Oracle® Solaris 11.1 관리: ZFS 파일 시스템

부품 번호: E36657–05
2013년 9월

ORACLE

Copyright © 2006, 2013, Oracle and/or its affiliates. All rights reserved.

본 소프트웨어와 관련 문서는 사용 제한 및 기밀 유지 규정을 포함하는 라이센스 계약서에 의거해 제공되며, 지적 재산법에 의해 보호됩니다. 라이센스 계약서 상에 명시적으로 허용되어 있는 경우나 법규에 의해 허용된 경우를 제외하고, 어떠한 부분도 복사, 재생, 번역, 방송, 수정, 라이센스, 전송, 배포, 진열, 실행, 발행, 또는 전시될 수 없습니다. 본 소프트웨어를 리버스 엔지니어링, 디스어셈블리 또는 디컴파일하는 것은 상호 운용에 대한 법규에 의해 명시된 경우를 제외하고는 금지되어 있습니다.

이 안의 내용은 사전 공지 없이 변경될 수 있으며 오류가 존재하지 않음을 보증하지 않습니다. 만일 오류를 발견하면 서면으로 통지해 주기 바랍니다.

만일 본 소프트웨어나 관련 문서를 미국 정부나 또는 미국 정부를 대신하여 라이센스한 개인이나 법인에게 배송하는 경우, 다음 공지 사항이 적용됩니다.

U.S. GOVERNMENT END USERS:

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

본 소프트웨어 혹은 하드웨어는 다양한 정보 관리 애플리케이션의 일반적인 사용을 목적으로 개발되었습니다. 본 소프트웨어 혹은 하드웨어는 개인적인 상해를 초래할 수 있는 애플리케이션을 포함한 본질적으로 위험한 애플리케이션에서 사용할 목적으로 개발되거나 그 용도로 사용될 수 없습니다. 만일 본 소프트웨어 혹은 하드웨어를 위험한 애플리케이션에서 사용할 경우, 라이센스 사용자는 해당 애플리케이션의 안전한 사용을 위해 모든 적절한 비상-안전, 백업, 대비 및 기타 조치를 반드시 취해야 합니다. Oracle Corporation과 그 자회사는 본 소프트웨어 혹은 하드웨어를 위험한 애플리케이션에서의 사용으로 인해 발생하는 어떠한 손해에 대해서도 책임지지 않습니다.

Oracle과 Java는 Oracle Corporation 및/또는 그 자회사의 등록 상표입니다. 기타의 명칭들은 각 해당 명칭을 소유한 회사의 상표일 수 있습니다.

Intel 및 Intel Xeon은 Intel Corporation의 상표 내지는 등록 상표입니다. SPARC 상표 일체는 라이센스에 의거하여 사용되며SPARC International, Inc.의 상표 내지는 등록 상표입니다. AMD, Opteron, AMD 로고, 및 AMD Opteron 로고는 Advanced Micro Devices의 상표 내지는 등록 상표입니다. UNIX는 The Open Group의 등록상표입니다.

본 소프트웨어 혹은 하드웨어와 관련문서(설명서)는 제 3자로부터 제공되는 컨텐츠, 제품 및 서비스에 접속할 수 있거나 정보를 제공합니다. Oracle Corporation과 그 자회사는 제 3자의 컨텐츠, 제품 및 서비스와 관련하여 어떠한 책임도 지지 않으며 명시적으로 모든 보증에 대해서도 책임을 지지 않습니다. Oracle Corporation과 그 자회사는 제 3자의 컨텐츠, 제품 및 서비스에 접속하거나 사용으로 인해 초래되는 어떠한 손실, 비용 또는 손해에 대해 어떠한 책임도 지지 않습니다.

130924@25097

목차

머리말 11

1 Oracle Solaris ZFS 파일 시스템(소개) 15
ZFS의 새로운 기능 15
향상된 ZFS 풀 장치 메시지 16
ZFS 파일 공유 향상 기능 16
공유 var 파일 시스템 17
EFI(GPT) 레이블 디스크에 대한 부트 지원 17
ZFS 명령 사용법 향상된 기능 18
ZFS 스냅샷 향상된 기능 18
ZFS 매뉴얼 페이지 변경(zfs.1m) 19
향상된 aclmode 등록 정보 19
물리적 위치로 풀 장치 식별 20
ZFS 그림자 마이그레이션 21
ZFS 파일 시스템 암호화 21
ZFS 전송 스트림의 향상된 기능 21
ZFS 스냅샷 차이(zfs diff) 22
ZFS 저장소 풀 복구 및 성능의 향상된 기능 22
ZFS 동기식 동작 조정 23
향상된 ZFS 풀 메시지 23
ZFS ACL 상호 운용성의 향상된 기능 24
미러링된 ZFS 저장소 풀 분할(zpool split) 25
ZFS iSCSI 변경 사항 25
새 ZFS 시스템 프로세스 25
ZFS 중복 제거 등록 정보 26
Oracle Solaris ZFS란? 26
ZFS 풀링된 저장소 27
트랜잭션 개념 27

체크섬 및 자체 치유 데이터 28
비교할 수 없는 확장성 28
ZFS 스냅샷 28
간소화된 관리 28
ZFS 용어 29
ZFS 구성 요소 명명 요구 사항 31
Oracle Solaris ZFS와 전통적인 파일 시스템의 차이 31
ZFS 파일 시스템 세분성 31
ZFS 디스크 공간 계산 32
ZFS 파일 시스템 마운트 33
기존 볼륨 관리 34
NFSv4 기반 Solaris ACL 모델 34

2 Oracle Solaris ZFS 시작하기 35
ZFS 권한 프로파일 35
ZFS 하드웨어 및 소프트웨어 요구 사항과 권장 사항 36
기본 ZFS 파일 시스템 만들기 36
기본 ZFS 저장소 풀 만들기 37
▼ ZFS 저장소 풀에 대한 저장소 요구 사항 식별 방법 37
▼ ZFS 저장소 풀을 만드는 방법 38
ZFS 파일 시스템 계층 만들기 38
▼ ZFS 파일 시스템 계층 확인 방법 39
▼ ZFS 파일 시스템을 만드는 방법 39

3 Oracle Solaris ZFS 저장소 풀 관리 43
ZFS 저장소 풀의 구성 요소 43
ZFS 저장소 풀의 디스크 사용 43
ZFS 저장소 풀에서 슬라이스 사용 45
ZFS 저장소 풀에서 파일 사용 46
ZFS 저장소 풀 고려 사항 47
ZFS 저장소 풀의 복제 기능 47
미러링된 저장소 풀 구성 48
RAID-Z 저장소 풀 구성 48
ZFS 하이브리드 저장소 풀 49
중복 구성에서 데이터 자가 치료 49

저장소 풀의 동적 스트라이프 49
ZFS 저장소 풀 만들기 및 삭제 50
ZFS 저장소 풀 만들기 50
저장소 풀 가상 장치 정보 표시 56
ZFS 저장소 풀 만들기 오류 처리 57
ZFS 저장소 풀 삭제 60
ZFS 저장소 풀의 장치 관리 61
저장소 풀에 장치 추가 61
저장소 풀에서 장치 연결 및 분리 66
미러링된 ZFS 저장소 풀을 분할하여 새로운 풀 만들기 67
저장소 풀에서 장치 온라인 및 오프라인 전환 70
저장소 풀 장치 오류 지우기 72
저장소 풀의 장치 교체 73
저장소 풀에서 핫 스패어 지정 75
ZFS 저장소 풀 등록 정보 관리 80
ZFS 저장소 풀 상태 질의 83
ZFS 저장소 풀에 대한 정보 표시 83
ZFS 저장소 풀에 대한 I/O 통계 보기 88
ZFS 저장소 풀의 건전성 상태 확인 90
ZFS 저장소 풀 마이그레이션 95
ZFS 저장소 풀 마이그레이션 준비 95
ZFS 저장소 풀 내보내기 96
가져올 수 있는 저장소 풀 결정 96
대체 디렉토리에서 ZFS 저장소 풀 가져오기 98
ZFS 저장소 풀 가져오기 98
삭제된 ZFS 저장소 풀 복구 102
ZFS 저장소 풀 업그레이드 104

4 ZFS 루트 풀 구성 요소 관리 107
ZFS 루트 풀 구성 요소 관리(개요) 107
ZFS 루트 풀 요구 사항 108
ZFS 루트 풀 관리 109
ZFS 루트 풀 설치 110
▼ ZFS 부트 환경 업데이트 방법 110
▼ 대체 BE를 마운트하는 방법 111

▼ 미러링된 루트 풀을 구성하는 방법(SPARC 또는 x86/VTOC) 112
▼ 미러링된 루트 풀을 구성하는 방법(x86/EFI(GPT)) 113
▼ ZFS 루트 풀의 디스크 교체 방법(SPARC 또는 x86/VTOC) 115
▼ ZFS 루트 풀의 디스크 교체 방법(SPARC 또는 x86/EFI(GPT)) 117
▼ 다른 루트 풀에 BE를 만드는 방법(SPARC 또는 x86/VTOC) 119
▼ 다른 루트 풀에 BE를 만드는 방법(SPARC 또는 x86/EFI(GPT)) 120
ZFS 스왑 및 덤프 장치 관리 121
ZFS 스왑 및 덤프 장치의 크기 조정 122
ZFS 덤프 장치 문제 해결 123
ZFS 루트 파일 시스템에서 부트 124
미러링된 ZFS 루트 풀의 대체 디스크에서 부트 124
SPARC 기반 시스템의 ZFS 루트 파일 시스템에서 부트 126
x86 기반 시스템의 ZFS 루트 파일 시스템에서 부트 127
복구를 위해 ZFS 루트 환경에서 부트 128

