
```

- 스왑 볼륨 크기 조정을 시도할 수 있지만 스왑 장치를 제거하는 것이 가장 좋을 수 있습니다. 그런 다음 스왑 장치를 다시 만듭니다. 예를 들면 다음과 같습니다.

```
# swap -d /dev/zvol/dsk/rpool/swap
# zfs create -V 2g rpool/swap
# swap -a /dev/zvol/dsk/rpool/swap
```

- 다음과 유사한 프로파일 구문을 사용하여 JumpStart 프로파일에서 스왑 및 덤프 볼륨의 크기를 조정할 수 있습니다.

```
install_type initial_install
cluster SUNWCXall
pool rpool 16g 2g 2g c0t0d0s0
```

  이 프로파일에서 두 개의 2g 항목이 스왑 볼륨 및 덤프 볼륨의 크기를 각각 2GB로 설정합니다.

- 이미 설치된 시스템에서 스왑 공간이 더 필요할 경우 다른 스왑 볼륨을 추가하면 됩니다. 예를 들면 다음과 같습니다.

```
# zfs create -V 2G rpool/swap2
```

  그런 다음 새 스왑 볼륨을 활성화합니다. 예를 들면 다음과 같습니다.

```
# swap -a /dev/zvol/dsk/rpool/swap2
# swap -l
swapfile                  dev  swaplo   blocks   free
/dev/zvol/dsk/rpool/swap  256,1      16 1058800 1058800
/dev/zvol/dsk/rpool/swap2 256,3      16 4194288 4194288
```

  마지막으로 두번째 스왑 볼륨에 대한 항목을 /etc/vfstab 파일에 추가합니다.

ZFS 스왑 및 덤프 볼륨 사용자 정의

기본 스왑 및 덤프 볼륨을 제거하고 비루트(데이터) 풀에 다시 만드는 경우 다음 사항에 유의하십시오.

- 비루트 풀에 스왑 및 덤프 장치를 만드는 경우 스왑 및 덤프 볼륨을 RAIDZ 풀에 만들지 마십시오. 풀에 스왑 및 덤프 볼륨이 포함되어 있는 경우 1-디스크 풀 또는 미러링된 풀이어야 합니다.
- Live Upgrade를 사용하여 시스템을 업데이트하는 경우 `-P` 옵션을 사용하여 PBE에서 ABE까지 덤프 장치를 보존하십시오. 예를 들면 다음과 같습니다.

```
# lucreate -n newBE -P
```

ZFS 덤프 장치 문제 해결

시스템 충돌 덤프를 캡처하거나 덤프 장치의 크기를 조정할 때 문제가 발생하면 다음을 검토하십시오.

- 충돌 덤프가 자동으로 만들어지지 않은 경우 `savecore` 명령을 사용하여 충돌 덤프를 저장할 수 있습니다.
- ZFS 루트 파일 시스템을 처음 설치하거나 ZFS 루트 파일 시스템으로 마이그레이션할 때 자동으로 덤프 볼륨이 만들어집니다. 대부분의 경우에는 기본 덤프 볼륨 크기가 너무 작은 경우에만 덤프 볼륨의 크기를 조정해야 합니다. 예를 들어, 메모리가 큰 시스템에서 덤프 볼륨 크기를 다음과 같이 40GB로 늘립니다.

```
# zfs set volsize=40G rpool/dump
```

  큰 덤프 볼륨의 크기를 조정하는 프로세스는 시간이 오래 걸릴 수 있습니다.

  특정 이유로 덤프 장치를 수동으로 만든 후 덤프 장치를 사용으로 설정해야 할 경우 다음과 유사한 구문을 사용하십시오.

```
# dumpadm -d /dev/zvol/dsk/rpool/dump
      Dump content: kernel pages
       Dump device: /dev/zvol/dsk/rpool/dump (dedicated)
Savecore directory: /var/crash/t2000
  Savecore enabled: yes
```

- 메모리가 128GB 이상인 시스템의 경우 기본적으로 만들어진 덤프 장치보다 큰 덤프 장치가 필요할 수 있습니다. 덤프 장치가 너무 작아 기존 충돌 덤프를 캡처할 수 없을 경우 다음과 유사한 메시지가 표시됩니다.

```
# dumpadm -d /dev/zvol/dsk/rpool/dump
dumpadm: dump device /dev/zvol/dsk/rpool/dump is too small to hold a system dump
dump size 36255432704 bytes, device size 34359738368 bytes
```

  스왑 및 덤프 장치 크기 지정에 대한 자세한 내용은 **System Administration Guide: Devices and File Systems**의 "Planning for Swap Space"을 참조하십시오.

- 지금은 최상위 장치가 여러 개인 풀에 덤프 장치를 추가할 수 없습니다. 다음과 유사한 메시지가 표시됩니다.

```
# dumpadm -d /dev/zvol/dsk/datapool/dump
dump is not supported on device '/dev/zvol/dsk/datapool/dump': 'datapool' has multiple top level vdevs
```

최상위 장치가 여러 개일 수 없는 루트 풀에 덤프 장치를 추가하십시오.

ZFS 루트 파일 시스템에서 부트

SPARC 기반 시스템과 x86 기반 시스템은 부트에 필요한 파일을 포함하는 파일 시스템 이미지인 부트 아카이브를 사용하는 새로운 스타일의 부트를 사용합니다. 시스템이 ZFS 루트 파일 시스템에서 부트되면 부트를 위해 선택된 루트 파일 시스템에서 부트 아카이브와 커널 파일의 경로 이름이 확인됩니다.

시스템이 설치를 위해 부트되면 전체 설치 프로세스에서 루트 파일 시스템에 RAM 디스크가 사용됩니다.

ZFS 파일 시스템에서 부트하는 것과 UFS 파일 시스템에서 부트하는 것은 다릅니다. ZFS를 사용하는 경우 부트 장치 지정자가 단일 루트 파일 시스템이 아닌 저장소 풀을 식별하기 때문입니다. 저장소 풀에는 여러 **부트 가능 데이터 세트** 또는 ZFS 루트 파일 시스템이 포함될 수 있습니다. ZFS에서 부트하는 경우 부트 장치로 식별된 풀에 있는 루트 파일 시스템과 부트 장치를 지정해야 합니다.

기본적으로 부트를 위해 선택된 데이터 세트는 풀의 `bootfs` 등록 정보로 식별됩니다. `boot -Z` 명령을 통해 대체 부트 가능 데이터 세트를 지정하여 이와 같이 기본적으로 선택된 항목을 대체할 수 있습니다.

미러링된 ZFS 루트 풀의 대체 디스크에서 부트

시스템을 설치할 때 미러링된 ZFS 루트 풀을 만들 수도 있고, 설치 후 디스크를 연결하여 미러링된 ZFS 루트 풀을 만들 수도 있습니다. 자세한 내용은 다음을 참조하십시오.

- 107 페이지 “ZFS 루트 파일 시스템 설치(Oracle Solaris 초기 설치)”
- 113 페이지 “미러링된 ZFS 루트 풀을 만드는 방법(사후 설치)”

미러링된 ZFS 루트 풀에 대해 알려진 다음 문제를 검토하십시오.

- `zpool replace` 명령을 사용하여 루트 풀 디스크를 교체할 경우 `installboot` 또는 `installgrub` 명령을 사용하여 새로 교체된 디스크에 부트 정보를 설치해야 합니다. 초기 설치 방법을 사용하여 미러링된 ZFS 루트 풀을 만들거나 `zpool attach` 명령을 사용하여 디스크를 루트 풀에 연결하는 경우 이 단계가 필요하지 않습니다. `installboot` 및 `installgrub` 명령 구문은 다음과 같습니다.
  - SPARC:

    ```
    sparc# installboot -F zfs /usr/platform/‘uname -i‘/lib/fs/zfs/bootblk
    ```
  - x86:

    ```
    x86# installgrub /boot/grub/stage1 /boot/grub/stage2 /dev/rdsk/c0t1d0s0
    ```

- 미러링된 ZFS 루트 풀의 여러 장치에서 부트할 수 있습니다. 하드웨어 구성에 따라 다른 부트 장치가 지정되도록 PROM 또는 BIOS를 업데이트해야 할 수도 있습니다.

  예를 들어, 다음 풀의 디스크(`c1t0d0s0` 또는 `c1t1d0s0`)에서 부트할 수 있습니다.

```
# zpool status rpool
  pool: rpool
 state: ONLINE
 scrub: none requested
config:

        NAME          STATE     READ WRITE CKSUM
        rpool         ONLINE       0     0     0
          mirror-0    ONLINE       0     0     0
            c1t0d0s0  ONLINE       0     0     0
            c1t1d0s0  ONLINE       0     0     0
```

- SPARC: `ok` 프롬프트에서 대체 디스크를 지정합니다. 예를 들면 다음과 같습니다.

```
ok boot /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0
```

  시스템이 재부트되면 활성 부트 장치를 확인합니다. 예를 들면 다음과 같습니다.

```
SPARC# prtconf -vp | grep bootpath
        bootpath:  '/pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0,0:a'
```

- x86: 적합한 BIOS 메뉴에서 미러링된 ZFS 루트 풀의 대체 디스크를 선택합니다.

  그런 후 다음과 유사한 구문을 사용하여 대체 디스크에서 부트되었는지 확인합니다.

```
x86# prtconf -v|sed -n '/bootpath/,/value/p'
        name='bootpath' type=string items=1
            value='/pci@0,0/pci8086,25f8@4/pci108e,286@0/disk@0,0:a'
```

SPARC: ZFS 루트 파일 시스템에서 부트

ZFS BE가 여러 개인 SPARC 기반 시스템에서 `luactivate` 명령을 사용하여 BE에서 부트할 수 있습니다.

Oracle Solaris OS 설치 및 Live Upgrade 프로세스 도중 `bootfs` 등록 정보를 통해 기본 ZFS 루트 파일 시스템이 자동으로 지정됩니다.

하나의 풀에 부트 가능 데이터 세트가 여러 개 존재할 수 있습니다. 기본적으로 */pool-name*`/boot/menu.lst` 파일의 부트 가능 데이터 세트 항목은 풀의 `bootfs` 등록 정보로 식별되지만, `menu.lst` 항목에 풀의 대체 데이터 세트를 지정하는 `bootfs` 명령이 포함될 수 있습니다. 따라서 `menu.lst` 파일에 풀 내 여러 루트 파일 시스템에 대한 항목이 포함될 수 있습니다.

ZFS 루트 파일 시스템이 있는 시스템이 설치되거나 시스템이 ZFS 루트 파일 시스템으로 마이그레이션되면 다음과 유사한 항목이 `menu.lst` 파일에 추가됩니다.

```
title zfsBE
bootfs rpool/ROOT/zfsBE
```

```
title zfs2BE
bootfs rpool/ROOT/zfs2BE
```

새 BE가 만들어지면 menu.lst 파일이 자동으로 업데이트됩니다.

SPARC 기반 시스템에서는 다음과 같은 두 가지 ZFS 부트 옵션을 사용할 수 있습니다.

- BE가 활성화된 후에는 boot -L 명령을 사용하여 ZFS 풀의 부트 가능 데이터 세트 목록을 표시할 수 있습니다. 그런 다음 목록에서 부트 가능 데이터 세트 중 하나를 선택할 수 있습니다. 해당 데이터 세트를 부트하는 것과 관련된 자세한 지침이 표시됩니다. 지침에 따라 선택한 데이터 세트를 부트할 수 있습니다.
- boot -Z *dataset* 명령을 사용하여 특정 ZFS 데이터 세트를 부트할 수 있습니다.

예 4–11 SPARC: 특정 ZFS 부트 환경에서 부트

시스템 부트 장치의 ZFS 저장소 풀에 ZFS BE가 여러 개 있을 경우 luactivate 명령을 사용하여 기본 BE를 지정할 수 있습니다.

예를 들어, 다음 lustatus 출력 결과는 두 개의 ZFS BE를 사용할 수 있음을 보여 줍니다.

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
zfsBE                      yes      no     no        yes    -
zfs2BE                     yes      yes    yes       no     -
```

SPARC 기반 시스템에 ZFS BE가 여러 개 있을 경우 boot -L 명령을 사용하여 기본 BE가 아닌 다른 BE에서 부트할 수 있습니다. 하지만 boot -L 세션에서 부트된 BE는 기본 BE로 재설정되지 않으며 bootfs 등록 정보도 업데이트되지 않습니다. boot -L 세션에서 부트된 BE를 기본 BE로 설정하려면 luactivate 명령을 사용하여 활성화해야 합니다.

예를 들면 다음과 같습니다.

```
ok boot -L
Rebooting with command: boot -L
Boot device: /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0  File and args: -L

1 zfsBE
2 zfs2BE
Select environment to boot: [ 1 - 2 ]: 1
To boot the selected entry, invoke:
boot [<root-device>] -Z rpool/ROOT/zfsBE

Program terminated
ok boot -Z rpool/ROOT/zfsBE
```

예 4–12 SPARC: 비상 안전 모드로 ZFS 파일 시스템 부트

SPARC 기반 시스템에서는 다음과 같이 /platform/‘uname -i‘/failsafe에 있는 비상 안전 아카이브에서 부트할 수 있습니다.

예 4–12 SPARC: 비상 안전 모드로 ZFS 파일 시스템 부트 *(계속)*

```
ok boot -F failsafe
```

특정 ZFS 부트 가능 데이터 세트에서 비상 안전 아카이브를 부트하려면 다음과 유사한 구문을 사용하십시오.

```
ok boot -Z rpool/ROOT/zfsBE -F failsafe
```

x86: ZFS 루트 파일 시스템에서 부트

Oracle Solaris OS 설치 또는 Live Upgrade 프로세스 도중 ZFS가 자동으로 부트되도록 다음 항목이 */pool-name* `/boot/grub/menu.lst` 파일에 추가됩니다.

```
title Solaris 10 1/13  X86
findroot (rootfs0,0,a)
kernel$ /platform/i86pc/multiboot -B $ZFS-BOOTFS
module /platform/i86pc/boot_archive
title Solaris failsafe
findroot (rootfs0,0,a)
kernel /boot/multiboot kernel/unix -s -B console=ttya
module /boot/x86.miniroot-safe
```

GRUB가 부트 장치로 식별한 장치에 ZFS 저장소 풀이 있을 경우 GRUB 메뉴를 만드는 데 `menu.lst` 파일이 사용됩니다.

ZFS BE가 여러 개인 x86 기반 시스템에서는 GRUB 메뉴에서 BE를 선택할 수 있습니다. 이 메뉴 항목에 해당하는 루트 파일 시스템이 ZFS 데이터 세트일 경우 다음 옵션이 추가됩니다.

```
-B $ZFS-BOOTFS
```

예 4–13 x86: ZFS 파일 시스템 부트

시스템이 ZFS 파일 시스템에서 부트되면 `-B $ZFS-BOOTFS` 부트 매개변수에 따라 루트 장치가 지정됩니다. 예를 들면 다음과 같습니다.

```
title Solaris 10 1/13  X86
findroot (pool_rpool,0,a)
kernel /platform/i86pc/multiboot -B $ZFS-BOOTFS
module /platform/i86pc/boot_archive
title Solaris failsafe
findroot (pool_rpool,0,a)
kernel /boot/multiboot kernel/unix -s -B console=ttya
module /boot/x86.miniroot-safe
```

예 4–14 x86: 비상 안전 모드로 ZFS 파일 시스템 부트

x86 비상 안전 아카이브는 `/boot/x86.miniroot-safe`이며 GRUB 메뉴에서 Solaris 비상 안전 항목을 선택하여 이 아카이브를 부트할 수 있습니다. 예를 들면 다음과 같습니다.

```
title Solaris failsafe
findroot (pool_rpool,0,a)
kernel /boot/multiboot kernel/unix -s -B console=ttya
module /boot/x86.miniroot-safe
```

부트가 성공하지 못하도록 하는 ZFS 마운트 지점 문제 해결(Solaris 10 10/08)

활성 BE(부트 환경)를 변경할 때는 `luactivate` 명령을 사용하는 것이 가장 좋습니다. 잘못된 패치 또는 구성 오류로 인해 활성 BE 부트가 실패할 경우 다른 BE에서 부트하는 유일한 방법은 부트 시 선택하는 것입니다. PROM(SPARC 기반 시스템의 경우)에서 또는 GRUB 메뉴(x86 기반 시스템의 경우)에서 명시적으로 부트하여 대체 BE를 선택할 수 있습니다.

Solaris 10 10/08 릴리스에서 Live Upgrade의 버그로 인해 BE 내 ZFS 데이터 세트 또는 영역의 ZFS 데이터 세트에 잘못된 마운트 지점이 있을 수 있으므로 비활성 BE의 부트가 실패할 수 있습니다. 또한 이 버그로 인해 BE에 별도의 `/var` 데이터 세트가 있을 경우 BE가 마운트되지 않습니다.

영역의 ZFS 데이터 세트에 잘못된 마운트 지점이 있을 경우 다음 단계를 수행하여 마운트 지점을 수정할 수 있습니다.

▼ ZFS 마운트 지점 문제 해결 방법

1 비상 안전 아카이브에서 시스템을 부트합니다.

2 풀을 가져옵니다.

예를 들면 다음과 같습니다.

```
# zpool import rpool
```

3 잘못된 임시 마운트 지점을 검색합니다.

예를 들면 다음과 같습니다.

```
# zfs list -r -o name,mountpoint rpool/ROOT/s10up

    NAME                              MOUNTPOINT
    rpool/ROOT/s10up                  /.alt.tmp.b-VP.mnt/
    rpool/ROOT/s10up/zones            /.alt.tmp.b-VP.mnt//zones
    rpool/ROOT/s10up/zones/zonerootA  /.alt.tmp.b-VP.mnt/zones/zonerootA
```

루트 BE 마운트 지점(rpool/ROOT/s10up)은 /여야 합니다.

/var 마운트 문제로 인해 부트가 실패하면 /var 데이터 세트에서 유사한 잘못된 임시 마운트 지점을 검색합니다.

4 **ZFS BE 및 해당 데이터 세트에 대한 마운트 지점을 재설정합니다.**

예를 들면 다음과 같습니다.

```
# zfs inherit -r mountpoint rpool/ROOT/s10up
# zfs set mountpoint=/ rpool/ROOT/s10up
```

5 **시스템을 재부트합니다.**

특정 BE를 부트할 수 있는 옵션이 표시되면 OpenBoot PROM 프롬프트 또는 GRUB 메뉴에서 마운트 지점이 수정된 부트 환경을 선택합니다.

복구를 위해 ZFS 루트 환경에서 부트

root 암호 분실 또는 유사한 문제를 복구할 수 있도록 시스템을 부트해야 할 경우 다음 절차를 사용하십시오.

오류 심각도에 따라 비상 안전 모드를 부트하거나 대체 매체에서 부트해야 합니다. 일반적으로 잊어버렸거나 모르는 root 암호를 복구하려는 경우 비상 안전 모드를 부트할 수 있습니다.

- 155 페이지 “ZFS 비상 안전 모드 부트 방법”
- 156 페이지 “대체 매체에서 ZFS를 부트하는 방법”

루트 풀 또는 루트 풀 스냅샷을 복구해야 할 경우 157 페이지 “ZFS 루트 풀 또는 루트 풀 스냅샷 복구”를 참조하십시오.

▼ ZFS 비상 안전 모드 부트 방법

1 **비상 안전 모드를 부트합니다.**

- SPARC 기반 시스템의 경우 ok 프롬프트에서 다음을 입력합니다.

  ```
  ok boot -F failsafe
  ```

- x86 시스템의 경우 GRUB 메뉴에서 비상 안전 모드를 선택합니다.

2 **프롬프트가 표시되면 /a에서 ZFS BE를 마운트합니다.**

```
.
.
.
ROOT/zfsBE was found on rpool.
Do you wish to have it mounted read-write on /a? [y,n,?] y
```

```
mounting rpool on /a
Starting shell.
```

3 **/a/etc 디렉토리로 변경합니다.**

```
# cd /a/etc
```

4 **필요한 경우 TERM 유형을 설정합니다.**

```
# TERM=vt100
# export TERM
```

5 **passwd 또는 shadow 파일을 수정합니다.**

```
# vi shadow
```

6 **시스템을 재부트합니다.**

```
# init 6
```

▼ 대체 매체에서 ZFS를 부트하는 방법

문제가 발생하여 시스템이 제대로 부트되지 않거나 다른 심각한 문제가 발생할 경우 네트워크 설치 서버 또는 Oracle Solaris 설치 DVD에서 부트하고, 루트 풀을 가져오고, ZFS BE를 마운트한 후 문제 해결을 시도해야 합니다.

1 **설치 DVD 또는 네트워크에서 부트합니다.**

- SPARC - 다음 부트 방법 중 하나를 선택합니다.

  ```
  ok boot cdrom -s
  ok boot net -s
  ```

  -s 옵션을 사용하지 않을 경우 설치 프로그램을 종료해야 합니다.
- x86 – 네트워크 부트 옵션을 선택하거나 로컬 DVD에서 부트합니다.

2 **루트 풀을 가져오고 대체 마운트 지점을 지정합니다. 예를 들면 다음과 같습니다.**

```
# zpool import -R /a rpool
```

3 **ZFS BE를 마운트합니다. 예를 들면 다음과 같습니다.**

```
# zfs mount rpool/ROOT/zfsBE
```

4 **/a 디렉토리에서 ZFS BE 컨텐츠에 액세스합니다.**

```
# cd /a
```

5 **시스템을 재부트합니다.**

```
# init 6
```

ZFS 루트 풀 또는 루트 풀 스냅샷 복구

다음 절에서는 관련 작업 수행 방법에 대해 설명합니다.

▼ ZFS 루트 풀의 디스크 교체 방법

다음과 같은 경우 루트 풀의 디스크를 교체해야 할 수 있습니다.

- 루트 풀이 너무 작은 상태에서 작은 디스크를 보다 큰 디스크로 교체하려는 경우
- 루트 풀 디스크가 실패하는 경우. 비중복 풀에서 디스크가 실패하여 시스템이 부트되지 않을 경우 루트 풀 디스크를 교체하기 전에 대체 매체(예: DVD 또는 네트워크)에서 부트해야 합니다.

미러링된 루트 풀 구성에서는 대체 매체에서 부트하지 않고도 디스크를 교체할 수 있습니다. `zpool replace` 명령을 사용하여 실패한 디스크를 교체할 수 있습니다. 또는 추가 디스크가 있을 경우 `zpool attach` 명령을 사용할 수 있습니다. 추가 디스크를 연결하고 루트 풀 디스크를 분리하는 예는 이 절의 절차를 참조하십시오.

일부 하드웨어의 경우 실패한 디스크를 교체하기 위해 `zpool replace` 작업을 시도하기 전에 디스크를 오프라인 상태로 만들고 구성을 해제해야 합니다. 예를 들면 다음과 같습니다.

```
# zpool offline rpool c1t0d0s0
# cfgadm -c unconfigure c1::dsk/c1t0d0
<Physically remove failed disk c1t0d0>
<Physically insert replacement disk c1t0d0>
# cfgadm -c configure c1::dsk/c1t0d0
# zpool replace rpool c1t0d0s0
# zpool online rpool c1t0d0s0
# zpool status rpool
<Let disk resilver before installing the boot blocks>
SPARC# installboot -F zfs /usr/platform/'uname -i'/lib/fs/zfs/bootblk /dev/rdsk/c1t0d0s0
x86# installgrub /boot/grub/stage1 /boot/grub/stage2 /dev/rdsk/c1t9d0s0
```

교체 디스크를 넣은 후 해당 디스크를 온라인 상태로 만들거나 재구성할 필요가 없는 하드웨어도 있습니다.

교체 디스크에서의 부트를 테스트하고 교체 디스크가 실패할 경우 기존 디스크에서 수동으로 부트할 수 있도록 현재 디스크와 새 디스크의 부트 장치 경로 이름을 식별해야 합니다. 다음 절차의 예에서 현재 루트 풀 디스크(`c1t10d0s0`)의 경로 이름은 다음과 같습니다.

```
/pci@8,700000/pci@3/scsi@5/sd@a,0
```

교체 부트 디스크(c1t9d0s0)의 경로 이름은 다음과 같습니다.

```
/pci@8,700000/pci@3/scsi@5/sd@9,0
```

1 교체 디스크(또는 새 디스크)를 물리적으로 연결합니다.

2 새 디스크의 레이블이 SMI, 슬라이스가 0인지 확인합니다.

루트 풀에 사용되는 디스크의 레이블을 재지정하는 방법은 다음 참조 자료를 참조하십시오.

- SPARC: **System Administration Guide: Devices and File Systems**의 "How to Set Up a Disk for a ZFS Root File System"
- x86: **System Administration Guide: Devices and File Systems**의 "How to Set Up a Disk for a ZFS Root File System"

3 새 디스크를 루트 풀에 연결합니다.

예를 들면 다음과 같습니다.

```
# zpool attach rpool c1t10d0s0 c1t9d0s0
```

4 루트 풀 상태를 확인합니다.

예를 들면 다음과 같습니다.

```
# zpool status rpool
  pool: rpool
 state: ONLINE
status: One or more devices is currently being resilvered.  The pool will
        continue to function, possibly in a degraded state.
action: Wait for the resilver to complete.
 scrub: resilver in progress, 25.47% done, 0h4m to go
config:

        NAME           STATE     READ WRITE CKSUM
        rpool          ONLINE       0     0     0
          mirror-0     ONLINE       0     0     0
            c1t10d0s0  ONLINE       0     0     0
            c1t9d0s0   ONLINE       0     0     0

errors: No known data errors
```

5 작은 루트 풀 디스크를 큰 디스크로 교체하는 경우 풀의 autoexpand 등록 정보를 설정하여 풀의 크기를 확장하십시오.

기존 rpool 풀 크기 결정:

```
# zpool list rpool
NAME   SIZE  ALLOC   FREE  CAP  DEDUP  HEALTH  ALTROOT
rpool  29.8G   152K  29.7G   0%  1.00x  ONLINE  -

# zpool set autoexpand=on rpool
```

확장된 rpool 풀 크기 검토:

```
# zpool list rpool
NAME   SIZE  ALLOC  FREE  CAP  DEDUP  HEALTH  ALTROOT
rpool  279G   146K  279G   0%  1.00x  ONLINE  -
```

6 **새 디스크에서 부트할 수 있는지 확인합니다.**

예를 들어, SPARC 기반 시스템에서는 다음과 유사한 구문을 사용합니다.

```
ok boot /pci@8,700000/pci@3/scsi@5/sd@9,0
```

7 **시스템이 새 디스크에서 부트되면 이전 디스크를 분리합니다.**

예를 들면 다음과 같습니다.

```
# zpool detach rpool c1t10d0s0
```

8 **기본 부트 장치를 재설정하여 새 디스크에서 자동으로 부트하도록 시스템을 설정합니다.**

- SPARC - SPARC 부트 PROM에서 eeprom 명령 또는 setenv 명령을 사용합니다.
- x86 - 시스템 BIOS를 재구성합니다.

▼ 루트 풀 스냅샷을 만드는 방법

복구를 위해 루트 풀 스냅샷을 만들 수 있습니다. 루트 풀 스냅샷을 만들 때는 순환적 루트 풀 스냅샷을 수행하는 것이 가장 좋습니다.

다음 절차에서는 순환적 루트 풀 스냅샷을 만든 후 해당 스냅샷을 파일로 저장하고 원격 시스템의 풀에 스냅샷으로 저장합니다. 루트 풀이 실패할 경우 NFS를 사용하여 원격 데이터 세트를 마운트할 수 있으며 다시 만들어진 풀로 스냅샷 파일을 수신할 수 있습니다. 또한 루트 풀 스냅샷을 원격 시스템의 풀에 실제 스냅샷으로 저장할 수 있습니다. 원격 시스템에서 스냅샷을 주고 받는 작업은 약간 복잡합니다. 손상을 복구할 시스템이 Oracle Solaris OS 미니 루트에서 부트되는 동안 ssh를 구성하거나 rsh를 사용해야 하기 때문입니다.

파일 또는 스냅샷으로 원격 저장된 스냅샷을 검증하는 것은 루트 풀 복구 과정에서 중요한 단계입니다. 어떤 방법이든 정기적으로(예: 풀 구성이 변경되는 경우 또는 Solaris OS가 업그레이드되는 경우) 스냅샷을 다시 만들어야 합니다.

다음 절차에서는 시스템이 zfsBE 부트 환경에서 부트됩니다.

1 **원격 시스템에서 스냅샷을 저장할 풀 및 파일 시스템을 만듭니다.**

예를 들면 다음과 같습니다.

```
remote# zfs create rpool/snaps
```

2 파일 시스템을 로컬 시스템과 공유합니다.

예를 들면 다음과 같습니다.

```
remote# zfs set sharenfs='rw=local-system,root=local-system' rpool/snaps
# share
-@rpool/snaps   /rpool/snaps   sec=sys,rw=local-system,root=local-system   ""
```

3 순환적 루트 풀 스냅샷을 만듭니다.

```
local# zfs snapshot -r rpool@snap1
local# zfs list -r rpool
# NAME                       USED  AVAIL  REFER  MOUNTPOINT
rpool                     15.1G   119G   106K  /rpool
rpool@snap1                   0      -   106K  -
rpool/ROOT                5.00G   119G    31K  legacy
rpool/ROOT@snap1              0      -    31K  -
rpool/ROOT/zfsBE          5.00G   119G  5.00G  /
rpool/ROOT/zfsBE@snap1        0      -  5.00G  -
rpool/dump                2.00G   120G  1.00G  -
rpool/dump@snap1              0      -  1.00G  -
rpool/export                63K   119G    32K  /export
rpool/export@snap1            0      -    32K  -
rpool/export/home           31K   119G    31K  /export/home
rpool/export/home@snap1       0      -    31K  -
rpool/swap                8.13G   123G  4.00G  -
rpool/swap@snap1              0      -  4.00G  -
```

4 루트 풀 스냅샷을 원격 시스템에 보냅니다.

예를 들어, 루트 풀 스냅샷을 파일로 원격 풀에 보내려면 다음과 유사한 구문을 사용합니다.

```
local# zfs send -Rv rpool@snap1 > /net/remote-system/rpool/snaps/rpool.snap1
sending from @ to rpool@snap1
sending from @ to rpool/ROOT@snap1
sending from @ to rpool/ROOT/s10zfsBE@snap1
sending from @ to rpool/dump@snap1
sending from @ to rpool/export@snap1
sending from @ to rpool/export/home@snap1
sending from @ to rpool/swap@snap1

local# zfs send -Rv rpool@snap1 > /net/remote-system/rpool/snaps/rpool.snap1
sending from @ to rpool@snap1
sending from @ to rpool/export@snap1
sending from @ to rpool/export/home@snap1
sending from @ to rpool/ROOT@snap1
sending from @ to rpool/ROOT/zfsBE@snap1
sending from @ to rpool/dump@snap1
sending from @ to rpool/swap@snap1
```

루트 풀 스냅샷을 스냅샷으로 원격 풀에 보내려면 다음과 유사한 구문을 사용합니다.

```
local# zfs send -Rv rpool@snap1 | ssh remote-system zfs receive
-Fd -o canmount=off tank/snaps
sending from @ to rpool@snap1
sending from @ to rpool/export@snap1
sending from @ to rpool/export/home@snap1
```

```
sending from @ to rpool/ROOT@snap1
sending from @ to rpool/ROOT/zfsBE@snap1
sending from @ to rpool/dump@snap1
sending from @ to rpool/swap@snap1
```

▼ ZFS 루트 풀을 다시 만들고 루트 풀 스냅샷을 복원하는 방법

이 절차에서는 다음과 같이 가정합니다.

- ZFS 루트 풀을 복구할 수 없습니다.
- ZFS 루트 풀 스냅샷이 원격 시스템에 저장되어 NFS를 통해 공유됩니다.

모든 단계는 로컬 시스템에서 수행됩니다.

1 설치 DVD 또는 네트워크에서 부트합니다.

- SPARC - 다음 부트 방법 중 하나를 선택합니다.

  ```
  ok boot net -s
  ok boot cdrom -s
  ```

  -s 옵션을 사용할 수 없을 경우 설치 프로그램을 종료해야 합니다.
- x86 – DVD에서 부트하거나 네트워크에서 부트하는 옵션을 선택합니다. 그런 다음 설치 프로그램을 종료합니다.

2 루트 풀 스냅샷을 파일로 원격 시스템에 보낸 경우 원격 스냅샷 파일 시스템을 마운트합니다.

예를 들면 다음과 같습니다.

```
# mount -F nfs remote-system:/rpool/snaps /mnt
```

네트워크 서비스가 구성되지 않은 경우 **원격 시스템**의 IP 주소를 지정해야 할 수 있습니다.

3 루트 풀 디스크가 교체되고 ZFS에서 사용 가능한 디스크 레이블을 포함하지 않는 경우 디스크 레이블을 재지정해야 합니다.

디스크 레이블 재지정에 대한 자세한 내용은 다음 참조 자료를 참조하십시오.

- SPARC: **System Administration Guide: Devices and File Systems**의 "How to Set Up a Disk for a ZFS Root File System"
- x86: **System Administration Guide: Devices and File Systems**의 "How to Set Up a Disk for a ZFS Root File System"

4 **루트 풀을 다시 만듭니다.**

예를 들면 다음과 같습니다.

```
# zpool create -f -o failmode=continue -R /a -m legacy -o cachefile=
/etc/zfs/zpool.cache rpool c1t1d0s0
```

5 **루트 풀 스냅샷을 복원합니다.**

이 단계는 오래 걸릴 수 있습니다. 예를 들면 다음과 같습니다.

```
# cat /mnt/rpool.snap1 | zfs receive -Fdu rpool
```

`-u` 옵션을 사용하면 `zfs receive` 작업이 완료될 때 복원된 아카이브가 마운트되지 않습니다.

원격 시스템의 풀에 저장된 실제 루트 풀 스냅샷을 복원하려면 다음과 유사한 구문을 사용합니다.

```
# ssh remote-system zfs send -Rb tank/snaps/rpool@snap1 | zfs receive -F rpool
```

6 **루트 풀 데이터 세트가 복원되었는지 확인합니다.**

예를 들면 다음과 같습니다.

```
# zfs list
```

7 **루트 풀 BE에서 `bootfs` 등록 정보를 설정합니다.**

예를 들면 다음과 같습니다.

```
# zpool set bootfs=rpool/ROOT/zfsBE rpool
```

8 **새 디스크에 부트 블록을 설치합니다.**

- SPARC:

  ```
  # installboot -F zfs /usr/platform/‘uname -i‘/lib/fs/zfs/bootblk /dev/rdsk/c1t1d0s0
  ```

- x86:

  ```
  # installgrub /boot/grub/stage1 /boot/grub/stage2 /dev/rdsk/c1t1d0s0
  ```

9 **시스템을 재부트합니다.**

```
# init 6
```

▼ 비상 안전 부트에서 루트 풀 스냅샷을 롤백하는 방법

이 절차에서는 기존 루트 풀 스냅샷을 사용할 수 있다고 가정합니다. 예에서는 로컬 시스템에서 이를 사용할 수 있습니다.

```
# zfs snapshot -r rpool@snap1
# zfs list -r rpool
NAME                       USED  AVAIL  REFER  MOUNTPOINT
```

```
rpool                      7.84G  59.1G   109K  /rpool
rpool@snap1                  21K      -   106K  -
rpool/ROOT                 4.78G  59.1G    31K  legacy
rpool/ROOT@snap1              0      -    31K  -
rpool/ROOT/s10zfsBE        4.78G  59.1G  4.76G  /
rpool/ROOT/s10zfsBE@snap1  15.6M      -  4.75G  -
rpool/dump                 1.00G  59.1G  1.00G  -
rpool/dump@snap1             16K      -  1.00G  -
rpool/export                 99K  59.1G    32K  /export
rpool/export@snap1           18K      -    32K  -
rpool/export/home            49K  59.1G    31K  /export/home
rpool/export/home@snap1      18K      -    31K  -
rpool/swap                 2.06G  61.2G    16K  -
rpool/swap@snap1              0      -    16K  -
```

1 시스템을 종료하고 비상 안전 모드를 부트합니다.

```
ok boot -F failsafe
ROOT/zfsBE was found on rpool.
Do you wish to have it mounted read-write on /a? [y,n,?] y
mounting rpool on /a

Starting shell.
```

2 각 루트 풀 스냅샷을 롤백합니다.

```
# zfs rollback rpool@snap1
# zfs rollback rpool/ROOT@snap1
# zfs rollback rpool/ROOT/s10zfsBE@snap1
```

3 다중 사용자 모드로 재부트합니다.

```
# init 6
```

◆ ◆ ◆ 5 장

5 Oracle Solaris ZFS 파일 시스템 관리

이 장에서는 Oracle Solaris ZFS 파일 시스템 관리에 대한 자세한 정보를 제공합니다. 여기에는 계층적 파일 시스템 레이아웃, 등록 정보 상속성 및 자동 마운트 지점 관리 및 공유 상호 작용과 같은 개념이 포함됩니다.

이 장에서는 다음과 같은 내용을 다룹니다.

ZFS 파일 시스템 관리(개요)

ZFS 파일 시스템은 저장소 풀을 기반으로 작성됩니다. 파일 시스템은 기본 디스크 공간을 할당하거나 포맷할 필요 없이 동적으로 만들고 삭제할 수 있습니다. 파일 시스템은 용량이 적고 ZFS에서 중앙 관리 지점이 되기 때문에 파일 시스템을 여러 개 만들어야 할 수 있습니다.

ZFS 파일 시스템은 `zfs` 명령을 사용하여 관리됩니다. `zfs` 명령은 파일 시스템에서 특정 작업을 수행하는 하위 명령 세트를 제공합니다. 이 장에서는 이러한 하위 명령에 대해 자세히 설명합니다. 스냅샷, 볼륨 및 복제본도 이 명령을 사용하여 관리되지만 이 장에서는 이러한 기능에 대해 간략하게만 설명합니다. 스냅샷 및 복제본에 대한 자세한 내용은 6 장, “Oracle Solaris ZFS 스냅샷 및 복제 작업”을 참조하십시오. ZFS 볼륨에 대한 자세한 내용은 261 페이지 “ZFS 볼륨”을 참조하십시오.

주 – 이 장에서 **데이터 세트**는 파일 시스템, 스냅샷, 복제본 또는 볼륨을 나타내는 일반 용어로 사용됩니다.

ZFS 파일 시스템 만들기, 삭제 및 이름 바꾸기

zfs create 및 zfs destroy 명령을 사용하면 ZFS 파일 시스템을 만들고 삭제할 수 있습니다. zfs rename 명령을 사용하면 ZFS 파일 시스템의 이름을 바꿀 수 있습니다.

ZFS 파일 시스템 만들기

ZFS 파일 시스템은 zfs create 명령을 사용하여 만들어집니다. create 하위 명령은 만들려는 파일 시스템의 이름을 단일 인수로 사용합니다. 파일 시스템 이름은 다음과 같이 풀 이름으로부터 시작되는 경로 이름으로 지정됩니다.

pool-name/[filesystem-name/]filesystem-name

경로에 있는 풀 이름 및 초기 파일 시스템 이름은 계층에서 새 파일 시스템이 만들어지는 위치를 식별합니다. 경로에서 마지막 이름은 만들려는 파일 시스템의 이름을 식별합니다. 파일 시스템 이름은 29 페이지 "ZFS 구성 요소 명명 요구 사항"의 조건을 충족해야 합니다.

다음 예제에서는 jeff라는 파일 시스템이 tank/home 파일 시스템에 만들어집니다.

```
# zfs create tank/home/jeff
```

ZFS는 파일 시스템이 성공적으로 만들어질 경우, 새로 만들어진 파일 시스템을 자동으로 마운트합니다. 기본적으로 파일 시스템은 create 하위 명령에서 파일 시스템 이름으로 제공된 경로를 사용하여 */dataset*로 마운트됩니다. 이 예제에서 새로 만들어진 jeff 파일 시스템은 /tank/home/jeff에 마운트됩니다. 자동으로 관리되는 마운트 지점에 대한 자세한 내용은 189 페이지 "ZFS 마운트 지점 관리"를 참조하십시오.

zfs create 명령에 대한 자세한 내용은 zfs(1M)을 참조하십시오.

파일 시스템이 만들어진 경우 파일 시스템 등록 정보를 설정할 수 있습니다.

다음 예제에서는 /export/zfs의 마운트 지점이 tank/home 파일 시스템에 대해 만들어집니다.

```
# zfs create -o mountpoint=/export/zfs tank/home
```

파일 시스템 등록 정보에 대한 자세한 내용은 169 페이지 “ZFS 등록 정보 소개”를 참조하십시오.

ZFS 파일 시스템 삭제

ZFS 파일 시스템을 삭제하려면 `zfs destroy` 명령을 사용합니다. 삭제된 파일 시스템은 자동으로 마운트 해제되고 공유 해제됩니다. 자동으로 관리되는 마운트 또는 자동으로 관리되는 공유에 대한 자세한 내용은 189 페이지 “자동 마운트 지점”을 참조하십시오.

다음 예제에서는 `tank/home/mark` 파일 시스템이 삭제됩니다.

```
# zfs destroy tank/home/mark
```

주의 – `destroy` 하위 명령의 경우 확인 프롬프트가 표시되지 않습니다. 이 명령을 사용할 때는 상당한 주의가 필요합니다.

삭제할 파일 시스템이 사용 중이거나 마운트 해제할 수 없는 경우 `zfs destroy` 명령이 실패합니다. 활성 파일 시스템을 삭제하려면 `-f` 옵션을 사용합니다. 이 옵션을 사용하면 활성 파일 시스템을 마운트 해제, 공유 해제 및 삭제하여 예상치 않은 응용 프로그램 동작이 발생할 수 있으므로 주의가 필요합니다.

```
# zfs destroy tank/home/matt
cannot unmount 'tank/home/matt': Device busy

# zfs destroy -f tank/home/matt
```

또한 파일 시스템에 종속 항목이 포함된 경우 `zfs destroy` 명령이 실패합니다. 파일 시스템과 모든 종속 항목을 반복해서 삭제하려면 `-r` 옵션을 사용합니다. 반복 삭제를 수행하면 스냅샷도 삭제되므로 이 옵션을 사용할 때는 주의가 필요합니다.

```
# zfs destroy tank/ws
cannot destroy 'tank/ws': filesystem has children
use '-r' to destroy the following datasets:
tank/ws/jeff
tank/ws/bill
tank/ws/mark
# zfs destroy -r tank/ws
```

삭제할 파일 시스템에 간접 종속 항목이 포함되어 있으면 반복 삭제 명령도 실패합니다. 대상 계층 외부의 복제된 파일 시스템을 포함하여 **모든** 종속 항목을 강제로 삭제하려면 `-R` 옵션을 사용해야 합니다. 이 옵션을 사용할 때는 상당한 주의가 필요합니다.

```
# zfs destroy -r tank/home/eric
cannot destroy 'tank/home/eric': filesystem has dependent clones
use '-R' to destroy the following datasets:
tank//home/eric-clone
# zfs destroy -R tank/home/eric
```

주의 – `zfs destroy` 명령에서 `-f`, `-r` 또는 `-R` 옵션에는 확인 프롬프트가 표시되지 않으므로 이러한 옵션을 사용할 때 주의가 필요합니다.

스냅샷 및 복제본에 대한 자세한 내용은 6 장, “Oracle Solaris ZFS 스냅샷 및 복제 작업”을 참조하십시오.

ZFS 파일 시스템 이름 바꾸기

`zfs rename` 명령을 사용하면 파일 시스템의 이름을 바꿀 수 있습니다. `rename` 하위 명령을 사용하면 다음 작업을 수행할 수 있습니다.

- 파일 시스템의 이름을 변경합니다.
- ZFS 계층 내에서 파일 시스템을 재배치합니다.
- 파일 시스템의 이름을 변경하고 ZFS 계층 내에 재배치합니다.

다음 예제에서는 `rename` 하위 명령을 사용하여 파일 시스템의 이름을 `eric`에서 `eric_old`로 바꿉니다.

```
# zfs rename tank/home/eric tank/home/eric_old
```

다음 예제에서는 `zfs rename`을 사용하여 파일 시스템을 재배치하는 방법을 보여 줍니다.

```
# zfs rename tank/home/mark tank/ws/mark
```

이 예제에서 `mark` 파일 시스템은 `tank/home`에서 `tank/ws`로 재배치됩니다. 이름 바꾸기를 통해 파일 시스템을 재배치할 때는 새 위치가 같은 풀 내에 있어야 하며, 이 새 파일 시스템을 저장하기 위한 충분한 디스크 공간이 있어야 합니다. 해당 쿼터에 도달하여 새 위치에 충분한 디스크 공간이 없으면 `rename` 작업이 실패합니다.

쿼터에 대한 자세한 내용은 194 페이지 “ZFS 쿼터 및 예약 설정”을 참조하십시오.

`rename` 작업을 수행하면 파일 시스템 및 모든 종속 파일 시스템의 시퀀스를 마운트 해제/재마운트하려고 시도합니다. 작업이 활성 파일 시스템을 마운트 해제할 수 없으면 `rename` 명령이 실패합니다. 이 문제가 발생하면 파일 시스템을 강제로 마운트 해제해야 합니다.

스냅샷 이름 바꾸기에 대한 자세한 내용은 206 페이지 “ZFS 스냅샷 이름 바꾸기”를 참조하십시오.

ZFS 등록 정보 소개

등록 정보는 파일 시스템, 볼륨, 스냅샷 및 복제본의 동작을 제어하기 위해 사용하는 기본 방식입니다. 특별한 설명이 없는 한 이 절에 정의된 등록 정보는 모든 데이터 세트 유형에 적용됩니다.

- 176 페이지 "ZFS 읽기 전용 고유 등록 정보"
- 177 페이지 "설정 가능한 ZFS 고유 등록 정보"
- 180 페이지 "ZFS 사용자 등록 정보"

등록 정보는 고유 등록 정보와 사용자 정의 등록 정보의 두 가지 유형으로 구분됩니다. 고유 등록 정보는 내부 통계를 제공하거나 ZFS 파일 시스템 동작을 제어합니다. 또한 고유 등록 정보는 설정 가능하거나 읽기 전용입니다. 사용자 등록 정보는 ZFS 파일 시스템 동작에 영향을 주지 않지만 이를 사용하여 해당 환경에 필요한 방식으로 데이터 세트에 주석을 달 수 있습니다. 사용자 등록 정보에 대한 자세한 내용은 180 페이지 "ZFS 사용자 등록 정보"를 참조하십시오.

대부분의 설정 가능한 등록 정보는 상속 가능성도 포함합니다. 상속 가능한 등록 정보는 부모 파일 시스템에 설정할 경우 모든 종속 항목으로 전파되는 등록 정보입니다.

모든 상속 가능한 등록 정보는 등록 정보를 가져온 방법을 나타내는 연관된 소스를 포함합니다. 등록 정보의 소스는 다음과 같은 값을 가질 수 있습니다.

`local`	184 페이지 "ZFS 등록 정보 설정"에 설명된 대로 `zfs set` 명령을 사용하여 데이터 세트에 등록 정보가 명시적으로 설정되었음을 나타냅니다.
`inherited from` *dataset-name*	등록 정보가 명명된 상위 항목으로부터 상속되었음을 나타냅니다.
`default`	등록 정보 값이 상속되지 않았거나 로컬로 설정되었음을 나타냅니다. 이 소스는 등록 정보가 소스 `local`로 설정된 상위 항목이 없기 때문에 발생한 결과입니다.

다음 표에서는 읽기 전용 및 설정 가능한 고유 ZFS 파일 시스템 등록 정보를 모두 보여 줍니다. 읽기 전용 고유 등록 정보는 읽기 전용으로 식별됩니다. 이 표에 나열된 다른 모든 고유 등록 정보는 모두 설정 가능으로 식별됩니다. 사용자 등록 정보에 대한 자세한 내용은 180 페이지 "ZFS 사용자 등록 정보"를 참조하십시오.

표 5–1 ZFS 고유 등록 정보 설명

등록 정보 이름	유형	기본값	설명
`aclinherit`	문자열	`secure`	파일 및 디렉토리를 만들 때 ACL 항목이 상속되는 방법을 제어합니다. 값은 `discard`, `noallow`, `secure` 및 `passthrough`입니다. 해당 값에 대한 설명은 228 페이지 "ACL 등록 정보"를 참조하십시오.

표 5–1 ZFS 고유 등록 정보 설명 (계속)

등록 정보 이름	유형	기본값	설명
aclmode	문자열	groupmask	chmod 작업 도중 ACL 항목을 수정하는 방법을 제어합니다. 값은 discard, groupmask 및 passthrough입니다. 이러한 값에 대한 자세한 내용은 228 페이지 "ACL 등록 정보"를 참조하십시오.
atime	부울	on	파일을 읽을 때 파일의 액세스 시간이 업데이트되는지 여부를 제어합니다. 이 등록 정보를 해제하면 파일을 읽을 때 쓰기 트래픽이 발생하는 것을 방지하여 성능상의 상당한 이점을 얻을 수 있지만 메일러 및 유사 유틸리티에 혼동을 줄 수도 있습니다.
available	숫자	해당 없음	풀에 다른 작업이 없다고 가정하고 파일 시스템 및 모든 자식에 사용 가능한 디스크 공간을 식별하는 읽기 전용 등록 정보입니다. 디스크 공간은 풀 내에서 공유되기 때문에 사용 가능한 공간은 실제 풀 크기, 쿼터, 예약 및 풀 내에 있는 기타 데이터 세트 등의 다양한 요소에 의해 제한될 수 있습니다. 이 등록 정보의 약어는 avail입니다. 디스크 공간 계산에 대한 자세한 내용은 30 페이지 "ZFS 디스크 공간 계산"을 참조하십시오.
canmount	부울	on	zfs mount 명령을 사용하여 파일 시스템을 마운트할 수 있는지 여부를 제어합니다. 이 등록 정보는 모든 파일 시스템에서 설정할 수 있으며 등록 정보 자체는 상속이 불가능합니다. 하지만 이 등록 정보가 off로 설정되어 있으면 마운트 지점을 종속 파일 시스템으로 상속할 수 있습니다. 그래도 파일 시스템 자체는 마운트되지 않습니다. 자세한 내용은 179 페이지 "canmount 등록 정보"를 참조하십시오.
checksum	문자열	on	데이터 무결성을 확인하기 위해 사용되는 체크섬을 제어합니다. 기본값은 적합한 알고리즘(현재까지 fletcher4)을 자동으로 선택하는 on입니다. 값은 on, off, fletcher2, fletcher4 및 sha256입니다. 값이 off면 사용자 데이터에 대한 무결성 검사가 사용 안함으로 설정됩니다. off 값은 권장되지 않습니다.