5 Oracle Solaris ZFS 파일 시스템 관리 133
ZFS 파일 시스템 관리(개요) 133
ZFS 파일 시스템 만들기, 삭제 및 이름 바꾸기 134
ZFS 파일 시스템 만들기 134
ZFS 파일 시스템 삭제 135
ZFS 파일 시스템 이름 바꾸기 136
ZFS 등록 정보 소개 137
ZFS 읽기 전용 고유 등록 정보 149
설정 가능한 ZFS 고유 등록 정보 150
ZFS 사용자 등록 정보 156
ZFS 파일 시스템 정보 질의 157
기본 ZFS 정보 나열 157
복잡한 ZFS 질의 만들기 158
ZFS 등록 정보 관리 160
ZFS 등록 정보 설정 160
ZFS 등록 정보 상속 161
ZFS 등록 정보 질의 162
ZFS 파일 시스템 마운트 165
ZFS 마운트 지점 관리 165
ZFS 파일 시스템 마운트 167

임시 마운트 등록 정보 사용 168
ZFS 파일 시스템 마운트 해제 169
ZFS 파일 시스템 공유 및 공유 해제 170
레거시 ZFS 공유 구문 171
새 ZFS 공유 구문 171
ZFS 공유 마이그레이션/전환 문제 177
ZFS 파일 시스템 공유 문제 해결 178
ZFS 쿼터 및 예약 설정 179
ZFS 파일 시스템에 대한 쿼터 설정 180
ZFS 파일 시스템에 대한 예약 설정 183
ZFS 파일 시스템 암호화 185
암호화된 ZFS 파일 시스템의 키 변경 187
암호화된 ZFS 파일 시스템 마운트 189
암호화된 ZFS 파일 시스템 업그레이드 189
ZFS 압축, 중복 제거 및 암호화 등록 정보 간의 상호 작용 190
ZFS 파일 시스템 암호화의 예 190
ZFS 파일 시스템 마이그레이션 192
▼ 파일 시스템을 ZFS 파일 시스템으로 마이그레이션하는 방법 193
ZFS 파일 시스템 마이그레이션 문제 해결 195
ZFS 파일 시스템 업그레이드 195

6 Oracle Solaris ZFS 스냅샷 및 복제 작업 197
ZFS 스냅샷 개요 197
ZFS 스냅샷 만들기 및 삭제 198
ZFS 스냅샷 표시 및 액세스 201
ZFS 스냅샷 롤백 202
ZFS 스냅샷 차이 식별(`zfs diff`) 203
ZFS 복제본 개요 204
ZFS 복제본 만들기 205
ZFS 복제본 삭제 205
ZFS 복제본으로 ZFS 파일 시스템 대체 205
ZFS 데이터 전송 및 수신 206
다른 백업 제품으로 ZFS 데이터 저장 207
ZFS 스냅샷 스트림 식별 207
ZFS 스냅샷 전송 209

ZFS 스냅샷 수신 210
ZFS 스냅샷 스트림에 다른 등록 정보 값 적용 211
복잡한 ZFS 스냅샷 스트림 전송 및 수신 213
ZFS 데이터 원격 복제 216

7 ACL 및 속성을 사용하여 Oracle Solaris ZFS 파일 보호 217
Solaris ACL 모델 217
ACL 설정을 위한 구문 설명 218
ACL 상속 222
ACL 등록 정보 223
ZFS 파일에 ACL 설정 224
Verbose 형식으로 ZFS 파일에서 ACL 설정 및 표시 226
Verbose 형식으로 ZFS 파일에서 ACL 상속 설정 231
Compact 형식으로 ZFS 파일에서 ACL 설정 및 표시 236
ZFS 파일에 특수 속성 적용 242

8 Oracle Solaris ZFS 위임 관리 245
ZFS 위임 관리 개요 245
ZFS 위임 권한을 사용 안함으로 설정 246
ZFS 권한 위임 246
ZFS 권한 위임(`zfs allow`) 249
ZFS 위임 권한 제거(`zfs unallow`) 249
ZFS 권한 위임(예) 250
ZFS 위임 권한 표시(예) 254
ZFS 위임 권한 제거(예) 255

9 Oracle Solaris ZFS 고급 주제 257
ZFS 볼륨 257
ZFS 볼륨을 스왑 또는 덤프 장치로 사용 258
ZFS 볼륨을 iSCSI LUN으로 사용 259
영역이 설치된 Solaris 시스템에서 ZFS 사용 260
비전역 영역에 ZFS 파일 시스템 추가 261
비전역 영역에 데이터 세트 위임 261
비전역 영역에 ZFS 볼륨 추가 262

영역 내에서 ZFS 저장소 풀 사용 263
영역 내에서 ZFS 등록 정보 관리 263
zoned 등록 정보 이해 264
다른 시스템에 영역 복사 265
ZFS 대체 루트 풀 사용 265
ZFS 대체 루트 풀 만들기 266
대체 루트 풀 가져오기 266

10 Oracle Solaris ZFS 문제 해결 및 풀 복구 267
ZFS 공간 문제 해결 267
ZFS 파일 시스템 공간 보고 267
ZFS 저장소 풀 공간 보고 268
ZFS 오류 식별 269
ZFS 저장소 풀에서 장치 누락 269
ZFS 저장소 풀에서 장치 손상 270
손상된 ZFS 데이터 270
ZFS 파일 시스템 무결성 검사 270
파일 시스템 복구 270
파일 시스템 검증 271
ZFS 데이터 스크러빙 제어 271
ZFS 관련 문제 해결 273
ZFS 저장소 풀에 문제가 있는지 확인 274
zpool status 출력 결과 검토 274
ZFS 오류 메시지에 대한 시스템 보고 277
손상된 ZFS 구성 복구 278
누락된 장치 해결 278
물리적으로 장치 재연결 281
ZFS에 장치 가용성 알림 281
손상된 장치 교체 또는 복구 282
장치 오류 유형 확인 282
일시적인 오류 지우기 283
ZFS 저장소 풀의 장치 교체 284
손상된 데이터 복구 291
데이터 손상 유형 식별 292
손상된 파일 또는 디렉토리 복구 293

ZFS 저장소 풀 전반의 손상 복구 294
부트할 수 없는 시스템 복구 296

11 스냅샷 아카이브 및 루트 풀 복구 297
ZFS 복구 프로세스 개요 297
ZFS 풀 복구 요구 사항 298
복구에 사용할 ZFS 스냅샷 아카이브 만들기 298
▼ ZFS 스냅샷 아카이브를 만드는 방법 299
루트 풀 다시 만들기 및 루트 풀 스냅샷 복구 300
▼ 복구 시스템에서 루트 풀을 다시 만드는 방법 300

12 Oracle Solaris ZFS 권장 방법 305
저장소 풀 권장 방법 305
일반 시스템 방법 305
ZFS 저장소 풀 만들기 방식 306
성능에 대한 저장소 풀 방법 310
ZFS 저장소 풀 유지 관리 및 모니터링 방법 311
파일 시스템 권장 방법 312
파일 시스템 생성 방법 312
ZFS 파일 시스템 모니터 방법 313

A Oracle Solaris ZFS 버전 설명 315
ZFS 버전 개요 315
ZFS 풀 버전 315
ZFS 파일 시스템 버전 317

색인 319

머리말

Oracle Solaris 11.1 ZFS 관리 설명서는 Oracle Solaris ZFS 파일 시스템 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, “ZFS 루트 풀 구성 요소 관리”	미러링된 루트 풀 구성, ZFS 부트 환경 업그레이드, 스왑 및 덤프 장치 크기 조정 등 ZFS 루트 풀 구성 요소를 관리하는 방법에 대해 설명합니다.

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, “스냅샷 아카이브 및 루트 풀 복구”	루트 풀 스냅샷을 아카이브하고 루트 풀 복구를 수행하는 방법에 대해 설명합니다.
12 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 서적

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

- **Oracle Solaris 11.1에서 시스템 정보, 프로세스 및 성능 관리**
- **Oracle Solaris 11.1에서 사용자 계정 및 사용자 환경 관리**
- **Oracle Solaris 11.1 관리: 장치 및 파일 시스템**
- **Oracle Solaris 11.1 관리: 보안 서비스**

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 설명서에서 사용되는 활자체 규약에 대해 설명합니다.

표 P–1 활자체 규약

활자체	설명	예
`AaBbCc123`	명령 및 파일, 디렉토리 이름; 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	위치 표시자: 실제 이름이나 값으로 바뀝니다.	`rm` *filename* 명령을 사용하여 파일을 제거합니다.
AaBbCc123	설명서 제목, 새 용어, 강조 표시할 용어입니다.	**사용자 설명서**의 6장을 읽으십시오. **캐시**는 로컬로 저장된 복사본입니다. 파일을 저장하면 **안 됩니다**. **주:** 일부 강조된 항목은 온라인에서 굵은체로 나타납니다.

명령 예의 셸 프롬프트

다음 표에는 Oracle Solaris OS에 포함된 셸의 UNIX 시스템 프롬프트 및 수퍼 유저 프롬프트가 나와 있습니다. 명령 예에서 셸 프롬프트는 명령을 일반 사용자가 실행해야 하는지 또는 권한 있는 사용자가 실행해야 하는지 나타냅니다.

표 P–2 셸 프롬프트

셸	프롬프트
Bash 셸, Korn 셸 및 Bourne 셸	`$`
수퍼 유저용 Bash 셸, Korn 셸 및 Bourne 셸	`#`
C 셸	`machine_name%`
수퍼 유저용 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.

- 15 페이지 “ZFS의 새로운 기능”
- 26 페이지 “Oracle Solaris ZFS란?”
- 29 페이지 “ZFS 용어”
- 31 페이지 “ZFS 구성 요소 명명 요구 사항”
- 31 페이지 “Oracle Solaris ZFS와 전통적인 파일 시스템의 차이”

ZFS의 새로운 기능

이 절에서는 ZFS 파일 시스템의 새 기능을 요약해서 보여 줍니다.

- 16 페이지 “향상된 ZFS 풀 장치 메시지”
- 16 페이지 “ZFS 파일 공유 향상 기능”
- 17 페이지 “공유 `var` 파일 시스템”
- 17 페이지 “EFI(GPT) 레이블 디스크에 대한 부트 지원”
- 18 페이지 “ZFS 명령 사용법 향상된 기능”
- 18 페이지 “ZFS 스냅샷 향상된 기능”
- 19 페이지 “ZFS 매뉴얼 페이지 변경(`zfs.1m`)”
- 19 페이지 “향상된 `aclmode` 등록 정보”
- 20 페이지 “물리적 위치로 풀 장치 식별”
- 21 페이지 “ZFS 그림자 마이그레이션”
- 21 페이지 “ZFS 파일 시스템 암호화”
- 21 페이지 “ZFS 전송 스트림의 향상된 기능”
- 22 페이지 “ZFS 스냅샷 차이(`zfs diff`)”
- 22 페이지 “ZFS 저장소 풀 복구 및 성능의 향상된 기능”
- 23 페이지 “ZFS 동기식 동작 조정”

- 23 페이지 “향상된 ZFS 풀 메시지”
- 24 페이지 “ZFS ACL 상호 운용성의 향상된 기능”
- 25 페이지 “미러링된 ZFS 저장소 풀 분할(zpool split)”
- 25 페이지 “ZFS iSCSI 변경 사항”
- 25 페이지 “새 ZFS 시스템 프로세스”
- 26 페이지 “ZFS 중복 제거 등록 정보”

향상된 ZFS 풀 장치 메시지

Oracle Solaris 11.1: zpool status 명령이 장치 오류에 대한 보다 자세한 정보를 제공하도록 향상되었습니다. 이 예에서 zpool status 출력 결과는 영구적인 오류로 인해 UNAVAIL 상태이고 교체해야 하는 풀 장치(c0t5000C500335F907Fd0)를 식별합니다.

```
# zpool status -v pond
  pool: pond
 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: Determine if the device needs to be replaced, and clear the errors
        using 'zpool clear' or 'fmadm repaired', or replace the device
        with 'zpool replace'.
  scan: scrub repaired 0 in 0h0m with 0 errors on Wed Jun 20 15:38:08 2012
config:

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

device details:

        c0t5000C500335F907Fd0    UNAVAIL          cannot open
        status: ZFS detected errors on this device.
                The device was missing.
           see: http://support.oracle.com/msg/ZFS-8000-LR for recovery


errors: No known data errors
```

ZFS 파일 공유 향상 기능

Oracle Solaris 11.1: 다음과 같은 주요 향상된 기능과 함께 ZFS 파일 시스템 공유가 개선되었습니다.

- 공유 구문이 간소화되었습니다. 새로운 share.nfs 또는 share.smb 등록 정보를 설정하여 파일 시스템을 공유할 수 있습니다.

- 종속된 파일 시스템에 대한 공유 등록 정보의 상속이 개선되었습니다.

파일 공유 개선 사항은 풀 버전 34와 연관되어 있습니다.

자세한 내용은 170 페이지 “ZFS 파일 시스템 공유 및 공유 해제”를 참조하십시오.

공유 var 파일 시스템

Oracle Solaris 11.1: Oracle Solaris 11.1을 설치하면 /var/share에 마운트되는 rpool/VARSHARE 파일 시스템이 자동으로 생성됩니다. 이 파일 시스템의 목적은 /var 디렉토리에서 모든 BE에 대해 필요한 공간을 줄일 수 있도록 부트 환경 간에 파일 시스템을 공유하기 위한 것입니다. 예를 들면 다음과 같습니다.

```
# ls /var/share
audit  cores  crash  mail
```

호환성 목적을 위해 위에 나열된 /var에서 /var/share 구성 요소로의 심볼릭 링크가 자동으로 생성됩니다. /var 구성 요소가 루트 파일 시스템을 채우지 않도록 확인하는 것을 제외하고는 일반적으로 이 파일 시스템에 대한 관리가 필요하지 않습니다.

Oracle Solaris 11 시스템을 Oracle Solaris 11.1 릴리스로 업데이트한 경우 원래 /var 디렉토리에서 /var/share 디렉토리로 데이터를 마이그레이션하는 데 시간이 오래 걸릴 수 있습니다.

EFI(GPT) 레이블 디스크에 대한 부트 지원

Oracle Solaris 11.1: 이 릴리스는 대부분의 경우 x86 기반 시스템에서 ZFS 루트 풀 디스크에 EFI(GPT) 디스크 레이블을 설치합니다. 예를 들면 다음과 같습니다.

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

    NAME      STATE     READ WRITE CKSUM
    rpool     ONLINE       0     0     0
      c2t0d0  ONLINE       0     0     0

errors: No known data errors
```

- GPT 지원 펌웨어가 포함된 SPARC 기반 시스템 또는 x86 기반 시스템에 Oracle Solaris 11.1을 설치하면 전체 디스크를 사용하는 루트 풀 디스크에 GPT 디스크 레이블이 적용됩니다. GPT 레이블 부트 디스크를 지원하는 SPARC 기반 시스템에서 GPT 지원 펌웨어를 적용하는 방법에 대한 자세한 내용은 Oracle Solaris 11.1 릴리스 노트를 참조하십시오. 그렇지 않으면 SPARC 시스템에 Oracle Solaris 11.1을 설치할 때 단일 슬라이스 0을 사용해서 루트 풀 디스크에 SMI(VTOC) 레이블이 적용됩니다.

- 대부분의 경우 x86 기반 시스템은 EFI(GPT) 레이블 디스크로 설치됩니다.
- 시스템을 설치한 후 루트 풀을 다시 만들어야 할 경우 `zpool create -B` 명령으로 해당 작업을 수행하기 위해 EFI(GPT) 디스크 레이블을 지원할 수 있도록 `zpool` 명령이 향상되었습니다. 이 새로운 명령 옵션을 사용하면 부트에 필요한 분할 영역 및 부트 정보를 만들 수 있습니다. 설치 후 루트 풀 만들기에 대한 자세한 내용은 119 페이지 "다른 루트 풀에 BE를 만드는 방법(SPARC 또는 x86/VTOC)"을 참조하십시오.

  루트 풀에서 EFI(GPT) 레이블로 디스크를 교체해야 할 경우 `zpool replace` 작업 후 이 명령을 실행해야 합니다.

  ```
  # bootadm install-bootloader
  ```
- Oracle Solaris 설치는 더 이상 x86 기반 시스템의 디스크에 대한 처음 2TiB로 제한되지 않습니다.

ZFS 명령 사용법 향상된 기능

Oracle Solaris 11: `zfs` 및 `zpool` 명령에는 `zfs` 및 `zpool` 하위 명령에 대한 추가 정보를 제공하고 지원되는 해당 옵션을 보기 위해 사용할 수 있는 `help` 하위 명령이 포함됩니다. 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 11: 이 릴리스에는 다음과 같은 ZFS 스냅샷 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 -e를 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.

ZFS 매뉴얼 페이지 변경(zfs.1m)

Oracle Solaris 11: 핵심 ZFS 파일 시스템 기능이 zfs.1m 페이지에 남아 있도록 zfs.1m 매뉴얼 페이지가 수정되었지만, 위임된 관리, 암호화, 공유 구문 및 예제는 다음 페이지에서 설명합니다.

- zfs_allow(1M)
- zfs_encrypt(1M)
- zfs_share(1M)

향상된 aclmode 등록 정보

Oracle Solaris 11: 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 7–14를 참조하십시오.

물리적 위치로 풀 장치 식별

Oracle Solaris 11: 이 Solaris 릴리스에서는 zpool status -l 명령을 사용하여 /dev/chassis 디렉토리에서 사용 가능한 풀 장치의 물리적 디스크 위치 정보를 표시합니다. 이 디렉토리에는 시스템의 장치에 대한 섀시, 저장소 및 점유자 값이 들어 있습니다.

또한 fmadm add-alias 명령을 사용하여 환경에 있는 디스크의 물리적 위치를 식별하는 데 도움이 되는 디스크 별칭을 포함시킬 수 있습니다. 예를 들면 다음과 같습니다.

```
# fmadm add-alias SUN-Storage-J4400.0912QAJ001 SUN-Storage-J4400.rack22
```

예를 들면 다음과 같습니다.