표 5–1 ZFS 고유 등록 정보 설명 *(계속)*

등록 정보 이름	유형	기본값	설명
compression	문자열	off	데이터 세트에 대한 압축을 사용 또는 사용 안함으로 설정합니다. 값은 on, off, lzjb, gzip 및 gzip-*N*입니다. 현재까지 이 등록 정보를 lzjb, gzip 또는 gzip-*N*으로 설정하는 것은 이 등록 정보를 on으로 설정하는 것과 효과가 동일합니다. 기존 데이터가 있는 파일 시스템에서 압축을 사용으로 설정하면 새 데이터만 압축됩니다. 기존 데이터는 압축되지 않은 상태로 남습니다. 이 등록 정보의 약어는 compress입니다.
compressratio	숫자	해당 없음	데이터 세트에 대해 얻은 압축 비율(배수로 표현)을 식별하는 읽기 전용 등록 정보입니다. zfs set compression=on *dataset* 명령을 통해 압축을 사용으로 설정할 수 있습니다. 값은 모든 파일의 논리적 크기 및 참조되는 실제 데이터의 양으로부터 계산됩니다. 여기에는 compression 등록 정보를 사용하여 얻는 명시적인 절약 공간이 포함됩니다.
copies	숫자	1	파일 시스템별 사용자 데이터의 복사본 수를 설정합니다. 사용 가능한 값은 1, 2 또는 3입니다. 이러한 복사본은 모든 풀 레벨의 중복성에 대해 추가로 설정됩니다. 사용자 데이터의 여러 복사본에 사용되는 디스크 공간은 해당 파일 및 데이터 세트에 포함되며 쿼터 및 예약에서 공제됩니다. 또한 used 등록 정보는 여러 복사본이 사용으로 설정될 때 업데이트됩니다. 기존 파일 시스템에서 이 등록 정보를 변경하면 새로 생성되는 데이터에만 영향을 주기 때문에 파일 시스템을 만들 때 이 등록 정보를 설정하는 것이 좋습니다.
creation	문자열	해당 없음	데이터 세트를 만든 날짜 및 시간을 식별하는 읽기 전용 등록 정보입니다.
devices	부울	on	파일 시스템의 장치 파일을 열 수 있는지 여부를 제어합니다.
exec	부울	on	파일 시스템의 프로그램 실행을 허용할지 여부를 제어합니다. 또한 off로 설정된 경우, PROT_EXEC를 사용한 mmap(2) 호출은 허용되지 않습니다.
mounted	부울	해당 없음	파일 시스템, 복제본 또는 스냅샷이 현재 마운트되었는지 여부를 나타내는 읽기 전용 등록 정보입니다. 이 등록 정보는 볼륨에 적용되지 않습니다. 값은 yes 또는 no일 수 있습니다.

표 5–1 ZFS 고유 등록 정보 설명 (계속)

등록 정보 이름	유형	기본값	설명
mountpoint	문자열	해당 없음	이 파일 시스템에 사용된 마운트 지점을 제어합니다. 특정 파일 시스템에서 mountpoint 등록 정보가 변경된 경우 해당 파일 시스템 및 마운트 지점을 상속하는 모든 종속 항목이 마운트 해제됩니다. 새 값이 legacy이면 마운트 해제된 상태로 유지됩니다. 그렇지 않으면 등록 정보가 이전에 legacy 또는 none인 경우 또는 등록 정보가 변경되기 전에 마운트된 경우 새 위치에 자동으로 재마운트됩니다. 또한 모든 공유 파일 시스템이 공유 해제되고 새 위치에서 공유됩니다. 이 등록 정보 사용에 대한 자세한 내용은 189 페이지 "ZFS 마운트 지점 관리"를 참조하십시오.
primarycache	문자열	all	기본 캐시(ARC)에 캐시되는 항목을 제어합니다. 가능한 값은 all, none 및 metadata입니다. all로 설정된 경우 사용자 데이터 및 메타 데이터가 모두 캐시됩니다. none으로 설정된 경우 사용자 데이터 또는 메타 데이터가 캐시되지 않습니다. metadata로 설정된 경우 메타 데이터만 캐시됩니다. 이러한 등록 정보가 기존 파일 시스템에 설정되어 있으면 해당 등록 정보의 값에 따라 새로운 I/O만 캐시됩니다. 일부 데이터베이스 환경은 사용자 데이터를 캐시하지 않는 것이 도움이 될 수 있습니다. 캐시 등록 정보 설정이 환경에 적합한지 여부를 결정해야 합니다.
origin	문자열	해당 없음	복제본이 만들어진 스냅샷을 식별하는 복제된 파일 시스템 또는 볼륨에 대한 읽기 전용 등록 정보입니다. 복제본이 존재하는 한 -r 또는 -f 옵션을 사용하더라도 원본을 삭제할 수 없습니다. 복제되지 않은 파일 시스템은 원본이 none입니다.
quota	숫자(또는 none)	none	파일 시스템 및 종속 항목이 사용할 수 있는 디스크 공간을 제한합니다. 이 등록 정보는 파일 시스템 및 스냅샷과 같이 종속 항목이 소비하는 모든 공간을 포함하여 사용되는 디스크 공간에 대한 하드 한계를 강제 적용합니다. 이미 쿼터가 있는 파일 시스템의 종속 항목에 대해 쿼터를 설정할 경우 상위 요소의 쿼터가 대체되는 대신 추가 제한이 적용됩니다. volsize 등록 정보는 암시적인 쿼터로 작동하므로 볼륨에 대해 쿼터를 설정할 수 없습니다. 쿼터 설정에 대한 자세한 내용은 195 페이지 "ZFS 파일 시스템에 대한 쿼터 설정"을 참조하십시오.

표 5–1 ZFS 고유 등록 정보 설명 *(계속)*

등록 정보 이름	유형	기본값	설명
readonly	부울	off	데이터 세트를 수정할 수 있는지 여부를 제어합니다. on으로 설정된 경우 항목을 수정할 수 없습니다. 이 등록 정보의 약어는 rdonly입니다.
recordsize	숫자	128K	파일 시스템에 있는 파일의 권장 블록 크기를 지정합니다. 이 등록 정보의 약어는 recsize입니다. 자세한 내용은 179 페이지 “recordsize 등록 정보”를 참조하십시오.
referenced	숫자	해당 없음	풀에 있는 다른 데이터 세트와 공유하거나 공유할 수 없는, 데이터 세트가 액세스할 수 있는 데이터의 양을 식별하는 읽기 전용 등록 정보입니다. 스냅샷 또는 복제본이 만들어지면 해당 컨텐츠가 동일하기 때문에 처음에 만들어진 파일 시스템 또는 스냅샷과 동일한 양의 디스크 공간을 참조합니다. 이 등록 정보의 약어는 refer입니다.
refquota	숫자(또는 none)	none	데이터 세트가 소비할 수 있는 디스크 공간을 설정합니다. 이 등록 정보는 사용되는 공간에 대한 하드 한계를 강제 적용합니다. 이 하드 한계에 스냅샷 및 복제본과 같은 종속 항목에서 사용되는 공간은 포함되지 않습니다.
refreservation	숫자(또는 none)	none	스냅샷 및 복제본과 같은 종속 항목을 제외하고 데이터 세트에 보장된 최소 디스크 공간을 설정합니다. 사용된 디스크 공간이 이 값 아래이면 데이터 세트가 refreservation으로 지정된 공간을 소비하는 것처럼 취급됩니다. refreservation 예약은 사용된 부모 데이터 세트의 디스크 공간으로 간주되며 부모 데이터 세트의 쿼터 및 예약에서 공제됩니다. refreservation이 설정된 경우에는 이 예약 외에도 데이터 세트에서 현재 **참조되는** 바이트 수를 수용할 수 있도록 충분한 여유 풀 공간을 사용할 수 있는 경우에만 스냅샷이 허용됩니다. 이 등록 정보의 약어는 refreserv입니다.

표 5–1 ZFS 고유 등록 정보 설명 (계속)

등록 정보 이름	유형	기본값	설명
reservation	숫자(또는 none)	none	파일 시스템 및 종속 항목에 보장되는 최소 디스크 공간을 설정합니다. 사용된 디스크 공간이 이 값 아래이면 파일 시스템이 해당 예약으로 지정된 공간을 사용 중인 것처럼 처리됩니다. 예약은 사용된 부모 파일 시스템의 디스크 공간으로 간주되며 부모 파일 시스템의 쿼터 및 예약 계산에 포함됩니다. 이 등록 정보의 약어는 reserv입니다. 자세한 내용은 198 페이지 “ZFS 파일 시스템에 대한 예약 설정”을 참조하십시오.
secondarycache	문자열	all	보조 캐시(L2ARC)에 캐시되는 항목을 제어합니다. 가능한 값은 all, none 및 metadata입니다. all로 설정된 경우 사용자 데이터 및 메타 데이터가 모두 캐시됩니다. none으로 설정된 경우 사용자 데이터 또는 메타 데이터가 캐시되지 않습니다. metadata로 설정된 경우 메타 데이터만 캐시됩니다.
setuid	부울	on	setuid 비트가 파일 시스템에서 보존되는지 여부를 제어합니다.
shareiscsi	문자열	Off	ZFS 볼륨이 iSCSI 대상으로 공유되는지 여부를 제어합니다. 등록 정보 값은 on, off 및 type=disk입니다. 파일 시스템 내의 모든 ZFS 볼륨이 기본적으로 공유되도록 파일 시스템에 대해 shareiscsi=on을 설정해야 할 수 있습니다. 하지만 파일 시스템에서는 이 등록 정보를 설정해도 직접적인 효과가 없습니다.
sharenfs	문자열	off	파일 시스템을 NFS에서 사용할 수 있는지 여부 그리고 사용되는 옵션을 제어합니다. on으로 설정된 경우 zfs share 명령이 옵션 없이 호출됩니다. 그렇지 않으면 zfs share 명령이 이 등록 정보의 컨텐츠와 상응하는 옵션으로 호출됩니다. off로 설정된 경우 파일 시스템은 레거시 share 및 unshare 명령과 dfstab 파일을 사용해서 관리됩니다. ZFS 파일 시스템 공유에 대한 자세한 내용은 193 페이지 “ZFS 파일 시스템 공유 및 공유 해제”를 참조하십시오.
snapdir	문자열	hidden	파일 시스템의 루트에서 .zfs 디렉토리를 숨기거나 표시할지를 제어합니다. 스냅샷 사용에 대한 자세한 내용은 203 페이지 “ZFS 스냅샷 개요”를 참조하십시오.

표 5–1 ZFS 고유 등록 정보 설명 *(계속)*

등록 정보 이름	유형	기본값	설명
type	문자열	해당 없음	데이터 세트 유형을 filesystem(파일 시스템 또는 복제본), volume 또는 snapshot으로 식별하는 읽기 전용 등록 정보입니다.
used	숫자	해당 없음	데이터 세트 및 모든 해당 종속 항목에서 소비되는 디스크 공간을 식별하는 읽기 전용 등록 정보입니다. 자세한 내용은 177 페이지 "used 등록 정보"를 참조하십시오.
usedbychildren	숫자	off	이 데이터 세트의 자식이 사용하는 디스크 공간(모든 데이터 세트의 자식이 삭제될 경우 비워짐)을 식별하는 읽기 전용 등록 정보입니다. 이 등록 정보의 약어는 usedchild입니다.
usedbydataset	숫자	off	먼저 모든 스냅샷을 삭제하고 모든 refreservation 예약을 제거한 후 데이터 세트 자체에 사용되는 디스크 공간(데이터 세트가 삭제된 후 비워짐)을 식별하는 읽기 전용 등록 정보입니다. 이 등록 정보의 약어는 usedds입니다.
usedbyrefreservation	숫자	off	데이터 세트에 설정된 refreservation에서 사용되는 디스크 공간(refreservation이 제거된 경우 비워짐)을 식별하는 읽기 전용 등록 정보입니다. 이 등록 정보의 약어는 usedrefreserv입니다.
usedbysnapshots	숫자	off	데이터 세트의 스냅샷에서 소비되는 디스크 공간을 식별하는 읽기 전용 등록 정보입니다. 특히 이 공간은 이 데이터 세트의 모든 스냅샷이 삭제되는 경우 비워지는 디스크 공간입니다. 여러 스냅샷에서 공간을 공유할 수 있으므로 이 값은 단순히 스냅샷의 used 등록 정보의 합계가 아닙니다. 이 등록 정보의 약어는 usedsnap입니다.
version	숫자	해당 없음	풀 버전과 독립적인 파일 시스템의 디스크 내장 버전을 식별합니다. 이 등록 정보는 지원되는 소프트웨어 릴리스에서 제공되는 이후 버전에만 설정할 수 있습니다. 자세한 내용은 zfs upgrade 명령을 참조하십시오.
volsize	숫자	해당 없음	볼륨에 대해 볼륨의 논리적 크기를 지정합니다. 자세한 내용은 179 페이지 "volsize 등록 정보"를 참조하십시오.

표 5–1 ZFS 고유 등록 정보 설명 (계속)

등록 정보 이름	유형	기본값	설명
volblocksize	숫자	8 KB	볼륨에 대해 볼륨의 블록 크기를 지정합니다. 블록 크기는 볼륨을 작성한 후 변경할 수 없으므로 볼륨을 만들 때 블록 크기를 설정하십시오. 볼륨의 기본 블록 크기는 8KB입니다. 유효한 값은 512바이트에서 128KB 사이의 모든 값에 대한 2배수입니다. 이 등록 정보의 약어는 volblock입니다.
zoned	부울	해당 없음	파일 시스템이 비전역 영역에 추가되었는지 여부를 나타냅니다. 이 등록 정보를 설정하면 전역 영역에서 마운트 지점이 보존되지 않으며, 요청이 있을 때 ZFS가 그러한 파일 시스템을 마운트할 수 없습니다. 영역을 처음 설치하면 추가되는 모든 파일 시스템에 대해 이 등록 정보가 설정됩니다. 설치된 영역에서 ZFS 사용에 대한 자세한 내용은 264 페이지 “영역이 설치된 Solaris 시스템에서 ZFS 사용”을 참조하십시오.
xattr	부울	on	이 파일 시스템에 대해 확장 속성이 사용(on) 또는 사용 안함(off)으로 설정되었는지를 나타냅니다.

ZFS 읽기 전용 고유 등록 정보

읽기 전용 고유 등록 정보는 검색할 수 있지만 설정할 수 없습니다. 읽기 전용 고유 등록 정보는 상속되지 않습니다. 일부 고유 등록 정보는 특정 데이터 세트의 유형에 한정됩니다. 표 5–1에는 이러한 데이터 유형에 대한 설명이 포함되어 있습니다.

읽기 전용 고유 등록 정보는 여기에 나열되어 있고 표 5–1에 설명되어 있습니다.

- available
- compressratio
- creation
- mounted
- origin
- referenced
- type
- used

  자세한 내용은 177 페이지 “used 등록 정보”를 참조하십시오.
- usedbychildren
- usedbydataset

- usedbyrefreservation
- usedbysnapshots

used , referenced 및 available 등록 정보를 포함한 디스크 공간 계산에 대한 자세한 내용은 30 페이지 “ZFS 디스크 공간 계산”을 참조하십시오.

used 등록 정보

used 등록 정보는 이 데이터 세트 및 모든 종속 항목에서 소비된 디스크 공간을 식별하는 읽기 전용 등록 정보입니다. 이 값은 데이터 세트의 쿼터 및 예약에서 공제됩니다. 이렇게 사용된 디스크 공간에는 데이터 세트의 예약이 포함되지 않지만 모든 종속 데이터 세트의 예약이 고려됩니다. 데이터 세트가 해당 부모로부터 소비하는 디스크 공간과 데이터 세트가 반복해서 삭제될 때 비워지는 디스크 공간은 사용된 공간과 해당 예약 중에서 큰 값입니다.

스냅샷을 만들면 처음에 스냅샷과 파일 시스템 간에 디스크 공간이 공유되며, 이전 스냅샷과 공유될 수도 있습니다. 파일 시스템이 변경되면 이전에 공유되던 디스크 공간이 해당 스냅샷의 고유 공간이 되고 스냅샷의 사용된 공간으로 계산됩니다. 스냅샷에서 사용되는 디스크 공간은 해당 고유 데이터로 계산됩니다. 또한 스냅샷을 삭제하면 다른 스냅샷에 고유한(그리고 다른 스냅샷에서 사용되는) 디스크 공간을 늘릴 수 있습니다. 스냅샷 및 공간 문제에 대한 자세한 내용은 31 페이지 “공간 부족 동작”을 참조하십시오.

사용된 디스크 공간, 사용 가능한 디스크 공간, 참조된 디스크 공간에는 보류 중인 변경 사항이 포함되지 않습니다. 일반적으로 변경 사항은 몇 초 동안 보류됩니다. fsync(3c) 또는 O_SYNC 함수를 사용하여 디스크에 변경 사항을 커밋해도 반드시 디스크 공간 사용 정보가 즉시 업데이트되는 것은 아닙니다.

usedbychildren, usedbydataset, usedbyrefreservation 및 usedbysnapshots 등록 정보는 zfs list -o space 명령을 사용하여 표시할 수 있습니다. 이러한 등록 정보는 used 등록 정보를 종속 항목에서 소비되는 디스크 공간으로 식별합니다. 자세한 내용은 표 5–1을 참조하십시오.

설정 가능한 ZFS 고유 등록 정보

설정 가능한 고유 등록 정보는 해당 값에 대해 검색 및 설정을 모두 수행할 수 있는 등록 정보입니다. 설정 가능한 고유 등록 정보는 184 페이지 “ZFS 등록 정보 설정”에 설명된 대로 zfs set 명령을 사용하거나 166 페이지 “ZFS 파일 시스템 만들기”에 설명된 대로 zfs create 명령을 사용하여 설정됩니다. 쿼터 및 예약을 제외하고, 설정 가능한 고유 등록 정보는 상속됩니다. 쿼터 및 예약에 대한 자세한 내용은 194 페이지 “ZFS 쿼터 및 예약 설정”을 참조하십시오.

일부 설정 가능한 고유 등록 정보는 특정 데이터 세트의 유형에 한정됩니다. 표 5–1에는 이러한 데이터 유형에 대한 설명이 포함되어 있습니다. 특별히 언급되지 않는 한 등록 정보는 파일 시스템, 볼륨, 복제본 및 스냅샷 등 모든 데이터 세트 유형에 적용됩니다.

설정 가능한 등록 정보는 여기에 나열되어 있고 표 5–1에 설명되어 있습니다.

- aclinherit
  자세한 내용은 228 페이지 “ACL 등록 정보”를 참조하십시오.
- aclmode
  자세한 내용은 228 페이지 “ACL 등록 정보”를 참조하십시오.
- atime
- canmount
- checksum
- compression
- copies
- devices
- exec
- mountpoint
- primarycache
- quota
- readonly
- recordsize
  자세한 내용은 179 페이지 “recordsize 등록 정보”를 참조하십시오.
- refquota
- refreservation
- reservation
- secondarycache
- shareiscsi
- setuid
- snapdir
- version
- volsize
  자세한 내용은 179 페이지 “volsize 등록 정보”를 참조하십시오.
- volblocksize
- zoned
- xattr

canmount 등록 정보

canmount 등록 정보가 off로 설정된 경우 zfs mount 또는 zfs mount -a 명령을 사용하여 파일 시스템을 마운트할 수 없습니다. 파일 시스템에 상속 가능한 일반적인 mountpoint 등록 정보가 계속 포함된다는 점만 제외하면 이 등록 정보를 off로 설정하는 것은 mountpoint 등록 정보를 none으로 설정하는 것과 비슷합니다. 예를 들어, 이 등록 정보를 off로 설정하고, 종속 파일 시스템에 대해 상속 가능한 등록 정보를 설정할 수 있지만 부모 파일 시스템 자체는 마운트되지 않으며 사용자가 액세스할 수도 없습니다. 이 경우 부모 파일 시스템은 사용자가 컨테이너에 대한 등록 정보를 설정할 수 있도록 *container*로 작동하지만 컨테이너 자체는 액세스할 수 없습니다.

다음 예제에서는 userpool을 만들고 해당 canmount 등록 정보를 off로 설정합니다. 종속 사용자 파일 시스템의 마운트 지점은 하나의 공통 마운트 지점인 /export/home으로 설정됩니다. 부모 파일 시스템에 설정되는 등록 정보는 종속 파일 시스템에 의해 상속되지만 부모 파일 시스템 자체는 마운트되지 않습니다.

```
# zpool create userpool mirror c0t5d0 c1t6d0
# zfs set canmount=off userpool
# zfs set mountpoint=/export/home userpool
# zfs set compression=on userpool
# zfs create userpool/user1
# zfs create userpool/user2
# zfs mount
userpool/user1                  /export/home/user1
userpool/user2                  /export/home/user2
```

recordsize 등록 정보

recordsize 등록 정보는 파일 시스템에서 파일의 권장 블록 크기를 지정합니다.

이 등록 정보는 고정 크기 레코드의 파일을 액세스하는 데이터베이스 작업 부하에서만 사용하도록 디자인되었습니다. ZFS는 일반적인 액세스 패턴에 맞게 최적화된 내부 알고리즘에 따라 블록 크기를 자동으로 조정합니다. 매우 큰 파일을 만들지만 작은 모든 청크로 파일을 액세스하는 데이터베이스의 경우 이러한 알고리즘은 최적의 방식이 아닐 수 있습니다. 데이터베이스의 레코드 크기보다 크거나 같은 recordsize 값을 지정하면 상당한 성능상의 이점을 얻을 수 있습니다. 일반 목적의 파일 시스템에서는 이 등록 정보를 사용하지 않는 것이 좋으며, 이 등록 정보를 사용할 경우 성능에 부정적인 영향을 줄 수 있습니다. 크기는 512바이트에서 128KB 이하의 값 중 모든 값에 대한 2배 값으로 지정해야 합니다. 파일 시스템의 recordsize 값을 변경하면 이후에 만들어지는 파일에만 영향을 줍니다. 기존 파일에는 영향을 주지 않습니다.

이 등록 정보의 약어는 recsize입니다.

volsize 등록 정보

volsize 등록 정보는 볼륨의 논리적 크기를 지정합니다. 기본적으로 볼륨을 만들면 동일 크기의 예약이 설정됩니다. volsize를 변경하면 예약에도 변경 사항이 동일하게 반영됩니다. 이러한 검사는 사용자에게 예상치 않은 동작이 발생하지 않도록 방지하기

위해 사용됩니다. 볼륨에 요청한 것보다 적은 공간이 포함된 경우 볼륨의 사용 방식에 따라 정의되지 않은 동작이 발생하거나 데이터 손상이 일어날 수 있습니다. 또한 볼륨을 사용 중일 때 볼륨 크기를 변경하는 경우, 특히 크기를 줄이는 경우에도 이러한 효과가 발생할 수 있습니다. 볼륨 크기를 조정할 때는 특히 주의가 필요합니다.

권장되는 방법은 아니지만 `-s` 플래그를 `zfs create -V`로 지정하거나 볼륨이 만들어진 후 예약을 변경하여 스파스(sparse) 볼륨을 만들 수 있습니다. **스파스***(sparse)* **볼륨**은 예약이 볼륨 크기와 같지 않은 볼륨입니다. 스파스 볼륨의 경우 `volsize`에 대한 변경 사항이 예약에 반영되지 않습니다.

볼륨 사용에 대한 자세한 내용은 261 페이지 “ZFS 볼륨”을 참조하십시오.

ZFS 사용자 등록 정보

고유 등록 정보 외에도 ZFS에서는 모든 사용자 등록 정보가 지원됩니다. 사용자 등록 정보는 ZFS 동작에 영향을 주지 않지만 이를 사용하여 해당 환경에 필요한 정보를 사용하여 데이터 세트에 주석으로 달 수 있습니다.

사용자 등록 정보 이름은 다음 규칙을 따라야 합니다.

- 고유 등록 정보와 구분될 수 있도록 콜론(':')을 포함해야 합니다.
- 소문자, 숫자 또는 ':', '+','.', '_' 구두점 문자를 포함해야 합니다.
- 사용자 등록 정보 이름의 최대 길이는 256자입니다.

일반적으로 등록 정보 이름은 다음 두 가지 구성 요소로 구분되어야 하지만 이러한 이름 공간이 ZFS에서 강제로 적용되는 것은 아닙니다.

module:property

사용자 등록 정보를 프로그래밍 방식으로 사용할 때는 서로 독립적으로 개발된 두 패키지가 다른 목적으로 동일한 등록 정보 이름을 사용할 수 있는 가능성을 줄이기 위해 등록 정보 이름의 *module* 구성 요소에 대해 예약된 DNS 도메인 이름을 사용하십시오. `com.oracle.`으로 시작되는 등록 정보 이름은 Oracle Corporation에서 사용할 목적으로 예약되어 있습니다.

사용자 등록 정보의 값은 다음 규칙을 따라야 합니다.

- 이러한 값은 항상 상속되며 검증되지 않는 모든 문자열로 구성되어야 합니다.
- 사용자 등록 정보 값의 최대 길이는 1024자입니다.

예를 들면 다음과 같습니다.

```
# zfs set dept:users=finance userpool/user1
# zfs set dept:users=general userpool/user2
# zfs set dept:users=itops userpool/user3
```

`zfs list`, `zfs get`, `zfs set` 등과 같이 등록 정보에 대해 작동하는 모든 명령은 고유 등록 정보와 사용자 등록 정보를 모두 조작하는 데 사용할 수 있습니다.

예를 들면 다음과 같습니다.

```
zfs get -r dept:users userpool
NAME            PROPERTY    VALUE           SOURCE
userpool        dept:users  all             local
userpool/user1  dept:users  finance         local
userpool/user2  dept:users  general         local
userpool/user3  dept:users  itops           local
```

사용자 등록 정보를 지우려면 `zfs inherit` 명령을 사용합니다. 예를 들면 다음과 같습니다.

```
# zfs inherit -r dept:users userpool
```

등록 정보가 어떠한 부모 데이터 세트에도 정의되어 있지 않으면 완전히 제거됩니다.

ZFS 파일 시스템 정보 질의

`zfs list` 명령은 데이터 세트 정보를 보고 질의하기 위한 확장 가능한 방식을 제공합니다. 이 절에서는 기본 질의 및 복합 질의 모두에 대해 설명합니다.

기본 ZFS 정보 나열

`zfs list` 명령을 옵션 없이 사용하여 기본 데이터 세트 정보를 나열할 수 있습니다. 이 명령은 시스템에 있는 모든 데이터 세트의 이름과 해당 `used`, `available`, `referenced` 및 `mountpoint` 등록 정보에 대한 값을 표시합니다. 이러한 등록 정보에 대한 자세한 내용은 169 페이지 “ZFS 등록 정보 소개”를 참조하십시오.

예를 들면 다음과 같습니다.

```
# zfs list
users                  2.00G  64.9G    32K  /users
users/home             2.00G  64.9G    35K  /users/home
users/home/cindy        548K  64.9G   548K  /users/home/cindy
users/home/mark        1.00G  64.9G  1.00G  /users/home/mark
users/home/neil        1.00G  64.9G  1.00G  /users/home/neil
```

또한 이 명령을 사용할 때 명령줄에 데이터 세트 이름을 제공하여 특정 데이터 세트를 표시할 수도 있습니다. `-r` 옵션을 사용하면 해당 데이터 세트의 모든 종속 항목을 반복해서 표시할 수도 있습니다. 예를 들면 다음과 같습니다.

```
# zfs list -t all -r users/home/mark
NAME                       USED  AVAIL  REFER  MOUNTPOINT
users/home/mark           1.00G  64.9G  1.00G  /users/home/mark
users/home/mark@yesterday     0      -  1.00G  -
users/home/mark@today         0      -  1.00G  -
```

파일 시스템의 마운트 지점에 `zfs list` 명령을 사용할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs list /user/home/mark
NAME              USED  AVAIL  REFER  MOUNTPOINT
users/home/mark  1.00G  64.9G  1.00G  /users/home/mark
```

다음 예에서는 tank/home/gina 및 모든 종속 파일 시스템에 대한 기본 정보를 표시하는 방법을 보여줍니다.

```
# zfs list -r users/home/gina
NAME                          USED  AVAIL  REFER  MOUNTPOINT
users/home/gina               2.00G  62.9G    32K  /users/home/gina
users/home/gina/projects      2.00G  62.9G    33K  /users/home/gina/projects
users/home/gina/projects/fs1  1.00G  62.9G  1.00G  /users/home/gina/projects/fs1
users/home/gina/projects/fs2  1.00G  62.9G  1.00G  /users/home/gina/projects/fs2
```

zfs list 명령에 대한 자세한 내용은 zfs(1M)을 참조하십시오.

복잡한 ZFS 질의 만들기

zfs list 결과는 -o, -t 및 -H 옵션을 사용하여 사용자 정의할 수 있습니다.

-o 옵션 및 콤마로 구분된 원하는 등록 정보 목록을 사용하여 등록 정보 값 결과를 사용자 정의할 수 있습니다. 모든 데이터 세트 등록 정보를 유효한 인수로 제공할 수 있습니다. 모든 지원되는 데이터 세트 등록 정보 목록을 보려면 169 페이지 “ZFS 등록 정보 소개”를 참조하십시오. 정의된 등록 정보 외에도 -o 옵션 목록에는 출력 결과에 데이터 세트 이름을 포함되도록 지정하는 name 리터럴이 포함될 수 있습니다.

다음 예제에서는 zfs list를 사용하여 sharenfs 및 mountpoint 등록 정보 값과 함께 데이터 세트 이름을 표시합니다.

```
# zfs list -r -o name,sharenfs,mountpoint users/home
NAME                          SHARENFS  MOUNTPOINT
users/home                    on        /users/home
users/home/cindy              on        /users/home/cindy
users/home/gina               on        /users/home/gina
users/home/gina/projects      on        /users/home/gina/projects
users/home/gina/projects/fs1  on        /users/home/gina/projects/fs1
users/home/gina/projects/fs2  on        /users/home/gina/projects/fs2
users/home/mark               on        /users/home/mark
users/home/neil               on        /users/home/neil
```

-t 옵션을 사용하면 표시할 데이터 세트의 유형을 지정할 수 있습니다. 유효한 유형은 다음 표에 설명되어 있습니다.

표 5–2 ZFS 객체 유형

유형	설명
filesystem	파일 시스템 및 복제본
volume	볼륨

표 5–2 ZFS 객체 유형 (계속)

유형	설명
share	파일 시스템 공유
snapshot	스냅샷

-t 옵션은 표시할 데이터 세트의 유형이 콤마로 구분된 목록을 받아 들입니다. 다음 예제에서는 -t 및 -o 옵션을 동시에 사용하여 모든 파일 시스템의 이름 및 used 등록 정보를 표시합니다.

```
# zfs list -r -t filesystem -o name,used users/home
NAME                          USED
users/home                   4.00G
users/home/cindy              548K
users/home/gina              2.00G
users/home/gina/projects     2.00G
users/home/gina/projects/fs1  1.00G
users/home/gina/projects/fs2  1.00G
users/home/mark              1.00G
users/home/neil              1.00G
```

-H 옵션을 사용하여 만들어진 출력 결과에서 zfs list 헤더를 생략할 수 있습니다. -H 옵션을 사용하면 모든 공백이 탭 문자로 교체됩니다. 이 옵션은 스크립팅과 같이 구문 분석 가능한 출력 결과가 필요한 경우에 유용합니다. 다음 예제에서는 -H 옵션으로 zfs list 명령을 사용했을 때 생성되는 출력 결과를 보여 줍니다.

```
# zfs list -r -H -o name users/home
users/home
users/home/cindy
users/home/gina
users/home/gina/projects
users/home/gina/projects/fs1
users/home/gina/projects/fs2
users/home/mark
users/home/neil
```

ZFS 등록 정보 관리

데이터 세트 등록 정보는 zfs 명령의 set, inherit 및 get 하위 명령을 통해 관리됩니다.

- 184 페이지 “ZFS 등록 정보 설정”
- 184 페이지 “ZFS 등록 정보 상속”
- 185 페이지 “ZFS 등록 정보 질의”

ZFS 등록 정보 설정

zfs set 명령을 사용하여 설정 가능한 모든 데이터 세트 등록 정보를 수정할 수 있습니다. 또는 데이터 세트를 만들 때 zfs create 명령을 사용하여 등록 정보를 설정할 수 있습니다. 설정 가능한 데이터 세트 등록 정보의 목록은 177 페이지 "설정 가능한 ZFS 고유 등록 정보"를 참조하십시오.

zfs set 명령은 *property=value*와 데이터 세트 이름의 형식으로 된 등록 정보/값 시퀀스를 받아 들입니다. 각 zfs set를 호출할 때는 등록 정보를 하나만 설정 또는 수정할 수 있습니다.

다음 예제에서는 tank/home에 대해 atime 등록 정보를 off로 설정합니다.

```
# zfs set atime=off tank/home
```

또한 파일 시스템을 만들 때 모든 파일 시스템 등록 정보를 설정할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs create -o atime=off tank/home
```

다음과 같이 이해하기 쉬운 접미어 BKMGTPEZ를 사용하여 숫자 등록 정보 값(증분 크기)을 지정할 수 있습니다. 이러한 접미어 다음에는 바이트를 나타내는 선택적인 b를 사용할 수 있지만 이미 바이트를 나타내는 B 접미어는 예외입니다. 다음 네 번의 zfs set 호출은 quota 등록 정보를 users/home/mark 파일 시스템에서 20GB 값으로 설정하는 숫자 표현식과 동일합니다.

```
# zfs set quota=20G users/home/mark
# zfs set quota=20g users/home/mark
# zfs set quota=20GB users/home/mark
# zfs set quota=20gb users/home/mark
```

완전히 가득 찬 파일 시스템에서 등록 정보를 설정하려고 시도하면 다음과 유사한 메시지가 표시됩니다.

```
# zfs set quota=20gb users/home/mark
cannot set property for '/users/home/mark': out of space
```

숫자가 아닌 등록 정보의 값은 대소문자를 구분하며 mountpoint 및 sharenfs를 제외하고 소문자여야 합니다. 이러한 등록 정보의 값은 대소문자가 혼합될 수 있습니다.

zfs set 명령에 대한 자세한 내용은 zfs(1M)를 참조하십시오.

ZFS 등록 정보 상속

쿼터 또는 예약이 종속된 파일 시스템에 명시적으로 설정되지 않은 경우 설정 가능한 모든 등록 정보는 쿼터 및 예약을 제외하고 부모 파일 시스템에서 해당 값을 상속합니다. 상위 항목에 상속된 등록 정보에 대해 설정된 명시적인 값이 없으면 등록 정보의

기본값이 사용됩니다. zfs inherit 명령을 사용하여 등록 정보 값을 지울 수 있으므로 부모 파일 시스템으로부터 값을 상속할 수 있습니다.

다음 예제에서는 zfs set 명령을 사용하여 tank/home/jeff 파일 시스템에 대한 압축을 설정합니다. 그런 다음 zfs inherit를 사용하여 compression 등록 정보를 지워서 해당 등록 정보가 기본값인 off를 상속하도록 만듭니다. home 또는 tank에 compression 등록 정보가 로컬로 설정되지 않으므로 기본값이 사용됩니다. 두 가지 항목 모두 압축이 사용으로 설정되어 있으면 가장 가까운 상위 요소에 설정된 값이 사용됩니다(이 예제의 경우 home).

```
# zfs set compression=on tank/home/jeff
# zfs get -r compression tank/home
NAME                  PROPERTY     VALUE     SOURCE
tank/home             compression  off       default
tank/home/eric        compression  off       default
tank/home/eric@today  compression  -         -
tank/home/jeff        compression  on        local
# zfs inherit compression tank/home/jeff
# zfs get -r compression tank/home
NAME                  PROPERTY     VALUE     SOURCE
tank/home             compression  off       default
tank/home/eric        compression  off       default
tank/home/eric@today  compression  -         -
tank/home/jeff        compression  off       default
```

-r 옵션을 지정하면 inherit 하위 명령이 반복해서 적용됩니다. 다음 예제에서 명령을 수행하면 tank/home 및 이 파일 시스템이 포함할 수 있는 모든 종속 항목에서 compression 등록 정보에 대한 값을 상속합니다.

```
# zfs inherit -r compression tank/home
```

주 – -r 옵션을 사용하면 모든 종속 파일 시스템의 현재 등록 정보 설정이 지워집니다.

zfs inherit 명령에 대한 자세한 내용은 zfs(1M)을 참조하십시오.

ZFS 등록 정보 질의

등록 정보 값을 질의하는 가장 간단한 방법은 zfs list 명령을 사용하는 것입니다. 자세한 내용은 181 페이지 “기본 ZFS 정보 나열”을 참조하십시오. 하지만 복잡한 질의 및 스크립팅의 경우 zfs get 명령을 사용하여 사용자 정의된 형식으로 보다 자세한 정보를 제공하십시오.

zfs get 명령을 사용하여 모든 데이터 세트 등록 정보를 검색할 수 있습니다. 다음 예제에서는 데이터 세트에서 단일 등록 정보 값을 검색하는 방법을 보여 줍니다.

```
# zfs get checksum tank/ws
NAME            PROPERTY       VALUE                    SOURCE
tank/ws         checksum       on                       default
```

네번째 열인 SOURCE는 이 등록 정보 값의 원본을 나타냅니다. 다음 표에서는 가능한 소스 값을 정의합니다.

표 5–3 가능한 SOURCE 값(zfs get 명령)

소스 값	설명
default	이 등록 정보 값은 이 데이터 세트 또는 해당 상위 항목에 대해 명시적으로 설정되지 않습니다. 이 등록 정보의 기본 값을 사용하는 중입니다.
inherited from *dataset-name*	이 등록 정보 값은 *dataset-name*에 지정된 부모 데이터 세트로부터 상속됩니다.
local	이 등록 정보 값은 zfs set를 사용하여 이 데이터 세트에 대해 명시적으로 설정되었습니다.
temporary	이 등록 정보 값은 zfs mount - o 옵션을 사용하여 설정되었으며 마운트 기간 동안만 유효합니다. 임시 마운트 지점 등록 정보에 대한 자세한 내용은 192 페이지 “임시 마운트 등록 정보 사용”을 참조하십시오.
-(없음)	이 등록 정보는 읽기 전용입니다. 해당 값은 ZFS에 의해 만들어집니다.

특수 키워드인 all을 사용하여 모든 데이터 세트 등록 정보 값을 검색할 수 있습니다. 다음 예제에서는 all 키워드가 사용됩니다.

주 – Oracle Solaris 10 릴리스에서는 Oracle Solaris SMB 서비스가 지원되지 않기 때문에 casesensitivity, nbmand, normalization, sharesmb, utf8only 및 vscan 등록 정보는 Oracle Solaris 10 릴리스에서 완전히 구동 가능하지 않습니다.

```
# zfs get all tank/home
NAME       PROPERTY              VALUE                  SOURCE
tank/home  type                  filesystem             -
tank/home  creation              Mon Dec  3 13:10 2012  -
tank/home  used                  291K                   -
tank/home  available             58.7G                  -
tank/home  referenced            291K                   -
tank/home  compressratio         1.00x                  -
tank/home  mounted               yes                    -
tank/home  quota                 none                   default
tank/home  reservation           none                   default
tank/home  recordsize            128K                   default
tank/home  mountpoint            /tank/home             default
tank/home  sharenfs              off                    default
tank/home  checksum              on                     default
tank/home  compression           off                    default
tank/home  atime                 on                     default
tank/home  devices               on                     default
tank/home  exec                  on                     default
```

```
tank/home  setuid               on                     default
tank/home  readonly             off                    default
tank/home  zoned                off                    default
tank/home  snapdir              hidden                 default
tank/home  aclmode              discard                default
tank/home  aclinherit           restricted             default
tank/home  canmount             on                     default
tank/home  shareiscsi           off                    default
tank/home  xattr                on                     default
tank/home  copies               1                      default
tank/home  version              5                      -
tank/home  utf8only             off                    -
tank/home  normalization        none                   -
tank/home  casesensitivity      mixed                  -
tank/home  vscan                off                    default
tank/home  nbmand               off                    default
tank/home  sharesmb             off                    default
tank/home  refquota             none                   default
tank/home  refreservation       none                   default
tank/home  primarycache         all                    default
tank/home  secondarycache       all                    default
tank/home  usedbysnapshots      0                      -
tank/home  usedbydataset        291K                   -
tank/home  usedbychildren       0                      -
tank/home  usedbyrefreservation 0                      -
tank/home  logbias              latency                default
tank/home  sync                 standard               default
tank/home  rekeydate            -                      default
tank/home  rstchown             on                     default
```

`zfs get`의 `-s` 옵션을 사용하면 소스 유형별로 표시할 등록 정보를 지정할 수 있습니다. 이 옵션은 원하는 소스 유형을 나타내는 콤마로 구분된 목록을 받아 들입니다. 지정된 소스 유형의 등록 정보만 표시됩니다. 유효한 소스 유형은 `local`, `default`, `inherited`, `temporary` 및 `none`입니다. 다음 예제에서는 `tank/ws`에 로컬로 설정된 모든 등록 정보를 보여 줍니다.

```
# zfs get -s local all tank/ws
NAME     PROPERTY              VALUE                  SOURCE
tank/ws  compression           on                     local
```

위 옵션 중 하나를 `-r` 옵션과 결합하여 지정한 파일 시스템의 모든 자식에 지정된 등록 정보를 순환적으로 표시할 수 있습니다. 다음 예에서는 `tank/home` 내의 모든 파일 시스템의 모든 임시 등록 정보가 순환적으로 표시됩니다.

```
# zfs get -r -s temporary all tank/home
NAME             PROPERTY       VALUE                      SOURCE
tank/home          atime          off                      temporary
tank/home/jeff     atime          off                      temporary
tank/home/mark     quota          20G                      temporary
```

대상 파일 시스템을 지정하지 않고 `zfs get` 명령을 사용하여 등록 정보 값을 질의할 수 있습니다. 즉, 명령이 모든 풀 또는 파일 시스템에서 수행됩니다. 예를 들면 다음과 같습니다.

```
# zfs get -s local all
tank/home               atime          off                    local
tank/home/jeff          atime          off                    local
tank/home/mark          quota          20G                    local
```

zfs get 명령에 대한 자세한 내용은 zfs(1M)를 참조하십시오.

스크립팅을 위한 ZFS 등록 정보 질의

zfs get 명령에는 스크립팅을 위해 디자인된 -H 및 -o 옵션이 지원됩니다. -H 옵션을 사용하면 헤더 정보를 생략하고 공백을 탭 문자로 바꿀 수 있습니다. 공백만 사용하면 데이터를 쉽게 구문 분석할 수 있습니다. 다음과 같은 방식으로 -o 옵션을 사용하여 출력 결과를 사용자 정의할 수 있습니다.

- 리터럴 name은 169 페이지 "ZFS 등록 정보 소개" 절에 정의된 대로 콤마로 구분된 등록 정보 목록에 사용할 수 있습니다.
- 공백과 인수(콤마로 구분된 등록 정보 목록)가 뒤에 이어서 출력되는 리터럴 필드 name, value, property 및 source의 콤마로 구분된 목록입니다.

다음 예제에서는 zfs get의 -H 및 -o 옵션을 사용하여 단일 값을 검색하는 방법을 보여 줍니다.

```
# zfs get -H -o value compression tank/home
on
```

-p 옵션은 해당 값을 정확한 숫자 값으로 보고합니다. 예를 들어, 1MB는 1000000으로 보고됩니다. 이 옵션은 다음과 같이 사용할 수 있습니다.

```
# zfs get -H -o value -p used tank/home
182983742
```

이전 옵션과 함께 -r 옵션을 사용하여 모든 종속 항목에 대해 요청된 값을 반복해서 검색할 수 있습니다. 다음 예에서는 -H, -o 및 -r 옵션을 사용하여 export/home 및 종속 항목의 파일 시스템 이름 및 used 등록 정보 값을 검색하고 헤더 출력은 생략합니다.

```
# zfs get -H -o name,value -r used export/home
```

ZFS 파일 시스템 마운트

이 절에서는 ZFS에서 파일 시스템을 마운트하는 방법에 대해 설명합니다.

- 189 페이지 "ZFS 마운트 지점 관리"
- 191 페이지 "ZFS 파일 시스템 마운트"
- 192 페이지 "임시 마운트 등록 정보 사용"
- 192 페이지 "ZFS 파일 시스템 마운트 해제"

ZFS 마운트 지점 관리

기본적으로 ZFS 파일 시스템은 만들어질 때 자동으로 마운트됩니다. 이 절에 설명된 대로 파일 시스템에 대한 특정 마운트 지점 동작을 결정할 수 있습니다.

`zpool create`의 `-m` 옵션을 사용하여 생성 시 풀의 파일 시스템에 대한 기본 마운트 지점을 설정할 수도 있습니다. 풀 만들기에 대한 자세한 내용은 49 페이지 “ZFS 저장소 풀 만들기”를 참조하십시오.

모든 ZFS 파일 시스템은 부트 시 SMF(서비스 관리 기능)의 `svc://system/filesystem/local` 서비스를 사용하여 ZFS에 의해 마운트됩니다. 파일 시스템은 */path*에 마운트됩니다. 여기서 *path*는 파일 시스템의 이름입니다.

`zfs set` 명령을 사용해서 `mountpoint` 등록 정보를 특정 경로로 설정하여 기본 마운트 지점을 대체할 수 있습니다. ZFS는 지정된 마운트 지점을 만들고 필요한 경우 연관된 파일 시스템을 자동으로 마운트합니다.

ZFS 파일 시스템은 사용자가 `/etc/vfstab` 파일을 편집할 필요 없이 부트 시에 자동으로 마운트됩니다.

`mountpoint` 등록 정보는 상속됩니다. 예를 들어, `pool/home`에서 `mountpoint` 등록 정보가 `/export/stuff`로 설정된 경우 `pool/home/user`는 해당 `mountpoint` 등록 정보 값에 대해 `/export/stuff/user`를 상속합니다.

파일 시스템이 마운트되지 않도록 방지하려면 `mountpoint` 등록 정보를 `none`으로 설정합니다. 또한 `canmount` 등록 정보를 사용하여 파일 시스템을 마운트할 수 있는지 여부를 제어할 수 있습니다. `canmount` 등록 정보에 대한 자세한 내용은 179 페이지 “canmount 등록 정보”를 참조하십시오.

또한 `zfs set`를 사용하여 `mountpoint` 등록 정보를 `legacy`로 설정해서 레거시 마운트 인터페이스를 통해 파일 시스템을 명시적으로 관리할 수도 있습니다. 이렇게 하면 ZFS가 파일 시스템을 자동으로 마운트하고 관리하는 것을 방지할 수 있습니다. `mount` 및 `umount` 명령을 포함하는 레거시 도구와 `/etc/vfstab` 파일을 대신 사용해야 합니다. 레거시 마운트에 대한 자세한 내용은 190 페이지 “레거시 마운트 지점”을 참조하십시오.

자동 마운트 지점

- `mountpoint` 등록 정보를 `legacy` 또는 `none`에서 특정 경로로 변경하면 ZFS가 파일 시스템을 자동으로 마운트합니다.
- ZFS에서 파일 시스템을 관리하지만 현재 마운트 해제되어 있고, `mountpoint` 등록 정보가 변경된 경우 파일 시스템이 마운트 해제된 상태로 유지됩니다.

`mountpoint` 등록 정보가 `legacy`가 아닌 파일 시스템은 모두 ZFS에서 관리됩니다. 다음 예에서는 마운트 지점이 ZFS에서 자동으로 관리되는 파일 시스템이 생성됩니다.

```
# zfs create pool/filesystem
# zfs get mountpoint pool/filesystem
NAME              PROPERTY       VALUE                      SOURCE
```

```
pool/filesystem  mountpoint    /pool/filesystem          default
# zfs get mounted pool/filesystem
NAME             PROPERTY      VALUE                     SOURCE
pool/filesystem  mounted       yes                       -
```

또한 다음 예제에 표시된 것처럼 mountpoint 등록 정보를 명시적으로 설정할 수도 있습니다.

```
# zfs set mountpoint=/mnt pool/filesystem
# zfs get mountpoint pool/filesystem
NAME             PROPERTY      VALUE                     SOURCE
pool/filesystem  mountpoint    /mnt                      local
# zfs get mounted pool/filesystem
NAME             PROPERTY      VALUE                     SOURCE
pool/filesystem  mounted       yes                       -
```

mountpoint 등록 정보가 변경되면 파일 시스템이 이전 마운트 지점에서 자동으로 마운트 해제되고 새 마운트 지점에 재마운트됩니다. 마운트 지점 디렉토리는 필요에 따라 만들어집니다. 파일 시스템이 활성 상태여서 ZFS가 파일 시스템을 마운트 해제할 수 없는 경우, 오류가 보고되며, 수동 마운트 해제를 강제로 수행해야 합니다.