```
% zpool status -l export
  pool: export
 state: ONLINE
  scan: resilvered 492G in 8h22m with 0 errors on Wed Aug  1 17:22:11 2012
config:

 NAME                                                    STATE  READ WRITE CKSUM
   export                                                ONLINE    0     0     0
     mirror-0                                            ONLINE    0     0     0
       /dev/chassis/SUN-Storage-J4400.rack22/SCSI_Device__2/disk  ONLINE    0     0     0
       /dev/chassis/SUN-Storage-J4400.rack22/SCSI_Device__3/disk  ONLINE    0     0     0
     mirror-1                                            ONLINE    0     0     0
       /dev/chassis/SUN-Storage-J4400.rack22/SCSI_Device__4/disk  ONLINE    0     0     0
       /dev/chassis/SUN-Storage-J4400.rack22/SCSI_Device__5/disk  ONLINE    0     0     0
     mirror-2                                            ONLINE    0     0     0
       /dev/chassis/SUN-Storage-J4400.rack22/SCSI_Device__6/disk  ONLINE    0     0     0
       /dev/chassis/SUN-Storage-J4400.rack22/SCSI_Device__7/disk  ONLINE    0     0     0
     mirror-3                                            ONLINE    0     0     0
       /dev/chassis/SUN-Storage-J4400.rack22/SCSI_Device__8/disk  ONLINE    0     0     0
       /dev/chassis/SUN-Storage-J4400.rack22/SCSI_Device__9/disk  ONLINE    0     0     0
     mirror-4                                            ONLINE    0     0     0
       /dev/chassis/SUN-Storage-J4400.rack22/SCSI_Device__10/disk ONLINE    0     0     0
       /dev/chassis/SUN-Storage-J4400.rack22/SCSI_Device__11/disk ONLINE    0     0     0
     spares
       /dev/chassis/SUN-Storage-J4400.rack22/SCSI_Device__0/disk  AVAIL
       /dev/chassis/SUN-Storage-J4400.rack22/SCSI_Device__1/disk  AVAIL

errors: No known data errors
```

또한 풀의 장치에 대한 물리적 위치 정보를 제공하기 위해 zpool iostat 명령이 업데이트되었습니다.

또한 diskinfo, format 및 prtconf 명령도 물리적 디스크 위치 정보를 제공합니다. 자세한 내용은 diskinfo(1M)을 참조하십시오.

ZFS 그림자 마이그레이션

Oracle Solaris 11: 이 릴리스에서는 마이그레이션 프로세스 도중 새 파일 시스템의 액세스와 수정을 허용하는 동시에 이전 파일 시스템의 데이터를 새 파일 시스템으로 마이그레이션할 수 있습니다.

새 ZFS 파일 시스템에 shadow 등록 정보를 설정하면 이전 데이터의 마이그레이션이 트리거됩니다. 다음 값 중 하나를 사용하여 로컬 시스템 또는 원격 시스템의 데이터를 마이그레이션하도록 shadow 등록 정보를 설정할 수 있습니다.

`file:///`*path*
`nfs://`*host:path*

자세한 내용은 192 페이지 “ZFS 파일 시스템 마이그레이션”을 참조하십시오.

ZFS 파일 시스템 암호화

Oracle Solaris 11: 이 릴리스에서는 ZFS 파일 시스템을 암호화할 수 있습니다.

예를 들어, `tank/cindy` 파일 시스템은 encryption 등록 정보를 사용으로 설정하여 생성됩니다. 기본 암호화 정책은 문장암호를 입력하는 프롬프트를 표시하는 것입니다. 문장암호는 최소 8자여야 합니다.

```
# zfs create -o encryption=on tank/cindy
Enter passphrase for 'tank/cindy': xxx
Enter again: xxx
```

암호화 정책은 ZFS 파일 시스템을 만들 때 설정됩니다. 파일 시스템의 암호화 정책은 종속 파일 시스템에 상속되며 제거할 수 없습니다.

자세한 내용은 185 페이지 “ZFS 파일 시스템 암호화”를 참조하십시오.

ZFS 전송 스트림의 향상된 기능

Oracle Solaris 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 211 페이지 “ZFS 스냅샷 스트림에 다른 등록 정보 값 적용”을 참조하십시오.

ZFS 스냅샷 차이(zfs diff)

Oracle Solaris 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 `M`은 디렉토리가 수정되었음을 나타냅니다. `+`는 `fileB`가 나중 스냅샷에 존재함을 나타냅니다.

자세한 내용은 203 페이지 “ZFS 스냅샷 차이 식별(zfs diff)”을 참조하십시오.

ZFS 저장소 풀 복구 및 성능의 향상된 기능

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

- `zpool import -m` 명령을 사용하여 누락된 로그로 풀을 가져올 수 있습니다. 자세한 내용은 100 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 101 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

Oracle Solaris 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 11: 이 릴리스에서는 -T 옵션을 통해 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 u를 사용하여 내부 형식으로 정보를 표시할 수 있습니다.

```
# zpool status -T d tank 3 2
Thu Jun 14 14:08:21 MDT 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.
        Run 'zpool status -v' to see device specific details.
  scan: resilver in progress since Thu Jun 14 14:08:05 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 11: 이 릴리스에서는 다음과 같은 ACL 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 `restricted`로 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 11: 이 릴리스에서는 `zpool split` 명령을 사용하여 미러링된 저장소 풀을 분할할 수 있습니다. 이 경우 미러링된 원래 풀에서 디스크가 분리되어 다른 동일한 풀을 만듭니다.

자세한 내용은 67 페이지 “미러링된 ZFS 저장소 풀을 분할하여 새로운 풀 만들기”를 참조하십시오.

ZFS iSCSI 변경 사항

Oracle Solaris 11: 이 릴리스에서는 iSCSI 대상 데몬이 COMSTAR(Common Multiprotocol SCSI Target) 대상 데몬을 사용하여 교체됩니다. 또한 이 변경 사항은 ZFS 볼륨을 iSCSI LUN으로 공유하는 데 사용된 `shareiscsi` 등록 정보를 더 이상 사용할 수 없음을 의미합니다. ZFS 볼륨을 iSCSI LUN으로 구성하고 공유하려면 `stmfadm` 명령을 사용합니다.

자세한 내용은 259 페이지 “ZFS 볼륨을 iSCSI LUN으로 사용”을 참조하십시오.

새 ZFS 시스템 프로세스

Oracle Solaris 11: 이 릴리스에서는 각 ZFS 저장소 풀에 관련 프로세스 `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)를 참조하십시오.

ZFS 중복 제거 등록 정보

Oracle Solaris 11: 이 릴리스에서는 중복 제거(dedup) 등록 정보를 사용하여 ZFS 파일 시스템에서 중복 데이터를 제거할 수 있습니다. 파일 시스템의 dedup 등록 정보가 사용으로 설정된 경우 중복 데이터 블록이 동기적으로 제거됩니다. 그 결과, 고유한 데이터만 저장되고 공통 구성 요소는 파일 간에 공유됩니다.

다음과 같이 이 등록 정보를 사용으로 설정할 수 있습니다.

```
# zfs set dedup=on tank/home
```

중복 제거는 파일 시스템 등록 정보로 설정되지만 범위가 풀 전체입니다. 예를 들어, 다음과 같이 중복 제거 비율을 식별할 수 있습니다.

```
# zpool list tank
NAME  SIZE  ALLOC  FREE  CAP  DEDUP  HEALTH  ALTROOT
tank  556G  4.19G  552G   0%  1.00x  ONLINE  -
```

zpool list 출력이 중복 제거 등록 정보를 지원하도록 업데이트되었습니다.

중복 제거 등록 정보 설정에 대한 자세한 내용은 154 페이지 “dedup 등록 정보”를 참조하십시오.

다음 고려 사항을 검토할 때까지 운용 시스템에 있는 파일 시스템에서 dedup 등록 정보를 사용으로 설정하지 마십시오.

- 중복 제거 공간 절약이 데이터에 유용한지 확인합니다.
- 시스템의 물리적 메모리가 중복 제거를 지원하기에 충분한지 확인합니다.
- 잠재적 시스템 성능 영향을 확인합니다.

이러한 고려 사항에 대한 자세한 내용은 154 페이지 “dedup 등록 정보”를 참조하십시오.

Oracle Solaris ZFS란?

Oracle Solaris ZFS 파일 시스템은 현재 사용 가능한 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.

부트 환경
: 부트 환경은 ZFS 루트 파일 시스템 및 선택적으로 그 아래에 마운트된 다른 파일 시스템으로 구성되는 부트 가능 Oracle Solaris 환경입니다. 정확히 한 번에 하나의 부트 환경만 활성 상태일 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.

clone
: 초기 컨텐츠가 스냅샷의 컨텐츠와 동일한 파일 시스템입니다.

  복제본에 대한 자세한 내용은 204 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.

RAID-Z
: 여러 디스크에 데이터 및 패리티를 저장하는 가상 장치입니다. RAID-Z에 대한 자세한 내용은 48 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 289 페이지 “리실버링 상태 보기”를 참조하십시오.

snapshot
: 지정된 시점의 파일 시스템 또는 볼륨의 읽기 전용 복사본입니다.

  스냅샷에 대한 자세한 내용은 197 페이지 “ZFS 스냅샷 개요”를 참조하십시오.

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

  가상 장치에 대한 자세한 내용은 56 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 257 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

- 31 페이지 “ZFS 파일 시스템 세분성”
- 32 페이지 “ZFS 디스크 공간 계산”
- 33 페이지 “ZFS 파일 시스템 마운트”
- 34 페이지 “기존 볼륨 관리”
- 34 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 180 페이지 "ZFS 파일 시스템에 대한 쿼터 설정"을 참조하십시오. 예약을 사용하여 파일 시스템이 지정된 디스크 공간을 사용하도록 보장할 수 있습니다. 예약에 대한 자세한 내용은 183 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 files는 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_SYNC를 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 du는 디스크에 저장된 파일의 실제 크기를 보고합니다. 이 크기에는 메타 데이터와 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 df는 파일 시스템 크기만이 아니라 풀 크기를 보고한다는 것에 주의합니다. 또한 df는 종속 파일 시스템 또는 스냅샷 존재 여부를 인식하지 못합니다. 압축, 할당량 등의 ZFS 등록 정보가 파일 시스템에 설정된 경우 df에서 보고된 공간 사용을 조정하기 어려울 수 있습니다.

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

- recordsize보다 큰 파일의 경우 파일의 마지막 블록은 일반적으로 약 1/2 정도 차 있습니다. 기본 recordsize가 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. recordsize가 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 recordsize를 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 165 페이지 "ZFS 파일 시스템 마운트"를 참조하십시오.

기존 볼륨 관리

27 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.

- 35 페이지 "ZFS 권한 프로파일"
- 36 페이지 "ZFS 하드웨어 및 소프트웨어 요구 사항과 권장 사항"
- 36 페이지 "기본 ZFS 파일 시스템 만들기"
- 37 페이지 "기본 ZFS 저장소 풀 만들기"
- 38 페이지 "ZFS 파일 시스템 계층 만들기"

ZFS 권한 프로파일

수퍼 유저(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 **Oracle Solaris 11.1 관리: 보안 서비스**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, 31 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 50 페이지 "ZFS 저장소 풀 만들기"를 참조하십시오. 사용 중인 장치 확인 방법에 대한 자세한 내용은 57 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 83 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 137 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 share.nfs=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 share.nfs=on -o compression=on tank/home
```

등록 정보 및 등록 정보 상속에 대한 자세한 내용은 137 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 134 페이지 "ZFS 파일 시스템 만들기"를 참조하십시오.

파일 시스템 마운트 및 공유에 대한 자세한 내용은 165 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 157 페이지 “ZFS 파일 시스템 정보 질의”를 참조하십시오.

디스크 공간 사용 및 계산 방법에 대한 자세한 내용은 32 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.

- 43 페이지 “ZFS 저장소 풀의 구성 요소”
- 47 페이지 “ZFS 저장소 풀의 복제 기능”
- 50 페이지 “ZFS 저장소 풀 만들기 및 삭제”
- 61 페이지 “ZFS 저장소 풀의 장치 관리”
- 80 페이지 “ZFS 저장소 풀 등록 정보 관리”
- 83 페이지 “ZFS 저장소 풀 상태 질의”
- 95 페이지 “ZFS 저장소 풀 마이그레이션”
- 104 페이지 “ZFS 저장소 풀 업그레이드”

ZFS 저장소 풀의 구성 요소

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

- 43 페이지 “ZFS 저장소 풀의 디스크 사용”
- 45 페이지 “ZFS 저장소 풀에서 슬라이스 사용”
- 46 페이지 “ZFS 저장소 풀에서 파일 사용”

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 분할 영역 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
```

Oracle Solaris 11.1이 설치된 경우 EFI(GPT) 레이블은 대부분의 경우 다음과 비슷한 x86 기반 시스템의 루트 풀 디스크에 적용됩니다.

```
Current partition table (original):
Total disk sectors available: 27246525 + 16384 (reserved sectors)

Part      Tag    Flag     First Sector        Size        Last Sector
  0  BIOS_boot    wm               256     256.00MB          524543
  1        usr    wm            524544      12.74GB         27246558
  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          27246559        8.00MB         27262942
```

위 출력 결과에서 분할 영역 0(`BIOS boot`)에 필요한 GPT 부트 정보가 포함됩니다. 분할 영역 8과 비슷하게 관리가 필요하지 않으며 수정하지 않아야 합니다. 루트 파일 시스템은 분할 영역 1에 포함됩니다.

Oracle Solaris 11.1에 설치된 업데이트된 펌웨어가 포함된 SPARC 시스템에서는 EFI(GPT) 디스크 레이블이 적용됩니다. 예를 들면 다음과 같습니다.

```
Current partition table (original):
Total disk sectors available: 143358320 + 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 **Oracle Solaris 11.1 관리: 장치 및 파일 시스템**의 "ZFS 파일 시스템에 사용할 디스크 설정(작업 맵)"을 참조하십시오.
- 단일 디스크로 저장소 풀을 만들 경우 ZFS는 EFI 레이블을 적용합니다. EFI 레이블에 대한 자세한 내용은 **Oracle Solaris 11.1 관리: 장치 및 파일 시스템**의 "EFI(GPT) 디스크 레이블"을 참조하십시오.
- Oracle Solaris 11.1 설치 프로그램은 대부분의 경우 GPT 지원 펌웨어가 포함된 SPARC 기반 시스템 및 x86 기반 시스템에서 루트 풀 디스크에 대해 EFI(GPT) 레이블을 적용합니다. 자세한 내용은 108 페이지 "ZFS 루트 풀 요구 사항"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 레거시 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 분할 영역은 /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.
  - 소프트웨어 기반 볼륨 관리자가 제공하는 볼륨을 사용하여 ZFS의 논리적 장치를 작성할 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 12 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.

- 48 페이지 “미러링된 저장소 풀 구성”
- 48 페이지 “RAID-Z 저장소 풀 구성”
- 49 페이지 “중복 구성에서 데이터 자가 치료”
- 49 페이지 “저장소 풀의 동적 스트라이프”
- 49 페이지 “ZFS 하이브리드 저장소 풀”

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 51 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 52 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 12 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.

- 50 페이지 “ZFS 저장소 풀 만들기”
- 56 페이지 “저장소 풀 가상 장치 정보 표시”
- 57 페이지 “ZFS 저장소 풀 만들기 오류 처리”
- 60 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 31 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 12 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 61 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 66 페이지 “저장소 풀에서 장치 연결 및 분리”를 참조하십시오.
- 기존 미러된 구성에서 디스크 교체(교체 디스크가 교체될 디스크의 크기보다 크거나 같은 경우에만). 자세한 내용은 73 페이지 “저장소 풀의 장치 교체”를 참조하십시오.
- 미러된 구성에서 디스크 분리(나머지 장치가 구성에 대한 충분한 중복성을 제공하는 경우에만). 자세한 내용은 66 페이지 “저장소 풀에서 장치 연결 및 분리”를 참조하십시오.
- 디스크 중 하나를 분리하여 미러된 구성을 분할하고 새로운 동일 풀 만들기. 자세한 내용은 67 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.

- Oracle Solaris 11.1에서 루트 풀에 사용되는 디스크는 x86 기반 시스템 또는 GPT 지원 펌웨어가 포함된 지원되는 SPARC 시스템에서 EFI(GPT) 레이블로 설치됩니다. 또는 GPT 지원 펌웨어가 포함되지 않은 SPARC 기반 시스템의 경우 SMI(VTOC) 레이블이 적용됩니다. Oracle Solaris 11.1 설치 프로그램은 가능한 경우 EFI(GPT) 레이블을 적용합니다. 설치 후 ZFS 루트 풀을 다시 만들어야 할 경우에는 다음 명령을 사용하여 EFI(GPT) 디스크 레이블 및 올바른 부트 정보를 적용할 수 있습니다.

  ```
  # zpool create -B rpool2 c1t0d0
  ```

- 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, "ZFS 루트 풀 구성 요소 관리"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. 예:

```
# 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 61 페이지 "저장소 풀에 장치 추가"를 참조하십시오.
- 기존 RAID-Z 구성에서 디스크 교체(교체 디스크가 교체될 디스크의 크기보다 크거나 같은 경우에만). 자세한 내용은 73 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 48 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. 예:

```
# 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.

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. 예:

```
# 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 165 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 102 페이지 "삭제된 ZFS 저장소 풀 복구"를 참조하십시오.

zpool destroy 명령으로 풀을 삭제할 경우 102 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.

암호화된 ZFS 파일 시스템을 만들어 파일 시스템 데이터를 기밀로 유지할 수도 있습니다. 암호화된 파일 시스템의 풀이 삭제되면 삭제된 풀이 복구된 경우에도 암호화 키 없이 데이터에 액세스할 수 없습니다. 자세한 내용은 185 페이지 "ZFS 파일 시스템 암호화"를 참조하십시오.

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 90 페이지 "ZFS 저장소 풀의 건전성 상태 확인"을 참조하십시오.

풀 가져오기에 대한 자세한 내용은 98 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.

- 61 페이지 “저장소 풀에 장치 추가”
- 66 페이지 “저장소 풀에서 장치 연결 및 분리”
- 67 페이지 “미러링된 ZFS 저장소 풀을 분할하여 새로운 풀 만들기”
- 70 페이지 “저장소 풀에서 장치 온라인 및 오프라인 전환”
- 72 페이지 “저장소 풀 장치 오류 지우기”
- 73 페이지 “저장소 풀의 장치 교체”
- 75 페이지 “저장소 풀에서 핫 스패어 지정”

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. 예:

```
# 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 57 페이지 “사용 중인 장치 감지”를 참조하십시오.

예 3–1 미러된 ZFS 구성에 디스크 추가

다음 예에서는 미러링된 기존 ZFS 구성에 다른 미러가 추가됩니다.