레거시 마운트 지점

mountpoint 등록 정보를 legacy로 설정하여 레거시 도구로 ZFS 파일 시스템을 관리할 수 있습니다. 레거시 파일 시스템은 mount 및 umount 명령과 /etc/vfstab 파일을 통해 관리해야 합니다. ZFS는 부트 시 레거시 파일 시스템을 자동으로 마운트하지 않으며, ZFS mount 및 umount 명령은 이 유형의 파일 시스템에서 작동하지 않습니다. 다음 예에서는 ZFS 파일 시스템을 레거시 모드로 설정 및 관리하는 방법을 보여줍니다.

```
# zfs set mountpoint=legacy tank/home/eric
# mount -F zfs tank/home/eschrock /mnt
```

부트 시에 레거시 파일 시스템을 자동으로 마운트하려면 /etc/vfstab 파일에 항목을 추가해야 합니다. 다음 예제에서는 /etc/vfstab 파일에 있는 항목이 어떻게 표시되는지를 보여 줍니다.

```
#device         device        mount           FS      fsck    mount   mount
#to mount       to fsck       point           type    pass    at boot options
#

tank/home/eric  -        /mnt           zfs        -        yes        -
```

fsck 명령을 ZFS 파일 시스템에 적용할 수 없으므로 device to fsck 및 fsck pass 항목은 -로 설정됩니다. ZFS 데이터 무결성에 대한 자세한 내용은 25 페이지 “트랜잭션 개념”을 참조하십시오.

ZFS 파일 시스템 마운트

ZFS는 파일 시스템을 만들 때 또는 시스템이 부트될 때 파일 시스템을 자동으로 마운트합니다. `zfs mount` 명령은 마운트 옵션을 변경해야 하거나 파일 시스템을 명시적으로 마운트하거나 마운트 해제해야 할 경우에만 사용해야 합니다.

인수 없이 `zfs mount` 명령을 실행하면 ZFS에서 관리되는 현재 마운트된 모든 파일 시스템이 표시됩니다. 관리되는 레거시 마운트 지점은 표시되지 않습니다. 예를 들면 다음과 같습니다.

```
# zfs mount | grep tank/home
zfs mount | grep tank/home
tank/home                       /tank/home
tank/home/jeff                  /tank/home/jeff
```

`-a` 옵션을 사용하면 ZFS에서 관리되는 모든 파일 시스템이 마운트됩니다. 관리되는 레거시 파일 시스템은 마운트되지 않습니다. 예를 들면 다음과 같습니다.

```
# zfs mount -a
```

기본적으로 ZFS는 비어 있지 않은 디렉토리에서 마운트를 수행할 수 없습니다. 예를 들면 다음과 같습니다.

```
# zfs mount tank/home/lori
cannot mount 'tank/home/lori': filesystem already mounted
```

레거시 마운트 지점은 레거시 도구를 통해 관리되어야 합니다. ZFS 도구를 사용하려고 시도하면 오류가 발생합니다. 예를 들면 다음과 같습니다.

```
# zfs mount tank/home/bill
cannot mount 'tank/home/bill': legacy mountpoint
use mount(1M) to mount this filesystem
# mount -F zfs tank/home/billm
```

파일 시스템이 마운트되면 파일 시스템과 연결된 등록 정보 값을 기준으로 하는 마운트 옵션 세트가 사용됩니다. 등록 정보와 마운트 옵션 간의 상관 관계는 다음과 같습니다.

표 5–4 ZFS 마운트 관련 등록 정보 및 마운트 옵션

등록 정보	마운트 옵션
atime	atime/noatime
devices	devices/nodevices
exec	exec/noexec
nbmand	nbmand/nonbmand
readonly	ro/rw
setuid	setuid/nosetuid

표 5–4 ZFS 마운트 관련 등록 정보 및 마운트 옵션 *(계속)*

등록 정보	마운트 옵션
xattr	xattr/noaxttr

마운트 옵션 nosuid는 nodevices,nosetuid에 대한 별칭입니다.

임시 마운트 등록 정보 사용

앞의 절에서 설명된 마운트 옵션 중 하나라도 zfs mount 명령에서 -o 옵션을 사용하여 명시적으로 설정된 경우 연관된 등록 정보 값이 임시로 대체됩니다. 이러한 등록 정보 값은 zfs get 명령에서 temporary로 보고되며 파일 시스템이 마운트 해제되면 원래 값으로 되돌아갑니다. 파일 시스템이 마운트된 동안 등록 정보 값을 변경하면 변경 사항이 즉시 적용되어 모든 임시 설정을 대체합니다.

다음 예에서는 읽기 전용 마운트 옵션이 tank/home/neil 파일 시스템에 임시로 설정됩니다. 파일 시스템은 마운트 해제된 것으로 가정합니다.

```
# zfs mount -o ro users/home/neil
```

현재 마운트된 파일 시스템에서 등록 정보 값을 임시로 변경하려면 특수한 remount 옵션을 사용해야 합니다. 다음 예제에서는 현재 마운트된 파일 시스템에 대해 atime 등록 정보가 임시로 off로 변경됩니다.

```
# zfs mount -o remount,noatime users/home/neil
NAME             PROPERTY  VALUE  SOURCE
users/home/neil  atime     off    temporary
# zfs get atime users/home/perrin
```

zfs mount 명령에 대한 자세한 내용은 zfs(1M)을 참조하십시오.

ZFS 파일 시스템 마운트 해제

zfs unmount 하위 명령을 사용하여 ZFS 파일 시스템을 마운트 해제할 수 있습니다. unmount 명령은 마운트 지점 또는 파일 시스템 이름을 인수로 받아 들일 수 있습니다.

다음 예제에서 파일 시스템은 해당 파일 시스템 이름으로 마운트됩니다.

```
# zfs unmount users/home/mark
```

다음 예제에서 파일 시스템은 해당 마운트 지점에 의해 마운트 해제됩니다.

```
# zfs unmount /users/home/mark
```

파일 시스템이 사용 중인 경우 unmount 명령이 실패합니다. 파일 시스템을 강제로 마운트 해제하려면 -f 옵션을 사용할 수 있습니다. 컨텐츠가 현재 사용 중인 경우 파일 시스템을 강제로 마운트 해제할 때 주의가 필요합니다. 예상치 않은 응용 프로그램 동작이 발생할 수 있습니다.

```
# zfs unmount tank/home/eric
cannot unmount '/tank/home/eric': Device busy
# zfs unmount -f tank/home/eric
```

이전 버전과의 호환성을 제공하기 위해 umount 명령을 사용하여 ZFS 파일 시스템을 마운트 해제할 수 있습니다. 예를 들면 다음과 같습니다.

```
# umount /tank/home/bob
```

zfs umount 명령에 대한 자세한 내용은 zfs(1M)을 참조하십시오.

ZFS 파일 시스템 공유 및 공유 해제

ZFS는 sharenfs 등록 정보를 설정하여 파일 시스템을 자동으로 공유할 수 있습니다. 이 등록 정보를 사용할 경우 새 파일 시스템을 공유할 때 /etc/dfs/dfstab 파일을 수정할 필요가 없습니다. sharenfs 등록 정보는 share 명령에 전달할 콤마로 구분된 옵션 목록입니다. 값 on은 누구나에게 read/write 권한을 제공하는 기본 공유 옵션에 대한 별칭입니다. 값 off는 파일 시스템이 ZFS에서 관리되지 않으며 /etc/dfs/dfstab 파일과 같은 기존 방식을 통해 파일 시스템을 공유할 수 있음을 나타냅니다. sharenfs 등록 정보가 off가 아닌 모든 파일 시스템은 부트 중에 공유됩니다.

공유 제어 개념

기본적으로 모든 파일 시스템이 공유 해제됩니다. 새 파일 시스템을 공유하려면 다음과 비슷한 zfs set 구문을 사용하십시오.

```
# zfs set sharenfs=on tank/home/eric
```

상속된 등록 정보가 off가 아니면 파일 시스템 생성 시 sharenfs 등록 정보가 상속되고 파일 시스템이 자동으로 공유됩니다. 예를 들면 다음과 같습니다.

```
# zfs set sharenfs=on tank/home
# zfs create tank/home/bill
# zfs create tank/home/mark
# zfs set sharenfs=ro tank/home/bob
```

tank/home/bill 및 tank/home/mark는 tank/home으로부터 sharenfs 등록 정보를 상속하기 때문에 처음에 쓰기 가능으로 공유됩니다. 등록 정보가 ro(읽기 전용)로 설정된 후 tank/home/mark는 tank/home에 대해 설정된 sharenfs 등록 정보에 관계없이 읽기 전용으로 공유됩니다.

ZFS 파일 시스템 공유 해제

대부분의 파일 시스템은 부트, 만들기 및 삭제 중에 자동으로 공유 또는 공유 해제되지만 일부 경우에는 파일 시스템을 명시적으로 공유 해제해야 합니다. 이렇게 하려면 `zfs unshare` 명령을 사용하십시오. 예를 들면 다음과 같습니다.

```
# zfs unshare tank/home/mark
```

이 명령은 `tank/home/mark` 파일 시스템을 공유 해제합니다. 시스템에서 모든 ZFS 파일 시스템을 공유 해제하려면 `-a` 옵션을 사용해야 합니다.

```
# zfs unshare -a
```

ZFS 파일 시스템 공유

대부분의 일반 작업의 경우 부트 및 생성 시 파일 시스템을 공유하는 것과 관련된 ZFS의 자동 동작만으로도 충분합니다. 특정 이유로 인해 파일 시스템을 공유 해제한 경우 `zfs share` 명령을 사용하여 다시 공유할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs share tank/home/mark
```

또한 `-a` 옵션을 사용하여 시스템에서 모든 ZFS 파일 시스템을 공유할 수도 있습니다.

```
# zfs share -a
```

레거시 공유 동작

`sharenfs` 등록 정보가 `off`로 설정되어 있으면 ZFS가 어떤 경우에도 파일 시스템에 대한 공유 또는 공유 해제를 시도하지 않습니다. 이 값을 사용하면 `/etc/dfs/dfstab` 파일과 같은 기존 방식을 통해 파일 시스템 공유를 관리할 수 있습니다.

레거시 `mount` 명령과 달리 레거시 `share` 및 `unshare` 명령은 ZFS 파일 시스템에서 계속 사용할 수 있습니다. 따라서 `sharenfs` 등록 정보의 옵션과 다른 옵션을 사용하여 파일 시스템을 수동으로 공유할 수 있습니다. 이 관리 모델은 사용하지 않는 것이 좋습니다. 완전히 ZFS를 사용하거나 `/etc/dfs/dfstab` 파일만 사용하여 NFS 공유를 관리하도록 선택하십시오. ZFS 관리 모델은 기존 모델보다 간단하고 작업도 덜 수행하도록 설계되었습니다.

ZFS 쿼터 및 예약 설정

`quota` 등록 정보를 사용하여 파일 시스템이 사용할 수 있는 디스크 공간에 대한 한도를 설정할 수 있습니다. 또한 `reservation` 등록 정보를 사용하여 지정된 디스크 공간을 파일 시스템에 사용할 수 있도록 보장할 수 있습니다. 두 가지 등록 정보는 모두 등록 정보가 설정된 파일 시스템 및 해당 파일 시스템의 모든 종속 항목에 적용됩니다.

즉, 쿼터가 `tank/home` 파일 시스템에 설정된 경우 `tank/home` 및 **모든 종속 항목**에서 사용되는 총 디스크 공간은 쿼터를 초과할 수 없습니다. 이와 비슷하게 `tank/home`에

예약이 제공된 경우 tank/home 및 **모든 종속 항목**은 해당 예약으로부터 공간을 가져옵니다. 파일 시스템 및 모든 종속 항목에서 사용되는 디스크 공간은 used 등록 정보로 보고됩니다.

refquota 및 refreservation 등록 정보는 스냅샷 및 복제본과 같은 종속 항목에서 소비되는 디스크 공간을 고려하지 않고 파일 시스템 공간을 관리하는 데 사용됩니다.

이 Solaris 릴리스에서는 특정 사용자 또는 그룹이 소유하는 파일에서 소비되는 디스크 공간에 대해 **사용자** 또는 **그룹** 쿼터를 설정할 수 있습니다. 사용자 및 그룹 쿼터 등록 정보는 볼륨, 파일 시스템 버전 4 이전의 파일 시스템, 풀 버전 15 이전의 풀에 설정할 수 없습니다.

파일 시스템 관리에 가장 효과적인 쿼터 및 예약 기능을 결정할 때는 다음과 같은 사항을 고려하십시오.

- quota 및 reservation 등록 정보는 파일 시스템 및 종속 항목에서 사용되는 디스크 공간을 관리하는 데 편리합니다.
- refquota 및 refreservation 등록 정보는 파일 시스템에서 사용되는 디스크 공간을 관리하는 데 적합합니다.
- refquota 또는 refreservation 등록 정보는 quota 또는 reservation 등록 정보보다 높게 설정해도 효과가 없습니다. quota 또는 refquota 등록 정보를 설정할 경우 어느 하나의 값을 초과하는 작업이 시도될 경우 작업이 실패합니다. refquota보다 큰 quota에서 초과가 발생할 수 있습니다. 예를 들어 일부 스냅샷 블록이 수정된 경우 refquota가 초과되기 전에 quota가 실제로 초과될 수 있습니다.
- 사용자 및 그룹 쿼터를 사용하면 대학 환경과 같이 사용자 계정이 많은 디스크 공간을 보다 쉽게 관리할 수 있습니다.

쿼터 및 예약 설정에 대한 자세한 내용은 195 페이지 “ZFS 파일 시스템에 대한 쿼터 설정” 및 198 페이지 “ZFS 파일 시스템에 대한 예약 설정”을 참조하십시오.

ZFS 파일 시스템에 대한 쿼터 설정

ZFS 파일 시스템에서 쿼터는 zfs set 및 zfs get 명령을 사용하여 설정하고 표시할 수 있습니다. 다음 예제에서는 tank/home/jeff에 10GB 쿼터를 설정합니다.

```
# zfs set quota=10G tank/home/jeff
# zfs get quota tank/home/jeff
NAME            PROPERTY  VALUE  SOURCE
tank/home/jeff  quota     10G    local
```

쿼터는 zfs list 및 df 명령의 출력 결과에도 영향을 줍니다. 예를 들면 다음과 같습니다.

```
# zfs list -r tank/home
NAME                USED  AVAIL  REFER  MOUNTPOINT
tank/home          1.45M  66.9G    36K  /tank/home
tank/home/eric      547K  66.9G   547K  /tank/home/eric
```

```
tank/home/jeff      322K  10.0G   291K  /tank/home/jeff
tank/home/jeff/ws    31K  10.0G    31K  /tank/home/jeff/ws
tank/home/lori      547K  66.9G   547K  /tank/home/lori
tank/home/mark       31K  66.9G    31K  /tank/home/mark
# df -h /tank/home/jeff
Filesystem            Size  Used Avail Use% Mounted on
tank/home/jeff         10G  306K   10G   1% /tank/home/jeff
```

tank/home에서 66.9GB의 디스크 공간을 사용할 수 있더라도 tank/home/jeff에 대한 쿼터로 인해 tank/home/jeff 및 tank/home/jeff/ws에는 각각 10GB의 디스크 공간만 사용할 수 있습니다.

파일 시스템에서 현재 사용 중인 공간보다 적은 양으로 쿼터를 설정할 수는 없습니다. 예를 들면 다음과 같습니다.

```
# zfs set quota=10K tank/home/jeff
cannot set property for 'tank/home/jeff':
size is less than current used or reserved space
```

파일 시스템에서 파일 시스템이 사용할 수 있는 디스크 공간을 제한하는 refquota를 설정할 수 있습니다. 이러한 하드 한계에는 종속 항목에서 소비되는 디스크 공간이 포함되지 않습니다. 예를 들어, studentA의 10GB 쿼터는 스냅샷에서 소비되는 공간의 영향을 받지 않습니다.

```
# zfs set refquota=10g students/studentA
# zfs list -t all -r students
NAME                         USED  AVAIL  REFER  MOUNTPOINT
students                     150M  66.8G    32K  /students
students/studentA            150M  9.85G   150M  /students/studentA
students/studentA@yesterday      0      -   150M  -
# zfs snapshot students/studentA@today
# zfs list -t all -r students
students                     150M  66.8G    32K  /students
students/studentA            150M  9.90G   100M  /students/studentA
students/studentA@yesterday  50.0M      -   150M  -
students/studentA@today          0      -   100M  -
```

추가 편의를 위해 스냅샷에서 사용되는 디스크 공간을 쉽게 관리할 수 있도록 파일 시스템에 또 다른 쿼터를 설정할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs set quota=20g students/studentA
# zfs list -t all -r students
NAME                         USED  AVAIL  REFER  MOUNTPOINT
students                     150M  66.8G    32K  /students
students/studentA            150M  9.90G   100M  /students/studentA
students/studentA@yesterday  50.0M      -   150M  -
students/studentA@today          0      -   100M  -
```

이 시나리오에서 studentA는 refquota(10GB) 하드 한계에 도달할 수 있지만 스냅샷이 존재하더라도 studentA가 파일을 제거하여 공간을 복구할 수 있습니다.

앞의 예제에서는 두 쿼터(10GB와 20GB) 중 작은 쿼터가 zfs list 출력 결과에 표시되었습니다. 두 쿼터의 값을 보려면 zfs get 명령을 사용합니다. 예를 들면 다음과 같습니다.

```
# zfs get refquota,quota students/studentA
NAME               PROPERTY  VALUE              SOURCE
students/studentA  refquota  10G                local
students/studentA  quota     20G                local
```

ZFS 파일 시스템에서 사용자 및 그룹 쿼터 설정

각각 `zfs userquota` 또는 `zfs groupquota` 명령을 사용하여 사용자 쿼터 또는 그룹 쿼터를 설정할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs create students/compsci
# zfs set userquota@student1=10G students/compsci
# zfs create students/labstaff
# zfs set groupquota@labstaff=20GB students/labstaff
```

현재 사용자 쿼터 또는 그룹 쿼터를 다음과 같이 표시합니다.

```
# zfs get userquota@student1 students/compsci
NAME               PROPERTY            VALUE               SOURCE
students/compsci   userquota@student1  10G                 local
# zfs get groupquota@labstaff students/labstaff
NAME               PROPERTY             VALUE                SOURCE
students/labstaff  groupquota@labstaff  20G                    local
```

다음 등록 정보를 질의하여 일반적인 사용자 디스크 공간 사용량 또는 그룹 디스크 공간 사용량을 표시할 수 있습니다.

```
# zfs userspace students/compsci
TYPE        NAME      USED  QUOTA
POSIX User  root      350M   none
POSIX User  student1  426M    10G
# zfs groupspace students/labstaff
TYPE         NAME      USED  QUOTA
POSIX Group  labstaff  250M    20G
POSIX Group  root      350M   none
```

개별 사용자 또는 그룹 디스크 공간 사용량을 식별하려면 다음 등록 정보를 질의합니다.

```
# zfs get userused@student1 students/compsci
NAME               PROPERTY           VALUE              SOURCE
students/compsci   userused@student1  550M               local
# zfs get groupused@labstaff students/labstaff
NAME               PROPERTY            VALUE               SOURCE
students/labstaff  groupused@labstaff  250                   local
```

사용자 및 그룹 쿼터 등록 정보는 다른 모든 파일 시스템 등록 정보의 목록을 표시하는 `zfs get all` *dataset* 명령을 사용하여 표시되지 않습니다.

다음과 같이 사용자 쿼터 또는 그룹 쿼터를 제거할 수 있습니다.

```
# zfs set userquota@student1=none students/compsci
# zfs set groupquota@labstaff=none students/labstaff
```

ZFS 파일 시스템에서 사용자 및 그룹 쿼터는 다음과 같은 기능을 제공합니다.

- 부모 파일 시스템에 설정된 사용자 쿼터 또는 그룹 쿼터는 종속된 파일 시스템에 의해 자동으로 상속되지 않습니다.
- 하지만 사용자 또는 그룹 쿼터는 사용자 또는 그룹 쿼터가 포함된 파일 시스템에서 복제본 또는 스냅샷을 만들 때 적용됩니다. 마찬가지로 `-R` 옵션이 없더라도 `zfs send` 명령을 사용하여 스트림을 만들면 사용자 또는 그룹 쿼터가 파일 시스템에 포함됩니다.
- 권한이 없는 사용자는 자신의 고유 디스크 공간 사용량만 액세스할 수 있습니다. 루트 사용자 또는 `userused` 또는 `groupused` 권한이 부여된 사용자는 모든 사용자 또는 그룹 디스크 공간 계산 정보에 액세스할 수 있습니다.
- `userquota` 및 `groupquota` 등록 정보는 ZFS 볼륨, 파일 시스템 버전 4 이전의 파일 시스템, 풀 버전 15 이전의 풀에 설정할 수 없습니다.

사용자 및 그룹 쿼터를 강제 적용하면 몇 초간 작업이 지연될 수 있습니다. 이러한 지연은 사용자가 자신의 쿼터를 초과한 후 시스템에서 쿼터 초과를 감지하여 `EDQUOT` 오류 메시지와 함께 추가 쓰기 작업을 거부할 수 있기 때문에 발생합니다.

ZFS 파일 시스템이 마운트된 것과 같은 NFS 환경에서 레거시 `quota` 명령을 사용하여 사용자 쿼터를 검토할 수 있습니다. 아무 옵션도 사용하지 않고 `quota` 명령을 실행하면 사용자의 쿼터가 초과되었는지를 나타내는 출력 결과만 표시됩니다. 예를 들면 다음과 같습니다.

```
# zfs set userquota@student1=10m students/compsci
# zfs userspace students/compsci
TYPE        NAME      USED  QUOTA
POSIX User  root      350M   none
POSIX User  student1  550M    10M
# quota student1
Block limit reached on /students/compsci
```

사용자 쿼터를 재설정할 때 쿼터 한도가 더 이상 초과되지 않으면 `quota -v` 명령을 사용하여 사용자의 쿼터를 검토할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs set userquota@student1=10GB students/compsci
# zfs userspace students/compsci
TYPE        NAME      USED  QUOTA
POSIX User  root      350M   none
POSIX User  student1  550M    10G
# quota student1
# quota -v student1
Disk quotas for student1 (uid 102):
Filesystem     usage  quota  limit    timeleft  files  quota  limit    timeleft
/students/compsci
              563287 10485760 10485760           -      -      -      -           -
```

ZFS 파일 시스템에 대한 예약 설정

ZFS **예약**은 데이터 세트에 사용할 수 있도록 보장되었고 풀에서 할당된 디스크 공간입니다. 따라서 풀에서 현재 사용할 수 있는 공간이 없으면 데이터 세트에 대한

디스크 공간을 예약할 수 없습니다. 모든 미해결된 미소비 예약의 총량은 풀에서 사용되지 않은 디스크 공간을 초과할 수 없습니다. ZFS 예약은 zfs set 및 zfs get 명령을 사용하여 설정 및 표시할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs set reservation=5G tank/home/bill
# zfs get reservation tank/home/bill
NAME            PROPERTY     VALUE   SOURCE
tank/home/bill  reservation  5G      local
```

예약은 zfs list 명령의 출력 결과에 영향을 줄 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs list -r tank/home
NAME                USED  AVAIL  REFER  MOUNTPOINT
tank/home          5.00G  61.9G    37K  /tank/home
tank/home/bill       31K  66.9G    31K  /tank/home/bill
tank/home/jeff      337K  10.0G   306K  /tank/home/jeff
tank/home/lori      547K  61.9G   547K  /tank/home/lori
tank/home/mark       31K  61.9G    31K  /tank/home/mark
```

tank/home 및 해당 종속 항목에서 참조하는 총 디스크 공간이 5GB보다 훨씬 적지만 tank/home에서는 5GB의 디스크 공간을 사용하는 중입니다. used 공간은 tank/home/bill에 대해 예약된 공간을 나타냅니다. 예약은 부모 파일 시스템에서 사용된 디스크 공간 계산에 포함되며 해당 쿼터, 예약 또는 두 항목 모두에서 공제됩니다.

```
# zfs set quota=5G pool/filesystem
# zfs set reservation=10G pool/filesystem/user1
cannot set reservation for 'pool/filesystem/user1': size is greater than
available space
```

풀에서 사용 가능한 예약되지 않은 공간이 있고 데이터 세트의 현재 사용량이 해당 쿼터 미만인 경우 데이터 세트는 해당 예약보다 많은 디스크 공간을 사용할 수 있습니다. 데이터 세트는 다른 데이터 세트에 대해 예약된 디스크 공간을 소비할 수 없습니다.

예약은 누적되지 않습니다. 즉, 예약을 설정하기 위해 zfs set를 두번째로 호출해도 해당 예약이 기존 예약에 추가되지 않습니다. 그 대신 첫번째 예약이 두번째 예약으로 대체됩니다. 예를 들면 다음과 같습니다.

```
# zfs set reservation=10G tank/home/bill
# zfs set reservation=5G tank/home/bill
# zfs get reservation tank/home/bill
NAME            PROPERTY     VALUE   SOURCE
tank/home/bill  reservation  5G      local
```

refreservation 예약을 설정하여 스냅샷 및 복제본이 소비하는 디스크 공간을 포함하지 않는 데이터 세트에 대한 디스크 공간을 보장할 수 있습니다. 이 예약은 부모 데이터 세트의 used 공간 계산에 포함되며 부모 데이터 세트의 쿼터 및 예약에서 공제됩니다. 예를 들면 다음과 같습니다.

```
# zfs set refreservation=10g profs/prof1
# zfs list
NAME                     USED  AVAIL  REFER  MOUNTPOINT
```

```
profs                   10.0G  23.2G    19K  /profs
profs/prof1               10G  33.2G    18K  /profs/prof1
```

또한 데이터 세트 공간과 스냅샷 공간을 보장할 수 있도록 동일한 데이터 세트에서 예약을 설정할 수도 있습니다. 예를 들면 다음과 같습니다.

```
# zfs set reservation=20g profs/prof1
# zfs list
NAME                     USED  AVAIL  REFER  MOUNTPOINT
profs                   20.0G  13.2G    19K  /profs
profs/prof1               10G  33.2G    18K  /profs/prof1
```

일반 예약은 부모의 used 공간 계산에 포함됩니다.

앞의 예제에서는 두 쿼터(10GB와 20GB) 중 작은 쿼터가 `zfs list` 출력 결과에 표시되었습니다. 두 쿼터의 값을 보려면 `zfs get` 명령을 사용합니다. 예를 들면 다음과 같습니다.

```
# zfs get reservation,refreserv profs/prof1
NAME         PROPERTY        VALUE        SOURCE
profs/prof1  reservation     20G          local
profs/prof1  refreservation  10G          local
```

`refreservation`이 설정된 경우에는 이 예약 외에도 데이터 세트에서 현재 **참조되는** 바이트 수를 수용할 수 있도록 예약되지 않은 충분한 풀 공간이 존재하는 경우에만 스냅샷이 허용됩니다.

ZFS 파일 시스템 업그레이드

이전 Solaris 릴리스의 ZFS 파일 시스템이 있는 경우 현재 릴리스의 파일 시스템 기능을 활용하기 위해 `zfs upgrade` 명령을 사용하여 파일 시스템을 업그레이드할 수 있습니다. 또한 이 명령을 사용하면 파일 시스템이 이전 버전을 실행 중인 경우 이를 사용자에게 알려 줍니다.

예를 들어 이 파일 시스템은 현재 버전인 5입니다.

```
# zfs upgrade
This system is currently running ZFS filesystem version 5.

All filesystems are formatted with the current version.
```

이 명령을 사용하면 각 파일 시스템 버전에서 제공되는 기능을 확인할 수 있습니다.

```
# zfs upgrade -v
The following filesystem versions are supported:

VER  DESCRIPTION
---  --------------------------------------------------------
 1   Initial ZFS filesystem version
 2   Enhanced directory entries
```

```
 3   Case insensitive and File system unique identifier (FUID)
 4   userquota, groupquota properties
 5   System attributes

For more information on a particular version, including supported releases,
see the ZFS Administration Guide.
```

◆ ◆ ◆ 6 장

6 Oracle Solaris ZFS 스냅샷 및 복제 작업

이 장에서는 Oracle Solaris ZFS 스냅샷 및 복제본을 만들고 관리하는 방법에 대해 설명합니다. 그리고 스냅샷 저장에 대해서도 다룹니다.

이 장에서는 다음과 같은 내용을 다룹니다.

ZFS 스냅샷 개요

스냅샷은 파일 시스템 또는 볼륨에 대한 읽기 전용 복사본입니다. 즉시 만들 수 있으며 처음에는 풀 내에서 추가 디스크 공간을 사용하지 않습니다. 그러나 활성 데이터 세트 내의 데이터가 변경되면 스냅샷은 기존 데이터를 계속 참조하기 위해 디스크 공간을 사용하므로 디스크 공간이 해제되지 않습니다.

ZFS 스냅샷의 기능은 다음과 같습니다.

- 시스템 재부트 후에도 지속됩니다.
- 이론상 최대 스냅샷 수는 2^{64}입니다.
- 스냅샷은 별도의 보조 저장소를 사용하지 않습니다. 스냅샷이 생성된 파일 시스템 또는 볼륨과 동일한 저장소 풀에서 직접 디스크 공간을 사용합니다.

- 순환 스냅샷은 하나의 기본 단위 작업으로 신속하게 생성됩니다. 스냅샷은 한꺼번에 동시에 생성되거나, 아니면 전혀 생성되지 않습니다. 기본 단위 스냅샷 작업의 이점은 종속 파일 시스템에서도 스냅샷 데이터를 항상 하나의 일관된 시간에 가져온다는 점입니다.

볼륨 스냅샷은 직접 액세스할 수는 있지만, 복제, 백업, 롤백 등은 불가능합니다. ZFS 스냅샷 백업에 대한 자세한 내용은 212 페이지 "ZFS 데이터 전송 및 수신"을 참조하십시오.

- 204 페이지 "ZFS 스냅샷 만들기 및 삭제"
- 207 페이지 "ZFS 스냅샷 표시 및 액세스"
- 209 페이지 "ZFS 스냅샷 롤백"

ZFS 스냅샷 만들기 및 삭제

스냅샷은 `zfs snapshot` 명령으로 생성되는데, 이 명령은 생성될 스냅샷의 이름만 인수로 사용합니다. 스냅샷은 이름은 다음과 같이 지정됩니다.

filesystem@snapname
volume@snapname

스냅샷 이름은 29 페이지 "ZFS 구성 요소 명명 요구 사항"의 조건을 충족해야 합니다.

다음 예에서는 `friday`라는 이름의 `tank/home/cindy` 스냅샷이 생성됩니다.

```
# zfs snapshot tank/home/cindy@friday
```

`-r` 옵션을 사용하면 모든 종속 파일 시스템에 대한 스냅샷을 만들 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs snapshot -r tank/home@snap1
# zfs list -t snapshot -r tank/home
NAME                   USED  AVAIL  REFER  MOUNTPOINT
tank/home@snap1           0      -  2.11G  -
tank/home/cindy@snap1     0      -   115M  -
tank/home/lori@snap1      0      -  2.00G  -
tank/home/mark@snap1      0      -  2.00G  -
tank/home/tim@snap1       0      -  57.3M  -
```

스냅샷에는 수정 가능한 등록 정보가 없습니다. 스냅샷에 적용할 수 있는 데이터 세트 등록 정보도 없습니다. 예를 들면 다음과 같습니다.

```
# zfs set compression=on tank/home/cindy@friday
cannot set property for 'tank/home/cindy@friday':
this property can not be modified for snapshots
```

스냅샷은 `zfs destroy` 명령을 사용하여 삭제됩니다. 예를 들면 다음과 같습니다.

```
# zfs destroy tank/home/cindy@friday
```

데이터 세트의 스냅샷이 있을 경우에는 데이터 세트를 삭제할 수 없습니다. 예를 들면 다음과 같습니다.

```
# zfs destroy tank/home/cindy
cannot destroy 'tank/home/cindy': filesystem has children
use '-r' to destroy the following datasets:
tank/home/cindy@tuesday
tank/home/cindy@wednesday
tank/home/cindy@thursday
```

또한 스냅샷에서 생성된 복제본은 스냅샷을 삭제하기 전에 삭제해야 합니다.

`destroy` 하위 명령에 대한 자세한 내용은 167 페이지 "ZFS 파일 시스템 삭제"를 참조하십시오.

ZFS 스냅샷 유지

보내는 측에 더 이상 있지 않다는 이유로 `zfs receive` 명령으로 오래된 스냅샷을 무단으로 삭제하는 것과 같은 여러 자동 스냅샷 정책을 사용하는 경우 스냅샷 유지 기능을 사용할 것을 고려해 볼 수 있습니다.

스냅샷 **유지** 기능을 사용하면 스냅샷이 삭제되지 않습니다. 또한 이 기능을 사용하면 클론이 있는 스냅샷의 경우 `zfs destroy -d` 명령을 사용하여 마지막 클론을 제거하지 않는 한 해당 스냅샷을 삭제할 수 있습니다. 각 스냅샷에는 연관된 user-reference 카운트가 있는데, 이 카운트는 0으로 초기화되어 있습니다. 이 카운트는 스냅샷이 유지될 때마다 1씩 늘어나고 유지가 해제될 때마다 1씩 줄어듭니다.

이전 Oracle Solaris 릴리스에서는 복제본이 없는 경우 `zfs destroy` 명령을 통해서만 스냅샷을 삭제할 수 있었습니다. 이 Oracle Solaris 릴리스에서는 스냅샷의 user-reference 카운트도 0이어야 합니다.

스냅샷 또는 스냅샷 세트를 유지할 수 있습니다. 예를 들어, 다음 구문은 `tank/home/cindy/snap@1`에 유지 태그인 `keep`을 삽입합니다.

```
# zfs hold keep tank/home/cindy@snap1
```

`-r` 옵션을 사용하면 모든 종속 파일 시스템의 스냅샷을 반복적으로 유지할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs snapshot -r tank/home@now
# zfs hold -r keep tank/home@now
```

이 구문은 단일 참조인 `keep`을 스냅샷 또는 스냅샷 세트에 추가합니다. 각 스냅샷에는 고유한 태그 이름 공간이 있으며 유지 태그는 해당 공간 내에서 고유해야 합니다. 유지된 스냅샷이 있는 경우 `zfs destroy` 명령을 사용하여 유지된 해당 스냅샷을 삭제하려고 하면 삭제되지 않습니다. 예를 들면 다음과 같습니다.

```
# zfs destroy tank/home/cindy@snap1
cannot destroy 'tank/home/cindy@snap1': dataset is busy
```

유지된 스냅샷을 삭제하려면 -d 옵션을 사용하십시오. 예를 들면 다음과 같습니다.

```
# zfs destroy -d tank/home/cindy@snap1
```

zfs holds 명령을 사용하면 유지된 스냅샷 목록이 표시됩니다. 예를 들면 다음과 같습니다.

```
# zfs holds tank/home@now
NAME           TAG   TIMESTAMP
tank/home@now  keep  Fri Aug  3 15:15:53 2012

# zfs holds -r tank/home@now
NAME                 TAG   TIMESTAMP
tank/home/cindy@now  keep  Fri Aug  3 15:15:53 2012
tank/home/lori@now   keep  Fri Aug  3 15:15:53 2012
tank/home/mark@now   keep  Fri Aug  3 15:15:53 2012
tank/home/tim@now    keep  Fri Aug  3 15:15:53 2012
tank/home@now        keep  Fri Aug  3 15:15:53 2012
```

zfs release 명령을 사용하면 스냅샷 또는 스냅샷 세트에 대한 유지가 해제됩니다. 예를 들면 다음과 같습니다.

```
# zfs release -r keep tank/home@now
```

스냅샷이 해제되면 zfs destroy 명령을 사용하여 스냅샷을 삭제할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs destroy -r tank/home@now
```

스냅샷 유지 정보는 다음 두 등록 정보로 식별할 수 있습니다.

- zfs destroy -d 명령을 사용하여 스냅샷이 삭제 지연으로 표시된 경우 defer_destroy 등록 정보가 on입니다. 그렇지 않은 경우 이 등록 정보는 off입니다.
- userrefs 등록 정보가 이 스냅샷에 대한 유지 수로 설정됩니다. 이를 *user-reference* 카운트라고도 합니다.

ZFS 스냅샷 이름 바꾸기

스냅샷의 이름을 바꿀 수는 있지만, 스냅샷이 생성된 것과 동일한 풀 및 데이터 세트 내에서 이름을 바꿔야 합니다. 예를 들면 다음과 같습니다.

```
# zfs rename tank/home/cindy@snap1 tank/home/cindy@today
```

또한 다음 단축 구문은 위의 구문과 같습니다.

```
# zfs rename tank/home/cindy@snap1 today
```

스냅샷이 생성된 풀 및 파일 시스템과 대상 풀 및 파일 시스템이 다르기 때문에 다음 스냅샷 **이름 바꾸기** 작업은 지원되지 않습니다.

```
# zfs rename tank/home/cindy@today pool/home/cindy@saturday
cannot rename to 'pool/home/cindy@today': snapshots must be part of same
dataset
```

zfs rename - r 명령을 사용하면 스냅샷의 이름을 반복적으로 바꿀 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs list -t snapshot -r users/home
NAME                         USED  AVAIL  REFER  MOUNTPOINT
users/home@now              23.5K      -  35.5K  -
users/home@yesterday            0      -    38K  -
users/home/lori@yesterday       0      -  2.00G  -
users/home/mark@yesterday       0      -  1.00G  -
users/home/neil@yesterday       0      -  2.00G  -
# zfs rename -r users/home@yesterday @2daysago
# zfs list -t snapshot -r users/home
NAME                         USED  AVAIL  REFER  MOUNTPOINT
users/home@now              23.5K      -  35.5K  -
users/home@2daysago             0      -    38K  -
users/home/lori@2daysago        0      -  2.00G  -
users/home/mark@2daysago        0      -  1.00G  -
users/home/neil@2daysago        0      -  2.00G  -
```

ZFS 스냅샷 표시 및 액세스

listsnapshots 풀 등록 정보를 사용하여 zfs list 출력 결과에서 스냅샷 목록 표시를 사용 또는 사용 안함으로 설정할 수 있습니다. 이 등록 정보는 기본적으로 사용으로 설정됩니다.

이 등록 정보를 사용 안함으로 설정하면 zfs list -t snapshot 명령을 사용하여 스냅샷 정보를 표시할 수 있습니다. 또는 listsnapshots 풀 등록 정보를 사용으로 설정하십시오. 예를 들면 다음과 같습니다.

```
# zpool get listsnapshots tank
NAME  PROPERTY       VALUE      SOURCE
tank  listsnapshots  on         default
# zpool set listsnapshots=off tank
# zpool get listsnapshots tank
NAME  PROPERTY       VALUE      SOURCE
tank  listsnapshots  off         local
```

파일 시스템 스냅샷은 파일 시스템 루트 내의 .zfs/snapshot 디렉토리에서 액세스할 수 있습니다. 예를 들어, tank/home/cindy가 /home/cindy에 마운트된 경우 tank/home/cindy@thursday 스냅샷 데이터는 /home/cindy/.zfs/snapshot/thursday 디렉토리에서 액세스할 수 있습니다.

```
# ls /tank/home/cindy/.zfs/snapshot
thursday   tuesday    wednesday
```

다음과 같이 스냅샷을 나열할 수 있습니다.

```
# zfs list -t snapshot -r tank/home
NAME                       USED  AVAIL  REFER  MOUNTPOINT
tank/home/cindy@tuesday     45K      -  2.11G  -
tank/home/cindy@wednesday   45K      -  2.11G  -
tank/home/cindy@thursday      0      -  2.17G  -
```

다음과 같이 특정 파일 시스템용으로 생성된 스냅샷을 나열할 수 있습니다.

```
# zfs list -r -t snapshot -o name,creation tank/home
NAME                       CREATION
tank/home/cindy@tuesday    Fri Aug  3 15:18 2012
tank/home/cindy@wednesday  Fri Aug  3 15:19 2012
tank/home/cindy@thursday   Fri Aug  3 15:19 2012
tank/home/lori@today       Fri Aug  3 15:24 2012
tank/home/mark@today       Fri Aug  3 15:24 2012
```

ZFS 스냅샷에 대한 디스크 공간 계산

스냅샷이 생성되면 처음에는 디스크 공간이 스냅샷과 파일 시스템 간에 공유됩니다. 이전 스냅샷의 경우에도 마찬가지입니다. 그러나 파일 시스템이 변경되면 이전에 공유되던 디스크 공간을 스냅샷에서만 사용하게 되므로, 스냅샷의 used 등록 정보가 카운트됩니다. 또한 스냅샷을 삭제하면 다른 스냅샷에서 사용하던 고유 디스크 공간의 양(*used* 등록 정보)이 늘어날 수 있습니다.

스냅샷 공간의 referenced 등록 정보 값은 스냅샷이 만들어졌을 때의 파일 시스템 등록 정보 값과 동일합니다.

used 등록 정보 값이 사용되는 방식에 대한 추가 정보를 확인할 수 있습니다. 새 읽기 전용 파일 시스템 등록 정보가 복제본, 파일 시스템 및 볼륨에 대한 디스크 공간 사용량을 설명합니다. 예를 들면 다음과 같습니다.

```
$ zfs list -o space -r rpool
NAME                       AVAIL   USED  USEDSNAP  USEDDS  USEDREFRESERV  USEDCHILD
rpool                      59.1G  7.84G       21K    109K              0      7.84G
rpool@snap1                    -    21K         -       -              -          -
rpool/ROOT                 59.1G  4.78G         0     31K              0      4.78G
rpool/ROOT@snap1               -     0          -       -              -          -
rpool/ROOT/zfsBE           59.1G  4.78G     15.6M   4.76G              0          0
rpool/ROOT/zfsBE@snap1         -  15.6M         -       -              -          -
rpool/dump                 59.1G  1.00G       16K   1.00G              0          0
rpool/dump@snap1               -    16K         -       -              -          -
rpool/export               59.1G    99K       18K     32K              0        49K
rpool/export@snap1             -    18K         -       -              -          -
rpool/export/home          59.1G    49K       18K     31K              0          0
rpool/export/home@snap1        -    18K         -       -              -          -
rpool/swap                 61.2G  2.06G         0     16K          2.06G          0
rpool/swap@snap1               -     0          -       -              -          -
```

이러한 등록 정보에 대한 설명은 표 5–1을 참조하십시오.

ZFS 스냅샷 롤백

zfs rollback 명령을 사용하면 특성 스냅샷이 생성된 이후에 발생한 파일 시스템의 변경 사항을 모두 무시할 수 있습니다. 파일 시스템의 상태는 스냅샷 생성 시점으로 되돌아갑니다. 기본적으로 이 명령은 가장 최근의 스냅샷만 롤백할 수 있습니다.

이전 스냅샷으로 롤백하려면 중간 스냅샷을 모두 삭제해야 합니다. -r 옵션을 사용하면 이전 스냅샷을 삭제할 수 있습니다.

중간 스냅샷 복제본이 있을 경우 -R 옵션을 지정하여 복제본도 삭제해야 합니다.

주 – 롤백하려는 현재 파일 시스템이 현재 마운트된 경우 마운트가 해제되었다가 다시 마운트됩니다. 파일 시스템을 마운트 해제할 수 없으면 롤백이 실패합니다. -f 옵션은 필요한 경우 파일 시스템을 강제로 마운트 해제합니다.

다음 예에서는 tank/home/cindy 파일 시스템이 tuesday 스냅샷으로 롤백됩니다.

```
# zfs rollback tank/home/cindy@tuesday
cannot rollback to 'tank/home/cindy@tuesday': more recent snapshots exist
use '-r' to force deletion of the following snapshots:
tank/home/cindy@wednesday
tank/home/cindy@thursday
# zfs rollback -r tank/home/cindy@tuesday
```

이 예에서는 이전 tuesday 스냅샷으로 롤백했으므로 wednesday 및 thursday 스냅샷이 삭제되었습니다.

```
# zfs list -r -t snapshot -o name,creation tank/home/cindy
NAME                      CREATION
tank/home/cindy@tuesday  Fri Aug  3 15:18 2012
```

ZFS 스냅샷 차이 식별(zfs diff)

ZFS 스냅샷 차이점은 zfs diff 명령을 사용하여 확인할 수 있습니다.

예를 들어, 다음 두 스냅샷이 생성된다고 가정해 보겠습니다.

```
$ ls /tank/home/tim
fileA
$ zfs snapshot tank/home/tim@snap1
$ ls /tank/home/tim
fileA  fileB
$ zfs snapshot tank/home/tim@snap2
```

예를 들어, 두 스냅샷 간의 차이를 확인하려면 다음과 비슷한 구문을 사용하십시오.

```
$ zfs diff tank/home/tim@snap1 tank/home/tim@snap2
M       /tank/home/tim/
+       /tank/home/tim/fileB
```

출력 결과에서 M은 디렉토리가 수정되었음을 나타냅니다. +는 fileB가 나중 스냅샷에 존재함을 나타냅니다.

다음 출력에서 R은 스냅샷에 있는 파일의 이름이 바뀌었음을 나타냅니다.

```
$ mv /tank/cindy/fileB /tank/cindy/fileC
$ zfs snapshot tank/cindy@snap2
$ zfs diff tank/cindy@snap1 tank/cindy@snap2
M       /tank/cindy/
R       /tank/cindy/fileB -> /tank/cindy/fileC
```

다음 표는 zfs diff 명령으로 식별되는 파일 또는 디렉토리 변경 사항을 요약하여 보여줍니다.

파일 또는 디렉토리 변경 사항	식별자
파일 또는 디렉토리가 수정되었거나 파일 또는 디렉토리 링크가 변경되었습니다.	M
파일 또는 디렉토리가 이전 스냅샷에만 있고 최근 스냅샷에는 없습니다.	–
파일 또는 디렉토리가 최근 스냅샷에만 있고 이전 스냅샷에는 없습니다.	+
파일 또는 디렉토리의 이름이 변경되었습니다.	R

자세한 내용은 zfs(1M)를 참조하십시오.

ZFS 복제본 개요

복제본은 쓰기가 가능한 볼륨 또는 파일 시스템으로, 초기 컨텐츠는 복제본이 생성된 데이터 세트와 동일합니다. 스냅샷과 마찬가지로, 복제본은 즉시 생성되며 처음에는 추가 디스크 공간을 사용하지 않습니다. 또한 복제본을 스냅샷할 수 있습니다.

복제본은 스냅샷에서만 만들 수 있습니다. 스냅샷이 복제되면 복제본과 스냅샷 간에 암시적 종속성이 생깁니다. 파일 시스템 계층 내의 다른 위치에 복제본을 만든 경우에도 복제본이 있는 한 원래 스냅샷을 삭제할 수 없습니다. origin 등록 정보가 이 종속성을 표시하며, zfs destroy 명령은 이러한 종속성이 있을 경우 이를 나열합니다.

복제본은 만들 때 사용된 데이터 세트의 등록 정보를 상속하지 않습니다. 복제된 데이터 세트의 등록 정보를 확인 및 변경하려면 zfs get 및 zfs set 명령을 사용하십시오. ZFS 데이터 세트 등록 정보 설정에 대한 자세한 내용은 184 페이지 “ZFS 등록 정보 설정”을 참조하십시오.

처음에는 복제본이 원본 스냅샷과 모든 디스크 공간을 공유하기 때문에 used 등록 정보 값이 처음에는 0입니다. 복제본이 변경되면 더 많은 디스크 공간을 사용합니다. 원본 스냅샷의 used 등록 정보에는 복제본이 사용하는 디스크 공간이 포함되지 않습니다.

ZFS 복제본 만들기

복제본을 만들려면 복제본을 만들 스냅샷 및 새 파일 시스템 또는 볼륨의 이름을 지정하여 `zfs clone` 명령을 사용하십시오. 새 파일 시스템 또는 볼륨은 ZFS 계층 내에 있습니다. 새 데이터 세트는 복제본이 생성된 스냅샷과 같은 유형(예: 파일 시스템 또는 볼륨)입니다. 원본 파일 시스템 스냅샷이 있는 풀이 아닌 다른 풀에는 파일 시스템 복제본을 만들 수 없습니다.

다음 예에서는 초기 컨텐츠가 `tank/ws/gate@yesterday` 스냅샷과 같은 새 복제본 `tank/home/matt/bug123`이 생성됩니다.

```
# zfs snapshot tank/ws/gate@yesterday
# zfs clone tank/ws/gate@yesterday tank/home/matt/bug123
```

다음 예에서는 임시 사용자에 대한 `projects/newproject@today` 스냅샷에서 복제된 작업 공간이 `projects/teamA/tempuser`로 생성되었습니다. 그런 다음 복제된 작업 공간에 대한 등록 정보가 설정되었습니다.

```
# zfs snapshot projects/newproject@today
# zfs clone projects/newproject@today projects/teamA/tempuser
# zfs set sharenfs=on projects/teamA/tempuser
# zfs set quota=5G projects/teamA/tempuser
```

ZFS 복제본 삭제

ZFS 복제본은 `zfs destroy` 명령을 사용하여 삭제됩니다. 예를 들면 다음과 같습니다.

```
# zfs destroy tank/home/matt/bug123
```

먼저 복제본을 삭제해야 부모 스냅샷을 삭제할 수 있습니다.