```
# 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
```

예 3–2 RAID-Z 구성에 디스크 추가 *(계속)*

```
            c1t2d0  ONLINE       0     0     0
            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
```

예 3–3 미러된 로그 장치 추가 및 제거 *(계속)*

```
            c0t5d0  ONLINE       0     0     0
          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. 예:

```
# 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 스패어, 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 55 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 미러링된 루트 풀을 만드는 경우 112 페이지 "미러링된 루트 풀을 구성하는 방법(SPARC 또는 x86/VTOC)"을 참조하십시오.

ZFS 루트 풀에서 디스크를 교체할 경우 115 페이지 "ZFS 루트 풀의 디스크 교체 방법(SPARC 또는 x86/VTOC)"을 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. 예:

```
# 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 305 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
   bringing device 'c1t0d0' offline
  ```

  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 73 페이지 "저장소 풀의 장치 교체"를 참조하십시오.

오프라인 장치는 풀 상태를 질의할 때 OFFLINE 상태에 있습니다. 풀 상태 질의에 대한 자세한 내용은 83 페이지 "ZFS 저장소 풀 상태 질의"를 참조하십시오.

장치 건전성에 대한 자세한 내용은 90 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.

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

```
SUNW-MSG-ID: ZFS-8000-LR, TYPE: Fault, VER: 1, SEVERITY: Major
EVENT-TIME: Wed Jun 20 11:35:26 MDT 2012
PLATFORM: ORCL,SPARC-T3-4, CSN: 1120BDRCCD, HOSTNAME: tardis
SOURCE: zfs-diagnosis, REV: 1.0
EVENT-ID: fb6699c8-6bfb-eefa-88bb-81479182e3b7
DESC: ZFS device 'id1,sd@n5000c500335dc60f/a' in pool 'pond' failed to open.
AUTO-RESPONSE: An attempt will be made to activate a hot spare if available.
IMPACT: Fault tolerance of the pool may be compromised.
REC-ACTION: Use 'fmadm faulty' to provide a more detailed view of this event.
Run 'zpool status -lx' for more information. Please refer to the associated
reference document at http://support.oracle.com/msg/ZFS-8000-LR for the latest
service procedures and policies regarding this diagnosis.
```

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

zpool online -e 명령을 사용하여 LUN을 확장할 수 있습니다. 기본적으로 풀에 추가된 LUN은 autoexpand 풀 등록 정보가 사용으로 설정되지 않은 경우 전체 크기로 확장되지 않습니다. LUN이 이미 온라인 상태이거나 LUN이 현재 오프라인 상태인 경우에도 zpool online -e 명령을 사용하여 LUN을 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 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 교체할 경우 115 페이지 "ZFS 루트 풀의 디스크 교체 방법(SPARC 또는 x86/VTOC)"을 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. `autoexpand` 풀 등록 정보 값은 디스크가 풀에 추가될 때 교체 LUN을 전체 크기로 확장할지 여부를 결정합니다. 기본적으로 `autoexpand` 등록 정보는 사용 안함으로 설정됩니다. 더 큰 LUN이 풀에 추가되기 전이나 후에 이 등록 정보를 사용으로 설정하여 LUN 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 77 페이지 “저장소 풀에서 핫 스패어 활성화 및 비활성화”를 참조하십시오.

장치 교체에 대한 자세한 내용은 278 페이지 “누락된 장치 해결” 및 282 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 스페어를 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 스페어는 `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`
- 스페어는 시스템 전체에서 공유될 수 없습니다.
- 동일 시스템에서 두 데이터 풀 사이에 스페어를 공유할 경우 두 풀 사이에 스페어 사용을 조정해야 합니다. 예를 들어, 풀 A에는 사용 중인 스페어가 있고 풀 A를 내보냈습니다. 풀 A를 내보내는 동안 풀 B는 이 사실을 모르고 스페어를 사용할 수 있습니다. 풀 A를 가져오면 두 풀에서 모두 동일 디스크를 사용하므로 데이터 손상이 발생할 수 있습니다.
- 루트 풀과 데이터 풀 사이에 스페어를 공유하지 마십시오.

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: Determine if the device needs to be replaced, and clear the errors
        using 'zpool clear' or 'fmadm repaired', or replace the device
        with 'zpool replace'.
        Run 'zpool status -v' to see device specific details.
  scan: resilvered 3.15G in 0h0m with 0 errors on Thu Jun 21 16:46:19 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 스패어를 비활성화할 수 있습니다.

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

예 3–8 실패한 디스크가 교체된 후 핫 스패어 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
```

예 3–8 실패한 디스크가 교체된 후 핫 스페어 분리 *(계속)*

```
        spares
          c0t5000C500335E106Bd0    AVAIL
```

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

```
# 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: Determine if the device needs to be replaced, and clear the errors
        using 'zpool clear' or 'fmadm repaired', or replace the device
        with 'zpool replace'.
        Run 'zpool status -v' to see device specific details.
  scan: scrub in progress since Thu Jun 21 17:01:49 2012
    1.07G scanned out of 6.29G at 220M/s, 0h0m to go
    0 repaired, 17.05% 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 3.15G in 0h0m with 0 errors on Thu Jun 21 17:02:35 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
```

예 3–9 실패한 디스크 분리 및 핫 스페어 사용 *(계속)*

```
          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
# zpool status zeepool
  pool: zeepool
 state: ONLINE
  scan: resilvered 3.15G in 0h0m with 0 errors on Thu Jun 21 17:02:35 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 zeepool
NAME     PROPERTY       VALUE                SOURCE
zeepool  allocated      6.29G                -
zeepool  altroot        -                    default
zeepool  autoexpand     off                  default
zeepool  autoreplace    off                  default
zeepool  bootfs         -                    default
zeepool  cachefile      -                    default
zeepool  capacity       1%                   -
zeepool  dedupditto     0                    default
zeepool  dedupratio     1.00x                -
zeepool  delegation     on                   default
zeepool  failmode       wait                 default
zeepool  free           550G                 -
zeepool  guid           7543986419840620672  -
```

```
zeepool  health         ONLINE                 -
zeepool  listshares     off                    default
zeepool  listsnapshots  off                    default
zeepool  readonly       off                    -
zeepool  size           556G                   -
zeepool  version        34                     default
```

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 12 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.

표 3–1 ZFS 풀 등록 정보 설명 (계속)

등록 정보 이름	유형	기본값	설명
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.
dedupditto	문자열	해당 없음	임계값을 설정하고, 중복 제거된 블록의 참조 수가 임계값을 초과할 경우 블록의 다른 복제 복사본이 자동으로 저장됩니다.
dedupratio	문자열	해당 없음	풀에 대해 얻은 읽기 전용 중복 제거 비율입니다(배수로 표현됨).
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.

표 3–1 ZFS 풀 등록 정보 설명 (계속)

등록 정보 이름	유형	기본값	설명
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	Off	이 풀과 연관된 스냅샷 정보가 zfs list 명령으로 표시되는지 여부를 제어합니다. 이 등록 정보가 사용 안함으로 설정된 경우 zfs list - t snapshot 명령을 사용하여 스냅샷 정보를 표시할 수 있습니다.
readonly	부울	Off	풀을 수정할 수 있는지 여부를 식별합니다. 이 등록 정보는 풀을 읽기 전용 모드로 가져온 경우에만 사용으로 설정됩니다. 사용으로 설정된 경우 풀을 읽기/쓰기 모드로 다시 가져올 때까지 계획 로그에만 있는 동기식 데이터에 액세스할 수 없습니다.
size	숫자	해당 없음	저장소 풀의 총 크기를 식별하는 읽기 전용 등록 정보입니다.
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.

- 83 페이지 “ZFS 저장소 풀에 대한 정보 표시”
- 88 페이지 “ZFS 저장소 풀에 대한 I/O 통계 보기 ”
- 90 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 32 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 90 페이지 “ZFS 저장소 풀의 건전성 상태 확인”을 참조하십시오.

ALTROOT — 풀의 대체 루트입니다(존재하는 경우).

대체 루트 풀에 대한 자세한 내용은 265 페이지 “ZFS 대체 루트 풀 사용”을 참조하십시오.

풀 이름을 지정하면 특정 풀에 대한 통계를 수집할 수도 있습니다. 예:

```
# 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  -
```

물리적 위치로 풀 장치 표시

zpool status -l 옵션을 사용하여 풀 장치의 물리적 위치 정보를 표시할 수 있습니다. 디스크를 물리적으로 제거하거나 교체해야 하는 경우 물리적 위치 정보를 검토하는 것이 유용합니다.