ZFS 복제본으로 ZFS 파일 시스템 대체

`zfs promote` 명령을 사용하여 활성 ZFS 파일 시스템을 해당 파일 시스템의 복제본으로 대체할 수 있습니다. 이 기능을 사용하여 **원본** 파일 시스템이 지정된 파일 시스템의 복제본이 되도록 파일 시스템을 복제하고 대체할 수 있습니다. 또한 이 기능으로 복제본이 원래 생성된 파일 시스템을 삭제할 수도 있습니다. 복제 프로모션이 없으면 활성 복제본의 원본 파일 시스템을 삭제할 수 없습니다. 복제본 삭제에 대한 자세한 내용은 211 페이지 "ZFS 복제본 삭제"를 참조하십시오.

다음 예에서는 `tank/test/productA` 파일 시스템이 복제된 다음 복제 파일 시스템 `tank/test/productAbeta`가 원본 `tank/test/productA` 파일 시스템이 됩니다.

```
# zfs create tank/test
# zfs create tank/test/productA
# zfs snapshot tank/test/productA@today
# zfs clone tank/test/productA@today tank/test/productAbeta
# zfs list -r tank/test
NAME                        USED  AVAIL  REFER  MOUNTPOINT
tank/test                   104M  66.2G    23K  /tank/test
tank/test/productA          104M  66.2G   104M  /tank/test/productA
tank/test/productA@today       0      -   104M  -
tank/test/productAbeta         0  66.2G   104M  /tank/test/productAbeta
# zfs promote tank/test/productAbeta
# zfs list -r tank/test
NAME                          USED  AVAIL  REFER  MOUNTPOINT
tank/test                     104M  66.2G    24K  /tank/test
tank/test/productA               0  66.2G   104M  /tank/test/productA
tank/test/productAbeta        104M  66.2G   104M  /tank/test/productAbeta
tank/test/productAbeta@today     0      -   104M  -
```

이 `zfs list` 출력 결과에서 원본 `productA` 파일 시스템에 대한 디스크 공간 계산 정보가 `productAbeta` 파일 시스템으로 대체되었습니다.

파일 시스템의 이름을 바꿔 복제 대체 프로세스를 완료할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs rename tank/test/productA tank/test/productAlegacy
# zfs rename tank/test/productAbeta tank/test/productA
# zfs list -r tank/test
```

선택적으로 레거시 파일 시스템을 제거할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs destroy tank/test/productAlegacy
```

ZFS 데이터 전송 및 수신

`zfs send` 명령은 표준 출력 결과에 기록될 스냅샷의 스트림 표현을 만듭니다. 기본적으로 전체 스트림이 생성됩니다. 출력을 파일 또는 다른 시스템으로 재지정할 수 있습니다. `zfs receive` 명령은 컨텐츠가 표준 출력에 제공된 스트림에 지정되어 있는 스냅샷을 만듭니다. 전체 스트림이 수신된 경우 새 파일 시스템도 생성됩니다. 이 명령으로 ZFS 스냅샷 데이터를 전송하고 ZFS 스냅샷 데이터 및 파일 시스템을 수신할 수 있습니다. 다음 절에 나와 있는 예제를 참조하십시오.

- 213 페이지 “다른 백업 제품으로 ZFS 데이터 저장”
- 215 페이지 “ZFS 스냅샷 전송”
- 216 페이지 “ZFS 스냅샷 수신”
- 217 페이지 “ZFS 스냅샷 스트림에 다른 등록 정보 값 적용”
- 219 페이지 “복잡한 ZFS 스냅샷 스트림 전송 및 수신”
- 222 페이지 “ZFS 데이터 원격 복제”

ZFS 데이터 저장을 위해 제공되는 백업 솔루션은 다음과 같습니다.

- **엔터프라이즈 백업 제품** – 다음 기능이 필요한 경우 엔터프라이즈 백업 솔루션을 고려하십시오.
  - 파일별 복원
  - 백업 매체 확인
  - 매체 관리
- **파일 시스템 스냅샷 및 스냅샷 롤백** – 파일 시스템 복사본을 손쉽게 만들고 필요한 경우 이전 파일 시스템 버전으로 되돌리려는 경우 `zfs snapshot` 및 `zfs rollback` 명령을 사용하십시오. 예를 들어 이전 버전의 파일 시스템에서 파일을 복원하려면 이 솔루션을 사용할 수 있습니다.

  스냅샷 만들기 및 롤백에 대한 자세한 내용은 203 페이지 “ZFS 스냅샷 개요”를 참조하십시오.
- **스냅샷 저장** – `zfs send` 및 `zfs receive` 명령을 사용하여 ZFS 스냅샷을 전송하고 수신할 수 있습니다. 스냅샷 간 증분 변경을 저장할 수 있지만 파일을 개별적으로 복원할 수는 없습니다. 전체 파일 시스템 스냅샷을 복원해야 합니다. 이러한 명령은 ZFS 데이터를 저장하기 위한 완벽한 백업 솔루션을 제공하지 않습니다.
- **원격 복제** – `zfs send` 및 `zfs receive` 명령을 사용하여 파일 시스템을 시스템 간에 복사할 수 있습니다. 이 프로세스는 WAN을 통해 장치를 미러링하는 기존의 볼륨 관리 제품과 다릅니다. 특별한 구성이나 하드웨어가 필요하지 않습니다. ZFS 파일 시스템 복제의 이점은 다른 시스템의 저장소 풀에 파일 시스템을 다시 만들고, 동일한 파일 시스템 데이터를 포함하되 새로 생성된 풀에 대해 다른 레벨의 구성(예: RAID-Z)을 지정할 수 있다는 점입니다.
- **아카이브 유틸리티** – `tar`, `cpio` , `pax` 등의 아카이브 유틸리티 또는 타사 백업 제품을 사용하여 ZFS 데이터를 저장할 수 있습니다. 현재 `tar` 및 `cpio`는 NFSv4 스타일 ACL을 올바르게 변환하지만, `pax`는 그렇지 못합니다.

다른 백업 제품으로 ZFS 데이터 저장

`zfs send` 및 `zfs receive` 명령 이외에도 `tar` 및 `cpio` 명령 등의 아카이브 유틸리티를 사용하여 ZFS 파일을 저장할 수도 있습니다. 이러한 유틸리티는 ZFS 파일 속성 및 ACL을 저장하고 복원합니다. `tar` 및 `cpio` 명령에 적합한 옵션을 확인하십시오.

ZFS 및 타사 백업 제품에 대한 최신 정보는 Solaris 10 릴리스 정보를 참조하십시오.

ZFS 스냅샷 스트림 식별

`zfs send` 명령을 사용하여 ZFS 파일 시스템 또는 볼륨의 스냅샷을 스냅샷 스트림으로 변환합니다. 그런 다음 `zfs receive` 명령을 통해 스냅샷 스트림을 사용하여 ZFS 파일 시스템 또는 볼륨을 다시 만들 수 있습니다.

스냅샷 스트림을 만드는 데 사용된 `zfs send` 옵션에 따라 다른 유형의 스트림 형식이 생성됩니다.

- 전체 스트림 – 데이터 세트가 생성된 시간부터 지정한 스냅샷까지의 모든 데이터 세트 컨텐츠로 구성됩니다.

  zfs send 명령으로 생성되는 기본 스트림이 전체 스트림입니다. 지정한 스냅샷까지 파일 시스템 또는 볼륨 한 개를 포함합니다. 명령줄에서 지정된 스냅샷이 아닌 스냅샷은 스트림에 포함되지 않습니다.
- 증분 스트림 – 한 스냅샷과 다른 스냅샷의 차이점으로 구성됩니다.

스트림 패키지는 전체 또는 증분 스트림이 한 개 이상 포함된 스트림 유형입니다. 다음 세 가지 유형의 스트림 패키지가 있습니다.

- 복제 스트림 패키지 – 지정한 데이터 세트 및 종속 항목으로 구성됩니다. 중간 스냅샷이 모두 포함됩니다. 복제된 데이터 세트의 원본이 명령줄에서 지정된 스냅샷의 종속 항목이 아닌 경우 원본 데이터 세트가 스트림 패키지에 포함되지 않습니다. 스트림을 받으려면 대상 저장소 풀에 원본 데이터 세트가 있어야 합니다.

  데이터 세트 및 해당 원본이 포함된 다음 목록을 고려해 보십시오. 아래 표시되는 순서대로 생성되었다고 가정합니다.

```
NAME                    ORIGIN
pool/a                  -
pool/a/1                -
pool/a/1@clone          -
pool/b                  -
pool/b/1                pool/a/1@clone
pool/b/1@clone2         -
pool/b/2                pool/b/1@clone2
pool/b@pre-send         -
pool/b/1@pre-send       -
pool/b/2@pre-send       -
pool/b@send             -
pool/b/1@send           -
pool/b/2@send           -
```

  다음 구문으로 생성된 복제 스트림 패키지가 있습니다.

```
# zfs send -R pool/b@send ....
```

  이 패키지는 다음과 같은 전체 및 증분 스트림으로 구성됩니다.

```
TYPE    SNAPSHOT                INCREMENTAL FROM
full    pool/b@pre-send         -
incr    pool/b@send             pool/b@pre-send
incr    pool/b/1@clone2         pool/a/1@clone
incr    pool/b/1@pre-send       pool/b/1@clone2
incr    pool/b/1@send           pool/b/1@send
incr    pool/b/2@pre-send       pool/b/1@clone2
incr    pool/b/2@send           pool/b/2@pre-send
```

  이전 출력에서 pool/a/1@clone 스냅샷은 복제 스트림 패키지에 포함되지 않습니다. 따라서 이미 pool/a/1@clone 스냅샷이 있는 풀에서만 이 복제 스트림 패키지를 받을 수 있습니다.

- 순환적 스트림 패키지 – 지정한 데이터 세트 및 종속 항목으로 구성됩니다. 복제 스트림 패키지와 달리 중간 스냅샷은 스트림에 포함된 복제된 데이터 세트의 원본이 아닌 경우 포함되지 않습니다. 기본적으로 데이터 세트의 원본이 명령줄에서 지정된 스냅샷의 종속 항목이 아닌 경우 복제 스트림과 유사하게 동작합니다. 하지만 아래에 설명된 독립적인 순환적 스트림은 외부 종속성이 없도록 생성됩니다.

  다음 구문으로 생성된 순환적 스트림 패키지가 있습니다.

  ```
  # zfs send -r pool/b@send ...
  ```

  이 패키지는 다음과 같은 전체 및 증분 스트림으로 구성됩니다.

  ```
  TYPE    SNAPSHOT                INCREMENTAL FROM
  full    pool/b@send             -
  incr    pool/b/1@clone2         pool/a/1@clone
  incr    pool/b/1@send           pool/b/1@clone2
  incr    pool/b/2@send           pool/b/1@clone2
  ```

  이전 출력에서 `pool/a/1@clone` 스냅샷은 순환적 스트림 패키지에 포함되지 않습니다. 따라서 이미 `pool/a/1@clone` 스냅샷이 있는 풀에서만 이 순환적 스트림 패키지를 받을 수 있습니다. 이 동작은 위에 설명된 복제 스트림 패키지 시나리오와 유사합니다.

- 독립적인 순환적 스트림 패키지 - 스트림 패키지에 포함되지 않은 데이터 세트에 종속되지 않습니다. 이 순환적 스트림 패키지는 다음 구문으로 생성됩니다.

  ```
  # zfs send -rc pool/b@send ...
  ```

  이 패키지는 다음과 같은 전체 및 증분 스트림으로 구성됩니다.

  ```
  TYPE    SNAPSHOT                INCREMENTAL FROM
  full    pool/b@send             -
  full    pool/b/1@clone2
  incr    pool/b/1@send           pool/b/1@clone2
  incr    pool/b/2@send           pool/b/1@clone2
  ```

  독립적인 순환적 스트림에는 `pool/b/1@clone2` 스냅샷의 전체 스트림이 있으므로 외부 종속성 없이 `pool/b/1` 스냅샷을 받을 수 있습니다.

ZFS 스냅샷 전송

`zfs send` 명령을 사용하여 스냅샷 스트림의 복사본을 전송하고 같은 시스템의 다른 풀 또는 백업 데이터를 백업하는 데 사용되는 다른 시스템의 다른 풀에 있는 스냅샷 스트림을 수신할 수 있습니다. 예를 들어, 같은 시스템의 다른 풀에 있는 스냅샷 스트림을 전송하려면 다음과 유사한 구문을 사용하십시오.

```
# zfs send tank/dana@snap1 | zfs recv spool/ds01
```

`zfs recv`를 `zfs receive` 명령에 대한 별명으로 사용할 수 있습니다.

스냅샷 스트림을 다른 시스템으로 전송하려는 경우 `ssh` 명령을 통해 `zfs send` 출력 결과를 파이프로 연결하십시오. 예를 들면 다음과 같습니다.

```
sys1# zfs send tank/dana@snap1 | ssh sys2 zfs recv newtank/dana
```

전체 스트림을 전송하는 경우 대상 파일 시스템이 존재하지 않아야 합니다.

zfs send -i 옵션을 사용하여 증분 데이터를 전송할 수 있습니다. 예를 들면 다음과 같습니다.

```
sys1# zfs send -i tank/dana@snap1 tank/dana@snap2 | ssh sys2 zfs recv newtank/dana
```

첫번째 인수(snap1)는 이전 스냅샷이고 두번째 인수(snap2)는 이후 스냅샷입니다. 이 경우 증분 수신이 성공하려면 newtank/dana 파일 시스템이 존재해야 합니다.

주 – 원래 수신된 파일 시스템에서 파일 정보에 액세스하면 다음과 비슷한 메시지와 함께 증분 스냅샷 수신 작업이 실패할 수 있습니다.

```
cannot receive incremental stream of tank/dana@snap2 into newtank/dana:
most recent snapshot of tank/dana@snap2 does not match incremental source
```

증분 스냅샷을 수신된 파일 시스템에도 수신해야 할 경우 원래 수신된 파일 시스템에서 파일 정보에 액세스해야 하면 atime 등록 정보를 off로 설정하십시오.

증분 *snap1* 소스는 스냅샷 이름의 마지막 구성 요소로 지정할 수 있습니다. 따라서 사용자가 *snap1*의 @ 기호 뒤에 이름을 지정하기만 하면 됩니다. snap1은 *snap2*와 같은 파일 시스템에서 생성된 것으로 간주됩니다. 예를 들면 다음과 같습니다.

```
sys1# zfs send -i snap1 tank/dana@snap2 | ssh sys2 zfs recv newtank/dana
```

이 단축 구문은 위 예의 증분 구문과 같습니다.

다른 파일 시스템 snapshot1에서 증분 스트림을 생성하려고 하면 다음과 같은 메시지가 표시됩니다.

```
cannot send 'pool/fs@name': not an earlier snapshot from the same fs
```

복사본을 여러 개 저장해야 할 경우에는 gzip 명령으로 ZFS 스냅샷 스트림 표현을 압축하십시오. 예를 들면 다음과 같습니다.

```
# zfs send pool/fs@snap | gzip > backupfile.gz
```

ZFS 스냅샷 수신

파일 시스템 스냅샷을 수신할 경우 다음 사항에 유념하십시오.

- 스냅샷과 파일 시스템이 모두 수신됩니다.
- 파일 시스템과 모든 종속 파일 시스템이 마운트 해제됩니다.
- 파일 시스템 수신 중에는 파일 시스템에 액세스할 수 없습니다.

- 수신할 원본 파일 시스템이 전송되는 동안에 존재하면 안됩니다.
- 파일 시스템 이름이 이미 존재할 경우 `zfs rename` 명령을 사용하여 파일 시스템의 이름을 바꿀 수 있습니다.

예를 들면 다음과 같습니다.

```
# zfs send tank/gozer@0830 > /bkups/gozer.083006
# zfs receive tank/gozer2@today < /bkups/gozer.083006
# zfs rename tank/gozer tank/gozer.old
# zfs rename tank/gozer2 tank/gozer
```

대상 파일 시스템을 변경하고 스냅샷에 대해 다른 증분 전송을 수행하려는 경우 먼저 수신 파일 시스템을 롤백해야 합니다.

다음 예를 고려하십시오. 먼저 다음과 같이 파일 시스템을 변경합니다.

```
sys2# rm newtank/dana/file.1
```

그런 다음 tank/dana@snap3에 대해 증분 전송을 수행합니다. 그러나 새 증분 스냅샷을 수신하려면 먼저 수신 파일 시스템을 롤백해야 합니다. 또는 `-F` 옵션을 사용하여 롤백 단계를 제거할 수 있습니다. 예를 들면 다음과 같습니다.

```
sys1# zfs send -i tank/dana@snap2 tank/dana@snap3 | ssh sys2 zfs recv -F newtank/dana
```

증분 스냅샷을 수신하는 경우 대상 파일 시스템이 이미 있어야 합니다.

파일 시스템을 변경한 다음 새 증분 스냅샷을 수신하기 위해 수신 파일 시스템을 롤백하지 않거나 `-F` 옵션을 사용하지 않을 경우, 다음과 비슷한 메시지가 표시됩니다.

```
sys1# zfs send -i tank/dana@snap4 tank/dana@snap5 | ssh sys2 zfs recv newtank/dana
cannot receive: destination has been modified since most recent snapshot
```

`-F` 옵션이 성공하기 전에 다음 검사가 수행됩니다.

- 가장 최근 스냅샷이 증분 소스와 일치하지 않을 경우 롤백과 수신이 모두 완료되지 않고 오류 메시지가 반환됩니다.
- `zfs receive` 명령에 지정된 증분 소스와 일치하지 않는 다른 파일 시스템의 이름을 실수로 제공할 경우 롤백과 수신이 모두 완료되지 않고 다음과 같은 오류 메시지가 반환됩니다.

  ```
  cannot send 'pool/fs@name': not an earlier snapshot from the same fs
  ```

ZFS 스냅샷 스트림에 다른 등록 정보 값 적용

특정 파일 시스템 등록 정보 값을 갖는 ZFS 스냅샷 스트림을 전송할 수 있지만, 스냅샷 스트림을 수신할 때 다른 로컬 등록 정보 값을 지정할 수 있습니다. 또는 스냅샷 스트림을 수신하여 원본 파일 시스템을 다시 만들 때 원본 등록 정보 값이 사용되도록 지정할 수 있습니다. 또한 스냅샷 스트림을 수신할 때 파일 시스템 등록 정보를 사용 안함으로 설정할 수 있습니다.

- 로컬 등록 정보 값을 수신 값(있는 경우)으로 되돌리려면 zfs inherit -S를 사용합니다. 등록 정보에 수신 값이 없는 경우 zfs inherit -S 명령은 -S 옵션 없이 zfs inherit 명령을 실행하는 것과 동일합니다. 등록 정보에 수신 값이 없으면 zfs inherit -S 명령을 실행하여 수신 값으로 되돌릴 때까지 zfs inherit 명령이 수신 값을 상속된 값으로 마스킹합니다.
- zfs get -o를 사용하여 새로운 기본값이 아닌 RECEIVED 열을 포함할 수 있습니다. 또는 zfs get -o all 명령을 사용하여 RECEIVED를 비롯한 모든 열을 포함할 수 있습니다.
- -R 옵션 없이 zfs send -p 옵션을 사용하면 송신 스트림에 등록 정보를 포함할 수 있습니다.
- zfs receive -e 옵션을 사용하면 전송된 스냅샷 이름의 마지막 요소를 사용하여 새 스냅샷 이름을 확인할 수 있습니다. 다음 예제에서는 poola/bee/cee@1 스냅샷을 poold/eee 파일 시스템에 전송하고 스냅샷 이름의 마지막 요소(cee@1)만 사용하여 수신된 파일 시스템 및 스냅샷을 만듭니다.

```
# zfs list -rt all poola
NAME              USED  AVAIL  REFER  MOUNTPOINT
poola             134K   134G    23K  /poola
poola/bee          44K   134G    23K  /poola/bee
poola/bee/cee      21K   134G    21K  /poola/bee/cee
poola/bee/cee@1      0      -    21K  -
# zfs send -R poola/bee/cee@1 | zfs receive -e poold/eee
# zfs list -rt all poold
NAME              USED  AVAIL  REFER  MOUNTPOINT
poold             134K   134G    23K  /poold
poold/eee          44K   134G    23K  /poold/eee
poold/eee/cee      21K   134G    21K  /poold/eee/cee
poold/eee/cee@1      0      -    21K  -
```

경우에 따라 전송 스트림의 파일 시스템 등록 정보가 수신 파일 시스템에 적용되지 않거나 로컬 파일 시스템 등록 정보(예: mountpoint 등록 정보 값)이 복원을 방해할 수 있습니다.

예를 들어 tank/data 파일 시스템의 경우 compression 등록 정보가 사용 안함으로 설정되어 있습니다. tank/data 파일 시스템의 스냅샷은 등록 정보(-p 옵션)을 사용하여 백업 풀에 전송되며 compression 등록 정보가 사용으로 설정된 상태로 수신됩니다.

```
# zfs get compression tank/data
NAME       PROPERTY     VALUE     SOURCE
tank/data  compression  off       default
# zfs snapshot tank/data@snap1
# zfs send -p tank/data@snap1 | zfs recv -o compression=on -d bpool
# zfs get -o all compression bpool/data
NAME        PROPERTY     VALUE     RECEIVED  SOURCE
bpool/data  compression  on        off       local
```

이 예에서 compression 등록 정보는 스냅샷이 bpool로 수신될 때 사용으로 설정됩니다. 따라서 bpool/data에 대한 compression 값은 on입니다.

복구를 위해 이 스냅샷 스트림이 새 풀인 restorepool로 전송될 경우 원본 스냅샷 등록 정보를 모두 유지하고자 할 수 있습니다. 이 경우 zfs send -b 명령을 사용하여 원본 스냅샷 등록 정보를 복원하십시오. 예를 들면 다음과 같습니다.

```
# zfs send -b bpool/data@snap1 | zfs recv -d restorepool
# zfs get -o all compression restorepool/data
NAME              PROPERTY     VALUE     RECEIVED  SOURCE
restorepool/data  compression  off       off       received
```

이 예에서 compression 값은 `off`인데, 이는 원본 `tank/data` 파일 시스템의 스냅샷 압축 값을 나타냅니다.

스냅샷 스트림에 로컬 파일 시스템 등록 정보 값이 있는데 이 스트림을 수신할 때 이 등록 정보를 사용 안함으로 설정하려면 `zfs receive -x` 명령을 사용하십시오. 예를 들어 다음 명령은 백업 풀에 예약된 모든 파일 시스템 등록 정보를 사용하여 홈 디렉토리 파일 시스템의 순환 스냅샷 스트림을 전송합니다. 이때 쿼터 등록 정보 값은 사용되지 않습니다.

```
# zfs send -R tank/home@snap1 | zfs recv -x quota bpool/home
# zfs get -r quota bpool/home
NAME                   PROPERTY  VALUE  SOURCE
bpool/home             quota     none   local
bpool/home@snap1       quota     -      -
bpool/home/lori        quota     none   default
bpool/home/lori@snap1  quota     -      -
bpool/home/mark        quota     none   default
bpool/home/mark@snap1  quota     -      -
```

`-x` 옵션을 사용하여 순환 스냅샷이 수신되지 않은 경우 수신된 파일 시스템에서 쿼터 등록 정보가 설정됩니다.

```
# zfs send -R tank/home@snap1 | zfs recv bpool/home
# zfs get -r quota bpool/home
NAME                   PROPERTY  VALUE  SOURCE
bpool/home             quota     none   received
bpool/home@snap1       quota     -      -
bpool/home/lori        quota     10G    received
bpool/home/lori@snap1  quota     -      -
bpool/home/mark        quota     10G    received
bpool/home/mark@snap1  quota     -      -
```

복잡한 ZFS 스냅샷 스트림 전송 및 수신

이 절에서는 `zfs send -I` 및 `-R` 옵션을 사용하여 보다 복잡한 스냅샷 스트림을 전송 및 수신하는 방법에 대해 설명합니다.

복잡한 ZFS 스냅샷 스트림을 전송 및 수신할 때는 다음 사항에 유의하십시오.

- `zfs send -I` 옵션을 사용하여 한 스냅샷의 모든 증분 스트림을 누적 스냅샷으로 전송할 수 있습니다. 또는 이 옵션으로 원본 스냅샷에서 증분 스트림을 전송하여 복제본을 만들 수 있습니다. 증분 스트림을 수락하려면 수신측에 원본 스냅샷이 있어야 합니다.
- `zfs send -R` 옵션을 사용하여 모든 종속 파일 시스템의 복제 스트림을 전송할 수 있습니다. 복제 스트림이 수신되면 등록 정보, 스냅샷, 종속 파일 시스템 및 복제본이 모두 유지됩니다.
- `zfs send -r` 옵션을 `-c` 옵션 없이 사용하고 `zfs send -R` 옵션 스트림 패키지를 사용하는 경우 상황에 따라 복제본의 `origin`을 생략합니다. 자세한 내용은 213 페이지 "ZFS 스냅샷 스트림 식별"을 참조하십시오.
- 두 옵션을 모두 사용하여 증분 복제 스트림을 전송할 수 있습니다.
  - 스냅샷 및 파일 시스템 **이름 바꾸기** 및 **삭제** 작업이 보존되므로 등록 정보에 대한 변경 사항이 보존됩니다.
  - 복제 스트림을 수신할 때 `zfs recv -F`가 지정되지 않은 경우 데이터 세트 **삭제** 작업이 무시됩니다. 이 경우 `zfs recv -F` 구문도 해당 **필요한 경우 롤백** 의미를 보존합니다.
  - 다른(`zfs send -R` 제외) `-i` 또는 `-I` 경우에도 `-I`가 사용된 경우 `snapA`와 `snapD` 간의 모든 스냅샷이 전송됩니다. `-i`가 사용된 경우 `snapD`(모든 종속 항목에 대한)만 전송됩니다.
- 이러한 새 유형의 `zfs send` 스트림을 수신하려면 수신 시스템이 스트림 전송을 지원하는 소프트웨어 버전을 실행 중이어야 합니다. 스트림 버전은 증분됩니다.

  그러나 최신 소프트웨어 버전을 사용할 경우 이전 풀 버전에서 스트림에 액세스할 수 없습니다. 예를 들어 최신 옵션을 사용하여 생성된 스트림은 버전 3 풀에서 또는 버전 3 풀로 전송 및 수신할 수 있습니다. 그러나 최신 옵션을 사용하여 전송된 스트림을 수신하려면 최신 소프트웨어가 실행 중이어야 합니다.

예 6–1 복잡한 ZFS 스냅샷 스트림 전송 및 수신

`zfs send -I` 옵션을 사용하여 증분 스냅샷 그룹을 하나의 스냅샷에 결합할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs send -I pool/fs@snapA pool/fs@snapD > /snaps/fs@all-I
```

그런 다음 `snapB`, `snapC` 및 `snapD`를 제거합니다.

```
# zfs destroy pool/fs@snapB
# zfs destroy pool/fs@snapC
# zfs destroy pool/fs@snapD
```

결합된 스냅샷을 수신하려면 다음 명령을 사용하십시오.

```
# zfs receive -d -F pool/fs < /snaps/fs@all-I
# zfs list
```

예 6–1 복잡한 ZFS 스냅샷 스트림 전송 및 수신 *(계속)*

```
NAME                     USED  AVAIL  REFER  MOUNTPOINT
pool                     428K  16.5G    20K  /pool
pool/fs                   71K  16.5G    21K  /pool/fs
pool/fs@snapA             16K      -  18.5K  -
pool/fs@snapB             17K      -    20K  -
pool/fs@snapC             17K      -  20.5K  -
pool/fs@snapD               0      -    21K  -
```

zfs send -I 명령으로 스냅샷과 복제 스냅샷을 결합하여 결합된 데이터 세트를 만들 수도 있습니다. 예를 들면 다음과 같습니다.

```
# zfs create pool/fs
# zfs snapshot pool/fs@snap1
# zfs clone pool/fs@snap1 pool/clone
# zfs snapshot pool/clone@snapA
# zfs send -I pool/fs@snap1 pool/clone@snapA > /snaps/fsclonesnap-I
# zfs destroy pool/clone@snapA
# zfs destroy pool/clone
# zfs receive -F pool/clone < /snaps/fsclonesnap-I
```

zfs send -R 명령을 사용하여 ZFS 파일 시스템과 모든 종속 파일 시스템 및 명명된 스냅샷까지 복제할 수 있습니다. 이 스트림이 수신되면 등록 정보, 스냅샷, 종속 파일 시스템 및 복제본이 모두 보존됩니다.

다음 예에서는 스냅샷이 사용자 파일 시스템용으로 생성되었습니다. 하나의 복제 스트림이 모든 사용자 스냅샷용으로 생성되었습니다. 다음으로, 원본 파일 시스템과 스냅샷이 삭제된 다음 복구되었습니다.

```
# zfs snapshot -r users@today
# zfs list
NAME                USED  AVAIL  REFER  MOUNTPOINT
users               187K  33.2G    22K  /users
users@today            0      -    22K  -
users/user1          18K  33.2G    18K  /users/user1
users/user1@today      0      -    18K  -
users/user2          18K  33.2G    18K  /users/user2
users/user2@today      0      -    18K  -
users/user3          18K  33.2G    18K  /users/user3
users/user3@today      0      -    18K  -
# zfs send -R users@today > /snaps/users-R
# zfs destroy -r users
# zfs receive -F -d users < /snaps/users-R
# zfs list
NAME                USED  AVAIL  REFER  MOUNTPOINT
users               196K  33.2G    22K  /users
users@today            0      -    22K  -
users/user1          18K  33.2G    18K  /users/user1
users/user1@today      0      -    18K  -
users/user2          18K  33.2G    18K  /users/user2
users/user2@today      0      -    18K  -
users/user3          18K  33.2G    18K  /users/user3
users/user3@today      0      -    18K  -
```

예 6–1 복잡한 ZFS 스냅샷 스트림 전송 및 수신 *(**계속**)*

다음 예에서는 `zfs send -R` 명령을 사용하여 `users` 파일 시스템과 종속 항목을 복제하고 복제된 스트림을 다른 풀인 `users2`로 보냈습니다.

```
# zfs create users2 mirror c0t1d0 c1t1d0
# zfs receive -F -d users2 < /snaps/users-R
# zfs list
NAME                  USED  AVAIL  REFER  MOUNTPOINT
users                 224K  33.2G    22K  /users
users@today              0      -    22K  -
users/user1           33K  33.2G    18K  /users/user1
users/user1@today     15K      -    18K  -
users/user2           18K  33.2G    18K  /users/user2
users/user2@today        0      -    18K  -
users/user3           18K  33.2G    18K  /users/user3
users/user3@today        0      -    18K  -
users2                188K  16.5G    22K  /users2
users2@today             0      -    22K  -
users2/user1          18K  16.5G    18K  /users2/user1
users2/user1@today       0      -    18K  -
users2/user2          18K  16.5G    18K  /users2/user2
users2/user2@today       0      -    18K  -
users2/user3          18K  16.5G    18K  /users2/user3
users2/user3@today       0      -    18K  -
```

ZFS 데이터 원격 복제

`zfs send` 및 `zfs recv` 명령을 사용하여 스냅샷 스트림 표현을 한 시스템에서 다른 시스템으로 원격을 복사할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs send tank/cindy@today | ssh newsys zfs recv sandbox/restfs@today
```

이 명령은 `tank/cindy@today` 스냅샷 데이터를 전송하고 `sandbox/restfs` 파일 시스템으로 수신합니다. 또한 `newsys` 시스템에 `restfs@today` 스냅샷도 만듭니다. 이 예에서는 사용자가 `ssh`를 원격 시스템에서 사용하도록 구성했습니다.

◆ ◆ ◆ 7 장

7 ACL 및 속성을 사용하여 Oracle Solaris ZFS 파일 보호

이 장에서는 액세스 제어 목록(ACL)을 사용하여 표준 UNIX 권한보다 훨씬 세부적인 권한을 통해 ZFS 파일을 보호하는 방법을 설명합니다.

이 장에서는 다음과 같은 내용을 다룹니다.

Solaris ACL 모델

이전 버전의 Solaris는 주로 POSIX 드래프트 ACL 사양에 기반한 ACL 구현을 지원했습니다. POSIX 드래프트 기반 ACL은 UFS 파일을 보호하는 데 사용되고 NFSv4 이전의 NFS 버전으로 변환됩니다.

NFSv4의 도입으로 새 ACL 모델은 UNIX와 비UNIX 클라이언트 간에 NFSv4가 제공하는 상호 운용성을 완벽히 지원합니다. 새 ACL 구현은 NFSv4 사양에 정의된 대로, NT 스타일 ACL에 기반한 다양한 의미론을 제공합니다.

새 ACL 모델의 주요 차이점은 다음과 같습니다.

- NFSv4 사양에 기반하며 NT 스타일 ACL과 비슷합니다.
- 보다 세부적인 액세스 권한 세트를 제공합니다. 자세한 내용은 표 7–2를 참조하십시오.
- `setfacl` 및 `getfacl` 명령이 아닌, `chmod` 및 `ls` 명령으로 설정 및 표시됩니다.
- 액세스 권한이 디렉토리에서 하위 디렉토리로 적용되는 방법을 지정하는 등 다양한 상속 의미론을 제공합니다. 자세한 내용은 228 페이지 “ACL 상속”을 참조하십시오.

양쪽 ACL 모델은 표준 파일 권한보다 훨씬 세분화된 액세스 제어를 제공합니다. POSIX 드래프트 ACL과 마찬가지로, 새 ACL은 여러 액세스 제어 항목(ACE)으로 구성됩니다.

POSIX 드래프트 스타일 ACL은 단일 항목을 사용하여 어떤 권한이 허용되고, 어떤 권한이 거부되는지 정의합니다. 새 ACL 모델에는 액세스 검사에 영향을 주는 ALLOW 및 DENY라는 두 가지 유형의 ACE가 있습니다. 따라서 권한 세트를 정의하는 단일 ACE로부터 ACE에 정의되지 않은 권한을 허용 또는 거부할지 여부를 유추할 수 없습니다.

NFSv4 스타일 ACL과 POSIX 드래프트 ACL 사이의 변환은 다음과 같습니다.

- cp, mv, tar, cpio, rcp 명령과 같은 ACL 인식 유틸리티를 사용하여 ACL이 포함된 UFS 파일을 ZFS 파일 시스템으로 전송하는 경우 POSIX 드래프트 ACL이 동등한 NFSv4 스타일 ACL로 변환됩니다.
- 일부 NFSv4 스타일 ACL은 POSIX 드래프트 ACL로 변환됩니다. NFSv4 스타일 ACL이 POSIX 드래프트 ACL로 변환되지 않으면 다음과 비슷한 메시지가 나타납니다.

  ```
  # cp -p filea /var/tmp
  cp: failed to set acl entries on /var/tmp/filea
  ```
- 현재 Solaris 릴리스를 실행하는 시스템에 보존 ACL 옵션(tar -p 또는 cpio -P)으로 UFS tar 또는 cpio 아카이브를 만들 경우, 이전 Solaris 릴리스를 실행하는 시스템에 아카이브를 추출할 때 ACL이 손실됩니다.

  파일은 모두 올바른 파일 모드로 추출되지만 ACL 항목이 무시됩니다.
- ufsrestore 명령을 사용하여 ZFS 파일 시스템에 데이터를 복원할 수 있습니다. 원래 데이터에 POSIX 스타일 ACL이 있는 경우 NFSv4 스타일 ACL로 변환됩니다.
- UFS 파일에 NFSv4 스타일 ACL을 설정하려고 시도하면 다음과 비슷한 메시지가 나타납니다.

  ```
  chmod: ERROR: ACL type's are different
  ```
- ZFS 파일에 POSIX 스타일 ACL을 설정하려고 시도하면 다음과 비슷한 메시지가 나타납니다.

  ```
  # getfacl filea
  File system doesn't support aclent_t style ACL's.
  See acl(5) for more information on Solaris ACL support.
  ```

다른 ACL 제한 사항 및 백업 제품에 대한 자세한 내용은 213 페이지 "다른 백업 제품으로 ZFS 데이터 저장"을 참조하십시오.

ACL 설정을 위한 구문 설명

다음과 같은 두 가지 기본 ACL 형식이 제공됩니다.

- **단순 ACL** – 기존의 UNIX user, group 및 owner 항목만 포함합니다.
- **복잡한 ACL** – 소유자, 그룹 및 모든 사용자뿐만 아니라 더 많은 항목을 포함하거나 상속 플래그 세트를 포함하거나 항목이 기존 방식이 아닌 다른 방식으로 정렬됩니다.

단순 ACL 설정을 위한 구문

```
chmod [options] A[index]{+|=}owner@ |group@ |everyone@:
access-permissions/...[:inheritance-flags]: deny | allow file
```

```
chmod [options] A-owner@, group@, everyone@:access-permissions
/...[:inheritance-flags]:deny | allow file ...
```

```
chmod [options] A[index]- file
```

복잡한 ACL 설정을 위한 구문

```
chmod [options] A[index]{+|=}user|group:name:access-permissions
/...[:inheritance-flags]:deny | allow file
```

```
chmod [options] A-user|group:name:access-permissions /...[:inheritance-flags]:deny |
allow file ...
```

```
chmod [options] A[index]- file
```

owner@, group@, everyone@
: 단순 ACL 구문을 위해 *ACL-entry-type*을 식별합니다. *ACL-entry-types*에 대한 설명은 표 7–1을 참조하십시오.

user or group:*ACL-entry-ID=username or groupname*
: 명시적 ACL 구문을 위해 *ACL-entry-type*을 식별합니다. 사용자 및 그룹 *ACL-entry-type*은 *ACL-entry-ID*와 *username* 또는 *groupname*도 포함해야 합니다. *ACL-entry-types*에 대한 설명은 표 7–1을 참조하십시오.

access-permissions/.../
: 부여 또는 거부되는 액세스 권한을 식별합니다. ACL 액세스 권한에 대한 설명은 표 7–2를 참조하십시오.

inheritance-flags
: ACL 상속 플래그의 선택적 목록을 식별합니다. ACL 상속 플래그에 대한 설명은 표 7–4를 참조하십시오.

deny | allow
: 액세스 권한이 부여 또는 거부되는지 여부를 식별합니다.

다음 예에서 owner@, group@, everyone@에 대한 *ACL-entry-ID* 값이 존재하지 않습니다.

```
group@:write_data/append_data/execute:deny
```

다음 예에서 특정 사용자(*ACL-entry-type*)가 ACL에 포함되므로 *ACL-entry-ID*가 있습니다.

```
0:user:gozer:list_directory/read_data/execute:allow
```

ACL 항목이 표시될 때 다음과 비슷하게 나타납니다.

```
2:group@:write_data/append_data/execute:deny
```

이 예에서 **2** 또는 *index-ID* 지정은 소유자, 특정 UID, 그룹, 모든 사람에 대한 여러 항목이 포함된 대형 ACL에서 ACL 항목을 식별합니다. `chmod` 명령에 *index-ID*를 지정하여 수정할 ACL의 부분을 식별할 수 있습니다. 예를 들어, 다음과 같이 인덱스 ID 3을 `chmod` 명령에 `A3`으로 식별할 수 있습니다.

```
chmod A3=user:venkman:read_acl:allow filename
```

다음 표에 ACL 항목 유형(owner/group/other의 ACL 표현)이 설명됩니다.

표 7–1 ACL 항목 유형

ACL 항목 유형	설명
`owner@`	객체의 소유자에 부여된 액세스를 지정합니다.
`group@`	객체의 소유 그룹에 부여된 액세스를 지정합니다.
`everyone@`	다른 ACL 항목과 일치하지 않는 모든 사용자나 그룹에 부여된 액세스를 지정합니다.
`user`	사용자 이름으로 객체의 추가 사용자에 부여된 액세스를 지정합니다. *ACL-entry-ID*에 *username* 또는 *userID*를 포함해야 합니다. 값이 유효한 숫자 UID나 *username*이 아닐 경우 잘못된 ACL 항목 유형입니다.
`group`	그룹 이름으로 객체의 추가 그룹에 부여된 액세스를 지정합니다. *ACL-entry-ID*에 *groupname* 또는 *groupID*를 포함해야 합니다. 값이 유효한 숫자 GID나 *groupname*이 아닐 경우 잘못된 ACL 항목 유형입니다.

다음 표에 ACL 액세스 권한이 설명됩니다.

표 7–2 ACL 액세스 권한

액세스 권한	Compact 액세스 권한	설명
`add_file`	`w`	새 파일을 디렉토리에 추가하는 권한입니다.
`add_subdirectory`	`p`	디렉토리에 하위 디렉토리를 만드는 권한입니다.
`append_data`	`p`	현재 구현되지 않습니다.
`delete`	`d`	파일을 삭제하는 권한입니다. 특정 `delete` 권한 동작에 대한 자세한 내용은 표 7–3을 참조하십시오.
`delete_child`	`D`	디렉토리 내의 파일 또는 디렉토리를 삭제하는 권한입니다. 특정 `delete_child` 권한 동작에 대한 자세한 내용은 표 7–3을 참조하십시오.
`execute`	`x`	파일을 실행하거나 디렉토리의 내용을 검색하는 권한입니다.
`list_directory`	`r`	디렉토리의 내용을 나열하는 권한입니다.
`read_acl`	`c`	ACL(`ls`)을 읽는 권한입니다.

표 7–2 ACL 액세스 권한 (계속)

액세스 권한	Compact 액세스 권한	설명
read_attributes	a	파일의 기본 속성(비ACL)을 읽는 권한입니다. 기본 속성은 stat 레벨 속성으로 간주됩니다. 이 액세스 마스크 비트를 허용하면 엔티티가 ls(1) 및 stat(2)를 실행할 수 있습니다.
read_data	r	파일의 내용을 읽는 권한입니다.
read_xattr	R	파일의 확장된 속성을 읽거나 파일의 확장된 속성 디렉토리에서 조회를 수행하는 권한입니다.
synchronize	s	현재 구현되지 않습니다.
write_xattr	W	확장된 속성을 만들거나 확장된 속성 디렉토리에 쓰는 권한입니다. 이 권한을 사용자에 부여하면 사용자가 파일의 확장된 속성 디렉토리를 만들 수 있습니다. 속성 파일의 권한은 사용자의 속성 액세스를 제어합니다.
write_data	w	파일의 내용을 수정하거나 바꾸는 권한입니다.
write_attributes	A	파일 또는 디렉토리와 연관된 시간을 모든 값으로 변경하는 권한입니다.
write_acl	C	ACL에 쓰는 권한, 또는 chmod 명령을 사용하여 ACL을 수정하는 능력입니다.
write_owner	o	파일의 소유자나 그룹을 변경하는 권한입니다. 또는 파일에 chown 또는 chgrp 명령을 실행하는 능력입니다. 파일의 소유권을 취하는 권한, 또는 파일의 그룹 소유권을 사용자가 구성원으로 속한 그룹으로 변경하는 권한입니다. 파일 또는 그룹 소유권을 모든 사용자나 그룹으로 변경하려면 PRIV_FILE_CHOWN 권한이 필요합니다.

다음 표에서는 ACL delete 및 delete_child 동작에 대해 자세히 설명합니다.

표 7–3 ACL delete 및 delete_child 권한 동작

부모 디렉토리 권한	대상 객체 권한		
	ACL에서 삭제 허용	ACL에서 삭제 거부	삭제 권한이 지정되지 않음
ACL에서 delete_child 허용	허용	허용	허용
ACL에서 delete_child 거부	허용	거부	거부
ACL에서 write 및 execute만 허용	허용	허용	허용

표 7–3 ACL delete 및 delete_child 권한 동작 (계속)

부모 디렉토리 권한	대상 객체 권한		
ACL에서 write 및 execute 거부	허용	거부	거부

ACL 상속

ACL 상속의 사용 목적은 새로 만든 파일 또는 디렉토리에서 부모 디렉토리의 기존 권한 비트를 무시하지 않고 ACL을 상속할 수 있도록 하는 것입니다.

기본적으로 ACL은 전파되지 않습니다. 디렉토리에 복잡한 ACL을 설정하면 후속 디렉토리로 상속되지 않습니다. 파일 또는 디렉토리에 ACL의 상속을 지정해야 합니다.

다음 표에 선택적 상속 플래그가 설명됩니다.

표 7–4 ACL 상속 플래그

상속 플래그	Compact 상속 플래그	설명
file_inherit	f	부모 디렉토리에서 디렉토리의 파일로만 ACL을 상속합니다.
dir_inherit	d	부모 디렉토리에서 디렉토리의 하위 디렉토리로만 ACL을 상속합니다.
inherit_only	i	부모 디렉토리에서 ACL을 상속하되, 디렉토리 자체가 아닌 새로 만든 파일 또는 하위 디렉토리에만 적용됩니다. 이 플래그에서 상속할 내용을 지정하려면 file_inherit 플래그나 dir_inherit 플래그(또는 둘 다)가 필요합니다.
no_propagate	n	부모 디렉토리에서 디렉토리의 첫번째 레벨 컨텐츠로만(두번째 레벨이나 후속 컨텐츠 아님) ACL을 상속합니다. 이 플래그에서 상속할 내용을 지정하려면 file_inherit 플래그나 dir_inherit 플래그(또는 둘 다)가 필요합니다.
-	해당 없음	부여된 권한이 없습니다.

더불어, aclinherit 파일 시스템 등록 정보를 사용하여 더 엄격한/덜 엄격한 기본 ACL 상속 정책을 파일 시스템에 설정할 수 있습니다. 자세한 내용은 다음 절을 참조하십시오.

ACL 등록 정보

ZFS 파일 시스템에는 ACL 상속의 특정 동작 및 chmod 작업과의 ACL 상호 작용을 결정하는 다음 ACL 등록 정보가 포함되어 있습니다.

- aclinherit – ACL 상속의 동작을 결정합니다. 다음과 같은 값이 있습니다.

- discard – 새 객체의 경우, 파일 또는 디렉토리를 만들 때 상속되는 ACL 항목이 없습니다. 파일 또는 디렉토리의 ACL은 파일 또는 디렉토리의 권한 모드와 같습니다.
- noallow – 새 객체의 경우, 액세스 유형이 deny인 상속 가능한 ACL 항목만 상속됩니다.
- restricted – 새 객체의 경우, ACL 항목을 상속할 때 write_owner 및 write_acl 권한이 제거됩니다.
- passthrough – 등록 정보 값이 passthrough로 설정된 경우 상속 가능한 ACE로 결정된 모드로 파일이 생성됩니다. 모드에 영향을 주는 상속 가능한 ACE가 없는 경우 응용 프로그램에서 요청한 모드에 따라 모드가 설정됩니다.
- passthrough-x – passthrough와 의미론이 같지만, 단 passthrough-x가 사용으로 설정된 경우 실행(x) 권한으로 파일이 생성됩니다(실행 권한이 파일 생성 모드로 설정되고 모드에 영향을 주는 상속 가능한 ACE가 있는 경우에 한함).

  aclinherit의 기본 모드는 restricted입니다.

- aclmode – 파일을 처음 만들 때 ACL 동작을 수정하거나 chmod 작업 도중 ACL 수정 방법을 제어합니다. 값은 다음과 같습니다.
  - discard – aclmode 등록 정보가 discard인 파일 시스템은 파일 모드를 나타내지 않는 ACL 항목을 모두 삭제합니다. 이것이 기본값입니다.
  - mask – aclmode 등록 정보가 mask인 파일 시스템은 사용자 또는 그룹 권한을 줄입니다. 파일 또는 디렉토리의 소유자와 동일한 UID를 가진 사용자 항목이 아닌 경우 그룹 권한 비트보다 크지 않도록 권한이 감소합니다. 이 경우 소유자 권한 비트보다 크지 않도록 ACL 권한이 감소합니다. 또한 명시적 ACL 세트 작업이 수행되지 않은 경우 모드 변경 후에도 마스크 값이 ACL을 유지합니다.
  - passthrough – aclmode 등록 정보가 passthrough인 파일 시스템은 파일 또는 디렉토리의 새 모드를 나타내는 데 필요한 ACL 항목의 생성 외에 ACL에 대한 변경 사항이 없음을 나타냅니다.

  aclmode의 기본 모드는 discard입니다.

  aclmode 등록 정보 사용에 대한 자세한 내용은 예 7–13을 참조하십시오.

ZFS 파일에 ACL 설정

ZFS로 구현된 대로 ACL은 ACL 항목의 배열로 구성됩니다. ZFS는 모든 파일에 ACL이 있는 **순수** ACL 모델을 제공합니다. 일반적으로 ACL은 **단순** 모델로, 기존 UNIX owner/group/other 항목만 나타냅니다.

ZFS 파일에 여전히 권한 비트와 모드가 있지만, 이러한 값이 ACL의 표현을 더욱 풍부하게 만듭니다. 따라서 파일의 권한을 변경하면 그에 따라 파일의 ACL이 업데이트됩니다. 더불어, 파일/디렉토리에 사용자 액세스가 부여된 복잡한 ACL을

제거할 경우, 그룹 또는 모든 사람에 액세스가 부여된 파일/디렉토리의 권한 비트 때문에 해당 사용자가 파일/디렉토리에 계속 액세스할 수 있었습니다. 모든 액세스 제어 결정이 파일 또는 디렉토리의 ACL에 표현된 권한으로 제어됩니다.

ZFS 파일의 ACL 액세스에 대한 기본 규칙은 다음과 같습니다.

- ZFS는 ACL에 나열된 순서대로, 위에서 아래로 ACL 항목을 처리합니다.
- 액세스 요청자와 일치하는 "사람"이 있는 ACL 항목만 처리됩니다.
- 일단 허용 권한이 부여된 후에는, 동일한 ACL 권한 세트에서 후속 ACL 거부 항목으로 권한을 거부할 수 없습니다.
- 파일 소유자는 권한이 명시적으로 거부되더라도 비조건부로 `write_acl` 권한이 부여됩니다. 그렇지 않으면 지정되지 않은 채 남은 권한은 거부됩니다.