또한 fmadm add-alias 명령을 사용하여 환경에 있는 디스크의 물리적 위치를 식별하는 데 도움이 되는 디스크 별칭을 포함시킬 수 있습니다. 예를 들면 다음과 같습니다.

```
# fmadm add-alias SUN-Storage-J4400.1002QCQ015 Lab10Rack5...

# zpool status -l tank
  pool: tank
 state: ONLINE
  scan: scrub repaired 0 in 0h0m with 0 errors on Fri Aug  3 16:00:35 2012
config:

        NAME                                          STATE     READ WRITE CKSUM
        tank                                          ONLINE       0     0     0
          mirror-0                                    ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_02/disk   ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_20/disk   ONLINE       0     0     0
          mirror-1                                    ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_22/disk   ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_14/disk   ONLINE       0     0     0
          mirror-2                                    ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_10/disk   ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_16/disk   ONLINE       0     0     0
          mirror-3                                    ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_01/disk   ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_21/disk   ONLINE       0     0     0
          mirror-4                                    ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_23/disk   ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_15/disk   ONLINE       0     0     0
          mirror-5                                    ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_09/disk   ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_04/disk   ONLINE       0     0     0
          mirror-6                                    ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_08/disk   ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_05/disk   ONLINE       0     0     0
          mirror-7                                    ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_07/disk   ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_11/disk   ONLINE       0     0     0
          mirror-8                                    ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_06/disk   ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_19/disk   ONLINE       0     0     0
          mirror-9                                    ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_00/disk   ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_13/disk   ONLINE       0     0     0
          mirror-10                                   ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_03/disk   ONLINE       0     0     0
            /dev/chassis/Lab10Rack5.../DISK_18/disk   ONLINE       0     0     0
        spares
          /dev/chassis/Lab10Rack5.../DISK_17/disk     AVAIL
          /dev/chassis/Lab10Rack5.../DISK_12/disk     AVAIL
```

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

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 84 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':
2012-04-06.14:02:55 zpool create -f rpool c3t0d0s0
2012-04-06.14:02:56 zfs create -p -o mountpoint=/export rpool/export
2012-04-06.14:02:58 zfs set mountpoint=/export rpool/export
2012-04-06.14:02:58 zfs create -p rpool/export/home
2012-04-06.14:03:03 zfs create -p -V 2048m rpool/swap
2012-04-06.14:03:08 zfs set primarycache=metadata rpool/swap
2012-04-06.14:03:09 zfs create -p -V 4094m rpool/dump
```

```
2012-04-06.14:26:47 zpool set bootfs=rpool/ROOT/s11u1 rpool
2012-04-06.14:31:15 zfs set primarycache=metadata rpool/swap
2012-04-06.14:31:46 zfs create -o canmount=noauto -o mountpoint=/var/share rpool/VARSHARE
2012-04-06.15:22:33 zfs set primarycache=metadata rpool/swap
2012-04-06.16:42:48 zfs set primarycache=metadata rpool/swap
2012-04-09.16:17:24 zfs snapshot -r rpool/ROOT@yesterday
2012-04-09.16:17:54 zfs snapshot -r rpool/ROOT@now
```

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

툴 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 32 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 툴에 대한 부트 이후 누적 통계를 표시합니다. 예:

```
# 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 3은 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. 예:

```
# 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.

풀의 가상 장치에 대한 물리적 위치 정보를 표시할 수도 있습니다. 예를 들면 다음과 같습니다.

```
# zpool iostat -lv
               capacity     operations    bandwidth
pool        alloc   free   read  write   read  write
----------  -----  -----  -----  -----  -----  -----
export      2.39T  2.14T     13     27  42.7K   300K
  mirror     490G   438G      2      5  8.53K  60.3K
    /dev/chassis/lab10rack15/SCSI_Device__2/disk      -      -      1      0  4.47K  60.3K
    /dev/chassis/lab10rack15/SCSI_Device__3/disk      -      -      1      0  4.45K  60.3K
  mirror     490G   438G      2      5  8.62K  59.9K
    /dev/chassis/lab10rack15/SCSI_Device__4/disk      -      -      1      0  4.52K  59.9K
    /dev/chassis/lab10rack15/SCSI_Device__5/disk      -      -      1      0  4.48K  59.9K
  mirror     490G   438G      2      5  8.60K  60.2K
    /dev/chassis/lab10rack15/SCSI_Device__6/disk      -      -      1      0  4.50K  60.2K
    /dev/chassis/lab10rack15/SCSI_Device__7/disk      -      -      1      0  4.49K  60.2K
  mirror     490G   438G      2      5  8.47K  60.1K
    /dev/chassis/lab10rack15/SCSI_Device__8/disk      -      -      1      0  4.42K  60.1K
    /dev/chassis/lab10rack15/SCSI_Device__9/disk      -      -      1      0  4.43K  60.1K
.
.
.
```

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 pond
  pool: pond
 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: Determine if the device needs to be replaced, and clear the errors
        using 'zpool clear' or 'fmadm repaired', or replace the device
        with 'zpool replace'.
  scan: scrub repaired 0 in 0h0m with 0 errors on Wed Jun 20 15:38:08 2012
config:

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

device details:

        c0t5000C500335F907Fd0    UNAVAIL          cannot open
        status: ZFS detected errors on this device.
                The device was missing.
           see: http://support.oracle.com/msg/ZFS-8000-LR for recovery


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 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: 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 206 페이지 “ZFS 데이터 전송 및 수신”을 참조하십시오.

- 95 페이지 “ZFS 저장소 풀 마이그레이션 준비”
- 96 페이지 “ZFS 저장소 풀 내보내기”
- 96 페이지 “가져올 수 있는 저장소 풀 결정”
- 98 페이지 “대체 디렉토리에서 ZFS 저장소 풀 가져오기”
- 98 페이지 “ZFS 저장소 풀 가져오기”
- 102 페이지 “삭제된 ZFS 저장소 풀 복구”

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 257 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: 4715259469716913940
 state: DEGRADED
status: One or more devices are unavailable.
action: The pool can be imported despite missing or damaged devices.  The
        fault tolerance of the pool may be compromised if imported.
config:

    tank                          DEGRADED
       mirror-0                   DEGRADED
         c0t5000C500335E106Bd0    ONLINE
         c0t5000C500335FC3E7d0    UNAVAIL  cannot open

device details:

         c0t5000C500335FC3E7d0    UNAVAIL   cannot open
         status: ZFS detected errors on this device.
                The device was missing.
```

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: mothership
    id: 3702878663042245922
 state: UNAVAIL
status: One or more devices are unavailable.
action: The pool cannot be imported due to unavailable devices or data.
config:

        mothership     UNAVAIL  insufficient replicas
```

```
          raidz1-0     UNAVAIL  insufficient replicas
            c8t0d0     UNAVAIL  cannot open
            c8t1d0     UNAVAIL  cannot open
            c8t2d0     ONLINE
            c8t3d0     ONLINE

device details:

        c8t0d0    UNAVAIL          cannot open
        status: ZFS detected errors on this device.
                The device was missing.

        c8t1d0    UNAVAIL          cannot open
        status: ZFS detected errors on this device.
                The device was missing.
```

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 265 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
  pool: dozer
    id: 16216589278751424645
 state: UNAVAIL
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://support.oracle.com/msg/ZFS-8000-6X
config:

        dozer              UNAVAIL  missing device
          mirror-0         ONLINE
            c8t0d0  ONLINE
            c8t1d0  ONLINE

device details:

        missing-1        UNAVAIL          corrupted data
        status: ZFS detected errors on this device.
                The device has bad label or disk contents.


        Additional devices are known to be part of this pool, though their
        exact configuration cannot be determined.
```

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 are unavailable in response to persistent errors.
        Sufficient replicas exist for the pool to continue functioning in a
        degraded state.
action: Determine if the device needs to be replaced, and clear the errors
        using 'zpool clear' or 'fmadm repaired', or replace the device
        with 'zpool replace'.
        Run 'zpool status -v' to see device specific details.
  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.

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. 예:

```
# zpool destroy dozer
# zpool import -D
```

```
  pool: dozer
    id: 4107023015970708695
 state: DEGRADED (DESTROYED)
status: One or more devices are unavailable.
action: The pool can be imported despite missing or damaged devices.  The
        fault tolerance of the pool may be compromised if imported.
config:

        dozer              DEGRADED
          raidz2-0         DEGRADED
            c8t0d0         ONLINE
            c8t1d0         ONLINE
            c8t2d0         ONLINE
            c8t3d0         UNAVAIL  cannot open
            c8t4d0         ONLINE

device details:

        c8t3d0    UNAVAIL          cannot open
        status: ZFS detected errors on this device.
                The device was missing.
# zpool import -Df dozer
# zpool status -x
  pool: dozer
 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: Determine if the device needs to be replaced, and clear the errors
        using 'zpool clear' or 'fmadm repaired', or replace the device
        with 'zpool replace'.
        Run 'zpool status -v' to see device specific details.
  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. 예:

```
# 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 33.

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  Deduplication
 22  Received properties
 23  Slim ZIL
```

```
 24  System attributes
 25  Improved scrub stats
 26  Improved snapshot deletion performance
 27  Improved snapshot creation performance
 28  Multiple vdev replacements
 29  RAID-Z/mirror hybrid allocator
 30  Encryption
 31  Improved 'zfs list' performance
 32  One MB blocksize
 33  Improved share support
 34  Sharing with inheritance

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.

◆ ◆ ◆ 4 장

4 ZFS 루트 풀 구성 요소 관리

이 장에서는 루트 풀 미러 연결, ZFS 부트 환경 복제 및 스왑 및 덤프 장치 크기 조정과 같이 Oracle Solaris ZFS 루트 풀 구성 요소를 관리하는 방법을 설명합니다.

이 장에서는 다음과 같은 내용을 다룹니다.

- 107 페이지 "ZFS 루트 풀 구성 요소 관리(개요)"
- 108 페이지 "ZFS 루트 풀 요구 사항"
- 109 페이지 "ZFS 루트 풀 관리"
- 121 페이지 "ZFS 스왑 및 덤프 장치 관리"
- 124 페이지 "ZFS 루트 파일 시스템에서 부트"

루트 풀 복구에 대한 자세한 내용은 11 장, "스냅샷 아카이브 및 루트 풀 복구"를 참조하십시오.