  거부 권한의 사례나 액세스 권한이 누락된 경우, 권한 부속 시스템이 파일 소유자 또는 수퍼유저에 대해 어떤 액세스 요청이 거부되는지 결정합니다. 이 방식 덕분에 파일 소유자가 자신의 파일이 잠기는 것을 방지하고 수퍼유저가 복구 목적으로 파일을 수정할 수 있습니다.

디렉토리에 복잡한 ACL을 설정하면 ACL이 디렉토리의 자식에 자동으로 상속되지 않습니다. 복잡한 ACL을 설정하고 디렉토리의 자식에 상속되도록 하려면 ACL 상속 플래그를 사용해야 합니다. 자세한 내용은 표 7–4 및 236 페이지 “Verbose 형식으로 ZFS 파일에서 ACL 상속 설정”을 참조하십시오.

새 파일을 만들고 `umask` 값에 의존하는 경우 다음과 비슷한 기본 단순 ACL이 적용됩니다.

```
$ ls -v file.1
-rw-r--r--   1 root     root      206663 Jun 23 15:06 file.1
     0:owner@:read_data/write_data/append_data/read_xattr/write_xattr
         /read_attributes/write_attributes/read_acl/write_acl/write_owner
         /synchronize:allow
     1:group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
     2:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

이 예에서 각 사용자 범주(`owner@`, `group@`, `everyone@`)에 ACL 항목이 있습니다.

이 파일 ACL의 설명은 다음과 같습니다.

`0:owner@` 소유자가 파일의 내용을 읽고 수정할 수 있습니다(`read_data/write_data/append_data/read_xattr`). 또한 소유자가 시간 기록, 확장된 속성, ACL과 같은 파일 속성을 수정할 수 있습니다(`write_xattr/read_attributes/write_attributes/read_acl/write_acl`). 더불어, 소유자가 파일의 소유권을 수정할 수 있습니다(`write_owner:allow`).

`synchronize` 액세스 권한은 현재 구현되지 않습니다.

`1:group@` 그룹에 파일 및 파일 속성에 대한 읽기 권한이 부여됩니다(`read_data/read_xattr/read_attributes/read_acl:allow`).

2:everyone@ 사용자/그룹이 아닌 모든 사람에게 파일 및 파일 속성에 대한 읽기 권한이 부여됩니다(read_data/read_xattr/read_attributes/read_acl/synchronize:allow). synchronize 액세스 권한은 현재 구현되지 않습니다.

새 디렉토리를 만들고 umask 값에 의존하는 경우 기본 디렉토리 ACL은 다음과 비슷합니다.

```
$ ls -dv dir.1
drwxr-xr-x   2 root     root           2 Jul 20 13:44 dir.1
     0:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/read_xattr/write_xattr/execute/delete_child
         /read_attributes/write_attributes/read_acl/write_acl/write_owner
         /synchronize:allow
     1:group@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
     2:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

이 디렉토리 ACL의 설명은 다음과 같습니다.

0:owner@ 소유자가 디렉토리 내용을 읽고 수정하고(list_directory/read_data/add_file/write_data/add_subdirectory/append_data) 시간 기록, 확장 속성, ACL과 같은 파일 속성을 읽고 수정할 수 있습니다(/read_xattr/write_xattr/read_attributes/write_attributes/read_write_acl). 또한 소유자는 컨텐츠를 검색(execute)하고, 파일 또는 디렉토리를 삭제(delete_child)하고, 디렉토리의 소유권을 수정(write_owner:allow)할 수 있습니다.

synchronize 액세스 권한은 현재 구현되지 않습니다.

1:group@ 그룹이 디렉토리 내용 및 디렉토리 속성을 나열하고 읽을 수 있습니다. 더불어, 그룹에 디렉토리 내용을 검색할 수 있는 실행 권한이 있습니다(list_directory/read_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow).

2:everyone@ 사용자/그룹이 아닌 모든 사람에게 디렉토리 내용 및 디렉토리 속성에 대한 읽기 및 실행 권한이 부여됩니다(list_directory/read_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow). synchronize 액세스 권한은 현재 구현되지 않습니다.

Verbose 형식으로 ZFS 파일에서 ACL 설정 및 표시

chmod 명령을 사용하여 ZFS 파일의 ACL을 수정할 수 있습니다. 다음과 같은 ACL 수정용 chmod 구문은 *acl-specification*을 사용하여 ACL 형식을 식별합니다. *acl-specification*에 대한 설명은 224 페이지 "ACL 설정을 위한 구문 설명"을 참조하십시오.

- ACL 항목 추가
  - 사용자의 ACL 항목 추가

    ```
    % chmod A+acl-specification filename
    ```
  - *index-ID*로 ACL 항목 추가

    ```
    % chmod Aindex-ID+acl-specification filename
    ```

    이 구문은 지정된 *index-ID* 위치에 새 ACL 항목을 삽입합니다.
- ACL 항목 바꾸기

  ```
  % chmod A=acl-specification filename
  ```

  ```
  % chmod Aindex-ID=acl-specification filename
  ```
- ACL 항목 제거
  - *index-ID*로 ACL 항목 제거

    ```
    % chmod Aindex-ID- filename
    ```
  - 사용자로 ACL 항목 제거

    ```
    % chmod A-acl-specification filename
    ```
  - 파일에서 모든 복잡한 ACE 제거

    ```
    % chmod A- filename
    ```

Verbose ACL 정보는 ls - v 명령을 사용하여 표시됩니다. 예를 들면 다음과 같습니다.

```
# ls -v file.1
-rw-r--r--   1 root     root      206695 Jul 20 13:43 file.1
     0:owner@:read_data/write_data/append_data/read_xattr/write_xattr
         /read_attributes/write_attributes/read_acl/write_acl/write_owner
         /synchronize:allow
     1:group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
     2:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

Compact ACL 형식 사용에 대한 자세한 내용은 241 페이지 "Compact 형식으로 ZFS 파일에서 ACL 설정 및 표시"를 참조하십시오.

예 7–1 ZFS 파일의 단순 ACL 수정

이 절에서는 기존 UNIX 항목, 사용자, 그룹 및 기타 항목이 ACL에 포함되는 단순 ACL을 설정하고 표시하는 예를 제공합니다.

예 7–1 ZFS 파일의 단순 ACL 수정 *(계속)*

다음 예에서 단순 ACL이 file.1에 존재합니다.

```
# ls -v file.1
-rw-r--r--   1 root     root      206695 Jul 20 13:43 file.1
     0:owner@:read_data/write_data/append_data/read_xattr/write_xattr
         /read_attributes/write_attributes/read_acl/write_acl/write_owner
         /synchronize:allow
     1:group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
     2:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

다음 예에서 group@에 대해 write_data 권한이 부여됩니다.

```
# chmod A1=group@:read_data/write_data:allow file.1
# ls -v file.1
-rw-rw-r--   1 root     root      206695 Jul 20 13:43 file.1
     0:owner@:read_data/write_data/append_data/read_xattr/write_xattr
         /read_attributes/write_attributes/read_acl/write_acl/write_owner
         /synchronize:allow
     1:group@:read_data/write_data:allow
     2:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

다음 예에서 file.1에 대한 권한이 644로 다시 설정됩니다.

```
# chmod 644 file.1
# ls -v file.1
-rw-r--r--   1 root     root      206695 Jul 20 13:43 file.1
     0:owner@:read_data/write_data/append_data/read_xattr/write_xattr
         /read_attributes/write_attributes/read_acl/write_acl/write_owner
         /synchronize:allow
     1:group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
     2:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

예 7–2 ZFS 파일에 복잡한 ACL 설정

이 절은 복잡한 ACL 설정 및 표시의 예를 제공합니다.

다음 예에서 test.dir 디렉토리의 사용자 gozer에 대해 read_data/execute 권한이 추가됩니다.

```
# chmod A+user:gozer:read_data/execute:allow test.dir
# ls -dv test.dir
drwxr-xr-x+  2 root     root           2 Jul 20 14:23 test.dir
     0:user:gozer:list_directory/read_data/execute:allow
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/read_xattr/write_xattr/execute/delete_child
         /read_attributes/write_attributes/read_acl/write_acl/write_owner
         /synchronize:allow
     2:group@:list_directory/read_data/read_xattr/execute/read_attributes
```

예 7–2 ZFS 파일에 복잡한 ACL 설정 (계속)

```
        /read_acl/synchronize:allow
     3:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
        /read_acl/synchronize:allow
```

다음 예에서 사용자 gozer에 대한 read_data/execute 권한이 제거됩니다.

```
# chmod A0- test.dir
# ls -dv test.dir
drwxr-xr-x   2 root     root           2 Jul 20 14:23 test.dir
     0:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
        /append_data/read_xattr/write_xattr/execute/delete_child
        /read_attributes/write_attributes/read_acl/write_acl/write_owner
        /synchronize:allow
     1:group@:list_directory/read_data/read_xattr/execute/read_attributes
        /read_acl/synchronize:allow
     2:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
        /read_acl/synchronize:allow
```

예 7–3 ZFS 파일의 권한과 ACL 상호 작용

다음 ACL 예에서 ACL 설정 후 파일/디렉토리의 권한 비트 변경 사이의 상호 작용을 보여줍니다.

다음 예에서 단순 ACL이 file.2에 존재합니다.

```
# ls -v file.2
-rw-r--r--   1 root     root        2693 Jul 20 14:26 file.2
     0:owner@:read_data/write_data/append_data/read_xattr/write_xattr
        /read_attributes/write_attributes/read_acl/write_acl/write_owner
        /synchronize:allow
     1:group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
     2:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
        :allow
```

다음 예에서 ACL allow 권한이 everyone@에서 제거됩니다.

```
# chmod A2- file.2
# ls -v file.2
-rw-r-----   1 root     root        2693 Jul 20 14:26 file.2
     0:owner@:read_data/write_data/append_data/read_xattr/write_xattr
        /read_attributes/write_attributes/read_acl/write_acl/write_owner
        /synchronize:allow
     1:group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
```

이 출력 결과에서 파일의 권한 비트는 644에서 640으로 재설정됩니다. everyone@에 대한 ACL 허용 권한을 제거할 때 everyone@에 대한 읽기 권한이 파일의 권한 비트에서 효과적으로 제거되었습니다.

다음 예에서 기존 ACL이 everyone@에 대한 read_data/write_data 권한으로 바뀝니다.

예 7–3 ZFS 파일의 권한과 ACL 상호 작용 *(계속)*

```
# chmod A=everyone@:read_data/write_data:allow file.3
# ls -v file.3
-rw-rw-rw-   1 root     root        2440 Jul 20 14:28 file.3
     0:everyone@:read_data/write_data:allow
```

이 출력 결과에서 chmod 구문은 효과적으로 기존 ACL을 owner/group/everyone@에 대한 read_data/write_data:allow 권한(읽기/쓰기)으로 바꿉니다. 이 모델에서 everyone@은 모든 사용자나 그룹에 대한 액세스를 지정합니다. 소유자 및 그룹에 대한 권한을 대체할 owner@ 또는 group@ ACL 항목이 존재하지 않으므로 권한 비트가 666으로 설정됩니다.

다음 예에서 기존 ACL이 사용자 gozer에 대한 읽기 권한으로 바뀝니다.

```
# chmod A=user:gozer:read_data:allow file.3
# ls -v file.3
----------+  1 root     root        2440 Jul 20 14:28 file.3
     0:user:gozer:read_data:allow
```

이 출력 결과에서는 전통적인 파일 권한 구성 요소인 owner@, group@, everyone@에 대한 ACL 항목이 존재하지 않으므로 파일 권한이 000으로 계산됩니다. 파일 소유자는 다음과 같이 권한(및 ACL)을 재설정하여 이 문제를 해결할 수 있습니다.

```
# chmod 655 file.3
# ls -v file.3
-rw-r-xr-x   1 root     root        2440 Jul 20 14:28 file.3
     0:owner@:execute:deny
     1:owner@:read_data/write_data/append_data/read_xattr/write_xattr
         /read_attributes/write_attributes/read_acl/write_acl/write_owner
         /synchronize:allow
     2:group@:read_data/read_xattr/execute/read_attributes/read_acl
         /synchronize:allow
     3:everyone@:read_data/read_xattr/execute/read_attributes/read_acl
         /synchronize:allow
```

예 7–4 ZFS 파일의 단순 ACL 복원

chmod 명령을 사용하여 파일 또는 디렉토리의 모든 복잡한 ACL을 제거할 수 있습니다.

다음 예에서 2개의 복잡한 ACE가 test5.dir에 존재합니다.

```
# ls -dv test5.dir
drwxr-xr-x+  2 root     root           2 Jul 20 14:32 test5.dir
     0:user:lp:read_data:file_inherit:deny
     1:user:gozer:read_data:file_inherit:deny
     2:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/read_xattr/write_xattr/execute/delete_child
         /read_attributes/write_attributes/read_acl/write_acl/write_owner
         /synchronize:allow
     3:group@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

예 7–4 ZFS 파일의 단순 ACL 복원 *(계속)*

```
     4:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

다음 예에서 사용자 gozer 및 lp에 대한 복잡한 ACL이 제거됩니다. 남은 ACL은 owner@, group@, everyone@에 대한 기본값을 포함합니다.

```
# chmod A- test5.dir
# ls -dv test5.dir
drwxr-xr-x   2 root     root           2 Jul 20 14:32 test5.dir
     0:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/read_xattr/write_xattr/execute/delete_child
         /read_attributes/write_attributes/read_acl/write_acl/write_owner
         /synchronize:allow
     1:group@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
     2:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

Verbose 형식으로 ZFS 파일에서 ACL 상속 설정

파일 및 디렉토리에 ACL을 상속하거나 상속하지 않는 방법을 결정할 수 있습니다. 기본적으로 ACL은 전파되지 않습니다. 디렉토리에 복잡한 ACL을 설정하면 ACL이 후속 디렉토리에서 상속되지 않습니다. 파일 또는 디렉토리에 ACL의 상속을 지정해야 합니다.

aclinherit 등록 정보는 파일 시스템에 전역적으로 설정할 수 있습니다. 기본적으로 aclinherit는 restricted로 설정됩니다.

자세한 내용은 228 페이지 “ACL 상속”을 참조하십시오.

예 7–5 기본 ACL 상속 부여

기본적으로 ACL은 디렉토리 구조를 통해 전파되지 않습니다.

다음 예에서 read_data/write_data/execute의 복잡한 ACE가 test.dir의 사용자 gozer에 적용됩니다.

```
# chmod A+user:gozer:read_data/write_data/execute:allow test.dir
# ls -dv test.dir
drwxr-xr-x+  2 root     root           2 Jul 20 14:53 test.dir
     0:user:gozer:list_directory/read_data/add_file/write_data/execute:allow
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/read_xattr/write_xattr/execute/delete_child
         /read_attributes/write_attributes/read_acl/write_acl/write_owner
         /synchronize:allow
     2:group@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

예 7–5 기본 ACL 상속 부여 *(계속)*

```
     3:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

test.dir 하위 디렉토리를 만들 경우 사용자 gozer에 대한 ACE가 전파되지 않습니다. 사용자 gozer는 자신에게 부여된 sub.dir 권한이 파일 소유자, 그룹 구성원 또는 everyone@으로 액세스하는 경우 sub.dir에만 액세스할 수 있습니다.

```
# mkdir test.dir/sub.dir
# ls -dv test.dir/sub.dir
drwxr-xr-x   2 root     root           2 Jul 20 14:54 test.dir/sub.dir
     0:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/read_xattr/write_xattr/execute/delete_child
         /read_attributes/write_attributes/read_acl/write_acl/write_owner
         /synchronize:allow
     1:group@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
     2:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

예 7–6 파일 및 디렉토리에 ACL 상속 부여

다음 일련의 예는 file_inherit 플래그를 설정할 때 적용되는 파일 및 디렉토리 ACE를 식별합니다.

다음 예에서 사용자 gozer에 대해 test2.dir 디렉토리의 파일에 대한 read_data/write_data 권한이 추가되므로 이 사용자가 새로 만든 파일에 읽기 액세스할 수 있습니다.

```
# chmod A+user:gozer:read_data/write_data:file_inherit:allow test2.dir
# ls -dv test2.dir
drwxr-xr-x+  2 root     root           2 Jul 20 14:55 test2.dir
     0:user:gozer:read_data/write_data:file_inherit:allow
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/read_xattr/write_xattr/execute/delete_child
         /read_attributes/write_attributes/read_acl/write_acl/write_owner
         /synchronize:allow
     2:group@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
     3:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

다음 예에서 사용자 gozer의 권한이 새로 만든 test2.dir/file.2 파일에 적용됩니다. 부여된 ACL 상속 read_data:file_inherit:allow로 인해 사용자 gozer가 새로 만든 파일의 내용을 읽을 수 있습니다.

```
# touch test2.dir/file.2
# ls -v test2.dir/file.2
-rw-r--r--+  1 root     root           0 Jul 20 14:56 test2.dir/file.2
     0:user:gozer:read_data:inherited:allow
     1:owner@:read_data/write_data/append_data/read_xattr/write_xattr
```

예 7–6 파일 및 디렉토리에 ACL 상속 부여 *(계속)*

```
        /read_attributes/write_attributes/read_acl/write_acl/write_owner
        /synchronize:allow
     2:group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
     3:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
        :allow
```

이 파일 시스템에 대한 aclinherit 등록 정보가 기본 모드 restricted로 설정되므로 사용자 gozer에 file.2에 대한 write_data 권한이 없습니다(파일의 그룹 권한이 허용하지 않음).

file_inherit 또는 dir_inherit 플래그를 설정할 때 적용되는 inherit_only 권한을 사용하여 디렉토리 구조를 통해 ACL을 전파할 수 있습니다. 따라서 사용자 gozer는 파일 소유자이거나 파일 그룹 소유자의 구성원이 아닌 한, everyone@에서만 권한이 부여되거나 거부됩니다. 예를 들면 다음과 같습니다.

```
# mkdir test2.dir/subdir.2
# ls -dv test2.dir/subdir.2
drwxr-xr-x+  2 root     root           2 Jun 23 15:21 test2.dir/subdir.2
     0:user:gozer:list_directory/read_data/add_file/write_data:file_inherit
        /inherit_only:allow
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
        /append_data/read_xattr/write_xattr/execute/read_attributes
        /write_attributes/read_acl/write_acl/write_owner/synchronize:allow
     2:group@:list_directory/read_data/read_xattr/execute/read_attributes
        /read_acl/synchronize:allow
     3:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
        /read_acl/synchronize:allow
```

다음 일련의 예는 file_inherit 및 dir_inherit 플래그를 설정할 때 적용되는 파일 및 디렉토리 ACL을 식별합니다.

다음 예에서 사용자 gozer에 대해 새로 만든 파일/디렉토리에 상속되는 읽기, 쓰기, 실행 권한이 부여됩니다.

```
# chmod A+user:gozer:read_data/write_data/execute:file_inherit/dir_inherit:allow
test3.dir
# ls -dv test3.dir
drwxr-xr-x+  2 root     root           2 Jul 20 15:00 test3.dir
     0:user:gozer:list_directory/read_data/add_file/write_data/execute
        :file_inherit/dir_inherit:allow
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
        /append_data/read_xattr/write_xattr/execute/delete_child
        /read_attributes/write_attributes/read_acl/write_acl/write_owner
        /synchronize:allow
     2:group@:list_directory/read_data/read_xattr/execute/read_attributes
        /read_acl/synchronize:allow
     3:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
        /read_acl/synchronize:allow

# touch test3.dir/file.3
# ls -v test3.dir/file.3
```

예 7–6 파일 및 디렉토리에 ACL 상속 부여 *(계속)*

```
-rw-r--r--+  1 root     root           0 Jun 23 15:25 test3.dir/file.3
     0:user:gozer:read_data:allow
     1:owner@:read_data/write_data/append_data/read_xattr/write_xattr
         /read_attributes/write_attributes/read_acl/write_acl/write_owner
         /synchronize:allow
     2:group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
     3:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow

# mkdir test3.dir/subdir.1
# ls -dv test3.dir/subdir.1
drwxr-xr-x+  2 root     root           2 Jun 23 15:26 test3.dir/subdir.1
     0:user:gozer:list_directory/read_data/execute:file_inherit/dir_inherit
         :allow
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/read_xattr/write_xattr/execute/read_attributes
         /write_attributes/read_acl/write_acl/write_owner/synchronize:allow
     2:group@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
     3:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

위 예에서 group@ 및 everyone@에 대한 부모 디렉토리의 권한 비트가 쓰기 및 실행 권한을 거부하므로 사용자 gozer에 쓰기 및 실행 권한이 거부됩니다. 기본 aclinherit 등록 정보는 restricted이며, write_data 및 execute 권한이 상속되지 않습니다.

다음 예에서 사용자 gozer에 대해 새로 만든 파일에 상속되는 읽기, 쓰기, 실행 권한이 부여되지만 후속 디렉토리 내용으로 전파되지 않습니다.

```
# chmod A+user:gozer:read_data/write_data/execute:file_inherit/no_propagate:allow
test4.dir
# ls -dv test4.dir
drwxr--r--+  2 root     root           2 Mar  1 12:11 test4.dir
     0:user:gozer:list_directory/read_data/add_file/write_data/execute
         :file_inherit/no_propagate:allow
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/read_xattr/write_xattr/execute/delete_child
         /read_attributes/write_attributes/read_acl/write_acl/write_owner
         /synchronize:allow
     2:group@:list_directory/read_data/read_xattr/read_attributes/read_acl
         /synchronize:allow
     3:everyone@:list_directory/read_data/read_xattr/read_attributes/read_acl
         /synchronize:allow
```

다음 예에 나타난 것처럼, gozer의 read_data/write_data/execute 권한이 소유 그룹의 권한에 따라 축소됩니다.

```
# touch test4.dir/file.4
# ls -v test4.dir/file.4
-rw-r--r--+  1 root     root           0 Jun 23 15:28 test4.dir/file.4
     0:user:gozer:read_data:allow
     1:owner@:read_data/write_data/append_data/read_xattr/write_xattr
```

예 7–6 파일 및 디렉토리에 ACL 상속 부여 *(계속)*

```
        /read_attributes/write_attributes/read_acl/write_acl/write_owner
        /synchronize:allow
     2:group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
     3:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
        :allow
```

예 7–7 ACL 상속 모드를 Pass Through로 설정한 채 ACL 상속

tank/cindy 파일 시스템의 aclinherit 등록 정보가 passthrough로 설정된 경우 사용자 gozer가 새로 생성된 file.5에 대해 test4.dir에 적용된 ACL을 상속합니다.

```
# zfs set aclinherit=passthrough tank/cindy
# touch test4.dir/file.4
# ls -v test4.dir/file.4
-rw-r--r--+  1 root     root           0 Jun 23 15:35 test4.dir/file.4
     0:user:gozer:read_data:allow
     1:owner@:read_data/write_data/append_data/read_xattr/write_xattr
        /read_attributes/write_attributes/read_acl/write_acl/write_owner
        /synchronize:allow
     2:group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
     3:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
        :allow
```

예 7–8 ACL 상속 모드를 Discard로 설정한 채 ACL 상속

파일 시스템의 aclinherit 등록 정보가 discard로 설정된 경우 디렉토리의 권한 비트가 변경될 때 ACL이 잠재적으로 무시될 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs set aclinherit=discard tank/cindy
# chmod A+user:gozer:read_data/write_data/execute:dir_inherit:allow test5.dir
# ls -dv test5.dir
drwxr-xr-x+  2 root     root           2 Jul 20 14:18 test5.dir
     0:user:gozer:list_directory/read_data/add_file/write_data/execute
        :dir_inherit:allow
     1:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
        /append_data/read_xattr/write_xattr/execute/delete_child
        /read_attributes/write_attributes/read_acl/write_acl/write_owner
        /synchronize:allow
     2:group@:list_directory/read_data/read_xattr/execute/read_attributes
        /read_acl/synchronize:allow
     3:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
        /read_acl/synchronize:allow
```

나중에 디렉토리의 권한 비트를 축소하기로 결정하면 복잡한 ACL이 무시됩니다. 예를 들면 다음과 같습니다.

```
# chmod 744 test5.dir
# ls -dv test5.dir
drwxr--r--   2 root     root           2 Jul 20 14:18 test5.dir
     0:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
        /append_data/read_xattr/write_xattr/execute/delete_child
        /read_attributes/write_attributes/read_acl/write_acl/write_owner
```

예 7–8 ACL 상속 모드를 Discard로 설정한 채 ACL 상속 *(계속)*

```
        /synchronize:allow
     1:group@:list_directory/read_data/read_xattr/read_attributes/read_acl
        /synchronize:allow
     2:everyone@:list_directory/read_data/read_xattr/read_attributes/read_acl
        /synchronize:allow
```

예 7–9 ACL 상속 모드를 Noallow로 설정한 채 ACL 상속

다음 예에서 2개의 복잡한 ACL이 파일 상속과 함께 설정됩니다. 한 ACL은 `read_data` 권한을 허용하고, 한 ACL은 `read_data` 권한을 거부합니다. 이 예는 동일한 `chmod` 명령에서 두 ACE를 지정하는 방법을 보여줍니다.

```
# zfs set aclinherit=noallow tank/cindy
# chmod A+user:gozer:read_data:file_inherit:deny,user:lp:read_data:file_inherit:allow
test6.dir
# ls -dv test6.dir
drwxr-xr-x+  2 root     root           2 Jul 20 14:22 test6.dir
     0:user:gozer:read_data:file_inherit:deny
     1:user:lp:read_data:file_inherit:allow
     2:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
         /append_data/read_xattr/write_xattr/execute/delete_child
         /read_attributes/write_attributes/read_acl/write_acl/write_owner
         /synchronize:allow
     3:group@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
     4:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

다음 예에 나타난 것처럼, 새 파일을 만들 때 `read_data` 권한을 허용하는 ACL이 무시됩니다.

```
# touch test6.dir/file.6
# ls -v test6.dir/file.6
-rw-r--r--+  1 root     root           0 Jun 15 12:19 test6.dir/file.6
     0:user:gozer:read_data:inherited:deny
     1:owner@:read_data/write_data/append_data/read_xattr/write_xattr
         /read_attributes/write_attributes/read_acl/write_acl/write_owner
         /synchronize:allow
     2:group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
     3:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
         :allow
```

Compact 형식으로 ZFS 파일에서 ACL 설정 및 표시

권한을 표현하는 14개 고유 문자를 사용하는 Compact 형식으로 ZFS 파일에서 권한을 설정 및 표시할 수 있습니다. Compact 권한을 나타내는 문자는 표 7–2 및 표 7–4에 나열되어 있습니다.

ls -V 명령을 사용하여 파일 및 디렉토리에 대한 Compact ACL 목록을 표시할 수 있습니다. 예를 들면 다음과 같습니다.

```
# ls -V file.1
-rw-r--r--   1 root     root      206663 Jun 23 15:06 file.1
           owner@:rw-p--aARWcCos:------:allow
           group@:r-----a-R-c--s:------:allow
        everyone@:r-----a-R-c--s:------:allow
```

다음은 Compact ACL 출력 결과에 대한 설명입니다.

owner@ 소유자가 파일의 내용을 읽고 수정할 수 있습니다(rw=read_data/write_data), (p= append_data). 또한 소유자가 시간 기록, 확장된 속성, ACL과 같은 파일 속성을 수정할 수 있습니다(a=read_attributes, W=write_xattr, R=read_xattr, A=write_attributes, c=read_acl, C=write_acl). 더불어, 소유자가 파일의 소유권을 수정할 수 있습니다(o= write_owner).

synchronize(s) 액세스 권한은 현재 구현되지 않습니다.

group@ 그룹에 파일(r= read_data) 및 파일 속성(a=read_attributes , R=read_xattr, c= read_acl)에 대한 읽기 권한이 부여됩니다.

synchronize(s) 액세스 권한은 현재 구현되지 않습니다.

everyone@ 사용자/그룹이 아닌 모든 사람에게 파일 및 파일 속성에 대한 읽기 권한이 부여됩니다(r=read_data, a=append_data, R=read_xattr , c=read_acl, and s= synchronize).

synchronize(s) 액세스 권한은 현재 구현되지 않습니다.

Compact ACL 형식은 Verbose ACL 형식과 비교해 다음과 같은 이점이 있습니다.

- chmod 명령에 위치 인수로 권한을 지정할 수 있습니다.
- 권한 없음을 나타내는 하이픈(-) 문자를 제거할 수 있고 필요한 문자만 지정해야 합니다.
- 권한 및 상속 플래그가 동일한 방식으로 설정됩니다.

Verbose ACL 형식 사용에 대한 자세한 내용은 232 페이지 “Verbose 형식으로 ZFS 파일에서 ACL 설정 및 표시”를 참조하십시오.

예 7–10 Compact 형식으로 ACL 설정 및 표시

다음 예에서 단순 ACL이 file.1에 존재합니다.

```
# ls -V file.1
-rw-r--r--   1 root     root      206663 Jun 23 15:06 file.1
           owner@:rw-p--aARWcCos:------:allow
           group@:r-----a-R-c--s:------:allow
```

예 7–10 Compact 형식으로 ACL 설정 및 표시 *(계속)*

```
         everyone@:r-----a-R-c--s:------:allow
```

이 예에서 `file.1`의 사용자 `gozer`에 대해 `read_data/execute` 권한이 추가됩니다.

```
# chmod A+user:gozer:rx:allow file.1
# ls -V file.1
-rw-r--r--+  1 root     root      206663 Jun 23 15:06 file.1
        user:gozer:r-x-----------:------:allow
            owner@:rw-p--aARWcCos:------:allow
            group@:r-----a-R-c--s:------:allow
         everyone@:r-----a-R-c--s:------:allow
```

다음 예에서 사용자 `gozer`에 대해 Compact ACL 형식을 사용하여 새로 만든 파일 및 디렉토리에 상속되는 읽기, 쓰기, 실행 권한이 부여됩니다.

```
# chmod A+user:gozer:rwx:fd:allow dir.2
# ls -dV dir.2
drwxr-xr-x+  2 root     root           2 Jun 23 16:04 dir.2
        user:gozer:rwx-----------:fd----:allow
            owner@:rwxp--aARWcCos:------:allow
            group@:r-x---a-R-c--s:------:allow
         everyone@:r-x---a-R-c--s:------:allow
```

`ls -V` 출력 결과에서 Compact `chmod` 형식으로 권한 및 상속 플래그를 잘라서 붙여 넣을 수 있습니다. 예를 들어, 사용자 `gozer`에 대한 `dir.2`의 권한 및 상속 플래그를 `dir.2`의 사용자 `cindy`로 복제하려면 권한 및 상속 플래그(`rwx-----------:fd----:allow`)를 복사하여 `chmod` 명령으로 붙여 넣으십시오. 예를 들면 다음과 같습니다.

```
# chmod A+user:cindy:rwx-----------:fd----:allow dir.2
# ls -dV dir.2
drwxr-xr-x+  2 root     root           2 Jun 23 16:04 dir.2
       user:cindy:rwx-----------:fd----:allow
        user:gozer:rwx-----------:fd----:allow
            owner@:rwxp--aARWcCos:------:allow
            group@:r-x---a-R-c--s:------:allow
         everyone@:r-x---a-R-c--s:------:allow
```

예 7–11 ACL 상속 모드를 Pass Through로 설정한 채 ACL 상속

`aclinherit` 등록 정보가 `passthrough`로 설정된 파일 시스템은 상속 당시 ACL 항목에 어떤 수정도 없이 모든 상속 가능한 ACL 항목을 상속받습니다. 이 등록 정보가 `passthrough`로 설정된 경우 상속 가능한 ACE로 결정된 권한 모드로 파일이 생성됩니다. 권한 모드에 영향을 주는 상속 가능한 ACE가 없는 경우 응용 프로그램에서 요청한 모드에 따라 권한 모드가 설정됩니다.

다음 예는 Compact ACL 구문을 사용하여 `aclinherit` 모드를 `passthrough`로 설정한 채 권한 비트를 상속하는 방법을 보여줍니다.

예 7–11 ACL 상속 모드를 Pass Through로 설정한 채 ACL 상속 *(계속)*

이 예에서 ACL은 상속을 강제하도록 `test1.dir`에 설정됩니다. 새로 만든 파일에 대해 `owner@`, `group@`, `everyone@` ACL 항목을 만듭니다. 새로 만든 디렉토리는 `@owner`, `group@`, `everyone@` ACL 항목을 상속합니다.

```
# zfs set aclinherit=passthrough tank/cindy
# pwd
/tank/cindy
# mkdir test1.dir

# chmod A=owner@:rwxpcCosRrWaAdD:fd:allow,group@:rwxp:fd:allow,everyone@::fd:allow
test1.dir
# ls -Vd test1.dir
drwxrwx---+  2 root     root           2 Jun 23 16:10 test1.dir
            owner@:rwxpdDaARWcCos:fd----:allow
            group@:rwxp----------:fd----:allow
         everyone@:--------------:fd----:allow
```

이 예에서 새로 만든 파일이 새로 만든 파일로 상속되도록 지정된 ACL을 상속합니다.

```
# cd test1.dir
# touch file.1
# ls -V file.1
-rwxrwx---+  1 root     root           0 Jun 23 16:11 file.1
            owner@:rwxpdDaARWcCos:------:allow
            group@:rwxp----------:------:allow
         everyone@:--------------:------:allow
```

이 예에서 새로 만든 디렉토리가 이 디렉토리에 대한 액세스를 제어하는 ACE와 앞으로 새로 만들 디렉토리의 자식으로 전파하기 위한 ACE를 모두 상속합니다.

```
# mkdir subdir.1
# ls -dV subdir.1
drwxrwx---+  2 root     root           2 Jun 23 16:13 subdir.1
            owner@:rwxpdDaARWcCos:fd----:allow
            group@:rwxp----------:fd----:allow
         everyone@:--------------:fd----:allow
```

`fd----` 항목은 상속을 전파하기 위한 것이며 액세스 제어 중 고려되지 않습니다. 이 예에서 상속된 ACE가 존재하지 않는 다른 디렉토리에 단순 ACL로 파일이 생성됩니다.

```
# cd /tank/cindy
# mkdir test2.dir
# cd test2.dir
# touch file.2
# ls -V file.2
-rw-r--r--   1 root     root           0 Jun 23 16:15 file.2
            owner@:rw-p--aARWcCos:------:allow
            group@:r-----a-R-c--s:------:allow
         everyone@:r-----a-R-c--s:------:allow
```

예 7–12 ACL 상속 모드를 Pass Through-X로 설정한 채 ACL 상속

aclinherit=passthrough-x가 사용으로 설정된 경우 owner@, group@, everyone@에 대한 실행(x) 권한으로 파일이 생성됩니다(실행 권한이 파일 생성 모드로 설정되고 모드에 영향을 주는 상속 가능한 ACE가 있는 경우에 한함).

다음 예는 aclinherit 모드를 passthrough-x로 설정한 채 실행 권한을 상속하는 방법을 보여줍니다.

```
# zfs set aclinherit=passthrough-x tank/cindy
```

다음 ACL은 /tank/cindy/test1.dir에 설정되어 owner@에 대한 실행 가능한 ACL 파일 상속을 제공합니다.

```
# chmod A=owner@:rwxpcCosRrWaAdD:fd:allow,group@:rwxp:fd:allow,everyone@::fd:allow test1.dir
# ls -Vd test1.dir
drwxrwx---+  2 root     root           2 Jun 23 16:17 test1.dir
            owner@:rwxpdDaARWcCos:fd----:allow
            group@:rwxp----------:fd----:allow
         everyone@:--------------:fd----:allow
```

파일(file1)이 요청된 권한 0666으로 생성됩니다. 결과 권한은 0660입니다. 생성 모드에서 요청하지 않아서 실행 권한은 상속되지 않았습니다.

```
# touch test1.dir/file1
# ls -V test1.dir/file1
-rw-rw----+  1 root     root           0 Jun 23 16:18 test1.dir/file1
             owner@:rw-pdDaARWcCos:------:allow
             group@:rw-p----------:------:allow
          everyone@:--------------:------:allow
```

그 다음, testdir 디렉토리에서 cc 컴파일러를 사용하여 t라는 실행 파일이 생성됩니다.

```
# cc -o t t.c
# ls -V t
-rwxrwx---+  1 root     root        7396 Dec  3 15:19 t
                owner@:rwxpdDaARWcCos:------:allow
                group@:rwxp----------:------:allow
             everyone@:--------------:------:allow
```

결과 권한은 0770입니다. cc가 권한 0777을 요청했기 때문이며, 이에 따라 owner@, group@, everyone@ 항목에서 실행 권한이 상속됩니다.

예 7–13 ZFS 파일에서 chmod 작업과의 ACL 상호 작용

다음 예에서는 특정 aclmode 및 aclinherit 등록 정보 값이 기존 ACL 권한을 소유 그룹과 일치하도록 줄이거나 확장하기 위해 파일 또는 디렉토리 권한을 변경하는 chmod 작업과 기존 ACL의 상호 작용에 영향을 주는 방법에 대해 설명합니다.

예 7–13 ZFS 파일에서 chmod 작업과의 ACL 상호 작용 *(계속)*

이 예에서 aclmode 등록 정보는 mask로 설정되어 있고, aclinherit 등록 정보는 restricted로 설정되어 있습니다. 이 예의 ACL 권한은 변경 중인 권한을 더 쉽게 보여주는 Compact 모드로 표시됩니다.

원본 파일 및 그룹 소유권과 ACL 권한은 다음과 같습니다.

```
# zfs set aclmode=mask pond/whoville
# zfs set aclinherit=restricted pond/whoville

# ls -lV file.1
-rwxrwx---+  1 root     root      206695 Aug 30 16:03 file.1
                user:amy:r-----a-R-c---:-------:allow
               user:rory:r-----a-R-c---:-------:allow
         group:sysadmin:rw-p--aARWc---:-------:allow
            group:staff:rw-p--aARWc---:-------:allow
                 owner@:rwxp--aARWcCos:-------:allow
                 group@:rwxp--aARWc--s:-------:allow
              everyone@:------a-R-c--s:-------:allow
```

chown 작업은 file.1에 대한 파일 소유권을 변경하며, 소유자 amy가 출력을 표시합니다. 예를 들면 다음과 같습니다.

```
# chown amy:staff file.1
# su - amy
$ ls -lV file.1
-rwxrwx---+  1 amy      staff     206695 Aug 30 16:03 file.1
                user:amy:r-----a-R-c---:-------:allow
               user:rory:r-----a-R-c---:-------:allow
         group:sysadmin:rw-p--aARWc---:-------:allow
            group:staff:rw-p--aARWc---:-------:allow
                 owner@:rwxp--aARWcCos:-------:allow
                 group@:rwxp--aARWc--s:-------:allow
              everyone@:------a-R-c--s:-------:allow
```

다음 chmod 작업은 권한을 더 제한적인 모드로 변경합니다. 이 예에서는 수정된 sysadmin 그룹 및 staff 그룹의 ACL 권한이 소유 그룹의 권한을 초과하지 않습니다.

```
$ chmod 640 file.1
$ ls -lV file.1
-rw-r-----+  1 amy      staff     206695 Aug 30 16:03 file.1
                user:amy:r-----a-R-c---:-------:allow
               user:rory:r-----a-R-c---:-------:allow
         group:sysadmin:r-----a-R-c---:-------:allow
            group:staff:r-----a-R-c---:-------:allow
                 owner@:rw-p--aARWcCos:-------:allow
                 group@:r-----a-R-c--s:-------:allow
              everyone@:------a-R-c--s:-------:allow
```

다음 chmod 작업은 권한을 덜 제한적인 모드로 변경합니다. 이 예에서는 수정된 sysadmin 그룹 및 staff 그룹의 ACL 권한이 복원되어 소유 그룹과 동일한 권한을 허용합니다.

예 7–13 ZFS 파일에서 chmod 작업과의 ACL 상호 작용 *(계속)*

```
$ chmod 770 file.1
$ ls -lV file.1
-rwxrwx---+  1 amy      staff     206695 Aug 30 16:03 file.1
              user:amy:r-----a-R-c---:-------:allow
             user:rory:r-----a-R-c---:-------:allow
         group:sysadmin:rw-p--aARWc---:-------:allow
            group:staff:rw-p--aARWc---:-------:allow
                 owner@:rwxp--aARWcCos:-------:allow
                 group@:rwxp--aARWc--s:-------:allow
              everyone@:------a-R-c--s:-------:allow
```

◆ ◆ ◆ 8 장

8 Oracle Solaris ZFS 위임 관리

이 장에서는 권한이 없는 사용자가 ZFS 관리 작업을 수행할 수 있도록 위임 관리를 사용하는 방법에 대해 설명합니다.

이 장에서는 다음과 같은 내용을 다룹니다.

ZFS 위임 관리 개요

ZFS 위임 관리를 통해 특정 사용자, 그룹 또는 모두에게 세분화된 권한을 분산할 수 있습니다. 두 가지 유형의 위임 권한이 지원됩니다.

- `create`, `destroy`, `mount`, `snapshot` 등의 개별 권한을 명시적으로 위임할 수 있습니다.
- **권한 세트**라는 권한 그룹을 정의할 수 있습니다. 나중에 권한 세트를 업데이트할 수 있으며 세트의 모든 소비자에게 변경 사항이 자동으로 전달됩니다. 권한 세트는 `@` 기호로 시작하며 64자로 제한됩니다. 세트 이름에서 `@` 기호 뒤에 나오는 나머지 문자에 대해서는 일반적인 ZFS 파일 시스템 이름과 동일한 제한 사항이 적용됩니다.

ZFS 위임 관리는 RBAC 보안 모델과 유사한 기능을 제공합니다. ZFS 위임은 ZFS 저장소 풀 및 파일 시스템을 관리하는 데 있어 다음과 같은 이점을 제공합니다.

- 권한은 풀이 마이그레이션될 때마다 ZFS 저장소 풀을 따릅니다.
- 파일 시스템을 통한 권한 전파 방식을 제어할 수 있는 동적 상속을 제공합니다.
- 파일 시스템 작성자만 파일 시스템을 삭제할 수 있도록 구성할 수 있습니다.
- 특정 파일 시스템에 권한을 위임할 수 있습니다. 새로 만들어진 파일 시스템은 자동으로 권한을 선택할 수 있습니다.

- NFS 관리를 간소화합니다. 예를 들어, 명시적 권한이 있는 사용자는 NFS를 통해 적합한 .zfs/snapshot 디렉토리에 스냅샷을 만들 수 있습니다.

ZFS 작업을 분산할 때 위임 관리를 사용하는 것이 좋습니다. RBAC를 사용하여 일반적인 Oracle Solaris 관리 작업을 관리하는 방법은 **System Administration Guide: Security Services**의 제III부, "Roles, Rights Profiles, and Privileges"을 참조하십시오.

ZFS 위임 권한을 사용 안함으로 설정

풀의 delegation 등록 정보를 사용하여 위임 관리 기능을 제어합니다. 예를 들면 다음과 같습니다.

```
# zpool get delegation users
NAME  PROPERTY    VALUE       SOURCE
users  delegation  on          default
# zpool set delegation=off users
# zpool get delegation users
NAME  PROPERTY    VALUE       SOURCE
users  delegation  off         local
```

기본적으로 delegation 등록 정보는 사용으로 설정됩니다.

ZFS 권한 위임

다음 방법으로 zfs allow 명령을 사용하여 ZFS 파일 시스템의 권한을 비루트 사용자에게 위임할 수 있습니다.

- 사용자, 그룹 또는 모두에게 개별 권한을 위임할 수 있습니다.
- 사용자, 그룹 또는 모두에게 개별 권한 그룹을 **권한 세트**로 위임할 수 있습니다.
- 로컬로 현재 파일 시스템에만 권한을 위임할 수도 있고, 현재 파일 시스템의 모든 종속 항목에 권한을 위임할 수도 있습니다.

다음 표에서는 위임할 수 있는 작업과 위임 작업 수행에 필요한 종속 권한에 대해 설명합니다.

권한(하위 명령)	설명	종속성
allow	본인의 권한을 다른 사용자에게 부여할 수 있는 권한입니다.	피허용 권한도 있어야 합니다.
clone	데이터 세트의 스냅샷을 복제할 수 있는 권한입니다.	원래 파일 시스템에 create 권한과 mount 권한도 있어야 합니다.
create	종속 데이터 세트를 만들 수 있는 권한입니다.	mount 권한도 있어야 합니다.

권한(하위 명령)	설명	종속성
destroy	데이터 세트를 삭제할 수 있는 권한입니다.	mount 권한도 있어야 합니다.
diff	데이터 세트 내의 경로를 식별하기 위한 권한입니다.	비루트 사용자가 zfs diff 명령을 사용하려면 이 권한이 필요합니다.
hold	스냅샷을 유지하기 위한 권한입니다.	
mount	파일 시스템을 마운트 및 마운트 해제하고 볼륨 장치 연결을 만들고 삭제할 수 있는 권한입니다.	
promote	복제본을 데이터 세트로 승격시킬 수 있는 권한입니다.	원래 파일 시스템에 mount 권한과 promote 권한도 있어야 합니다.
receive	zfs receive 명령을 사용하여 종속 파일 시스템을 만들 수 있는 권한입니다.	mount 권한과 create 권한도 있어야 합니다.
release	스냅샷 유지를 해제하기 위한 권한입니다. 스냅샷을 삭제할 수 있습니다.	
rename	데이터 세트의 이름을 바꿀 수 있는 권한입니다.	새 상위에 create 권한과 mount 권한도 있어야 합니다.
rollback	스냅샷을 롤백할 수 있는 권한입니다.	
send	스냅샷 스트림을 보낼 수 있는 권한입니다.	
share	파일 시스템을 공유하고 공유를 해제할 수 있는 권한입니다.	NFS 공유를 만들려면 share 및 sharenfs를 모두 포함해야 합니다. SMB 공유를 만들려면 share 및 sharesmb를 모두 포함해야 합니다.
snapshot	데이터 세트의 스냅샷을 만들 수 있는 권한입니다.	

다음과 같은 권한 세트를 위임할 수 있지만 권한이 액세스, 읽기 또는 변경 권한으로 제한될 수 있습니다.

- groupquota
- groupused
- userprop
- userquota
- userused

또한 다음과 같은 ZFS 등록 정보 관리를 비루트 사용자에게 위임할 수 있습니다.

- aclinherit
- aclmode
- atime
- canmount
- casesensitivity
- checksum
- compression
- copies
- devices
- exec
- logbias
- mountpoint
- nbmand
- normalization
- primarycache
- quota
- readonly
- recordsize
- refquota
- refreservation
- reservation
- rstchown
- secondarycache
- setuid
- sharenfs
- sharesmb
- snapdir
- sync
- utf8only
- version
- volblocksize
- volsize
- vscan
- xattr
- zoned

이러한 등록 정보 중 일부는 데이터 세트를 만들 때만 설정할 수 있습니다. 이러한 등록 정보에 대한 자세한 내용은 169 페이지 “ZFS 등록 정보 소개”를 참조하십시오.

ZFS 권한 위임(zfs allow)

zfs allow 구문은 다음과 같습니다.

```
zfs allow -[ldugecs] everyone|user|group[,...] perm|@setname,...] filesystem| volume
```

굵게 표시된 다음 zfs allow 구문은 권한이 위임된 대상을 식별합니다.

zfs allow [-uge]|user|group|everyone [,...] *filesystem* | *volume*

쉼표로 구분된 목록으로 여러 항목을 지정할 수 있습니다. -uge 옵션을 지정하지 않을 경우 인수가 키워드 everyone, 사용자 이름, 그룹 이름 순으로 해석됩니다. 사용자 또는 그룹 이름을 "everyone"으로 지정하려면 -u 또는 -g 옵션을 사용하십시오. 그룹 이름을 사용자 이름과 동일하게 지정하려면 -g 옵션을 사용하십시오. -c 옵션은 만들 당시의 권한을 위임합니다.

굵게 표시된 다음 zfs allow 구문은 권한 및 권한 세트 지정 방식을 식별합니다.

zfs allow [-s] ... perm|@setname [,...] *filesystem* | *volume*

쉼표로 구분된 목록으로 여러 권한을 지정할 수 있습니다. 권한 이름은 ZFS 하위 명령 및 등록 정보와 동일합니다. 자세한 내용은 선행 절을 참조하십시오.

권한은 **권한 세트**로 통합될 수 있으며 -s 옵션으로 식별됩니다. 지정된 파일 시스템 및 종속 항목에 대해 다른 zfs allow 명령이 권한 세트를 사용할 수 있습니다. 권한 세트는 동적으로 평가되므로 세트에 대한 변경 사항이 즉시 업데이트됩니다. 권한 세트는 ZFS 파일 시스템과 동일한 명명 요구 사항을 따르지만 이름은 기호(@)로 시작해야 하며 64자를 초과할 수 없습니다.

굵게 표시된 다음 zfs allow 구문은 권한 위임 방식을 식별합니다.

zfs allow [-ld] ... ... *filesystem* | *volume*

-d 옵션이 함께 지정되지 않은 경우 -l 옵션은 권한이 지정된 파일 시스템에 대해서만 허용되며 종속 항목에 대해서는 허용되지 않음을 나타냅니다. -l 옵션이 함께 지정되지 않은 경우 -d 옵션은 권한이 종속 파일 시스템에 대해서만 허용되며 해당 파일 시스템에 대해서는 허용되지 않음을 나타냅니다. 어떤 옵션도 지정하지 않을 경우 권한이 파일 시스템 또는 볼륨 및 모든 종속 항목에 대해 허용됩니다.

ZFS 위임 권한 제거(zfs unallow)

zfs unallow 명령을 사용하여 이전에 위임된 권한을 제거할 수 있습니다.

예를 들어, 다음과 같이 create, destroy, mount 및 snapshot 권한을 위임했다고 가정합니다.

```
# zfs allow cindy create,destroy,mount,snapshot tank/home/cindy
# zfs allow tank/home/cindy
---- Permissions on tank/home/cindy --------------------------------
Local+Descendent permissions:
        user cindy create,destroy,mount,snapshot
```

권한을 제거하려면 다음 구문을 사용합니다.

```
# zfs unallow cindy tank/home/cindy
# zfs allow tank/home/cindy
```

ZFS 권한 위임(예)

예 8–1 개별 사용자에게 권한 위임

개별 사용자에게 create 및 mount 권한을 위임할 때는 사용자가 기본 마운트 지점에 대한 권한을 가지도록 해야 합니다.

예를 들어, 사용자 mark에게 tank 파일 시스템에 대한 create 및 mount 권한을 위임하려면 먼저 권한을 설정하십시오.

```
# chmod A+user:mark:add_subdirectory:fd:allow /tank/home
```

그런 다음 zfs allow 명령을 사용하여 create, destroy 및 mount 권한을 위임하십시오. 예를 들면 다음과 같습니다.

```
# zfs allow mark create,destroy,mount tank/home
```

그러면 사용자 mark가 tank/home 파일 시스템에 고유한 파일 시스템을 만들 수 있습니다. 예를 들면 다음과 같습니다.

```
# su mark
mark$ zfs create tank/home/mark
mark$ ^D
# su lp
$ zfs create tank/home/lp
cannot create 'tank/home/lp': permission denied
```

예 8–2 그룹에 create 및 destroy 권한 위임

다음 예에서는 staff 그룹의 모두가 tank/home 파일 시스템에 파일 시스템을 만들고 마운트하며 고유한 파일 시스템을 삭제할 수 있도록 파일 시스템을 설정하는 방법을 보여줍니다. 단, staff 그룹 구성원이 다른 사람의 파일 시스템을 삭제할 수는 없습니다.

```
# zfs allow staff create,mount tank/home
# zfs allow -c create,destroy tank/home
# zfs allow tank/home
---- Permissions on tank/home --------------------------------------
Create time permissions:
        create,destroy
Local+Descendent permissions:
        group staff create,mount
# su cindy
cindy% zfs create tank/home/cindy/files
cindy% exit
# su mark
mark% zfs create tank/home/mark/data
mark% exit
```

예 8–2 그룹에 create 및 destroy 권한 위임 *(계속)*

```
cindy% zfs destroy tank/home/mark/data
cannot destroy 'tank/home/mark/data': permission denied
```

예 8–3 올바른 파일 시스템 레벨에서 권한 위임

올바른 파일 시스템 레벨에서 사용자 권한을 위임해야 합니다. 예를 들어, 사용자 mark에게는 로컬 및 종속 파일 시스템에 대한 create, destroy 및 mount 권한이 위임되었습니다. 사용자 mark는 tank/home 파일 시스템의 스냅샷을 만들 수 있는 로컬 권한을 위임 받았지만 고유 파일 시스템의 스냅샷을 만들 수 없습니다. 따라서 올바른 파일 시스템 레벨에서 snapshot 권한을 위임 받은 것이 아닙니다.

```
# zfs allow -l mark snapshot tank/home
# zfs allow tank/home
---- Permissions on tank/home --------------------------------------
Create time permissions:
        create,destroy
Local permissions:
        user mark snapshot
Local+Descendent permissions:
        group staff create,mount
# su mark
mark$ zfs snapshot tank/home@snap1
mark$ zfs snapshot tank/home/mark@snap1
cannot create snapshot 'tank/home/mark@snap1': permission denied
```

종속 파일 시스템 레벨에서 사용자 mark에게 권한을 위임하려면 zfs allow -d 옵션을 사용하십시오. 예를 들면 다음과 같습니다.

```
# zfs unallow -l mark snapshot tank/home
# zfs allow -d mark snapshot tank/home
# zfs allow tank/home
---- Permissions on tank/home --------------------------------------
Create time permissions:
        create,destroy
Descendent permissions:
        user mark snapshot
Local+Descendent permissions:
        group staff create,mount
# su mark
$ zfs snapshot tank/home@snap2
cannot create snapshot 'tank/home@snap2': permission denied
$ zfs snapshot tank/home/mark@snappy
```

그러면 사용자 mark가 tank/home 파일 시스템 레벨 아래에만 스냅샷을 만들 수 있습니다.

예 8–4 복잡한 위임 권한 정의 및 사용

사용자 또는 그룹에 특정 권한을 위임할 수 있습니다. 예를 들어, 다음 zfs allow 명령은 staff 그룹에 특정 권한을 위임합니다. 또한 tank/home 파일 시스템이 만들어진 후 destroy 및 snapshot 권한이 위임됩니다.

예 8–4 복잡한 위임 권한 정의 및 사용 *(계속)*

```
# zfs allow staff create,mount tank/home
# zfs allow -c destroy,snapshot tank/home
# zfs allow tank/home
---- Permissions on tank/home --------------------------------------
Create time permissions:
        create,destroy,snapshot
Local+Descendent permissions:
        group staff create,mount
```

사용자 mark는 staff 그룹에 속하므로 tank/home에서 파일 시스템을 만들 수 있습니다. 또한 사용자 mark는 관련 권한을 가지고 있으므로 tank/home/mark2의 스냅샷을 만들 수 있습니다. 예를 들면 다음과 같습니다.

```
# su mark
$ zfs create tank/home/mark2
$ zfs allow tank/home/mark2
---- Permissions on tank/home/mark2 -------------------------------
Local permissions:
        user mark create,destroy,snapshot
---- Permissions on tank/home --------------------------------------
Create time permissions:
        create,destroy,snapshot
Local+Descendent permissions:
        group staff create,mount
```

하지만 사용자 mark는 관련 권한을 가지고 있지 않으므로 tank/home/mark에서 스냅샷을 만들 수 없습니다. 예를 들면 다음과 같습니다.

```
$ zfs snapshot tank/home/mark@snap1
cannot create snapshot 'tank/home/mark@snap1': permission denied
```

이 예에서 사용자 mark는 자신의 홈 디렉토리에 대한 create 권한을 가지고 있으므로 스냅샷을 만들 수 있습니다. 이 시나리오는 파일 시스템이 NFS 마운트 시스템인 경우 유용합니다.

```
$ cd /tank/home/mark2
$ ls
$ cd .zfs
$ ls
shares snapshot
$ cd snapshot
$ ls -l
total 3
drwxr-xr-x   2 mark    staff          2 Sep 27 15:55 snap1
$ pwd
/tank/home/mark2/.zfs/snapshot
$ mkdir snap2
$ zfs list
# zfs list -r tank/home
NAME                   USED  AVAIL  REFER  MOUNTPOINT
tank/home/mark          63K  62.3G    32K  /tank/home/mark
tank/home/mark2         49K  62.3G    31K  /tank/home/mark2
```

예 8–4 복잡한 위임 권한 정의 및 사용 *(계속)*

```
tank/home/mark2@snap1    18K      -    31K  -
tank/home/mark2@snap2      0      -    31K  -
$ ls
snap1  snap2
$ rmdir snap2
$ ls
snap1
```

예 8–5 ZFS 위임 권한 세트 정의 및 사용

다음 예에서는 권한 세트 @myset를 만들고 tank 파일 시스템에 대해 staff 그룹에 이 권한 세트와 rename 권한을 위임하는 방법을 보여 줍니다. 사용자 cindy는 staff 그룹에 속하며 tank에서 파일 시스템을 만들 수 있는 권한을 가지고 있지만, 사용자 lp는 tank에서 파일 시스템을 만들 수 있는 권한을 가지고 있지 않습니다.

```
# zfs allow -s @myset create,destroy,mount,snapshot,promote,clone,readonly tank
# zfs allow tank
---- Permissions on tank --------------------------------------------
Permission sets:
        @myset clone,create,destroy,mount,promote,readonly,snapshot
# zfs allow staff @myset,rename tank
# zfs allow tank
---- Permissions on tank --------------------------------------------
Permission sets:
        @myset clone,create,destroy,mount,promote,readonly,snapshot
Local+Descendent permissions:
        group staff @myset,rename
# chmod A+group:staff:add_subdirectory:fd:allow tank
# su cindy
cindy% zfs create tank/data
cindy% zfs allow tank
---- Permissions on tank --------------------------------------------
Permission sets:
        @myset clone,create,destroy,mount,promote,readonly,snapshot
Local+Descendent permissions:
        group staff @myset,rename
cindy% ls -l /tank
total 15
drwxr-xr-x   2 cindy   staff          2 Jun 24 10:55 data
cindy% exit
# su lp
$ zfs create tank/lp
cannot create 'tank/lp': permission denied
```

ZFS 위임 권한 표시(예)

다음 명령을 사용하여 권한을 표시할 수 있습니다.

 # zfs allow *dataset*

이 명령은 지정된 데이터 세트에서 설정되거나 허용되는 권한을 표시합니다. 출력 결과에는 다음 구성 요소가 포함됩니다.

- 권한 세트
- 개별 권한 또는 만들 당시의 권한
- 로컬 데이터 세트
- 로컬 및 종속 데이터 세트
- 종속 데이터 세트만

예 8–6 기본적인 위임 관리 권한 표시

다음 출력 결과는 사용자 `cindy`가 `tank/cindy` 파일 시스템에 대한 `create`, `destroy`, `mount`, snapshot 권한을 가지고 있음을 나타냅니다.

```
# zfs allow tank/cindy
       ---------------------------------------------------------------
       Local+Descendent permissions on (tank/cindy)
               user cindy create,destroy,mount,snapshot
```

예 8–7 복잡한 위임 관리 권한 표시

이 예의 출력 결과는 `pool/fred` 및 `pool` 파일 시스템에 대한 다음 권한을 나타냅니다.

`pool/fred` 파일 시스템의 경우:

- 다음과 같은 두 가지 권한 세트가 정의됩니다.
  - `@eng`(`create`, `destroy`, `snapshot`, `mount`, `clone`, `promote`, `rename`)
  - `@simple`(`create`, `mount`)
- `@eng` 권한 세트와 `mountpoint` 등록 정보에 대해 만들 당시의 권한이 설정됩니다. "만들 당시"는 파일 시스템 세트가 생성된 후 `@eng` 권한 세트와 `mountpoint` 등록 정보를 설정할 수 있는 권한이 위임됨을 의미합니다.
- 사용자 `tom`에게 `@eng` 권한 세트가 위임되고 사용자 `joe`에게 로컬 파일 시스템에 대한 `create`, `destroy` 및 `mount` 권한이 부여됩니다.
- 사용자 `fred`에게 로컬 및 종속 파일 시스템에 대한 `@basic` 권한 세트와 `share` 및 `rename` 권한이 위임됩니다.
- 사용자 `barney`와 `staff` 그룹에 종속 파일 시스템에 대해서만 `@basic` 권한 세트가 위임됩니다.

`pool` 파일 시스템의 경우:

- `@simple`(create, destroy, mount) 권한 세트가 정의됩니다.

예 8–7 복잡한 위임 관리 권한 표시 *(계속)*

- staff 그룹에 로컬 파일 시스템에 대한 @simple 권한 세트가 부여됩니다.

다음은 이 예에 대한 출력 결과입니다.

```
$ zfs allow pool/fred
---- Permissions on pool/fred --------------------------------------
Permission sets:
        @eng create,destroy,snapshot,mount,clone,promote,rename
        @simple create,mount
Create time permissions:
        @eng,mountpoint
Local permissions:
        user tom @eng
        user joe create,destroy,mount
Local+Descendent permissions:
        user fred @basic,share,rename
        user barney @basic
        group staff @basic
---- Permissions on pool -------------------------------------------
Permission sets:
        @simple create,destroy,mount
Local permissions:
        group staff @simple
```

ZFS 위임 권한 제거(예)

zfs unallow 명령을 사용하여 위임 권한을 제거할 수 있습니다. 예를 들어, 사용자 cindy가 tank/cindy 파일 시스템에 대한 create, destroy, mount 및 snapshot 권한을 가지고 있습니다.

```
# zfs allow cindy create,destroy,mount,snapshot tank/home/cindy
# zfs allow tank/home/cindy
---- Permissions on tank/home/cindy ----------------------------------
Local+Descendent permissions:
        user cindy create,destroy,mount,snapshot
```

다음 zfs unallow 구문은 사용자 cindy의 snapshot 권한을 tank/home/cindy 파일 시스템에서 제거합니다.

```
# zfs unallow cindy snapshot tank/home/cindy
# zfs allow tank/home/cindy
---- Permissions on tank/home/cindy ----------------------------------
Local+Descendent permissions:
        user cindy create,destroy,mount
cindy% zfs create tank/home/cindy/data
cindy% zfs snapshot tank/home/cindy@today
cannot create snapshot 'tank/home/cindy@today': permission denied
```

다른 예로, 사용자 mark가 tank/home/mark 파일 시스템에 대한 다음 권한을 가지고 있습니다.

```
# zfs allow tank/home/mark
---- Permissions on tank/home/mark -------------------------------------
Local+Descendent permissions:
        user mark create,destroy,mount
--------------------------------------------------------------------
```

다음 zfs unallow 구문은 사용자 mark에 대한 모든 권한을 tank/home/mark 파일 시스템에서 제거합니다.

```
# zfs unallow mark tank/home/mark
```

다음 zfs unallow 구문은 tank 파일 시스템에 대한 권한 세트를 제거합니다.

```
# zfs allow tank
---- Permissions on tank --------------------------------------------
Permission sets:
        @myset clone,create,destroy,mount,promote,readonly,snapshot
Create time permissions:
        create,destroy,mount
Local+Descendent permissions:
        group staff create,mount
# zfs unallow -s @myset tank
# zfs allow tank
---- Permissions on tank --------------------------------------------
Create time permissions:
        create,destroy,mount
Local+Descendent permissions:
        group staff create,mount
```

◆ ◆ ◆ 9 장

9 Oracle Solaris ZFS 고급 주제

이 장에서는 ZFS 볼륨, 영역이 설치된 Solaris 시스템에서 ZFS 사용, ZFS 대체 루트 풀 및 ZFS 권한 프로파일을 설명합니다.

이 장에서는 다음과 같은 내용을 다룹니다.

ZFS 볼륨

ZFS 볼륨은 블록 장치를 나타내는 데이터 세트입니다. ZFS 볼륨은 `/dev/zvol/{dsk,rdsk}/pool` 디렉토리에서 장치로 식별됩니다.

다음 예에서 5GB ZFS 볼륨 `tank/vol`이 만들어집니다.

```
# zfs create -V 5gb tank/vol
```

볼륨을 만들 때 예상치 않은 동작이 발생하지 않도록 예약이 볼륨의 초기 크기로 자동으로 설정됩니다. 예를 들어, 볼륨 크기를 줄이면 데이터 손상이 발생할 수 있습니다. 볼륨 크기를 변경할 때는 주의해야 합니다.

더불어, 크기가 변하는 볼륨의 스냅샷을 만드는 경우 스냅샷을 롤백하거나 스냅샷에서 복제본을 만들려고 시도하면 불일치가 일어날 수 있습니다.

볼륨에 적용할 수 있는 파일 시스템 등록 정보에 대한 자세한 내용은 표 5–1을 참조하십시오.

`zfs get` 또는 `zfs get all` 명령을 사용하여 ZFS 볼륨의 등록 정보를 표시할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs get all tank/vol
```

zfs get 출력의 volsize에 대해 표시된 물음표(?)는 I/O 오류가 발생했기 때문에 알 수 없는 값을 나타냅니다. 예를 들면 다음과 같습니다.

```
# zfs get -H volsize tank/vol
tank/vol        volsize ?       local
```

I/O 오류는 일반적으로 풀 장치에 문제가 발생했음을 나타냅니다. 풀 장치 문제 해결에 대한 자세한 내용은 274 페이지 "ZFS 저장소 풀을 사용하여 문제 식별"을 참조하십시오.

영역이 설치된 Solaris 시스템을 사용하는 경우 비전역 영역에서 ZFS 볼륨을 만들거나 복제할 수 없습니다. 이를 시도하면 실패하게 됩니다. 전역 영역에서 ZFS 볼륨 사용에 대한 자세한 내용은 266 페이지 "비전역 영역에 ZFS 볼륨 추가"를 참조하십시오.

ZFS 볼륨을 스왑 또는 덤프 장치로 사용

ZFS 루트 파일 시스템을 설치하거나 UFS 루트 파일 시스템에서 마이그레이션하는 동안, ZFS 루트 풀의 ZFS 볼륨에 스왑 장치가 만들어집니다. 예를 들면 다음과 같습니다.

```
# swap -l
swapfile                  dev    swaplo   blocks     free
/dev/zvol/dsk/rpool/swap 253,3        16  8257520  8257520
```

ZFS 루트 파일 시스템을 설치하거나 UFS 루트 파일 시스템에서 마이그레이션하는 동안, ZFS 루트 풀의 ZFS 볼륨에 덤프 장치가 만들어집니다. 덤프 장치가 설정된 후에는 관리가 필요 없습니다. 예를 들면 다음과 같습니다.

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

시스템이 설치된 후에 스왑 영역이나 덤프 장치를 변경해야 하는 경우 이전 Solaris 릴리스처럼 swap 및 dumpadm 명령을 사용합니다. 추가 스왑 볼륨을 만들어야 하는 경우 특정 크기의 ZFS 볼륨을 만든 후 해당 장치에 스왑을 사용으로 설정하십시오. 그런 다음 /etc/vfstab 파일에 새 스왑 장치에 대한 항목을 추가합니다. 예를 들면 다음과 같습니다.

```
# zfs create -V 2G rpool/swap2
# swap -a /dev/zvol/dsk/rpool/swap2
# swap -l
swapfile                   dev  swaplo blocks   free
/dev/zvol/dsk/rpool/swap  256,1      16 2097136 2097136
/dev/zvol/dsk/rpool/swap2 256,5      16 4194288 4194288
```

ZFS 파일 시스템의 파일로 스왑하지 마십시오. ZFS 스왑 파일 구성은 지원되지 않습니다.

스왑 및 덤프 볼륨의 크기 조정에 대한 자세한 내용은 147 페이지 "ZFS 스왑 장치 및 덤프 장치의 크기 조정"을 참조하십시오.

ZFS 볼륨을 Solaris iSCSI 대상으로 사용

shareiscsi 등록 정보를 볼륨에 설정하여 ZFS 볼륨을 iSCSI 대상으로 쉽게 만들 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs create -V 2g tank/volumes/v2
# zfs set shareiscsi=on tank/volumes/v2
# iscsitadm list target
Target: tank/volumes/v2
    iSCSI Name: iqn.1986-03.com.sun:02:984fe301-c412-ccc1-cc80-cf9a72aa062a
    Connections: 0
```

iSCSI 대상을 만든 후에는 iSCSI 초기화 프로그램을 설정합니다. Solaris iSCSI 대상 및 개시자에 대한 자세한 내용은 **System Administration Guide: Devices and File Systems**의 12 장, “Configuring Oracle Solaris iSCSI Targets (Tasks)”을 참조하십시오.

주 – Solaris iSCSI 대상은 iscsitadm 명령으로도 만들고 관리할 수 있습니다. ZFS 볼륨에 shareiscsi 등록 정보를 설정한 경우 동일한 대상 장치를 생성하기 위해 iscsitadm 명령을 사용하지 마십시오. 그렇지 않으면 동일한 장치에 대해 중복된 대상 정보가 만들어집니다.

---

iSCSI 대상으로서 ZFS 볼륨은 다른 ZFS 데이터 세트처럼 관리합니다. 그러나 rename, export, import 작업이 iSCSI 대상에 약간 다르게 작동합니다.

- ZFS 볼륨 이름을 바꿀 때 iSCSI 대상 이름은 그대로 유지됩니다. 예를 들면 다음과 같습니다.

  ```
  # zfs rename tank/volumes/v2 tank/volumes/v1
  # iscsitadm list target
  Target: tank/volumes/v1
      iSCSI Name: iqn.1986-03.com.sun:02:984fe301-c412-ccc1-cc80-cf9a72aa062a
      Connections: 0
  ```

- 공유 ZFS 볼륨을 포함하는 풀을 내보내면 대상이 제거됩니다. 공유 ZFS 볼륨을 포함하는 풀을 가져오면 대상이 공유됩니다. 예를 들면 다음과 같습니다.

  ```
  # zpool export tank
  # iscsitadm list target
  # zpool import tank
  # iscsitadm list target
  Target: tank/volumes/v1
      iSCSI Name: iqn.1986-03.com.sun:02:984fe301-c412-ccc1-cc80-cf9a72aa062a
      Connections: 0
  ```

모든 iSCSI 대상 구성 정보는 데이터 세트에 저장됩니다. NFS 공유 파일 시스템과 마찬가지로, 여러 시스템에 가져온 iSCSI 대상은 적절히 공유됩니다.

영역이 설치된 Solaris 시스템에서 ZFS 사용

다음 절은 Oracle Solaris 영역이 있는 시스템에서 ZFS를 사용하는 방법을 설명합니다.

Oracle Solaris Live Upgrade로 마이그레이션/패치될 ZFS 루트 파일 시스템이 포함된 시스템에 영역을 구성하는 방법은 131 페이지 "Live Upgrade를 사용하여 영역이 있는 시스템 마이그레이션 또는 업그레이드(Solaris 10 10/08)" 또는 136 페이지 "Oracle Solaris Live Upgrade를 사용하여 영역이 있는 시스템 마이그레이션 또는 업그레이드(Solaris 10 5/09 이상)"를 참조하십시오.

ZFS 데이터 세트를 영역과 연관시킬 때 다음 사항에 유의하십시오.

- 관리 컨트롤러 위임 여부에 상관없이 ZFS 파일 시스템을 추가하거나 비전역 영역에 복제할 수 있습니다.
- 비전역 영역에 ZFS 볼륨을 장치로 추가할 수 있습니다.
- 지금은 ZFS 스냅샷을 영역과 연관시킬 수 없습니다.

다음 절에서 ZFS 데이터 세트는 파일 시스템이나 복제본을 말합니다.

데이터 세트를 추가하면 (영역 관리자가 기본 파일 시스템 계층에서 등록 정보를 제어하거나 새 파일 시스템을 만들 수 없더라도) 비전역 영역에서 디스크 공간을 전역 영역과 공유할 수 있습니다. 이 작업은 다른 유형의 파일 시스템을 영역에 추가하는 것과 같으며, 주 목적이 전적으로 공통 디스크 공간을 공유하는 것일 때 사용해야 합니다.

ZFS에서는 데이터 세트가 비전역 영역에 위임되므로 영역 관리자로 데이터 세트와 모든 자식을 완전히 제어할 수 있습니다. 영역 관리자는 파일 시스템을 만들고 삭제하거나, 해당 데이터 세트 내에 복제하거나, 데이터 세트의 등록 정보를 수정할 수 있습니다. 영역 관리자는 위임된 데이터 세트에 설정된 최상위 레벨의 쿼터 초과를 비롯하여 영역에 추가되지 않은 데이터 세트에 영향을 미칠 수 없습니다.

Oracle Solaris 영역이 설치된 시스템에서 ZFS로 작업할 때 다음 사항을 고려하십시오.

- 비전역 영역에 추가된 ZFS 파일 시스템에서 `mountpoint` 등록 정보가 `legacy`로 설정되어야 합니다.
- 비전역 영역을 구성할 때 ZFS 데이터 세트를 비전역 영역에 추가하지 마십시오. 대신, 영역을 설치한 후에 ZFS 데이터 세트를 추가하십시오.
- 소스 `zonepath`와 대상 `zonepath`가 모두 ZFS 파일 시스템에 상주하고 동일한 풀에 있는 경우 `zoneadm clone`이 영역 복제를 위해 ZFS 복제를 자동으로 사용합니다. `zoneadm clone` 명령은 소스 `zonepath`의 ZFS 스냅샷을 만들고 대상 `zonepath`를 설정합니다.

영역 복제를 위해 zfs clone 명령을 사용할 수 없습니다. 자세한 내용은 **시스템 관리 설명서: Oracle Solaris Containers-리소스 관리 및 Oracle Solaris 영역**의 제II부, "영역"을 참조하십시오.

- ZFS 파일 시스템을 비전역 영역에 위임하는 경우 Oracle Solaris Live Upgrade를 사용하기 전에 비전역 영역에서 해당 파일 시스템을 제거해야 합니다. 그렇지 않으면 읽기 전용 파일 시스템 오류로 인해 Oracle Live Upgrade를 실패합니다.

비전역 영역에 ZFS 파일 시스템 추가

주 목적이 전적으로 전역 영역과 공간을 공유하는 것일 때 ZFS 파일 시스템을 일반 파일 시스템으로 추가할 수 있습니다. 비전역 영역에 추가된 ZFS 파일 시스템에서 mountpoint 등록 정보가 legacy로 설정되어야 합니다. 예를 들어, tank/zone/zion 파일 시스템을 비전역 영역에 추가할 경우 다음과 같이 전역 영역에 mountpoint 등록 정보를 설정합니다.

```
# zfs set mountpoint=legacy tank/zone/zion
```

zonecfg 명령의 add fs 하위 명령을 사용하여 비전역 영역에 ZFS 파일 시스템을 추가할 수 있습니다.

다음 예에서 전역 영역 관리자에 의해 전역 영역에서 비전역 영역으로 ZFS 파일 시스템이 추가됩니다.

```
# zonecfg -z zion
zonecfg:zion> add fs
zonecfg:zion:fs> set type=zfs
zonecfg:zion:fs> set special=tank/zone/zion
zonecfg:zion:fs> set dir=/opt/data
zonecfg:zion:fs> end
```

이 구문은 ZFS 파일 시스템 tank/zone/zion을 이미 구성된 zion 영역에 추가합니다. 이 영역은 /opt/data에 마운트되어 있습니다. 파일 시스템의 mountpoint 등록 정보는 legacy로 설정해야 하고 파일 시스템은 다른 위치에 이미 마운트될 수 없습니다. 영역 관리자는 파일 시스템 내에서 파일을 만들고 삭제할 수 있습니다. 파일 시스템은 다른 위치에 다시 마운트할 수 없고 영역 관리자가 atime, readonly, compression 등의 등록 정보를 파일 시스템에서 변경할 수 없습니다. 전역 영역 관리자는 파일 시스템의 등록 정보를 설정하고 제어할 책임이 있습니다.

zonecfg 명령 및 zonecfg로 리소스 유형을 구성하는 방법에 대한 자세한 내용은 **시스템 관리 설명서: Oracle Solaris Containers-리소스 관리 및 Oracle Solaris 영역**의 제II부, "영역"을 참조하십시오.

비전역 영역에 데이터 세트 위임

저장소 관리를 영역에 위임하는 주 목적을 충족하기 위해 ZFS는 zonecfg 명령의 add dataset 하위 명령을 통해 비전역 영역에 데이터 세트를 추가할 수 있도록 지원합니다.

다음 예에서 전역 영역 관리자에 의해 전역 영역에서 비전역 영역으로 ZFS 파일 시스템이 위임됩니다.

```
# zonecfg -z zion
zonecfg:zion> add dataset
zonecfg:zion:dataset> set name=tank/zone/zion
zonecfg:zion:dataset> end
```

파일 시스템 추가와 달리, 이 구문에서 ZFS 파일 시스템 tank/zone/zion이 이미 구성된 zion 영역에 표시됩니다. 영역 관리자는 파일 시스템 등록 정보를 설정하고 자손 파일 시스템을 만들 수 있습니다. 더불어, 영역 관리자는 스냅샷을 만들거나 복제하고, 아니면 전체 파일 시스템 계층을 제어할 수 있습니다.

파일 시스템 추가와 달리, 이 구문에서 ZFS 파일 시스템 tank/zone/zion이 이미 구성된 zion 영역에 표시됩니다. zion 영역 내에서 이 파일 시스템은 tank/zone/zion으로 액세스할 수 없지만 tank라는 **가상 풀**로 액세스할 수 있습니다. 위임된 파일 시스템 별칭은 원래 풀의 뷰를 영역에 가상 풀로 제공합니다. 별칭 등록 정보는 가상 풀의 이름을 지정합니다. 별칭을 지정하지 않으면 파일 시스템 이름의 마지막 구성 요소와 일치하는 기본 별칭이 사용됩니다. 특정 별칭을 제공하지 않은 경우 위 예의 기본 별칭은 zion입니다.

위임된 데이터 세트 내에서 영역 관리자는 파일 시스템 등록 정보를 설정하고 종속 파일 시스템을 만들 수 있습니다. 더불어, 영역 관리자는 스냅샷을 만들거나 복제하고, 아니면 전체 파일 시스템 계층을 제어할 수 있습니다. 위임된 파일 시스템 내에 ZFS 볼륨을 만들 경우 장치 리소스로 추가된 ZFS 볼륨과 충돌할 수 있습니다. 자세한 내용은 다음 절을 참조하십시오.

Oracle Solaris Live Upgrade를 사용하여 ZFS BE를 비전역 영역으로 업그레이드하는 경우 먼저 위임된 데이터 세트를 제거하십시오. 그렇지 않으면 읽기 전용 파일 시스템 오류로 인해 Oracle Solaris Live Upgrade를 실패합니다. 예를 들면 다음과 같습니다.

```
zonecfg:zion>
zonecfg:zion> remove dataset name=tank/zone/zion
zonecfg:zion1> exit
```

영역 내에서 허용되는 작업에 대한 자세한 내용은 267 페이지 "영역 내에서 ZFS 등록 정보 관리"를 참조하십시오.

비전역 영역에 ZFS 볼륨 추가

zonecfg 명령의 add dataset 하위 명령을 사용하여 ZFS 볼륨을 비전역 영역에 추가할 수 없습니다. 그러나 zonecfg 명령의 add device 하위 명령을 사용하면 볼륨을 영역에 추가할 수 있습니다.

다음 예에서 전역 영역 관리자에 의해 전역 영역에서 비전역 영역으로 ZFS 볼륨이 추가됩니다.

```
# zonecfg -z zion
zion: No such zone configured
Use 'create' to begin configuring a new zone.
zonecfg:zion> create
zonecfg:zion> add device
zonecfg:zion:device> set match=/dev/zvol/dsk/tank/vol
zonecfg:zion:device> end
```

이 구문은 tank/vol 볼륨을 zion 영역에 추가합니다.

원시 볼륨을 영역에 추가하면 볼륨이 물리적 장치에 해당하지 않더라도 보안 위험에 노출될 수 있습니다. 특히, 영역 관리자가 마운트를 시도할 때 시스템 장애를 일으키는 잘못된 형태의 파일 시스템을 만들 수 있습니다. 영역에 장치 추가와 관련된 보안 위험에 대한 자세한 내용은 268 페이지 "zoned 등록 정보 이해"를 참조하십시오.

영역에 장치 추가에 대한 자세한 내용은 **시스템 관리 설명서: Oracle Solaris Containers-리소스 관리 및 Oracle Solaris 영역**의 제II부, "영역"을 참조하십시오.

영역 내에서 ZFS 저장소 풀 사용

ZFS 저장소 풀은 영역 내에서 만들거나 수정할 수 없습니다. 위임 관리 모델은 중앙집중 방식으로 전역 영역 내에서 실제 저장 장치를 제어하고 비전역 영역에서 가상 저장소를 제어합니다. 풀 레벨 데이터 세트를 영역에 추가할 수 있더라도 장치 만들기, 추가, 제거 등 풀의 물리적 특성을 수정하는 명령은 영역 내에서 허용되지 않습니다. zonecfg 명령의 add device 하위 명령을 사용하여 물리적 장치를 영역에 추가하거나 파일을 사용하더라도 zpool 명령으로 영역 내에서 새 풀을 만들 수 없습니다.

영역 내에서 ZFS 등록 정보 관리

데이터 세트가 영역에 위임된 후에 영역 관리자가 특정 데이터 세트 등록 정보를 제어할 수 있습니다. 데이터 세트가 영역에 위임된 후에는 모든 조상이 읽기 전용 데이터 세트로 표시되고, 모든 자손인 데이터 세트 자체가 쓰기 가능합니다. 예를 들어, 다음 구성을 고려하십시오.

```
global# zfs list -Ho name
tank
tank/home
tank/data
tank/data/matrix
tank/data/zion
tank/data/zion/home
```

tank/data/zion이 영역에 추가된 경우 각 데이터 세트는 다음 등록 정보를 갖게 됩니다.

데이터 세트	표시 가능	쓰기 가능	변경할 수 없는 등록 정보
tank	예	아니오	-

데이터 세트	표시 가능	쓰기 가능	변경할 수 없는 등록 정보
tank/home	아니오	-	-
tank/data	예	아니오	-
tank/data/matrix	아니오	-	-
tank/data/zion	예	예	sharenfs, zoned, quota, reservation
tank/data/zion/home	예	예	sharenfs, zoned

tank/zone/zion의 모든 부모는 읽기 전용으로 표시되고, 모든 자손은 쓰기 가능하고, 부모 계층에 속하지 않는 데이터 세트는 전혀 보이지 않습니다. 비전역 영역은 NFS 서버로 작동할 수 없으므로 영역 관리자가 sharenfs 등록 정보를 변경할 수 없습니다. 영역 관리자는 zoned 등록 정보를 변경할 수 없습니다. 그러면 다음 절에 설명된 대로 보안 위험에 노출되기 때문입니다.

영역의 권한 사용자는 quota 및 reservation 등록 정보를 제외한, 다른 설정 가능한 등록 정보를 변경할 수 있습니다. 이 동작으로 전역 영역 관리자가 비전역 영역에서 사용된 모든 데이터 세트의 디스크 공간 소비를 제어할 수 있습니다.

더불어, 데이터 집합이 비전역 영역에 위임된 후에는 전역 영역 관리자가 sharenfs 및 mountpoint 등록 정보를 변경할 수 없습니다.

zoned 등록 정보 이해

데이터 집합이 비전역 영역에 위임될 때 특정 등록 정보가 전역 영역의 컨텍스트에서 해석되지 않도록 데이터 집합을 특별히 표시해야 합니다. 데이터 집합이 비전역 영역에 위임된 후에 전역 관리자의 통제 아래에 있으면 해당 컨텐츠를 더 이상 신뢰할 수 없습니다. 다른 파일 시스템과 마찬가지로, 전역 영역의 보안에 악영향을 미치는 setuid 이진, 심볼릭 링크나 의심스러운 컨텐츠가 있을 수 있습니다. 더불어, mountpoint 등록 정보는 전역 영역의 컨텍스트에서 해석할 수 없습니다. 그렇지 않으면 영역 관리자가 전역 영역의 이름 공간에 영향을 미칠 수 있습니다. 후자를 처리하기 위해 ZFS는 zoned 등록 정보를 사용하여 특정 시점에 데이터 집합이 비전역 영역에 위임되었는지 나타냅니다.

zoned 등록 정보는 ZFS 데이터 집합을 포함하는 영역을 처음 부트할 때 자동으로 켜지는 부울값입니다. 영역 관리자가 이 등록 정보를 수동으로 켤 필요가 없습니다. zoned 등록 정보가 설정된 경우 전역 영역에서 데이터 집합을 마운트하거나 공유할 수 없습니다. 다음 예에서 tank/zone/zion은 영역에 위임되었고 tank/zone/global은 위임되지 않았습니다.

```
# zfs list -o name,zoned,mountpoint -r tank/zone
NAME                  ZONED  MOUNTPOINT
tank/zone/global        off  /tank/zone/global
```

```
tank/zone/zion            on  /tank/zone/zion
# zfs mount
tank/zone/global            /tank/zone/global
tank/zone/zion              /export/zone/zion/root/tank/zone/zion
```

mountpoint 등록 정보와 tank/zone/zion 데이터 세트가 현재 마운트된 디렉토리의 차이점에 유의하십시오. mountpoint 등록 정보는 데이터 세트가 현재 시스템에 마운트된 위치가 아니라, 디스크에 저장할 당시의 등록 정보를 반영합니다.

데이터 세트를 영역에서 제거하거나 영역을 삭제할 때 zoned 등록 정보가 자동으로 지워지지 **않습니다**. 이 동작은 이러한 작업과 연관된 고유의 보안 위험 때문입니다. 신뢰할 수 없는 사용자가 데이터 세트와 그 자손에 전체 액세스할 수 있기 때문에 mountpoint 등록 정보가 잘못된 값으로 설정되거나 setuid 이진이 파일 시스템에 존재할 수 있습니다.

돌발적 보안 위험을 방지를 위해, 어떤 식으로든 데이터 세트를 재사용하려면 전역 영역 관리자가 zoned 등록 정보를 수동으로 지워야 합니다. zoned 등록 정보를 off로 설정하기 전에 데이터 세트와 모든 자손에 대한 mountpoint 등록 정보가 적합한 값으로 설정되고 setuid 이진이 존재하지 않거나 setuid 등록 정보가 꺼져 있는지 확인하십시오.

보안 취약점이 없는지 확인한 후에 zfs set 또는 zfs inherit 명령을 사용하여 zoned 등록 정보를 끌 수 있습니다. zoned 등록 정보를 껐지만 데이터 세트가 영역 내에서 사용 중인 경우 시스템 동작을 예측할 수 없습니다. 데이터 세트가 비전역 영역에서 더 이상 사용되지 않는 경우에만 등록 정보를 변경하십시오.

ZFS 대체 루트 풀 사용

풀을 만들 때 본능적으로 호스트 시스템에 묶입니다. 호스트 시스템은 풀에 대한 정보를 유지 관리하므로 풀이 사용 불가능한 때를 감지할 수 있습니다. 정상 작업에 유용하더라도 대체 매체에서 부트하거나 이동식 매체에서 풀을 만들 때 이 정보가 장애가 될 수 있습니다. 이 문제를 해결하기 위해 ZFS는 **대체 루트** 풀 기능을 제공합니다. 대체 루트 풀은 시스템 재부트에서 지속되지 않고, 모든 마운트 지점이 풀 루트의 상대 경로로 수정됩니다.

ZFS 대체 루트 풀 만들기

대체 루트 풀을 만드는 가장 흔한 이유는 이동식 매체에 사용하기 위한 것입니다. 이러한 상황에서 사용자는 일반적으로 단일 파일 시스템을 선호하고 대상 시스템 어디에서든 마운트하기를 원합니다. zpool create -R 옵션을 사용하여 대체 루트 풀을 만들 때 루트 파일 시스템의 마운트 지점이 대체 루트 값에 해당하는 /로 자동으로 설정됩니다.

다음 예에서 대체 루트 경로로 /mnt를 사용하여 morpheus라는 풀을 만듭니다.

```
# zpool create -R /mnt morpheus c0t0d0
# zfs list morpheus
NAME                    USED  AVAIL  REFER  MOUNTPOINT
morpheus               32.5K  33.5G     8K  /mnt
```

단일 파일 시스템 morpheus에서 마운트 지점이 풀의 대체 루트인 /mnt라는 사실에 유의하십시오. 디스크에 저장되는 마운트 지점은 /이고 /mnt의 전체 경로는 풀 생성의 초기 컨텍스트에서만 해석됩니다. 그런 다음, 이 파일 시스템을 -R *alternate root value* 구문을 사용하여 다른 시스템의 모든 대체 루트 풀에 내보내거나 가져올 수 있습니다.

```
# zpool export morpheus
# zpool import morpheus
cannot mount '/': directory is not empty
# zpool export morpheus
# zpool import -R /mnt morpheus
# zfs list morpheus
NAME                    USED  AVAIL  REFER  MOUNTPOINT
morpheus               32.5K  33.5G     8K  /mnt
```

대체 루트 풀 가져오기

대체 루트를 사용하여 풀을 가져올 수도 있습니다. 이 기능은 현재 루트의 컨텍스트에서 마운트 지점을 해석할 수 없지만 복구가 수행되는 임시 디렉토리 아래에 해석할 수 있는 복구 상황에 유용합니다. 또한 이전 절에서 설명된 대로 이동식 매체를 마운트할 때 이 기능을 사용할 수 있습니다.

다음 예에서 대체 루트 경로로 /mnt를 사용하여 morpheus라는 풀을 가져옵니다. 이 예에서 morpheus를 이전에 내보낸 것으로 가정합니다.

```
# zpool import -R /a pool
# zpool list morpheus
NAME   SIZE   ALLOC  FREE    CAP  HEALTH  ALTROOT
pool  44.8G    78K  44.7G     0%  ONLINE  /a
# zfs list pool
NAME   USED  AVAIL  REFER  MOUNTPOINT
pool  73.5K  44.1G    21K  /a/pool
```

◆ ◆ ◆ 10장

10 Oracle Solaris ZFS 문제 해결 및 풀 복구

이 장에서는 ZFS 오류를 식별하고 복구하는 방법에 대해 설명합니다. 오류 방지를 위한 정보도 제공됩니다.

이 장에서는 다음과 같은 내용을 다룹니다.

전체 루트 풀 복구에 대한 자세한 내용은 157 페이지 "ZFS 루트 풀 또는 루트 풀 스냅샷 복구"를 참조하십시오.

ZFS 문제 식별

ZFS는 결합된 파일 시스템 및 볼륨 관리자로서 여러 가지 오류가 발생할 수 있습니다. 이 장에서는 일반 하드웨어 오류를 진단하고 풀 장치 및 파일 시스템 문제를 해결하는 방법에 대해 간단히 설명합니다. 발생 가능한 문제 유형:

- **일반 하드웨어 문제** – 하드웨어 문제는 풀 성능 및 풀 데이터의 가용성에 영향을 줄 수 있습니다. 풀 및 파일 시스템 등 상위 레벨에서 문제를 확인하기 전에 잘못된 구성 요소 및 메모리와 같은 일반 하드웨어 문제를 제외합니다.
- **ZFS 저장소 풀 문제**
  - 274 페이지 "ZFS 저장소 풀을 사용하여 문제 식별"
  - 278 페이지 "ZFS 저장 장치 문제 해결"
- 291 페이지 "ZFS 파일 시스템 문제 해결"
- 300 페이지 "손상된 ZFS 구성 복구"

- 300 페이지 “부트할 수 없는 시스템 복구”

하나의 풀에서 이 세 가지 오류가 모두 발생할 수 있으므로, 전체 복구 절차는 하나의 오류를 찾아 해결한 다음 그 다음 오류로 진행하는 방식으로 진행됩니다.

일반 하드웨어 문제 해결

다음 섹션을 검토하여 풀 문제 또는 파일 시스템 비가용성이 잘못된 시스템 보드, 메모리, 장치, HBA 또는 잘못된 구성과 같은 하드웨어 문제와 관련된 것인지 확인하십시오.

예를 들어, 사용량이 많은 ZFS 풀에서 디스크에 오류 또는 결함이 발생하면 전체 시스템 성능을 크게 저하시킬 수 있습니다.

모든 하드웨어를 검색하고 체크아웃하는 것보다 쉬운 방식으로 하드웨어 문제를 먼저 진단하고 식별할 경우에는 이 장의 남은 부분의 설명에 따라 풀 및 파일 시스템 문제 진단으로 이동할 수 있습니다. 하드웨어, 풀 및 파일 시스템 구성이 건전한 경우, 응용 프로그램 문제를 진단해보십시오. 이 절차는 일반적으로 해결하는 데 조금 더 복잡하며 이 설명서에서 다뤄지지 않습니다.

하드웨어 및 장치 결함 식별

Solaris Fault Manager는 오류 로그에서 특정 증상을 나타내는 오류 원격 측정 정보를 식별한 후 오류 증상으로 인해 실제 결함이 발생할 경우 실제 결함 진단을 보고하는 방식으로 소프트웨어, 하드웨어 및 특정 장치 문제를 추적합니다.

다음 명령은 모든 소프트웨어 또는 하드웨어 관련 결함을 식별합니다.

```
# fmadm faulty
```

위 명령을 일반적으로 사용해서 실패한 서비스 또는 장치를 식별합니다.

하드웨어 또는 장치 관련 오류를 식별하려면 다음 명령을 일반적으로 사용합니다.

```
# fmdump -eV | more
```

`vdev.open_failed`, `checksum` 또는 `io_failure` 문제를 설명하는 이 로그 파일의 오류 메시지는 문제를 해결하지 않는 경우 `fmadm faulty` 명령을 사용하여 표시되는 실제 결함으로 발전할 수 있습니다.

위 명령으로 장치에 오류가 발생하고 있음을 나타낼 경우 교체용 장치가 있는지 확인하는 것이 좋습니다.

또한 `iostat` 명령을 사용하여 추가 장치 오류를 추적할 수도 있습니다. 다음 구문을 사용하여 오류 통계에 대한 요약을 식별합니다.

```
# iostat -en
  ---- errors ---
  s/w h/w trn tot device
    0   0   0   0 c0t5000C500335F95E3d0
    0   0   0   0 c0t5000C500335FC3E7d0
    0   0   0   0 c0t5000C500335BA8C3d0
    0  12   0  12 c2t0d0
    0   0   0   0 c0t5000C500335E106Bd0
    0   0   0   0 c0t50015179594B6F11d0
    0   0   0   0 c0t5000C500335DC60Fd0
    0   0   0   0 c0t5000C500335F907Fd0
    0   0   0   0 c0t5000C500335BD117d0
```

위 출력에서 오류는 내부 디스크 `c2t0d0`에 보고됩니다. 다음 구문을 사용하여 보다 자세한 장치 오류를 표시합니다.

```
# iostat -En
c0t5000C500335F95E3d0 Soft Errors: 0 Hard Errors: 0 Transport Errors: 0
Vendor: SEAGATE  Product: ST930003SSUN300G Revision: 0B70 Serial No: 110672QFSB
Size: 300.00GB <300000000000 bytes>
Media Error: 0 Device Not Ready: 0 No Device: 0 Recoverable: 0
Illegal Request: 0 Predictive Failure Analysis: 0
c0t5000C500335FC3E7d0 Soft Errors: 0 Hard Errors: 0 Transport Errors: 0
Vendor: SEAGATE  Product: ST930003SSUN300G Revision: 0B70 Serial No: 110672TE67
Size: 300.00GB <300000000000 bytes>
Media Error: 0 Device Not Ready: 0 No Device: 0 Recoverable: 0
Illegal Request: 0 Predictive Failure Analysis: 0
c0t5000C500335BA8C3d0 Soft Errors: 0 Hard Errors: 0 Transport Errors: 0
Vendor: SEAGATE  Product: ST930003SSUN300G Revision: 0B70 Serial No: 110672SDF4
Size: 300.00GB <300000000000 bytes>
Media Error: 0 Device Not Ready: 0 No Device: 0 Recoverable: 0
Illegal Request: 0 Predictive Failure Analysis: 0
c2t0d0           Soft Errors: 0 Hard Errors: 12 Transport Errors: 0
Vendor: AMI      Product: Virtual CDROM    Revision: 1.00 Serial No:
Size: 0.00GB <0 bytes>
Media Error: 0 Device Not Ready: 12 No Device: 0 Recoverable: 0
Illegal Request: 2 Predictive Failure Analysis: 0
```

ZFS 오류 메시지에 대한 시스템 보고

ZFS는 풀 내에서 오류를 지속적으로 추적하는 것 이외에도 관심 있는 이벤트가 발생할 때 `syslog` 메시지를 표시합니다. 다음 시나리오에서는 알림 이벤트를 생성합니다.

- **장치 상태 전환** – 장치가 `FAULTED` 상태가 되면 ZFS는 풀의 결함 허용이 손상되었을 수 있음을 나타내는 메시지를 기록합니다. 장치가 나중에 온라인 상태가 되면 비슷한 메시지가 전송되어 풀을 정상 상태로 복원합니다.
- **데이터 손상** – 데이터 손상이 발견될 경우 ZFS는 손상이 발견된 시기 및 위치를 설명하는 메시지를 기록합니다. 이 메시지는 처음 발견되었을 때만 기록되고, 이후 액세스는 메시지를 생성하지 않습니다.
- **풀 오류 및 장치 오류** – 풀 오류 및 장치 오류가 발생하면 결함 관리자 데몬이 `syslog` 메시지 및 `fmdump` 명령을 통해 이 오류를 보고합니다.

ZFS에서 장치 오류를 발견하고 이를 자동으로 복구한 경우에는 알림이 표시되지 않습니다. 이 오류는 풀 중복성 오류 또는 데이터 무결성 오류에 해당하지 않습니다. 또한 이 오류는 보통 자체 오류 메시지 세트에서 표시하는 드라이버 문제의 결과로 발생합니다.

ZFS 저장소 풀을 사용하여 문제 식별

다음 절에서는 ZFS 파일 시스템 또는 저장소 풀 관련 문제를 식별하고 해결하는 방법에 대해 설명합니다.

- 275 페이지 “ZFS 저장소 풀에 문제가 있는지 확인”
- 275 페이지 “zpool status 출력 결과 검토”
- 273 페이지 “ZFS 오류 메시지에 대한 시스템 보고”

다음 기능을 사용하여 ZFS 구성 관련 문제를 식별할 수 있습니다.

- 자세한 ZFS 저장소 풀 정보는 zpool status 명령을 사용하여 표시할 수 있습니다.
- 풀 및 장치 오류는 ZFS/FMA 진단 메시지를 통해 보고됩니다.
- 풀 상태 정보를 수정한 이전 ZFS 명령은 zpool history 명령을 사용하여 표시할 수 있습니다.

대부분의 ZFS 문제 해결은 zpool status 명령과 관련됩니다. 이 명령은 시스템에서 발생한 다양한 오류를 분석하고 가장 심각한 문제를 식별하여 권장되는 조치와 자세한 정보를 볼 수 있는 지식 문서에 대한 링크를 표시합니다. 여러 개의 문제가 존재하더라도 이 명령은 풀과 관련된 한 개의 문제만 식별합니다. 예를 들어 데이터 손상 오류는 일반적으로 장치 중 하나에서 오류가 발생했음을 암시하지만, 오류가 발생한 장치를 교체한다고 해서 모든 데이터 손상 문제가 해결되는 것은 아닐 수 있습니다.

또한 ZFS 진단 엔진이 풀 오류 및 장치 오류를 진단하고 보고합니다. 이러한 오류와 연관된 체크섬, I/O, 장치 및 풀 오류도 보고됩니다. fmd에 의해 보고된 ZFS 오류는 콘솔과 시스템 메시지 파일에 표시됩니다. 대부분의 경우 fmd 메시지를 통해 자세한 복구 지침을 제공하는 zpool status 명령으로 이동할 수 있습니다.

기본 복구 프로세스는 다음과 같습니다.

- 해당하는 경우 zpool history 명령을 사용하여 해당 오류 시나리오 이전에 사용된 ZFS 명령을 식별하십시오. 예를 들면 다음과 같습니다.

  ```
  # zpool history tank
  History for 'tank':
  2012-11-12.13:01:31 zpool create tank mirror c0t1d0 c0t2d0 c0t3d0
  2012-11-12.13:28:10 zfs create tank/eric
  2012-11-12.13:37:48 zfs set checksum=off tank/eric
  ```

  이 출력 결과에서는 tank/eric 파일 시스템에 대한 체크섬이 사용 안함으로 설정되었습니다. 이 구성은 권장되지 않는 구성입니다.
- 시스템 콘솔 또는 /var/adm/messages 파일에 표시되는 fmd 메시지를 통해 오류를 식별합니다.

- zpool status -x 명령을 사용하여 자세한 복구 지침을 찾습니다.
- 다음 단계를 수행하여 오류를 복구합니다.
  - 사용할 수 없거나 누락된 장치를 교체하고 온라인으로 설정합니다.
  - 결함이 있는 구성이나 손상된 데이터를 백업에서 복원합니다.
  - zpool status -x 명령을 사용하여 복구를 확인합니다.
  - 해당하는 경우 복원된 구성을 백업합니다.

이 절에서는 발생 가능한 오류 유형을 진단하기 위해 zpool status 출력 결과를 해석하는 방법에 대해 설명합니다. 대부분의 작업은 명령에 의해 자동으로 수행되지만 오류를 진단하기 위해서는 식별하려는 문제가 무엇인지 정확하게 이해하는 것이 중요합니다. 이후 절에서는 발생 가능한 여러 문제를 복구하는 방법에 대해 설명합니다.

ZFS 저장소 풀에 문제가 있는지 확인

시스템에 알려진 문제가 있는지 확인하는 가장 쉬운 방법은 zpool status -x 명령을 사용하는 것입니다. 이 명령은 문제가 발생한 풀만 설명합니다. 비정상적인 있는 풀이 시스템에 없을 경우 이 명령은 다음과 같은 내용을 표시합니다.

```
# zpool status -x
all pools are healthy
```

-x 플래그를 사용하지 않을 경우, 이 명령은 정상적인 풀이더라도 모든 풀(명령줄에 지정된 경우 요청된 풀)에 대한 전체 상태를 표시합니다.

zpool status 명령의 명령줄 옵션에 대한 자세한 내용은 82 페이지 “ZFS 저장소 풀 상태 질의”를 참조하십시오.

zpool status 출력 결과 검토

전체 zpool status 출력 결과는 다음과 비슷합니다.

```
# zpool status tank
   pool: tank
 state: DEGRADED
status: One or more devices could not be opened.  Sufficient replicas exist for
        the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
 scan: scrub repaired 0 in 0h3m with 0 errors on Mon Nov 12 15:17:02 2012
config:

        NAME        STATE     READ WRITE CKSUM
        tank        DEGRADED     0     0     0
          mirror-0  DEGRADED     0     0     0
            c1t1d0  ONLINE       0     0     0
            c1t2d0  UNAVAIL      0     0     0  cannot open

errors: No known data errors
```

이 출력 결과는 다음 절에서 설명합니다.

전체 풀 상태 정보

zpool status 출력 결과에서 이 절은 다음과 같은 필드로 구성됩니다. 이 중 일부는 문제가 발생한 풀의 경우에만 표시됩니다.

pool 풀 이름을 식별합니다.

state 풀의 현재 상태를 나타냅니다. 이 정보는 풀이 필요한 복제 레벨을 제공할 수 있는지만 나타냅니다.

status 풀에서 발생한 문제를 설명합니다. 오류가 발견되지 않을 경우 이 필드는 생략됩니다.

action 오류 복구를 위해 권장되는 조치입니다. 오류가 발견되지 않을 경우 이 필드는 생략됩니다.

see 자세한 복구 정보를 포함하는 지식 문서를 나타냅니다. 온라인 문서가 이 가이드가 업데이트할 수 있는 것보다 더 자주 업데이트됩니다. 따라서 최신 복구 절차는 항상 이 문서를 참조하십시오. 오류가 발견되지 않을 경우 이 필드는 생략됩니다.

scrub 스크러빙 작업의 현재 상태를 식별합니다. 이 정보에는 스크러빙이 마지막으로 완료된 날짜 및 시간, 스크러빙이 진행 중인 날짜 및 시간 또는 스크러빙이 요청되지 않은 경우 날짜 및 시간이 포함될 수 있습니다.

errors 알려진 데이터 오류 또는 알려진 데이터 오류가 없음을 식별합니다.

ZFS 저장소 풀 구성 정보

zpool status 출력의 config 필드는 풀에 있는 장치의 구성 및 장치에서 생성된 오류와 상태에 대해 설명합니다. 상태는 ONLINE, FAULTED, DEGRADED 또는 SUSPENDED 중 하나일 수 있습니다. ONLINE 상태가 아닐 경우 풀의 결함 허용이 손상됩니다.

구성 출력의 두번째 절에는 오류 통계가 표시됩니다. 이러한 오류는 다음 세 범주로 구분됩니다.

- READ – 읽기 요청을 발행하는 중 발생한 I/O 오류입니다.
- WRITE – 쓰기 요청을 발행하는 중 발생한 I/O 오류입니다.
- CKSUM – 읽기 요청의 결과로 장치에서 손상된 데이터를 반환함을 의미하는 체크섬 오류입니다.

이러한 오류를 사용하여 손상이 영구적인지 확인할 수 있습니다. I/O 오류 수가 적으면 일시적인 작동 중단을 나타내지만, 오류 수가 많으면 영구적인 장치 문제를 나타낼 수 있습니다. 이러한 오류가 반드시 응용 프로그램에서 해석한 데이터 손상과 일치하지는 않습니다. 장치가 중복 구성일 경우 해결할 수 없는 오류가 표시될 수 있지만, 미러 또는

RAID-Z 장치 레벨에서는 오류가 표시되지 않습니다. 이 경우 ZFS에서 정상적인 데이터를 성공적으로 검색하여 기존 복제본에서 손상된 데이터를 치료하려고 시도했습니다.

이러한 오류 해석에 대한 자세한 내용은 281 페이지 "장치 오류 유형 확인"을 참조하십시오.

끝으로, 추가 보조 정보가 `zpool status` 출력의 마지막 열에 표시됩니다. 이 정보는 오류 진단을 지원하기 위해 `state` 필드에서 확장됩니다. 장치가 `UNAVAIL`인 경우 이 필드는 장치에 액세스할 수 없는지 여부 또는 장치 데이터가 손상되었는지 여부를 나타냅니다. 장치에서 리실버링이 진행 중인 경우 이 필드에 현재 진행률이 표시됩니다.

리실버링 진행률 모니터링에 대한 자세한 내용은 290 페이지 "리실버링 상태 보기"를 참조하십시오.

ZFS 저장소 풀 스크러빙 상태

`zpool status` 출력의 scrub 절은 명시적 스크러빙 작업의 현재 상태에 대해 설명합니다. 이 정보를 사용하여 데이터 손상 오류 보고가 정확한지 확인할 수는 있지만, 이 정보는 시스템에서 오류가 발견되었는지 여부와는 다른 별개의 정보입니다. 최근에 마지막으로 스크러빙이 종료되었다면 알려진 데이터 손상이 발견되었을 가능성이 높습니다.

스크러빙 완료 메시지는 시스템 재부트 후에도 보존됩니다.

데이터 스크러빙 및 이 정보를 해석하는 방법에 대한 자세한 내용은 291 페이지 "ZFS 파일 시스템 무결성 검사"를 참조하십시오.

ZFS 데이터 손상 오류

`zpool status` 명령은 알려진 오류가 풀과 연관되는지 여부도 표시합니다. 이러한 오류는 데이터 스크러빙 또는 일반 작업 중에 발견되었을 수 있습니다. ZFS는 풀과 연관된 모든 데이터 오류의 영구 로그를 유지 관리합니다. 이 로그는 시스템의 전체 스크러빙이 완료될 때마다 교체됩니다.

데이터 손상 오류는 항상 치명적입니다. 이러한 오류가 있다는 것은 풀 내의 손상된 데이터로 인해 적어도 하나의 응용 프로그램에서 I/O 오류가 발생했음을 나타냅니다. 중복 풀 내의 장치 오류는 데이터 손상을 일으키지 않으므로 이 로그의 일부로 기록되지 않습니다. 기본적으로 발견된 오류 수만 표시됩니다. 전체 오류 목록 및 구체적인 정보는 `zpool status -v` 옵션을 사용하여 찾을 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool status -v
  pool: tank
 state: UNAVAIL
status: One or more devices are faulted in response to IO failures.
action: Make sure the affected devices are connected, then run 'zpool clear'.
   see: http://www.sun.com/msg/ZFS-8000-HC
 scrub: scrub completed after 0h0m with 0 errors on Tue Feb  2 13:08:42 2010
```

```
config:

        NAME        STATE     READ WRITE CKSUM
        tank        UNAVAIL      0     0     0  insufficient replicas
          c1t0d0    ONLINE       0     0     0
          c1t1d0    UNAVAIL      4     1     0  cannot open

errors: Permanent errors have been detected in the following files:

/tank/data/aaa
/tank/data/bbb
/tank/data/ccc
```

fmd에 의해 시스템 콘솔 및 /var/adm/messages 파일에도 비슷한 메시지가 표시됩니다. 이러한 메시지는 fmdump 명령을 사용하여 추적할 수도 있습니다.

데이터 손상 오류 해석에 대한 자세한 내용은 296 페이지 “데이터 손상 유형 식별”을 참조하십시오.

ZFS 저장 장치 문제 해결

다음 섹션을 검토하여 누락, 제거 또는 결함이 있는 장치를 해결합니다.

누락되었거나 제거된 장치 해결

장치를 열 수 없는 경우 zpool status 출력에 UNAVAIL 상태가 표시됩니다. 이 상태는 풀에 처음 액세스할 때 ZFS에서 장치를 열 수 없거나 장치를 사용할 수 없게 되었음을 의미합니다. 장치로 인해 최상위 가상 장치를 사용할 수 없게 될 경우 풀에 있는 어떠한 장치에도 액세스할 수 없습니다. 그렇지 않은 경우 풀의 결함 허용이 손상될 수 있습니다. 어떤 경우든지 장치를 시스템에 다시 연결하여 일반 작업을 복원해야 합니다. 결함이 발생하여 UNAVAIL 상태인 장치를 교체해야 할 경우 284 페이지 “ZFS 저장소 풀의 장치 교체”를 참조하십시오.

장치가 루트 풀 또는 미러링된 루트 풀에서 UNAVAIL인 경우 다음 참조 자료를 참조하십시오.

- **미러링된 루트 풀 디스크 실패** – 150 페이지 “미러링된 ZFS 루트 풀의 대체 디스크에서 부트”
- **루트 풀에서 디스크 교체**
  - 157 페이지 “ZFS 루트 풀의 디스크 교체 방법”
- **전체 루트 풀 재해 복구** – 157 페이지 “ZFS 루트 풀 또는 루트 풀 스냅샷 복구”

예를 들어, 장치 오류 후 fmd에서 다음과 비슷한 메시지를 표시할 수 있습니다.

```
SUNW-MSG-ID: ZFS-8000-FD, TYPE: Fault, VER: 1, SEVERITY: Major
EVENT-TIME: Thu Jun 24 10:42:36 PDT 2010
PLATFORM: SUNW,Sun-Fire-T200, CSN: -, HOSTNAME: daleks
SOURCE: zfs-diagnosis, REV: 1.0
EVENT-ID: a1fb66d0-cc51-cd14-a835-961c15696fcb
DESC: The number of I/O errors associated with a ZFS device exceeded
acceptable levels.  Refer to http://sun.com/msg/ZFS-8000-FD for more information.
AUTO-RESPONSE: The device has been offlined and marked as faulted.  An attempt
will be made to activate a hot spare if available.
IMPACT: Fault tolerance of the pool may be compromised.
REC-ACTION: Run 'zpool status -x' and replace the bad device.
```

장치 문제 및 해결 방법에 대한 자세한 정보를 보려면 `zpool status -x` 명령을 사용하십시오. 예를 들면 다음과 같습니다.

```
# zpool status -x
  pool: tank
 state: DEGRADED
status: One or more devices could not be opened.  Sufficient replicas exist for
        the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
  scan: scrub repaired 0 in 0h0m with 0 errors on Tue Sep 27 16:59:07 2011
config:

        NAME        STATE     READ WRITE CKSUM
        tank        DEGRADED     0     0     0
          mirror-0  DEGRADED     0     0     0
            c2t2d0  ONLINE       0     0     0
            c2t1d0  UNAVAIL      0     0     0  cannot open

errors: No known data errors
```

이 출력에서 `c2t1d0` 장치가 작동하지 않음을 확인할 수 있습니다. 이 장치에 오류가 있다고 판단되면 해당 장치를 교체하십시오.

필요한 경우 `zpool online` 명령을 사용하여 교체한 장치를 온라인 상태로 설정합니다. 예를 들면 다음과 같습니다.

```
# zpool online tank c2t1d0
```

`fmadm faulty` 출력에서 장치 오류가 식별되는 경우 장치가 교체되었음을 FMA에 알립니다. 예를 들면 다음과 같습니다.

```
# fmadm faulty
--------------- ------------------------------------  -------------- ---------
TIME            EVENT-ID                              MSG-ID         SEVERITY
--------------- ------------------------------------  -------------- ---------
Sep 27 16:58:50 e6bb52c3-5fe0-41a1-9ccc-c2f8a6b56100  ZFS-8000-D3    Major

Host        : neo
Platform    : SUNW,Sun-Fire-T200        Chassis_id  :
Product_sn  :
```

```
Fault class : fault.fs.zfs.device
Affects     : zfs://pool=tank/vdev=c75a8336cda03110
                  faulted and taken out of service
Problem in  : zfs://pool=tank/vdev=c75a8336cda03110
                  faulted and taken out of service

Description : A ZFS device failed.  Refer to http://sun.com/msg/ZFS-8000-D3 for
              more information.

Response    : No automated response will occur.

Impact      : Fault tolerance of the pool may be compromised.

Action      : Run 'zpool status -x' and replace the bad device.
```

```
# fmadm repaired zfs://pool=tank/vdev=c75a8336cda03110
```

끝으로, 교체된 장치를 포함하는 풀이 정상적으로 작동하는지 확인하십시오. 예를 들면 다음과 같습니다.

```
# zpool status -x tank
pool 'tank' is healthy
```

제거된 장치 해결

시스템에서 장치가 완전히 제거되면 ZFS는 해당 장치를 열 수 없음을 감지하고 장치를 REMOVED 상태로 설정합니다. 풀의 데이터 복제 레벨에 따라 이 제거로 인해 전체 풀이 사용하지 못하게 될 수도 있고 그렇지 않을 수도 있습니다. 미러링된 장치나 RAID-Z 장치의 한 디스크만 제거되면 풀에 계속 액세스할 수 있습니다. 다음과 같은 조건에서는 풀이 UNAVAIL 상태가 될 수 있습니다. 이 경우 장치가 다시 연결될 때까지 데이터에 액세스할 수 없습니다.

- 미러의 모든 구성 요소가 제거된 경우
- RAID-Z(raidz1) 장치에서 하나 이상의 장치가 제거된 경우
- 단일 디스크 구성에서 최상위 장치가 제거된 경우

물리적으로 장치 재연결

누락된 장치가 다시 연결되는 방식에 따라 문제가 발생하는 장치가 달라집니다. 장치가 네트워크 연결 드라이브일 경우 네트워크 연결을 복원해야 합니다. 장치가 USB 장치이거나 기타 이동식 매체일 경우 시스템에 다시 연결해야 합니다. 장치가 로컬 디스크일 경우 컨트롤러에서 오류가 발생하여 장치가 더 이상 시스템에 표시되지 않을 수 있습니다. 이 경우 컨트롤러를 교체해야 합니다. 그러면 디스크를 다시 사용할 수 있게 됩니다. 다른 문제가 존재할 수 있으며 이러한 문제는 하드웨어 및 하드웨어 구성의 유형에 따라 달라집니다. 드라이버에서 오류가 발생하여 시스템에 더 이상 표시되지 않을 경우 장치를 손상된 장치로 간주해야 합니다. 281 페이지 “손상된 장치 교체 또는 복구”에 설명된 절차를 수행하십시오.

장치 연결이 손상된 경우 풀이 SUSPENDED 상태일 수 있습니다. 풀은 장치 문제가 해결될 때까지 SUSPENDED 상태를 유지합니다. 예를 들면 다음과 같습니다.

```
# zpool status cybermen
  pool: cybermen
 state: SUSPENDED
status: One or more devices are unavailable in response to IO failures.
        The pool is suspended.
action: Make sure the affected devices are connected, then run 'zpool clear' or
        'fmadm repaired'.
    see: http://www.sun.com/msg/ZFS-8000-HC
  scan: none requested
config:

        NAME           STATE     READ WRITE CKSUM
        cybermen       UNAVAIL      0    16     0
            c8t3d0     UNAVAIL      0     0     0
            c8t1d0     UNAVAIL      0     0     0
```

장치 연결이 복원된 후 풀 또는 장치 오류를 지우십시오.

```
# zpool clear cybermen
# fmadm repaired zfs://pool=name/vdev=guid
```

ZFS에 장치 가용성 알림

장치를 시스템에 다시 연결하면 ZFS에서 자동으로 해당 장치가 사용 가능한지를 감지할 수도 있고 그렇지 않을 수도 있습니다. 풀이 이전에 UNAVAIL 또는 SUSPENDED 상태였거나 시스템이 attach 프로시저의 일부로 재부트된 경우에는 ZFS에서 풀을 열려고 할 때 자동으로 모든 장치를 다시 스캔합니다. 풀이 디그레이드되어 시스템 실행 중에 장치를 교체한 경우 zpool online 명령을 사용하여 이제 장치를 사용할 수 있으며 다시 열 준비가 되었음을 ZFS에 알려야 합니다. 예를 들면 다음과 같습니다.

```
# zpool online tank c0t1d0
```

장치를 온라인으로 설정하는 방법은 70 페이지 "온라인으로 장치 설정"을 참조하십시오.

손상된 장치 교체 또는 복구

이 절에서는 장치 오류 유형 확인, 일시적인 오류 지우기 및 장치 교체 방법에 대해 설명합니다.

장치 오류 유형 확인

손상된 장치라는 용어는 다소 모호하므로 여러 가지 가능한 상황을 들어 설명할 수 있습니다.

- **비트 로트** – 시간이 경과하면 자기적 영향 및 우주선(cosmic ray)과 같은 모든 이벤트로 인해 디스크에 저장된 비트가 플립됩니다. 이러한 이벤트는 상대적으로 발생할 확률이 거의 없지만, 대형 시스템 또는 장기 실행 중인 시스템에서 데이터 손상을 일으킬 가능성이 있습니다.
- **잘못 지정된 읽기 또는 쓰기** – 펌웨어 버그 또는 하드웨어 결함으로 인해 전체 블록의 읽기 또는 쓰기가 디스크의 잘못된 위치를 참조할 수 있습니다. 이러한 오류는 보통 일시적이지만, 이 오류의 대부분이 고장난 드라이브를 나타낼 수 있습니다.
- **관리자 오류** – 관리자가 실수로 디스크의 일부분을 잘못된 데이터로 덮어쓸 수 있습니다(예: `/dev/zero`를 디스크 일부분으로 복사). 이로 인해 디스크에 영구적인 손상이 발생할 수 있습니다. 이 오류는 항상 일시적입니다.
- **일시적인 작동 중단** – 일정 기간 동안 디스크를 사용할 수 없게 되어 I/O가 실패합니다. 이 상황은 보통 네트워크 연결 장치와 관련이 있지만, 로컬 디스크에서 일시적인 작동 중단이 발생할 수 있습니다. 이 오류는 일시적일 수도 있고 일시적이지 않을 수도 있습니다.
- **잘못되거나 이상한 하드웨어** – 이 상황은 일관된 I/O 오류, 모든 손상을 일으키는 잘못된 전송 또는 여러 오류와 같이 고장난 하드웨어에서 발생하는 다양한 문제에 모두 적용됩니다. 이 오류는 보통 영구적입니다.
- **오프라인 장치** – 장치가 오프라인일 경우 장치에 결함이 있어 관리자가 해당 장치를 이 상태로 설정한 것으로 간주합니다. 장치를 이 상태로 설정한 관리자는 이러한 가정이 정확한지 확인할 수 있습니다.

장치에서 발생한 문제가 무엇인지 정확하게 확인하는 것은 어려운 프로세스일 수 있습니다. 첫번째 단계는 `zpool status` 출력에서 오류 카운트를 확인하는 것입니다. 예를 들면 다음과 같습니다.

```
# zpool status -v tank
  pool: tank
 state: ONLINE
status: One or more devices has experienced an error resulting in data
        corruption.  Applications may be affected.
action: Restore the file in question if possible.  Otherwise restore the
        entire pool from backup.
   see: http://www.sun.com/msg/ZFS-8000-8A
  scan: scrub in progress since Tue Sep 27 17:12:40 2011
    63.9M scanned out of 528M at 10.7M/s, 0h0m to go
    0 repaired, 12.11% done
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       2     0     0
          mirror-0  ONLINE       2     0     0
            c2t2d0  ONLINE       2     0     0
            c2t1d0  ONLINE       2     0     0

errors: Permanent errors have been detected in the following files:

        /tank/words
```

오류는 I/O 오류와 체크섬 오류로 분류되는데, 둘 다 발생 가능한 오류 유형을 나타냅니다. 일반적인 작업의 경우 매우 적은 수의 오류(긴 기간 동안 소수의 오류)가 발생할 것으로 예측됩니다. 많은 수의 오류가 표시될 경우 이는 임박한 장치 오류나 전체 장치 오류를 나타내는 것일 수 있습니다. 그러나 관리자 오류로 인해 많은 오류 카운트가 발생할 수도 있습니다. 정보의 다른 소스는 `syslog` 시스템 로그입니다. 로그에 많은 수의 SCSI 또는 Fibre Channel 드라이버 메시지가 표시될 경우 이는 심각한 하드웨어 문제를 나타내는 것일 수 있습니다. `syslog` 메시지가 생성되지 않을 경우 일시적인 손상일 수 있습니다.

목표는 다음 질문에 답하는 것입니다.

이 장치에서 또 다른 오류가 발생할 가능성이 있습니까?

한 번만 발생하는 오류는 **일시적**인 오류로 간주되고 잠재적인 실패를 나타내지 않습니다. 잠재적인 하드웨어 실패를 나타낼 수 있는 영구적이거나 심각한 오류는 **치명적**인 오류로 간주됩니다. 오류의 유형을 확인하는 작업은 현재 ZFS에서 제공하는 자동 소프트웨어의 범위를 벗어나는 것이므로, 관리자가 수동으로 많은 부분을 수행해야 합니다. 확인이 이루어진 후에 적절한 조치를 취할 수 있습니다. 일시적인 오류를 지우거나 치명적인 오류가 발생한 장치를 교체하십시오. 이러한 복구 절차는 다음 절에서 설명합니다.

장치 오류가 일시적인 오류로 간주되더라도 풀 내에서 해결할 수 없는 데이터 오류를 일으킬 수도 있습니다. 이 경우 기본 장치가 정상적으로 작동하거나 복구되었다 하더라도 이러한 오류에는 특별한 복구 절차가 필요합니다. 데이터 오류 복구에 대한 자세한 내용은 296 페이지 “손상된 데이터 복구”를 참조하십시오.

일시적인 데이터 오류 지우기

장치 오류가 일시적인 오류로 간주될 경우, 장치의 이후 상태에 영향을 줄 가능성이 거의 없으므로 치명적인 오류가 발생하지 않음을 나타내도록 안전하게 오류를 지우십시오. RAID-Z 또는 미러링된 장치에 대한 오류 카운터를 지우려면 `zpool clear` 명령을 사용하십시오. 예를 들면 다음과 같습니다.

```
# zpool clear tank c1t1d0
```

이 구문은 장치 오류를 지우고 해당 장치와 연관된 데이터 오류 카운트를 지웁니다.

풀의 가상 장치와 연관된 모든 오류를 지우고 해당 풀과 연관된 데이터 오류 카운트를 지우려면 다음 구문을 사용하십시오.

```
# zpool clear tank
```

풀 오류 지우기에 대한 자세한 내용은 71 페이지 “저장소 풀 장치 오류 지우기”를 참조하십시오.

ZFS 저장소 풀의 장치 교체

장치 손상이 영구적이거나 이후에 영구적인 손상이 될 가능성이 있을 경우 장치를 교체해야 합니다. 장치를 교체할 수 있는 여부는 구성에 따라 달라집니다.

- 284 페이지 “교체 가능한 장치인지 확인”
- 285 페이지 “교체할 수 없는 장치”
- 285 페이지 “ZFS 저장소 풀의 장치 교체”
- 290 페이지 “리실버링 상태 보기”

교체 가능한 장치인지 확인

교체할 장치가 중복 구성의 일부일 경우 정상적인 데이터를 검색할 수 있는 충분한 복제본이 있어야 합니다. 예를 들어 4방향 미러의 두 디스크가 UNAVAIL 상태인 경우 정상 상태인 복제본을 사용할 수 있으므로 둘 중 한 디스크를 교체할 수 있습니다. 그러나 4방향 RAID-Z(raidz1) 가상 장치의 두 디스크가 UNAVAIL 상태인 경우, 데이터를 검색할 수 있는 충분한 복제본이 없으므로 어떠한 디스크도 교체할 수 없습니다. 장치가 손상되었으나 온라인일 경우 풀이 UNAVAIL 상태가 아니라면 장치를 교체할 수 있습니다. 그러나 정상적인 데이터를 포함하는 복제본이 부족할 경우 장치의 손상된 데이터가 새 장치로 복사됩니다.

다음 구성에서 c1t1d0 디스크는 교체할 수 있으며, 풀의 데이터는 정상 복제본 c1t0d0에서 복사됩니다.

```
    mirror            DEGRADED
    c1t0d0             ONLINE
    c1t1d0             FAULTED
```

c1t0d0 디스크도 교체할 수 있지만, 정상적인 복제본을 사용할 수 없으므로 데이터 자체 치료가 수행되지 않습니다.

다음 구성에서는 UNAVAIL 디스크를 교체할 수 없습니다. ONLINE 디스크는 풀 자체가 UNAVAIL 상태이므로 교체할 수 없습니다.

```
    raidz             FAULTED
    c1t0d0            ONLINE
    c2t0d0            FAULTED
    c3t0d0            FAULTED
    c4t0d0            ONLINE
```

다음 구성에서 최상위 디스크는 교체할 수 있지만, 디스크에 있는 잘못된 데이터가 새 디스크로 복사됩니다.

```
c1t0d0         ONLINE
c1t1d0         ONLINE
```

디스크가 UNAVAIL인 경우 풀 자체가 UNAVAIL이기 때문에 교체를 수행할 수 없습니다.

교체할 수 없는 장치

장치 손실로 인해 풀이 UNAVAIL 상태가 되거나 장치의 비중복 구성에 너무 많은 데이터 오류가 있는 경우 장치를 안전하게 교체할 수 없습니다. 충분한 중복성을 사용하지 않으면 손상된 장치를 치료할 수 있는 정상적인 데이터가 생기지 않습니다. 이 경우 유일한 옵션은 풀을 삭제하고 구성을 다시 생성한 다음 백업 복사본에서 데이터를 복원하는 것입니다.

전체 풀 복원에 대한 자세한 내용은 298 페이지 "ZFS 저장소 풀 전반의 손상 복구"를 참조하십시오.

ZFS 저장소 풀의 장치 교체

교체할 수 있다고 장치로 확인되었으면 zpool replace 명령을 사용하여 장치를 교체하십시오. 손상된 장치를 다른 장치로 교체할 경우 다음과 비슷한 구문을 사용하십시오.

```
# zpool replace tank c1t1d0 c2t0d0
```

이 명령은 손상된 장치 또는 풀의 다른 장치(중복 구성인 경우)에서 새 장치로 데이터를 마이그레이션합니다. 명령이 완료되면 구성에서 손상된 장치가 분리됩니다. 따라서 시스템에서 해당 당치를 제거할 수 있습니다. 아미 장치를 제거하고 같은 위치의 새 장치로 교체한 경우 명령의 단일 장치 양식을 사용하십시오. 예를 들면 다음과 같습니다.

```
# zpool replace tank c1t1d0
```

이 명령은 포맷되지 않은 디스크를 가져와서 적절하게 포맷한 다음 나머지 구성에서 데이터를 리실버링합니다.

zpool replace 명령에 대한 자세한 내용은 72 페이지 "저장소 풀의 장치 교체"를 참조하십시오.

예 10–1 ZFS 저장소 풀의 SATA 디스크 교체

다음 예에서는 SATA 장치가 있는 시스템의 미러링된 저장소 풀 tank에서 장치(c1t3d0)를 교체하는 방법을 보여줍니다. c1t3d0 디스크를 같은 위치(c1t3d0)의 새 디스크로 교체하려면 디스크 교체를 시도하기 전에 디스크를 구성 해제해야 합니다. 교체할 디스크가 SATA 디스크가 아니면 72 페이지 "저장소 풀의 장치 교체"를 참조하십시오.

기본 단계는 다음과 같습니다.

- 교체할 디스크(c1t3d0)를 오프라인으로 설정합니다. 현재 사용 중인 SATA 디스크는 구성 해제할 수 없습니다.
- cfgadm 명령을 사용하여 구성 해제할 SATA 디스크(c1t3d0)를 식별한 다음 구성 해제합니다. 이 미러링된 구성의 오프라인 디스크로 풀이 디그레이드되지만 풀은 계속 사용 가능합니다.

예 10–1 ZFS 저장소 풀의 SATA 디스크 교체 *(계속)*

- 디스크(c1t3d0)를 물리적으로 교체합니다. UNAVAIL 드라이브를 물리적으로 제거하기 전에 Ready to Remove 파란색 LED가 켜졌는지 확인합니다(사용 가능한 경우).
- SATA 디스크(c1t3d0)를 다시 구성합니다.
- 새 디스크(c1t3d0)를 온라인으로 설정합니다.
- zpool replace 명령을 사용하여 디스크(c1t3d0)를 교체합니다.

주 – 이전에 풀 등록 정보 autoreplace를 on으로 설정한 경우, 새 장치가 이전에 풀에 속해 있던 장치와 동일한 물리적 위치에서 발견되면 zpool replace 명령을 사용하지 않고 자동으로 포맷되고 교체됩니다. 이 기능은 일부 하드웨어에서만 지원될 수 있습니다.

- 실패한 디스크가 자동으로 핫 스페어로 교체된 경우 실패한 디스크가 교체된 후 핫 스페어를 분리해야 합니다. 예를 들어 실패한 디스크를 교체한 후에도 여전히 c2t4d0가 활성 핫 스페어인 경우 이를 분리하십시오.

  # zpool detach tank c2t4d0

- FMA에서 장애가 발생한 장치를 보고하는 경우 장치 오류를 지워야 합니다.

  # fmadm faulty
  # fmadm repaired *zfs://pool=name/vdev=guid*

다음 예는 ZFS 저장소 풀에서 디스크를 교체하는 단계를 보여줍니다.

```
# zpool offline tank c1t3d0
# cfgadm | grep c1t3d0
sata1/3::dsk/c1t3d0            disk         connected    configured   ok
# cfgadm -c unconfigure sata1/3
Unconfigure the device at: /devices/pci@0,0/pci1022,7458@2/pci11ab,11ab@1:3
This operation will suspend activity on the SATA device
Continue (yes/no)? yes
# cfgadm | grep sata1/3
sata1/3                        disk         connected    unconfigured ok
<Physically replace the failed disk c1t3d0>
# cfgadm -c configure sata1/3
# cfgadm | grep sata1/3
sata1/3::dsk/c1t3d0            disk         connected    configured   ok
# zpool online tank c1t3d0
# zpool replace tank c1t3d0
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: resilver completed after 0h0m with 0 errors on Tue Feb  2 13:17:32 2010
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
```

예 10–1 ZFS 저장소 풀의 SATA 디스크 교체 *(계속)*

```
            c0t1d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0
          mirror-1  ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c1t2d0  ONLINE       0     0     0
          mirror-2  ONLINE       0     0     0
            c0t3d0  ONLINE       0     0     0
            c1t3d0  ONLINE       0     0     0

errors: No known data errors
```

위 `zpool output`의 경우 *replacing* 제목 아래에 새 디스크와 이전 디스크를 모두 표시할 수 있습니다. 예를 들면 다음과 같습니다.

```
replacing     DEGRADED     0     0    0
  c1t3d0s0/o  FAULTED      0     0    0
  c1t3d0      ONLINE       0     0    0
```

이 텍스트는 교체 프로세스가 진행 중이며 새 디스크를 리실버링하는 중임을 의미합니다.

디스크(`c1t3d0`)를 다른 디스크(`c4t3d0`)로 교체하려는 경우에는 `zpool replace` 명령을 실행하기만 하면 됩니다. 예를 들면 다음과 같습니다.

```
# zpool replace tank c1t3d0 c4t3d0
# zpool status
  pool: tank
 state: DEGRADED
 scrub: resilver completed after 0h0m with 0 errors on Tue Feb  2 13:35:41 2010
config:

        NAME             STATE     READ WRITE CKSUM
        tank             DEGRADED     0     0     0
          mirror-0       ONLINE       0     0     0
            c0t1d0       ONLINE       0     0     0
            c1t1d0       ONLINE       0     0     0
          mirror-1       ONLINE       0     0     0
            c0t2d0       ONLINE       0     0     0
            c1t2d0       ONLINE       0     0     0
          mirror-2       DEGRADED     0     0     0
            c0t3d0       ONLINE       0     0     0
            replacing    DEGRADED     0     0     0
              c1t3d0     OFFLINE      0     0     0
              c4t3d0     ONLINE       0     0     0

errors: No known data errors
```

디스크 교체가 완료될 때까지 `zpool status` 명령을 여러 번 실행해야 할 수도 있습니다.

```
# zpool status tank
  pool: tank
 state: ONLINE
```

예 10–1 ZFS 저장소 풀의 SATA 디스크 교체 *(계속)*

```
 scrub: resilver completed after 0h0m with 0 errors on Tue Feb  2 13:35:41 2010
config:

        NAME          STATE     READ WRITE CKSUM
        tank          ONLINE       0     0     0
          mirror-0    ONLINE       0     0     0
            c0t1d0    ONLINE       0     0     0
            c1t1d0    ONLINE       0     0     0
          mirror-1    ONLINE       0     0     0
            c0t2d0    ONLINE       0     0     0
            c1t2d0    ONLINE       0     0     0
          mirror-2    ONLINE       0     0     0
            c0t3d0    ONLINE       0     0     0
            c4t3d0    ONLINE       0     0     0
```

예 10–2 실패한 로그 장치 교체

ZFS는 `zpool status` 명령 출력에서 계획 로그 오류를 식별합니다. FMA(Fault Management Architecture)에서는 이러한 오류도 보고됩니다. ZFS 및 FMA 모두 의도 로그 오류에서 복구하는 방법을 설명합니다.

다음 예는 저장소 풀(`pool`)의 실패한 로그 장치(`c0t5d0`)에서 복구하는 방법을 보여줍니다. 기본 단계는 다음과 같습니다.

- `zpool status -x` 출력 결과 및 여기에 표시된 FMA 진단 메시지를 검토합니다.

  https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&doctype=REFERENCE&alias=EVENT:ZFS-8000-K4
- 실패한 로그 장치를 물리적으로 교체합니다.
- 새 로그 장치를 온라인으로 설정합니다.
- 풀의 오류 조건을 지웁니다.
- FMA 오류를 지웁니다.

```
# zpool status -x
  pool: pool
 state: FAULTED
status: One or more of the intent logs could not be read.
        Waiting for adminstrator intervention to fix the faulted pool.
action: Either restore the affected device(s) and run 'zpool online',
        or ignore the intent log records by running 'zpool clear'.
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        pool        FAULTED      0     0     0 bad intent log
          mirror    ONLINE       0     0     0
            c0t1d0  ONLINE       0     0     0
            c0t4d0  ONLINE       0     0     0
```

예 10–2 실패한 로그 장치 교체 *(계속)*

```
        logs        FAULTED      0     0     0 bad intent log
          c0t5d0    UNAVAIL      0     0     0 cannot open
<Physically replace the failed log device>
# zpool online pool c0t5d0
# zpool clear pool
```

예를 들어 별도의 로그 장치가 있는 풀에 동기식 쓰기 작업이 커밋되기 전에 시스템이 갑자기 종료된 경우에는 다음과 비슷한 메시지가 표시됩니다.

```
# zpool status -x
  pool: pool
 state: FAULTED
status: One or more of the intent logs could not be read.
        Waiting for adminstrator intervention to fix the faulted pool.
action: Either restore the affected device(s) and run 'zpool online',
        or ignore the intent log records by running 'zpool clear'.
 scrub: none requested
config:

        NAME          STATE     READ WRITE CKSUM
        pool          FAULTED      0     0     0 bad intent log
          mirror-0    ONLINE       0     0     0
            c0t1d0    ONLINE       0     0     0
            c0t4d0    ONLINE       0     0     0
        logs          FAULTED      0     0     0 bad intent log
          c0t5d0      UNAVAIL      0     0     0 cannot open
<Physically replace the failed log device>
# zpool online pool c0t5d0
# zpool clear pool
# fmadm faulty
# fmadm repair zfs://pool=name/vdev=guid
```

로그 장치 오류는 다음과 같은 방식으로 해결할 수 있습니다.

- 로그 장치를 교체하거나 복구합니다. 이 예제에서 로그 장치는 `c0t5d0`입니다.
- 로그 장치를 다시 온라인으로 설정합니다.

  ```
  # zpool online pool c0t5d0
  ```

- 실패한 로그 장치 오류 조건을 재설정합니다.

  ```
  # zpool clear pool
  ```

실패한 로그 장치를 교체하지 않고 이 오류로부터 복구하려면 `zpool clear` 명령을 사용하여 오류를 지울 수 있습니다. 이 시나리오에서 풀은 성능 저하 모드로 작동하며 별도의 로그 장치가 교체될 때까지 로그 레코드가 기본 풀에 작성됩니다.

로그 장치 오류 시나리오를 방지하려면 미러링된 로그 장치를 사용하는 것이 좋습니다.

리실버링 상태 보기

장치를 교체하는 프로세스는 장치의 크기 및 풀에 포함된 데이터의 양에 따라 상당한 시간이 걸릴 수 있습니다. 데이터를 한 장치에서 다른 장치로 이동하는 프로세스를 **리실버링**이라고 하며 `zpool status` 명령을 사용하여 모니터링할 수 있습니다.

기존 파일 시스템은 블록 레벨에서 데이터를 리실버링합니다. ZFS에서는 볼륨 관리자의 인공 계층이 제거되므로 훨씬 더 강력하고 제어된 방식으로 리실버링을 수행할 수 있습니다. 이 기능의 두 가지 주요 이점은 다음과 같습니다.

- ZFS에서는 최소한으로 필요한 데이터만 리실버링합니다. 단기 작동 중단(전체 장치 교체와 반대)의 경우 전체 디스크를 간단하게 리실버링할 수 있습니다. 전체 디스크를 교체할 경우 디스크에 사용된 데이터의 양에 비례하여 리실버링 프로세스 시간이 걸립니다. 풀의 사용된 디스크 공간이 몇 GB에 불과할 경우 몇 초면 500GB 디스크를 교체할 수 있습니다.
- 리실버링은 중단 가능하며 안전합니다. 시스템 전원이 끊기거나 시스템이 재부트될 경우 사용자의 개입 없이도 정확히 중단되었던 부분부터 리실버링 프로세스가 재개됩니다.

리실버링 프로세스를 확인하려면 `zpool status` 명령을 사용하십시오. 예를 들면 다음과 같습니다.

```
# zpool status tank
  pool: tank
 state: DEGRADED
status: One or more devices is currently being resilvered.  The pool will
        continue to function, possibly in a degraded state.
action: Wait for the resilver to complete.
 scrub: resilver in progress for 0h0m, 22.60% done, 0h1m to go
config:
        NAME                  STATE     READ WRITE CKSUM
        tank                DEGRADED     0     0     0
          mirror-0          DEGRADED     0     0     0
            replacing-0     DEGRADED     0     0     0
              c1t0d0        UNAVAIL      0     0     0  cannot open
              c2t0d0        ONLINE       0     0     0  85.0M resilvered
            c1t1d0          ONLINE       0     0     0

errors: No known data errors
```

이 예에서 `c1t0d0` 디스크가 `c2t0d0`으로 교체되는 중입니다. 상태 출력 결과에서 구성에 `replacing` 가상 장치가 있다면 이 이벤트를 확인할 수 있습니다. 이 장치는 실제가 아니므로 이 장치를 사용하여 풀을 만들 수 없습니다. 아 장치의 목적은 리실버링 진행률을 표시하고 교체할 장치를 식별하기 위한 것입니다.

리실버링 프로세스가 완료될 때까지 풀이 원하는 레벨의 중복성을 제공할 수 없기 때문에 현재 리실버링이 진행 중인 풀은 `ONLINE` 또는 `DEGRADED` 상태입니다. 리실버링은 가능한 빠르게 진행되지만, 시스템에 미치는 영향을 최소화하기 위해 I/O는 항상

사용자가 요청한 I/O보다 낮은 우선 순위로 예약됩니다. 리실버링이 완료되면 구성이 완료된 새 구성으로 되돌아갑니다. 예를 들면 다음과 같습니다.

```
# zpool status tank
  pool: tank
 state: ONLINE
 scrub: resilver completed after 0h1m with 0 errors on Tue Feb  2 13:54:30 2010
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c2t0d0  ONLINE       0     0     0  377M resilvered
            c1t1d0  ONLINE       0     0     0

errors: No known data errors
```

풀이 다시 ONLINE 상태가 되고, 실패한 원래 디스크(c1t0d0)가 구성에서 제거되었습니다.

ZFS 파일 시스템 문제 해결

ZFS 저장소 풀의 데이터 문제 해결

다음을 포함한 데이터 문제 예는 다음과 같습니다.

- 잘못된 디스크 또는 컨트롤러로 인한 일시적인 I/O 오류
- 우주선(cosmic ray)으로 인한 디스크 내장 데이터 손상
- 드라이버 버그로 인해 잘못된 위치에서 또는 잘못된 위치로 데이터 전송
- 사용자가 실수로 물리적 장치의 일부분을 덮어쓰는 경우

경우에 따라 모든 I/O 오류와 같이 컨트롤러에 문제가 발생하는 동안에 발생하는 일시적인 오류일 수도 있고, 디스크 손상과 같은 영구적인 손상일 수도 있습니다. 그러나 영구적인 손상이라고 해서 이 오류가 반드시 다시 발생하는 것은 아닙니다. 예를 들어, 실수로 디스크의 일부를 덮어쓴 경우 어떠한 유형의 하드웨어 고장도 발생하지 않았으므로 장치를 교체할 필요가 없습니다. 장치와 관련한 정확한 문제를 식별하는 것은 쉬운 작업이 아니므로 뒤 절에서 자세히 설명합니다.

ZFS 파일 시스템 무결성 검사

fsck에 해당하는 유틸리티가 ZFS에는 없습니다. 이 유틸리티는 일반적으로 파일 시스템 복구 및 파일 시스템 검증이라는 두 가지 목적을 제공합니다.

파일 시스템 복구

기존 파일 시스템에서는 데이터가 기록되는 방식이 본래 파일 시스템 불일치를 일으키는 예상치 않은 오류에 취약합니다. 기존 파일 시스템은 트랜잭션이 아니므로 참조되지 않은 블록, 잘못된 링크 카운트 또는 기타 불일치한 파일 시스템 구조가 발생할 수 있습니다. 저널링을 추가하면 이러한 문제가 해결되지만, 로그를 롤백할 수 없는 경우 추가로 문제가 발생할 수 있습니다. 하드웨어 오류(이 경우 중복된 풀을 사용해야 함)가 발생하거나 ZFS 소프트웨어에 버그가 존재하는 경우에만 불일치 데이터가 ZFS 구성의 디스크에 존재하게 됩니다.

`fsck` 유틸리티는 UFS 파일 시스템에만 해당하는 알려진 문제를 복구합니다. 대부분의 ZFS 저장소 풀 문제는 일반적으로 하드웨어 오류 또는 전원 오류와 관련이 있습니다. 따라서 중복된 풀을 사용하여 문제를 방지할 수 있습니다. 하드웨어 오류 또는 정전으로 인해 풀이 손상된 경우 298 페이지 “ZFS 저장소 풀 전반의 손상 복구”를 참조하십시오.

중복된 풀이 아닌 경우 파일 시스템 손상으로 인해 데이터의 일부 또는 전체에 액세스할 수 없게 될 위험이 항상 존재합니다.

파일 시스템 검증

`fsck` 유틸리티는 파일 시스템 복구를 수행하는 것 외에도, 디스크의 데이터에 문제가 없는지 검증합니다. 일반적으로 이 작업을 수행하려면 파일 시스템을 마운트 해제하고 `fsck` 유틸리티를 실행하며 가능하면 프로세스 중에 시스템을 단일 사용자 모드로 전환해야 합니다. 이 경우 검사할 파일 시스템의 크기에 비례하여 작동 중지 시간이 발생합니다. ZFS는 필요한 검사를 수행하기 위한 명시적 유틸리티 대신 모든 불일치에 대해 루틴 검사를 수행하는 방식을 제공합니다. **스크러빙**이라고 하는 이 기능은 일반적으로 메모리 및 기타 시스템에서 하드웨어 또는 소프트웨어 오류가 발생하기 전에 오류를 발견하고 방지하는 한 방법으로 사용됩니다.

ZFS 데이터 스크러빙 제어

ZFS에서 오류가 발생할 때마다 스크러빙을 통해 또는 요구 시 파일에 액세스할 경우 오류가 내부적으로 기록되므로, 사용자는 풀 내에서 발생한 모든 알려진 오류를 한눈에 확인할 수 있습니다.

명시적 ZFS 데이터 스크러빙

데이터 무결성을 검사하는 가장 간단한 방법은 풀 내에 있는 모든 데이터의 명시적 스크러빙을 시작하는 것입니다. 이 작업은 풀 내의 모든 데이터를 한 번 탐색한 다음 모든 블록을 읽을 수 있는지 확인합니다. 스크러빙은 장치에서 허용하는 한 가능한 빠르게 진행되지만, I/O 우선 순위는 일반 작업보다 낮습니다. 스크러빙이 수행되는 동안 풀 데이터는 사용하지 않은 상태로 유지되고 거의 응답하지만 이 작업은 성능에 부정적인 영향을 미칠 수 있습니다. 명시적 스크러빙을 시작하려면 `zpool scrub` 명령을 사용하십시오. 예를 들면 다음과 같습니다.

```
# zpool scrub tank
```

현재 스크러빙 작업의 상태는 zpool status 명령을 사용하여 표시할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool status -v tank
  pool: tank
 state: ONLINE
 scrub: scrub completed after 0h7m with 0 errors on Tue Tue Feb  2 12:54:00 2010
config:
        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c1t0d0  ONLINE       0     0     0
            c1t1d0  ONLINE       0     0     0

errors: No known data errors
```

풀당 하나의 활성 스크러빙 작업만 한 번에 수행될 수 있습니다.

-s 옵션을 사용하여 진행 중인 스크러빙 작업을 중지할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool scrub -s tank
```

대부분의 경우 데이터 무결성을 확인하는 스크러빙 작업을 계속 수행하여 완료해야 합니다. 스크러빙 작업이 시스템 성능에 영향을 미치는 경우 재량으로 이 작업을 중지할 수 있습니다.

루틴 스크러빙을 수행하면 시스템의 모든 디스크에 대한 연속 I/O가 보장됩니다. 루틴 스크러빙은 유휴 디스크를 절전 모드로 전환하지 못하도록 하는 부작용이 있습니다. 시스템이 일반적으로 I/O를 항상 수행하는 경우 또는 전력 소비가 중요하지 않은 경우에는 이 문제를 무시해도 안전합니다.

zpool status 출력 결과 해석에 대한 자세한 내용은 82 페이지 “ZFS 저장소 풀 상태 질의”를 참조하십시오.

ZFS 데이터 스크러빙 및 리실버링

장치가 교체되면 리실버링 작업이 시작되어 양호한 복사본의 데이터를 새 장치로 이동합니다. 이 작업은 디스크 스크러빙의 일종입니다. 따라서 지정된 시간에 풀에서 한 번만 수행될 수 있습니다. 스크러빙 작업이 진행 중인 경우 리실버링 작업이 현재 스크러빙을 일시 중지한 다음 리실버링이 완료된 후 다시 시작합니다.

리실버링에 대한 자세한 내용은 290 페이지 “리실버링 상태 보기”를 참조하십시오.

손상된 ZFS 데이터

데이터 손상은 하나 이상의 장치 오류(하나 이상의 장치 누락 또는 손상을 나타냄)가 최상위 가상 장치에 영향을 미칠 때 발상합니다. 예를 들어 미러의 반쪽에서 수천 개의 장치가 오류가 발생했지만 데이터 손상이 발생하지 않을 수 있습니다. 그러나 미러 다른 쪽의 정확히 같은 위치에서 오류가 발생할 경우에는 데이터가 손상됩니다.

데이터 손상은 항상 영구적이므로 복구 중 특별한 고려가 필요합니다. 기본 장치를 복구하거나 교체해도 원본 데이터는 영구 손실됩니다. 대부분 이 경우에는 백업에서 데이터를 복원해야 합니다. 데이터 오류는 발생할 때 기록되므로, 다음 절에 설명된 루틴 풀 스크러빙을 통해 제어할 수 있습니다. 손상된 블록이 제거되면 다음 스크러빙 단계에서 더 이상 손상된 부분이 없음을 파악하여 시스템에서 오류 추적을 제거합니다.

ZFS 공간 문제 해결

ZFS의 파일 시스템 및 풀 공간 계산 보고 방법이 확실하지 않은 경우 다음 절을 참조하십시오. 또한 30 페이지 “ZFS 디스크 공간 계산”도 참조하십시오.

ZFS 파일 시스템 공간 보고

`zpool list` 및 `zfs list` 명령은 사용 가능한 풀과 파일 시스템 공간을 확인하는 데 있어 이전 `df` 및 `du` 명령보다 낫습니다. 레거시 명령을 사용하면 풀 공간과 파일 시스템 공간을 쉽게 구별할 수 없으며, 종속 파일 시스템이나 스냅샷에서 사용하는 공간을 확인할 수도 없습니다.

예를 들어 다음 루트 풀(`rpool`)에는 5.46GB가 할당되었으며 68.5GB가 사용 가능합니다.

```
# zpool list rpool
NAME   SIZE  ALLOC   FREE  CAP  DEDUP  HEALTH  ALTROOT
rpool   74G  5.46G  68.5G   7%  1.00x  ONLINE  -
```

개별 파일 시스템의 `USED` 열을 검토하여 풀 공간 계산을 파일 시스템 공간 계산과 비교할 경우 `ALLOC`에 보고된 풀 공간이 파일 시스템의 `USED` 공간에 대해 고려되었음을 알 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs list -r rpool
NAME                     USED  AVAIL  REFER  MOUNTPOINT
rpool                   5.41G  67.4G  74.5K  /rpool
rpool/ROOT              3.37G  67.4G    31K  legacy
rpool/ROOT/solaris      3.37G  67.4G  3.07G  /
rpool/ROOT/solaris/var   302M  67.4G   214M  /var
rpool/dump              1.01G  67.5G  1000M  -
rpool/export            97.5K  67.4G    32K  /rpool/export
rpool/export/home       65.5K  67.4G    32K  /rpool/export/home
rpool/export/home/admin 33.5K  67.4G  33.5K  /rpool/export/home/admin
rpool/swap              1.03G  67.5G  1.00G  -
```

ZFS 저장소 풀 공간 보고

zpool list 명령이 보고하는 SIZE 값은 일반적으로 풀의 물리적 디스크 공간의 양이지만 풀의 중복성 레벨에 따라 달라집니다. 다음 예제를 참조하십시오. zfs list 명령은 파일 시스템에서 사용 가능한 공간을 나열하는데, 이는 디스크 공간에서 ZFS 풀 중복성 메타데이터 오버헤드(있는 경우)를 뺀 값입니다.

- **중복되지 않은 저장소 풀** – 풀이 한 개의 136GB 디스크로 만들어진 경우 zpool list 명령은 SIZE 및 초기 FREE 값을 136GB로 보고합니다. zfs list 명령이 보고하는 초기 AVAIL 공간은 풀 메타 데이터 오버헤드의 양이 작기 때문에 134GB입니다. 예를 들면 다음과 같습니다.

```
# zpool create tank c0t6d0
# zpool list tank
NAME   SIZE  ALLOC   FREE    CAP  DEDUP  HEALTH  ALTROOT
tank   136G  95.5K   136G     0%  1.00x  ONLINE  -
# zfs list tank
NAME   USED  AVAIL  REFER  MOUNTPOINT
tank    72K   134G    21K  /tank
```

- **미러링된 저장소 풀**– 풀이 두 개의 136GB 디스크로 만들어진 경우 zpool list 명령은 SIZE를 136GB로 보고하고 초기 FREE 값을 136GB로 보고합니다. 이 보고를 **압축** 공간 값이라고 합니다. zfs list 명령이 보고하는 초기 AVAIL 공간은 풀 메타 데이터 오버헤드의 양이 작기 때문에 134GB입니다. 예를 들면 다음과 같습니다.

```
# zpool create tank mirror c0t6d0 c0t7d0
# zpool list tank
NAME   SIZE  ALLOC   FREE    CAP  DEDUP  HEALTH  ALTROOT
tank   136G  95.5K   136G     0%  1.00x  ONLINE  -
# zfs list tank
NAME   USED  AVAIL  REFER  MOUNTPOINT
tank    72K   134G    21K  /tank
```

- **RAID-Z 저장소 풀** – raidz2 풀이 세 개의 136GB 디스크로 만들어진 경우 zpool list 명령은 SIZE를 408GB로 보고하고 초기 FREE 값을 408GB로 보고합니다. 이 보고를 **압축** 공간 값이라고 하는데, 여기에는 중복성 오버헤드(예: 패리티 정보)가 포함됩니다. zfs list 명령이 보고하는 초기 AVAIL 공간은 풀 중복성 오버헤드로 인해 133GB입니다. RAID-Z 풀에 대한 zpool list와 zfs list 출력 결과 사이의 공간 차이는 zpool list가 압축 풀 공간을 보고하기 때문입니다.

```
# zpool create tank raidz2 c0t6d0 c0t7d0 c0t8d0
# zpool list tank
NAME   SIZE  ALLOC   FREE    CAP  DEDUP  HEALTH  ALTROOT
tank   408G   286K   408G     0%  1.00x  ONLINE  -
# zfs list tank
NAME   USED  AVAIL  REFER  MOUNTPOINT
tank  73.2K   133G  20.9K  /tank
```

손상된 데이터 복구

다음 절에서는 데이터 손상 유형을 식별하고 데이터를 복구하는 방법에 대해 설명합니다.

ZFS는 체크섬, 중복성 및 자체 치료 데이터를 사용하여 데이터 손상 위험을 최소화합니다. 그럼에도 불구하고, 풀이 중복되지 않은 경우, 풀 디그레이드 중에 손상이 발생한 경우 또는 일련의 이벤트가 동시에 발생하여 여러 데이터 복사본이 손상되는 경우 데이터 손상이 발생할 수 있습니다. 소스와 관계없이 결과는 같습니다. 즉, 데이터가 손상되어 더 이상 액세스할 수 없습니다. 수행할 조치는 손상된 데이터의 유형 및 관련 값에 따라 달라집니다. 두 가지 기본 유형의 데이터가 손상될 수 있습니다.

- 풀 메타 데이터 - 풀을 열고 데이터 세트에 액세스하기 위해서는 구문 분석할 특정한 양의 데이터가 ZFS에 필요합니다. 이 데이터가 손상될 경우 전체 풀 또는 데이터 세트 계층의 일부분을 사용할 수 없게 됩니다.
- 객체 데이터 – 이 경우 특정 파일 또는 디렉토리 내에서 손상이 발생합니다. 이 문제로 인해 파일 또는 디렉토리의 일부분에 액세스할 수 없게 되거나 객체가 모두 손상됩니다.

일반 작업 중이나 스크러빙을 통해 데이터를 확인할 수 있습니다. 풀 데이터의 무결성을 확인하는 방법에 대한 자세한 내용은 291 페이지 "ZFS 파일 시스템 무결성 검사"를 참조하십시오.

데이터 손상 유형 식별

기본적으로 `zpool status` 명령은 손상이 발생했다는 것만 표시하고 이 손상이 발생한 위치는 표시하지 않습니다. 예를 들면 다음과 같습니다.:

```
# zpool status monkey
  pool: monkey
 state: ONLINE
status: One or more devices has experienced an error resulting in data
        corruption.  Applications may be affected.
action: Restore the file in question if possible.  Otherwise restore the
        entire pool from backup.
   see: http://www.sun.com/msg/ZFS-8000-8A
 scrub: scrub completed after 0h0m with 8 errors on Tue Jul 13 13:17:32 2010
config:

        NAME        STATE     READ WRITE CKSUM
        monkey      ONLINE       8     0     0
          c1t1d0    ONLINE       2     0     0
          c2t5d0    ONLINE       6     0     0

errors: 8 data errors, use '-v' for a list
```

각 오류는 지정된 시점에서 오류가 발생했다는 사실만 표시할 뿐 반드시 시스템에 아직도 이러한 오류가 있다는 것은 아닙니다. 그러나 이는 정상적인 상황에서 적용되는 내용입니다. 일시적인 특정 작동 중단은 데이터 손상을 일으키지만 이러한 손상은 작동 중단이 종료되면 자동으로 복구됩니다. 풀의 모든 활성 블록을 검사하기 위해 전체 풀 스크러빙이 보장됩니다. 따라서 스크러빙이 완료될 때마다 오류 로그가 재설정됩니다. 더 이상 오류가 존재하지 않는 것으로 확인되어 스크러빙이 완료될 때까지 기다리지 않아도 되는 경우 `zpool online` 명령을 사용하여 풀의 모든 오류를 재설정하십시오.

데이터 손상이 풀 전체 메타 데이터에서 발생한 경우 출력 결과가 약간 다릅니다. 예를 들면 다음과 같습니다.

```
# zpool status -v morpheus
  pool: morpheus
    id: 1422736890544688191
 state: FAULTED
status: The pool metadata is corrupted.
action: The pool cannot be imported due to damaged devices or data.
   see: http://www.sun.com/msg/ZFS-8000-72
config:

        morpheus    FAULTED   corrupted data
          c1t10d0   ONLINE
```

풀 전체 손상의 경우 풀이 원하는 중복성 레벨을 제공할 수 없기 때문에 풀이 `FAULTED` 상태가 됩니다.

손상된 파일 또는 디렉토리 복구

파일 또는 디렉토리가 손상된 경우 손상 유형에 따라 시스템이 계속 작동할 수 있습니다. 시스템에 정상적인 데이터 복사본이 없을 경우 결과적으로 손상을 복구할 수 없습니다. 중요한 데이터일 경우 영향을 받는 데이터를 백업에서 복원해야 합니다. 이 경우에도 전체 풀을 복원하지 않고 이 손상에서 복구할 수 있습니다.

파일 데이터 블록에서 손상이 발생한 경우 파일을 안전하게 제거하면 시스템에서 오류가 지워집니다. 영구적인 오류가 발생한 파일 이름 목록을 표시하려면 `zpool status -v` 명령을 사용하십시오. 예를 들면 다음과 같습니다.

```
# zpool status -v
  pool: monkey
 state: ONLINE
status: One or more devices has experienced an error resulting in data
        corruption.  Applications may be affected.
action: Restore the file in question if possible.  Otherwise restore the
        entire pool from backup.
   see: http://www.sun.com/msg/ZFS-8000-8A
 scrub: scrub completed after 0h0m with 8 errors on Tue Jul 13 13:17:32 2010
config:

        NAME        STATE     READ WRITE CKSUM
        monkey      ONLINE       8     0     0
```

```
          c1t1d0    ONLINE       2     0     0
          c2t5d0    ONLINE       6     0     0

errors: Permanent errors have been detected in the following files:

/monkey/a.txt
/monkey/bananas/b.txt
/monkey/sub/dir/d.txt
monkey/ghost/e.txt
/monkey/ghost/boo/f.txt
```

영구적인 오류가 발생한 파일 이름 목록은 다음과 같이 설명될 수 있습니다.

- 전체 파일 경로가 발견되고 데이터 세트가 마운트된 경우, 전체 파일 경로가 표시됩니다. 예를 들면 다음과 같습니다.

  ```
  /monkey/a.txt
  ```

- 전체 파일 경로가 발견되었지만 데이터 세트가 마운트되지 않은 경우, 앞에 슬래시(/)가 붙지 않고 데이터 세트 내의 파일에 대한 경로가 이어지는 데이터 세트 이름이 표시됩니다. 예를 들면 다음과 같습니다.

  ```
  monkey/ghost/e.txt
  ```

- `dnode_t`의 경우와 같이, 오류로 인해 또는 객체에 연관된 실제 파일 경로가 없어 파일 경로에 대한 객체 수를 성공적으로 변환할 수 없는 경우 데이터 세트 이름 뒤에 객체 번호가 표시됩니다. 예를 들면 다음과 같습니다.

  ```
  monkey/dnode:<0x0>
  ```

- MOS(Metaobject Set)에 있는 객체가 손상된 경우 특수 태그 `<metadata>` 뒤에 객체 번호가 표시됩니다.

손상이 디렉토리 또는 파일의 메타 데이터 내에서 발생한 경우 파일을 다른 곳으로 이동할 수만 있습니다. 파일이나 디렉토리를 덜 편리한 위치로 안전하게 이동하면 원본 객체를 해당 위치에서 복원할 수 있습니다.

여러 블록 참조를 사용하여 손상된 데이터 복구

손상된 파일 시스템에 스냅샷과 같이 여러 블록 참조가 있는 손상된 데이터가 있는 경우 `zpool status -v` 명령은 **모든** 손상된 데이터 경로를 표시하지 않습니다. 손상된 데이터에 대한 현재 `zpool status` 보고는 메타 데이터 손상 정도에 따라 제한되며 `zpool status` 명령 실행 후 블록이 재사용된 경우로 제한됩니다. 중복 제거된 블록은 모든 손상된 데이터를 더욱 복잡하게 만듭니다.

손상된 데이터가 있고 `zpool status -v` 명령으로 스냅샷 데이터가 영향을 받은 것으로 확인된 경우 다음 명령을 실행하여 추가 손상된 경로가 있는지 식별하는 것이 좋습니다.

ZFS 저장소 풀 전반의 손상 복구

풀 메타 데이터에서 손상이 발생했는데 이 손상으로 인해 풀을 열거나 가져올 수 없는 경우 다음 옵션을 사용할 수 있습니다.

- zpool clear -F 명령 또는 zpool import - F 명령을 사용하여 풀을 복구할 수 있습니다. 이 명령은 마지막 몇 개의 풀 트랜잭션을 작동 상태로 롤백하려고 시도합니다. zpool status 명령을 사용하여 손상된 풀 및 권장되는 복구 단계를 검토할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool status
  pool: tpool
 state: FAULTED
status: The pool metadata is corrupted and the pool cannot be opened.
action: Recovery is possible, but will result in some data loss.
        Returning the pool to its state as of Wed Jul 14 11:44:10 2010
        should correct the problem.  Approximately 5 seconds of data
        must be discarded, irreversibly.  Recovery can be attempted
        by executing 'zpool clear -F tpool'.  A scrub of the pool
        is strongly recommended after recovery.
   see: http://www.sun.com/msg/ZFS-8000-72
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
         tpool      FAULTED      0     0     1  corrupted data
          c1t1d0    ONLINE       0     0     2
          c1t3d0    ONLINE       0     0     4
```

위 출력 결과에 설명된 복구 프로세스는 다음 명령에 사용하기 위한 것입니다.

```
# zpool clear -F tpool
```

손상된 저장소 풀을 가져오려고 시도하면 다음과 비슷한 메시지가 표시됩니다.

```
# zpool import tpool
cannot import 'tpool': I/O error
        Recovery is possible, but will result in some data loss.
        Returning the pool to its state as of Wed Jul 14 11:44:10 2010
        should correct the problem.  Approximately 5 seconds of data
        must be discarded, irreversibly.  Recovery can be attempted
        by executing 'zpool import -F tpool'.  A scrub of the pool
        is strongly recommended after recovery.
```

위 출력 결과에 설명된 복구 프로세스는 다음 명령에 사용하기 위한 것입니다.

```
# zpool import -F tpool
Pool tpool returned to its state as of Wed Jul 14 11:44:10 2010.
Discarded approximately 5 seconds of transactions
```

손상된 풀이 zpool.cache 파일에 있는 경우, 시스템을 부트하면 문제가 발견되고 손상된 풀이 zpool status 명령에 보고됩니다. 이 풀이 zpool.cache 파일에 없는 경우 풀을 가져오거나 열 수 없으므로 해당 풀을 가져오려고 시도하면 손상된 풀 메시지가 표시됩니다.

- 손상된 풀은 읽기 전용 모드로 가져올 수 있습니다. 이 방법으로 풀을 가져와서 데이터에 액세스할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool import -o readonly=on tpool
```

풀을 읽기 전용으로 가져오는 방법은 98 페이지 "읽기 전용 모드로 풀 가져오기"를 참조하십시오.

- `zpool import -m` 명령을 사용하여 누락된 로그 장치가 있는 풀을 가져올 수 있습니다. 자세한 내용은 97 페이지 "누락된 로그 장치가 있는 풀 가져오기"를 참조하십시오.
- 어떠한 풀 복구 방법으로도 풀을 복구할 수 없는 경우 백업 복사본에서 풀과 모든 데이터를 복원해야 합니다. 풀 구성 및 백업 전략에 따라 사용하는 방식이 달라집니다. 먼저, 풀이 삭제된 후 구성을 다시 만들 수 있도록 `zpool status` 명령에 의해 표시된 대로 구성을 저장합니다. 그런 다음 `zpool destroy -f` 명령을 사용하여 풀을 삭제합니다.

  또한 풀에 액세스할 수 없게 될 경우 데이터 세트의 레이아웃 및 로컬에 설정된 다양한 등록 정보에도 액세스할 수 없게 되므로 이 정보를 안전한 위치에 보관하십시오. 풀 구성 및 데이터 세트 레이아웃을 사용하면 풀 삭제 후 전체 구성을 재구성할 수 있습니다. 그런 다음 백업 또는 복원 전략을 사용하여 데이터를 채울 수 있습니다.

손상된 ZFS 구성 복구

ZFS는 루트 파일 시스템에서 활성 풀 및 구성 캐시를 유지 관리합니다. 이 캐시 파일이 손상되거나 디스크에 저장된 구성 정보와 동기화되지 않을 경우 풀을 더 이상 열 수 없습니다. ZFS는 이 상황을 방지하려고 시도하지만, 기본 저장소의 품질을 감안할 때 임의적인 손상이 항상 발생할 수 있습니다. 이 상황은 보통 풀이 사용 가능해야 할 때 시스템에서 사라지도록 만듭니다. 또한 알 수 없는 개수의 최상위 가상 장치가 누락된 부분 구성으로 표시될 수도 있습니다. 어떤 경우든지 풀(표시되는 경우)을 내보낸 다음 다시 가져와서 구성을 복구할 수 있습니다.

풀 가져오기 및 내보내기에 대한 자세한 내용은 92 페이지 "ZFS 저장소 풀 마이그레이션"을 참조하십시오.

부트할 수 없는 시스템 복구

ZFS는 오류가 발생해도 강력하고 안정적이도록 설계되었습니다. 그렇지만 소프트웨어 버그 또는 예상치 않은 특정 문제로 인해 풀에 액세스할 때 시스템이 패닉 상태가 될 수 있습니다. 부트 프로세스의 일부로 각 풀을 열어야 하는데, 이는 오류 발생 시 시스템이 패닉-재부트를 반복하게 된다는 것을 의미합니다. 이 상황에서 복구하려면 시작 시 풀을 검색하지 않도록 ZFS에 알려야 합니다.

ZFS는 `/etc/zfs/zpool.cache`에서 사용 가능한 풀 및 구성의 내부 캐시를 유지 관리합니다. 이 파일의 위치와 내용은 사용자마다 다르며 변경될 수 있습니다. 시스템을 부트할 수 없는 경우 `-m milestone=none` 부트 옵션을 사용하여 마일스톤 `none`으로 부트하십시오. 시스템이 작동하면 루트 파일 시스템을 쓰기 가능 상태로 다시 마운트한 다음 `/etc/zfs/zpool.cache` 파일의 이름을 바꾸거나 다른 위치로 이동하십시오. 이와

같이 하면 ZFS에서 시스템에 풀이 있다는 것을 인식하지 못해 비상적인 풀에 액세스하여 문제를 일으키는 것을 방지할 수 있습니다. 그런 다음 `svcadm milestone all` 명령을 실행하여 정상 시스템 상태를 계속 유지합니다. 대체 루트에서 부트하여 복구를 수행하는 경우에도 이와 비슷한 프로세스를 사용할 수 있습니다.

시스템이 작동되면 `zpool import` 명령을 사용하여 풀을 가져올 수 있습니다. 그러나 이 명령은 풀 액세스 시 동일한 방식을 사용하므로 부트 중에 발생한 것과 같은 오류가 발생할 수 있습니다. 시스템에 풀이 여러 개 있을 경우 다음을 수행하십시오.

- 위 텍스트에서 설명한 것과 같이 `zpool.cache` 파일의 이름을 바꾸거나 다른 위치로 이동합니다.
- `fmdump -eV` 명령으로 보고된 치명적인 오류가 있는 풀을 표시하여 문제가 발생한 풀을 확인합니다.
- `fmdump` 출력 결과에 설명된 문제가 있는 풀은 건너뛰면서 풀을 하나씩 가져옵니다.

◆ ◆ ◆ 11 장

11 Oracle Solaris ZFS 권장 방법

이 장에서는 ZFS 저장소 풀과 파일 시스템을 만들고, 모니터 및 유지 관리하는 권장 방법에 대해 설명합니다.

이 장에서는 다음과 같은 내용을 다룹니다.

저장소 풀 권장 방법

다음 절에서는 ZFS 저장소 풀을 만들고 모니터하는 권장 방법을 제공합니다. 저장소 풀 문제 해결에 대한 자세한 내용은 10 장, "Oracle Solaris ZFS 문제 해결 및 풀 복구"를 참조하십시오.

일반 시스템 방법

- 최신 Solaris 릴리스와 패치를 사용하여 시스템을 최신 상태로 유지합니다.
- 풀 장치를 변경하거나 미러링된 저장소 풀을 분할하기 전에 데이터가 안전하게 기록되도록 보장하기 위해 컨트롤러가 캐시 비우기 명령을 따르는지 확인합니다. Oracle/Sun 하드웨어에서는 일반적으로 문제가 되지 않지만 하드웨어의 캐시 비우기 설정이 사용으로 설정되었는지 확인하는 것이 좋습니다.
- 실제 시스템 작업 부하에 대한 크기 메모리 요구 사항을 충족합니다.
  - 데이터베이스 응용 프로그램 등에 대해 알려진 응용 프로그램 메모리 단위를 사용하여 ARC 크기 상한을 제한하면 응용 프로그램이 ZFS 캐시에서 필요한 메모리를 재생 이용할 필요가 없습니다.
  - 중복 제거 메모리 요구 사항을 고려해 보십시오.
  - 다음 명령으로 ZFS 메모리 사용량을 식별합니다.

```
# mdb -k
> ::memstat
Page Summary                Pages                MB  %Tot
------------     ----------------  ----------------  ----
Kernel                     388117              1516   19%
ZFS File Data               81321               317    4%
Anon                        29928               116    1%
Exec and libs                1359                 5    0%
Page cache                   4890                19    0%
Free (cachelist)             6030                23    0%
Free (freelist)           1581183              6176   76%

Total                     2092828              8175
Physical                  2092827              8175
> $q
```

  - 메모리 손상을 방지하지 위해 ECC 메모리를 사용하는 것이 좋습니다. 기록되지 않은 메모리 손상으로 인해 데이터가 손상될 수 있습니다.

- 정기 백업 수행 – ZFS 중복을 사용하여 만든 풀은 하드웨어 고장으로 인한 작동 중지 시간을 줄이는 데 도움이 되지만 하드웨어 고장, 정전 또는 연결 해제된 케이블의 영향을 받습니다. 정기적으로 데이터를 백업해야 합니다. 데이터가 중요한 경우 백업해야 합니다. 데이터 복사본을 제공하는 여러 가지 방법은 다음과 같습니다.
  - 정기 또는 일별 ZFS 스냅샷
  - ZFS 풀 데이터의 주별 백업. `zpool split` 명령을 사용하여 미러링된 ZFS 저장소 풀의 정확한 복제본을 만들 수 있습니다.
  - 엔터프라이즈 레벨 백업 제품을 사용한 월별 백업
- 하드웨어 RAID
  - ZFS가 저장소와 중복을 관리할 수 있도록 하드웨어 RAID 대신 JBOD 모드를 저장소 배열에 사용하는 것이 좋습니다.
  - 하드웨어 RAID나 ZFS 중복(또는 둘 다)을 사용합니다.
  - ZFS 중복 사용 시 여러 가지 이점 – 운용 환경의 경우 데이터 불일치를 복구할 수 있도록 ZFS를 구성합니다. 기본 저장 장치에 구현된 RAID 레벨에 관계없이 RAID-Z, RAID-Z-2, RAID-Z-3, 미러와 같은 ZFS 중복을 사용합니다. 이러한 중복을 사용하면 기본 저장 장치나 그 호스트 연결에 결함이 발생할 경우 ZFS에서 복구하고 수리할 수 있습니다.

  307 페이지 “로컬 또는 네트워크 연결 저장소 어레이에서 풀 만들기 실행 방법”도 참조하십시오.
- 충돌 덤프는 물리적 메모리 범위의 1/2- 3/4 크기로 추가 디스크 공간을 사용합니다.

ZFS 저장소 풀 만들기 방식

다음 절에서는 일반 및 특정 풀 방법을 제공합니다.

일반 저장소 풀 방법

- 전체 디스크를 사용하여 디스크 쓰기 캐시를 사용으로 설정하고 유지 관리를 용이하게 합니다. 슬라이스에 풀을 만들면 디스크 관리 및 복구가 더 복잡해집니다.
- ZFS가 데이터 불일치를 복구할 수 있도록 ZFS 중복을 사용합니다.
  - 비중복 풀을 만들면 다음 메시지가 표시됩니다.

    ```
    # zpool create tank c4t1d0 c4t3d0
    'tank' successfully created, but with no redundancy; failure
    of one device will cause loss of the pool
    ```
  - 미러링된 풀의 경우 미러링된 디스크 쌍을 사용합니다.
  - RAID-Z 풀의 경우 VDEV당 3-9개 디스크 그룹으로 묶습니다.
  - 동일한 풀 내에서는 RAID-Z 및 미러링된 구성 요소를 혼합하지 마십시오. 이러한 풀은 관리하기 어렵고 성능도 저하될 수 있습니다.
- 핫 스페어를 사용하여 하드웨어 고장으로 인한 작동 중지 시간을 줄입니다.
- 장치 간에 I/O가 균형을 이루도록 유사한 크기의 디스크를 사용합니다.
  - 작은 LUN을 큰 LUN으로 확장할 수 있습니다.
  - 최적의 metaslab 크기를 유지하려면 차이가 심한 경우(예: 128MB와 2TB) LUN을 확장하지 마십시오.
- 더 빠른 시스템 복구를 지원하기 위해 작은 루트 풀과 큰 데이터 풀을 만드는 것이 좋습니다.
- 권장되는 최소 풀 크기는 8GB입니다. 최소 풀 크기는 64MB이지만 8GB 미만의 경우 무료 풀 공간을 할당 및 회수하기가 어려워집니다.
- 권장되는 최대 풀 크기는 작업 부하 또는 데이터 크기에 적합해야 합니다. 정기적으로 백업할 수 있는 것보다 많은 데이터를 저장하려고 시도하지 마십시오. 그렇지 않으면 예측하지 못한 이벤트로 인해 데이터가 위험해질 수 있습니다.

루트 풀 생성 방법

- s* 식별자를 사용하여 슬라이스로 루트 풀을 만듭니다. p* 식별자는 사용하지 마십시오. 일반적으로 시스템의 ZFS 루트 풀은 시스템 설치 시 만들어집니다. 다른 루트 풀을 만들거나 루트 풀을 다시 만드는 경우 다음과 비슷한 구문을 사용합니다.

  ```
  # zpool create rpool c0t1d0s0
  ```

  또는 미러링된 루트 풀을 만듭니다. 예를 들면 다음과 같습니다.

  ```
  # zpool create rpool mirror c0t1d0s0 c0t2d0s0
  ```
  - 루트 풀은 미러된 구성 또는 단일 디스크 구성으로 만들어야 합니다. RAID-Z 또는 스트라이프 구성은 모두 지원되지 않습니다. `zpool add` 명령을 사용하여 디스크를 추가함으로써 여러 미러된 최상위 레벨 가상 장치를 만들 수 없지만, `zpool attach` 명령을 사용하여 미러된 가상 장치를 확장할 수는 있습니다.
  - 루트 풀은 별도의 로그 장치를 가질 수 없습니다.

- AI 설치 도중 풀 등록 정보를 설정할 수 있지만 gzip 압축 알고리즘은 루트 풀에서 지원되지 않습니다.
- 초기 설치로 루트 풀을 만든 후에는 루트 풀 이름을 바꾸지 마십시오. 루트 풀의 이름을 바꾸면 시스템이 부트되지 않을 수 있습니다.
- 루트 풀 디스크는 특히 기업 환경에서 연속된 작업을 위해 매우 중요하므로 운용 시스템을 위한 루트 풀을 USB 메모리에 만들지 마십시오. 루트 풀에는 시스템의 내장 디스크를 사용하고 비루트 데이터에 대해 사용하는 것과 최소한 동일한 품질의 디스크를 사용하십시오. 또한 USB 메모리가 물리적 메모리 크기의 1/2 이상에 해당하는 덤프 볼륨 크기를 지원할 수 있을 정도로 크지 않을 수 있습니다.

비루트 풀 생성 방법

- d* 식별자를 사용하여 전체 디스크로 비루트 풀을 만듭니다. p* 식별자는 사용하지 마십시오.
  - ZFS는 추가 볼륨 관리 소프트웨어 없이도 잘 작동합니다.
  - 최상의 성능을 위해 개별 디스크 또는 소수의 디스크로 구성된 최소 LUN을 사용합니다. ZFS에 LUN 설정을 보다 자세히 표시하면 ZFS가 더 나은 I/O 일정 잡기 결정을 내릴 수 있습니다.
  - 여러 컨트롤러에서 중복 풀 구성을 만들어 컨트롤러 오류로 인한 작동 중지 시간을 줄입니다.
  - **미러링된 저장소 풀** – 추가 디스크 공간을 사용하지만 일반적으로 모든 읽기가 작을 때 성능이 더 좋습니다.

    ```
    # zpool create tank mirror c1d0 c2d0 mirror c3d0 c4d0
    ```

  - **RAID-Z 저장소 풀** – 패리티가 1(raidz), 2(raidz2) 또는 3(raidz3)인 3개의 패리티 전략을 사용하여 만들 수 있습니다. RAID-Z 구성은 디스크 공간을 최대화하며 일반적으로 데이터를 큰 청크(128K 이상)로 쓰고 읽을 때 성능이 향상됩니다.
    - 각각 3개 디스크(2+1)의 2개 VDEV가 포함된 단일 패리티 RAID-Z(raidz) 구성을 고려해 보십시오.

      ```
      # zpool create rzpool raidz1 c1t0d0 c2t0d0 c3t0d0 raidz1 c1t1d0 c2t1d0 c3t1d0
      ```

    - RAIDZ-2 구성은 더 향상된 데이터 가용성을 제공하며, RAID-Z와 비슷한 성능을 제공합니다. 또한 RAID-Z 또는 양방향 미러에 비해 상당히 향상된 MTTDL(Mean Time To Data Loss)을 제공합니다. 6개의 디스크(4+2)에 이중 패리티 RAID-Z(raidz2) 구성을 만듭니다.

      ```
      # zpool create rzpool raidz2 c0t1d0 c1t1d0 c4t1d0 c5t1d0 c6t1d0 c7t1d0
      raidz2 c0t2d0 c1t2d0 c4t2d0 c5t2d0 c6t2d0 c7t2d
      ```

    - RAIDZ-3 구성은 디스크 공간을 최대화하며, 세 개의 디스크 오류를 견딜 수 있으므로 뛰어난 가용성을 제공합니다. 9개 디스크(6+3)에서 삼중 패리티 RAID-Z(raidz3) 구성을 만듭니다.

      ```
      # zpool create rzpool raidz3 c0t0d0 c1t0d0 c2t0d0 c3t0d0 c4t0d0
      c5t0d0 c6t0d0 c7t0d0 c8t0d0
      ```

로컬 또는 네트워크 연결 저장소 어레이에서 풀 만들기 실행 방법

로컬 또는 원격으로 연결된 저장소 어레이에서 ZFS 저장소 풀을 만들 때는 다음과 같은 저장소 풀 실행 방법을 참조하십시오.

- SAN 장치에 풀을 만들고, 네트워크 연결 속도가 느린 경우, 풀의 장치가 일정 기간 동안 `UNAVAIL` 상태가 될 수 있습니다. 네트워크 연결 상태가 연속적으로 데이터를 제공하는 데 적합한지 평가해야 합니다. 또한 루트 풀용으로 SAN 장치를 사용할 경우 시스템을 부트하는 즉시 해당 장치를 사용할 수 없고 루트 풀의 장치가 `UNAVAIL` 상태가 될 수 있습니다.
- ZFS가 쓰기 캐시 비우기를 요청한 후 디스크 어레이가 해당 캐시를 비우지 않는지 해당 어레이 제조업체에 확인하십시오.
- Oracle Solaris ZFS가 로컬의 작은 디스크 캐시를 활성화하여 적절한 시간에 비워질 수 있도록 디스크 슬라이스가 아닌 전체 디스크를 저장소 풀 장치로 사용하십시오.
- 최상의 성능을 위해서는 어레이의 각 물리적 디스크에 대해 하나의 LUN을 만드십시오. 큰 LUN을 하나만 사용할 경우 ZFS가 너무 적은 수의 읽기 I/O 작업을 대기열에 넣어서 실제로 저장소에 대한 최적의 성능을 얻을 수 없습니다. 반대로 작은 LUN을 너무 많이 사용하면 보류 중인 읽기 I/O 작업 수가 너무 많아져서 저장소를 가득 채울 수 있습니다.
- 가상 공간 할당을 위해 동적(또는 씬) 프로비저닝 소프트웨어를 사용하는 저장소 어레이는 Oracle Solaris ZFS에 권장되지 않습니다. Oracle Solaris ZFS가 수정된 데이터를 빈 공간에 기록할 때는 전체 LUN에 기록합니다. Oracle Solaris ZFS 쓰기 프로세스는 저장소 어레이의 관점에서 볼 때 모든 가상 공간을 할당하므로 동적 프로비저닝의 이점과는 반대가 됩니다.

  ZFS를 사용할 때는 동적 프로비저닝 소프트웨어가 필요하지 않을 수 있습니다.

  - 기존 ZFS 저장소 풀에서 LUN을 확장하여 새 공간을 사용할 수 있습니다.
  - 작은 LUN을 큰 LUN으로 바꿀 경우에도 비슷한 결과를 얻을 수 있습니다.
  - 풀의 저장소 요구를 평가하고 필요한 저장소 요구에 따라 더 작은 LUN으로 풀을 만들 경우에도 더 많은 공간이 필요할 때는 항상 LUN을 더 큰 크기로 확장할 수 있습니다.
- 어레이가 개별 장치를 표시할 수 있는 경우(JBOD 모드), ZFS가 데이터 불일치를 보고하고 수정할 수 있도록 이 유형의 어레이에 중복된 ZFS 저장소 풀(미러 또는 RAID-Z)을 만드는 것이 좋습니다.

Oracle 데이터베이스에 대한 풀 생성 방법

Oracle 데이터베이스를 만드는 경우 다음 저장소 풀 방법을 고려해 보십시오.

- 미러링된 풀 또는 하드웨어 RAID를 풀에 사용합니다.
- RAID-Z 풀은 일반적으로 모든 읽기 작업 부하에 권장되지 않습니다.
- 데이터베이스 리두 로그에 별도의 로그 장치를 사용하여 작은 개별 풀을 만듭니다.
- 아카이브 로그에 대한 작은 개별 풀을 만듭니다.

자세한 내용은 다음 백서를 참조하십시오.

http://blogs.oracle.com/storage/entry/new_white_paper_configuring_oracle

VirtualBox에서 ZFS 저장소 풀 사용

- Virtual Box는 기본적으로 기본 저장소의 캐시 비우기 명령을 무시하도록 구성되어 있습니다. 즉, 시스템 충돌이나 하드웨어 오류의 경우 데이터가 손실될 수 있습니다.
- Virtual Box에서 캐시 비우기를 사용으로 설정하려면 다음 명령을 실행합니다.

```
VBoxManage setextradata <VM_NAME> "VBoxInternal/Devices/<type>/0/LUN#<n>/Config/IgnoreFlush" 0
```

  - <VM_NAME>은 가상 시스템의 이름입니다.
  - <type>은 컨트롤러 유형이며, piix3ide(일반적인 IDE 가상 컨트롤러를 사용하는 경우) 또는 ahci(SATA 컨트롤러를 사용하는 경우)입니다.
  - <n>은 디스크 번호입니다.

성능에 대한 저장소 풀 방법

- 최고 성능을 얻으려면 풀 용량을 80% 아래로 유지합니다.
- 모든 읽기/쓰기 작업 부하의 경우 미러링된 풀이 RAID-Z 풀보다 권장됩니다.
- 별도의 로그 장치
  - 동기식 쓰기 성능 향상을 위해 권장됩니다.
  - 높은 동기식 쓰기 부하 상태에서 기본 풀에 많은 로그 블록을 쓰지 않도록 단편화를 최소화합니다.
- 읽기 성능을 향상시키려면 별도의 캐시 장치를 사용하는 것이 좋습니다.
- 스크러빙/리실버링 - 수많은 장치가 있는 매우 큰 RAID-Z 풀은 스크러빙 및 리실버링 시간이 오래 걸립니다.
- 풀 성능이 느립니다. zpool status 명령을 사용하여 풀 성능 문제를 발생시키는 하드웨어 문제를 제외합니다. zpool status 명령에서 문제가 표시되지 않는 경우 fmdump 명령을 사용하여 하드웨어 오류를 표시하거나 fmdump -eV 명령을 사용하여 보고된 결함을 초래하지 않은 하드웨어 오류를 모두 검토합니다.

ZFS 저장소 풀 유지 관리 및 모니터링 방법

- 최상의 성능을 위해 풀 용량이 80% 미만인지 확인합니다.

  풀이 가득 차 있고 파일 시스템이 자주 업데이트되는 경우(예: 활발한 메일 서버) 풀 성능이 저하될 수 있습니다. 가득 찬 풀은 성능 저하를 일으킬 수 있지만 다른 문제는 없습니다. 주요 작업 부하가 변경할 수 없는 파일인 경우 풀 사용률을 95-96% 범위로 유지합니다. 95-96% 범위에서는 가장 정적인 컨텐츠조차 쓰기, 읽기, 리실버링 성능이 악화될 수 있습니다.
  - 풀 및 파일 시스템 공간이 가득 차지 않도록 이러한 공간을 모니터링합니다.

    - 파일 시스템 공간이 풀 용량의 80%를 초과하지 않도록 ZFS 쿼터 및 예약을 사용하십시오.
- 풀 건전성 모니터
    - 중복 풀의 경우 매주 `zpool status` 및 `fmdump`를 사용하여 풀을 모니터합니다.
    - 비중복 풀의 경우 격주로 `zpool status` 및 `fmdump`를 사용하여 풀을 모니터합니다.
- 정기적으로 `zpool scrub`을 실행하여 데이터 무결성 문제를 식별합니다.
    - 소비자 품질의 드라이브가 있는 경우, 주 단위 스크러빙 일정을 고려합니다.
    - 데이터 센터 품질의 드라이브가 있는 경우, 월 단위 스크러빙 일정을 고려합니다.
    - 모든 장치가 현재 작동하는지 확인하려면 장치를 교체하거나 풀 중복성을 일시적으로 줄이기 전에 스크러빙을 실행해야 합니다.
- 풀 또는 장치 오류 모니터링 - 아래 설명된 대로 `zpool status`를 사용합니다. `fmdump` 또는 `fmdump -eV`를 사용하여 장치 결함이나 오류가 발생했는지도 확인합니다.
    - 중복 풀의 경우 매주 `zpool status` 및 `fmdump`를 사용하여 풀 건전성을 모니터합니다.
    - 비중복 풀의 경우 격주로 `zpool status` 및 `fmdump`를 사용하여 풀 건전성을 모니터합니다.
- 풀 장치가 `UNAVAIL` 또는`OFFLINE`입니다. 풀 장치를 사용할 수 없는 경우 장치가 `format` 명령 출력에 나열되는지 확인합니다. 장치가 `format` 출력에 나열되지 않는 경우 ZFS에 표시되지 않습니다.

  풀 장치가 `UNAVAIL` 또는 `OFFLINE`인 경우 장치에서 장애가 발생했거나 케이블 연결이 해제되었거나 불량 케이블 또는 불량 컨트롤러와 같은 기타 하드웨어 문제로 인해 장치에 액세스할 수 없는 것입니다.
- 저장소 풀 공간을 모니터합니다. `zpool list` 명령과 `zfs list` 명령을 사용하여 파일 시스템 데이터에서 사용하는 디스크 크기를 식별합니다. ZFS 스냅샷은 디스크 공간을 사용할 수 있으며, `zfs list` 명령으로 나열되지 않는 경우 자동으로 디스크 공간을 사용할 수 있습니다. `zfs list -t` 스냅샷 명령을 사용하여 스냅샷에서 사용되는 디스크 공간을 식별합니다.

파일 시스템 권장 방법

다음 절에서는 파일 시스템 권장 방법에 대해 설명합니다.

파일 시스템 생성 방법

다음 절에서는 ZFS 파일 시스템 생성 방법에 대해 설명합니다.

- 홈 디렉토리에 대해 사용자당 파일 시스템 한 개를 만듭니다.
- 파일 시스템 쿼터와 예약을 사용하여 중요한 파일 시스템의 디스크 공간을 관리하고 예약하는 것이 좋습니다.

- 사용자 및 그룹 쿼터를 사용하여 여러 사용자가 있는 환경의 디스크 공간을 관리하는 것이 좋습니다.
- ZFS 등록 정보 상속을 사용하여 많은 종속 파일 시스템에 등록 정보를 적용합니다.

Oracle 데이터베이스에 대한 파일 시스템 생성 방법

Oracle 데이터베이스를 만드는 경우 다음 파일 시스템 방법을 고려해 보십시오.

- ZFS `recordsize` 등록 정보를 Oracle `db_block_size`와 일치시킵니다.
- 8KB `recordsize` 및 기본 `primarycache` 값을 사용하여 기본 데이터베이스 풀에 데이터베이스 테이블과 인덱스 파일 시스템을 만듭니다.
- 기본 `recordsize` 및 `primarycache` 값을 사용하여 기본 데이터베이스 풀에 임시 데이터를 만들고 테이블 공간 파일 시스템을 실행 취소합니다.
- 압축을 사용으로 설정하고 기본 `recordsize` 값과 `primarycache`를 `metadata`로 설정하여 아카이브 풀에 아카이브 로그 파일 시스템을 만듭니다.

자세한 내용은 다음 백서를 참조하십시오.

`http://blogs.oracle.com/storage/entry/new_white_paper_configuring_oracle`

ZFS 파일 시스템 모니터 방법

ZFS 파일 시스템을 모니터하여 사용 가능한지 확인하고 공간 사용 문제를 식별합니다.

- 매주 `zpool list` 및 `zfs list` 명령을 사용하여 파일 시스템 공간 가용성을 모니터합니다. 레거시 명령인 `du` 및 `df`는 종속 파일 시스템이나 스냅샷에서 소비하는 공간을 고려하지 않으므로 사용하지 마십시오.

  자세한 내용은 294 페이지 “ZFS 공간 문제 해결”을 참조하십시오.
- `zfs list -o space` 명령을 사용하여 파일 시스템 공간 사용을 표시합니다.
- 모르는 사이에 스냅샷이 파일 시스템 공간을 사용할 수 있습니다. 다음 구문을 사용하면 모든 데이터 세트 정보를 표시할 수 있습니다.

  ```
  # zfs list -t all
  ```
- 시스템을 설치할 때 별도의 `/var` 파일 시스템이 자동으로 생성되지만 이 파일 시스템에서 쿼터 및 예약을 설정하여 모르는 사이에 루트 풀 공간을 사용하지 않도록 해야 합니다.
- 또한 `fsstat` 명령을 사용하여 ZFS 파일 시스템의 파일 작업을 표시할 수 있습니다. 마운트 지점이나 파일 시스템 유형별로 작업을 보고할 수 있습니다. 다음 예제에서는 일반적인 ZFS 파일 시스템 작업을 보여 줍니다.

  ```
  # fsstat /
   new  name   name  attr  attr lookup rddir  read read  write write
   file remov  chng   get   set    ops   ops   ops bytes   ops bytes
    832   589   286  837K 3.23K  2.62M 20.8K 1.15M 1.75G 62.5K  348M /
  ```

- 백업
  - 파일 시스템 스냅샷을 보관합니다.
  - 엔터프라이즈 레벨 소프트웨어는 주별/월별 백업을 고려하십시오.
  - 베어 메탈 복구를 위해 루트 풀 스냅샷을 원격 시스템에 저장합니다.

◆ ◆ ◆ 부 록 A

Oracle Solaris ZFS 버전 설명

이 부록은 사용 가능한 ZFS 버전, 각 버전의 기능 및 ZFS 버전과 기능을 제공하는 Solaris OS에 대해 설명합니다.

이 부록은 다음과 같은 절으로 구성됩니다.

ZFS 버전 개요

새로운 ZFS 풀 및 파일 시스템 기능이 새로 도입되어 Solaris 릴리스에서 제공하는 특정 ZFS 버전을 통해 액세스할 수 있습니다. `zpool upgrade` 또는 `zfs upgrade`를 사용하여 풀 또는 파일 시스템이 현재 실행 중인 Solaris 릴리스에서 제공하는 버전보다 낮은 버전인지 식별할 수 있습니다. 이 명령을 사용하여 풀 및 파일 시스템 버전을 업그레이드할 수도 있습니다.

`zpool upgrade` 및 `zfs upgrade` 명령 사용에 대한 자세한 내용은 200 페이지 “ZFS 파일 시스템 업그레이드” 및 101 페이지 “ZFS 저장소 풀 업그레이드”를 참조하십시오.

ZFS 풀 버전

다음 표에서는 Oracle Solaris 릴리스에서 제공하는 ZFS 풀 버전 목록을 보여줍니다.

버전	Solaris 10	설명
1	Solaris 10 6/06	초기 ZFS 버전
2	Solaris 10 11/06	Ditto 블록(복제된 메타 데이터)

버전	Solaris 10	설명
3	Solaris 10 11/06	핫 스패어 및 이중 패리티 RAID-Z
4	Solaris 10 8/07	`zpool history`
5	Solaris 10 10/08	`gzip` 압축 알고리즘
6	Solaris 10 10/08	`bootfs` 풀 등록 정보
7	Solaris 10 10/08	분리된 계획 로그 장치
8	Solaris 10 10/08	위임 관리
9	Solaris 10 10/08	`refquota` 및 `refreservation` 등록 정보
10	Solaris 10 5/09	캐시 장치
11	Solaris 10 10/09	향상된 스크러빙 성능
12	Solaris 10 10/09	스냅샷 등록 정보
13	Solaris 10 10/09	`snapused` 등록 정보
14	Solaris 10 10/09	`aclinherit passthrough-x` 등록 정보
15	Solaris 10 10/09	사용자 및 그룹 공간 계산
16	Solaris 10 9/10	`stmf` 등록 정보 지원
17	Solaris 10 9/10	삼중 패리티 RAID-Z
18	Solaris 10 9/10	스냅샷 사용자 유지
19	Solaris 10 9/10	로그 장치 제거
20	Solaris 10 9/10	`Compression using zle`(0 길이 인코딩)
21	Solaris 10 9/10	예약됨
22	Solaris 10 9/10	수신된 등록 정보
23	Solaris 10 8/11	Slim ZIL
24	Solaris 10 8/11	시스템 속성
25	Solaris 10 8/11	향상된 스크러빙 통계
26	Solaris 10 8/11	향상된 스냅샷 삭제 성능
27	Solaris 10 8/11	향상된 스냅샷 만들기 성능
28	Solaris 10 8/11	다중 vdev 대체
29	Solaris 10 8/11	RAID-Z/미러 하이브리드 할당자
30	Solaris 10 1/13	예약됨

버전	Solaris 10	설명
31	Solaris 10 1/13	향상된 `zfs list` 성능
32	Solaris 10 1/13	1MB 블록 크기

ZFS 파일 시스템 버전

다음 표에서는 Oracle Solaris 릴리스에서 제공하는 ZFS 파일 시스템 버전을 보여줍니다. 특정 파일 시스템 버전에서 제공되는 기능은 해당 특정 풀 버전이 필요합니다.

버전	Solaris 10	설명
1	Solaris 10 6/06	초기 ZFS 파일 시스템 버전
2	Solaris 10 10/08	향상된 디렉토리 항목
3	Solaris 10 10/08	대소문자 비구분 및 파일 시스템 고유 식별자(FUID)
4	Solaris 10 10/09	`userquota` 및 `groupquota` 등록 정보
5	Solaris 10 8/11	시스템 속성

색인

가

간

감

개

명

문

미

별

복

수

순

스

식

알

액

저

전

제

조

지

체

추

충

캐

쿼