최근 문제의 경우 Oracle Solaris 11.1 릴리스 정보를 참조하십시오.

ZFS 루트 풀 구성 요소 관리(개요)

ZFS는 Oracle Solaris 11 릴리스의 기본 루트 파일 시스템입니다. Oracle Solaris 릴리스를 설치할 때 다음 고려 사항을 검토하십시오.

- **설치** – Oracle Solaris 11 릴리스에서는 다음 방법으로 ZFS 루트 파일 시스템을 설치하고 부트할 수 있습니다.
  - Live CD(x86에만 해당) – 단일 디스크에 ZFS 루트 풀을 설치합니다. 설치 중 `fdisk` 분할 영역 메뉴를 사용하여 사용자 환경에 맞게 디스크를 분할할 수 있습니다.
  - 텍스트 설치(SPARC 및 x86) – 매체에서 또는 네트워크를 통해 ZFS 루트 풀을 단일 디스크에 설치합니다. 설치 중 `fdisk` 분할 영역 메뉴를 사용하여 사용자 환경에 맞게 디스크를 분할할 수 있습니다.
  - 자동 설치 프로그램(AI)(SPARC 및 x86) – ZFS 루트 풀을 자동으로 설치합니다. AI 매니페스트를 사용하여 ZFS 루트 풀에 사용할 디스크 및 디스크 분할 영역을 확인할 수 있습니다.

- **스왑 및 덤프 장치** – 위의 설치 방법 모두 ZFS 루트 풀의 ZFS 볼륨에 자동으로 만들어집니다. ZFS 스왑 및 덤프 장치 관리에 대한 자세한 내용은 121 페이지 "ZFS 스왑 및 덤프 장치 관리"를 참조하십시오.
- **미러링된 루트 풀 구성** – 자동 설치 도중 미러링된 루트 풀을 구성할 수 있습니다. 설치 후 미러링된 루트 풀 구성에 대한 자세한 내용은 112 페이지 "미러링된 루트 풀을 구성하는 방법(SPARC 또는 x86/VTOC)"을 참조하십시오.
- **루트 풀 공간 관리** – 시스템을 설치한 후 루트 파일 시스템이 채워지지 않도록 ZFS 루트 파일 시스템에 쿼터를 설정하는 것이 좋습니다. 현재까지 전체 파일 시스템에 대한 안전망으로 ZFS 루트 풀 공간이 예약되지 않았습니다. 예를 들어, 루트 풀에 68GB 디스크가 있는 경우 ZFS 루트 파일 시스템(`rpool/ROOT/solaris`)에 67GB 쿼터를 설정하여 파일 시스템 공간을 1GB 남겨두는 것이 좋습니다. 쿼터 설정에 대한 자세한 내용은 180 페이지 "ZFS 파일 시스템에 대한 쿼터 설정"을 참조하십시오.

ZFS 루트 풀 요구 사항

ZFS 루트 풀 공간 및 구성 요구 사항에 대해 설명하는 다음 절의 내용을 검토하십시오.

ZFS 루트 풀 공간 요구 사항

시스템을 설치할 때 스왑 볼륨 및 덤프 볼륨의 크기는 물리적 메모리의 크기에 따라 달라집니다. 부트 가능 ZFS 루트 파일 시스템에 대한 최소 풀 공간은 물리적 메모리, 사용 가능한 디스크 공간 및 만들려는 BE(부트 환경) 수에 따라 다릅니다.

다음 ZFS 저장소 풀 공간 요구 사항을 검토하십시오.

- 다른 설치 방법의 메모리 요구 사항에 대한 설명은 **Oracle Solaris 11.1 릴리스 노트**를 참조하십시오.
- 권장되는 최소 디스크 공간은 7-13GB입니다. 이 공간은 다음과 같이 사용됩니다.
  - **스왑 영역 및 덤프 장치** – Solaris 설치 프로그램에서 만드는 스왑 및 덤프 볼륨의 기본 크기는 시스템의 메모리 크기 및 다른 변수에 따라 달라집니다. 덤프 장치 크기는 시스템의 작업에 따라 물리적 메모리 크기의 약 절반 이상입니다.

    스왑 및 덤프 볼륨의 크기를 선택한 크기로 조정할 수 있습니다. 단, 새 크기가 설치 도중 또는 설치 후 시스템 작동을 지원해야 합니다. 자세한 내용은 122 페이지 "ZFS 스왑 및 덤프 장치의 크기 조정"을 참조하십시오.
  - **BE(부트 환경)** – ZFS BE는 약 4-6GB입니다. 다른 ZFS BE에서 복제된 각 ZFS BE에는 추가 디스크 공간이 필요 없습니다. 업데이트에 따라 BE를 업데이트하면 BE 크기가 증가합니다. 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 시스템에 있어야 합니다. 스왑 및 덤프 장치를 제외한 모든 Solaris OS 구성 요소도 루트 풀에 상주해야 합니다.

ZFS 루트 풀 구성 요구 사항

다음 ZFS 저장소 풀 구성 요구 사항을 검토하십시오.

- Oracle Solaris 11.1에서 루트 풀용으로 사용되는 디스크는 x86 기반 시스템의 EFI(GPT) 또는 SMI(VTOC)나 SPARC 시스템의 SMI(VTOC) 레이블을 가질 수 있습니다.
  - 업데이트된 GPT 지원 펌웨어가 포함된 SPARC 시스템은 루트 풀 디스크에 EFI(GPT) 디스크 레이블을 설치합니다. SPARC 시스템에 업데이트된 펌웨어가 포함되지 않은 경우에는 루트 풀 디스크에 SMI(VTOC) 디스크 레이블이 설치됩니다.
  - x86 기반 시스템은 대부분의 경우 루트 풀 디스크의 EFI(GPT) 레이블로 설치됩니다.

  EFI(GPT) 레이블이 x86 기반 시스템에서 어떻게 표시되는지에 대한 자세한 내용은 43 페이지 “ZFS 저장소 풀의 디스크 사용”을 참조하십시오.
- 풀은 SPARC 기반 시스템 또는 x86 기반 시스템의 SMI(VTOC) 레이블 디스크인 경우 미러링되는 디스크 슬라이스에 존재해야 합니다. 또는 루트 풀 디스크가 EFI(GPT)로 레이블 지정된 경우 풀이 전체 디스크 또는 미러링된 전체 디스크에 존재할 수 있습니다. `beadm` 작업 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 51 페이지 “ZFS 루트 풀 만들기”를 참조하십시오.
- x86 기반 시스템에서는 디스크에 Solaris `fdisk` 분할 영역이 포함되어야 합니다. Solaris `fdisk` 분할 영역은 x86 기반 시스템이 설치될 때 자동으로 만들어집니다. Solaris `fdisk` 분할 영역에 대한 자세한 내용은 **Oracle Solaris 11.1 관리: 장치 및 파일 시스템**의 “fdisk 분할 영역을 만드는 방법에 대한 지침”을 참조하십시오.
- 자동 설치 도중 루트 풀에 풀 등록 정보 또는 파일 시스템 등록 정보를 설정할 수 있습니다. 루트 풀에서는 `gzip` 압축 알고리즘이 지원되지 않습니다.
- 초기 설치로 루트 풀을 만든 후에는 루트 풀 이름을 바꾸지 마십시오. 루트 풀의 이름을 바꾸면 시스템이 부트되지 않을 수 있습니다.

ZFS 루트 풀 관리

다음 절에서는 ZFS 루트 풀 설치 및 업데이트와 미러링된 루트 풀 구성에 대한 정보를 제공합니다.

ZFS 루트 풀 설치

Oracle Solaris 11 Live CD 설치 방법을 사용하면 단일 디스크에 기본 ZFS 루트 풀이 설치됩니다. Oracle Solaris 11 AI(자동 설치) 방법을 사용할 경우 ZFS 루트 풀의 디스크 또는 미러링된 디스크를 식별할 AI 매니페스트를 만들 수 있습니다.

AI 설치 프로그램은 기본 부트 디스크 또는 사용자가 식별한 대상 디스크에 ZFS 루트 풀을 설치하는 유연성을 제공합니다. `c1t0d0`과 같은 논리적 장치 또는 물리적 장치 경로를 지정할 수 있습니다. 또한 설치할 장치에 대해 MPxIO 식별자 또는 장치 ID를 사용할 수 있습니다.

설치 후 ZFS 저장소 풀 및 파일 시스템 정보를 검토하십시오. 이는 설치 유형 및 사용자 정의에 따라 다를 수 있습니다. 예를 들면 다음과 같습니다.

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

        NAME        STATE     READ WRITE CKSUM
        rpool       ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c8t0d0  ONLINE       0     0     0
            c8t1d0  ONLINE       0     0     0
# zfs list
NAME                      USED  AVAIL  REFER  MOUNTPOINT
rpool                    11.8G  55.1G  4.58M  /rpool
rpool/ROOT               3.57G  55.1G    31K  legacy
rpool/ROOT/solaris       3.57G  55.1G  3.40G  /
rpool/ROOT/solaris/var    165M  55.1G   163M  /var
rpool/VARSHARE           42.5K  55.1G  42.5K  /var/share
rpool/dump               6.19G  55.3G  6.00G  -
rpool/export               63K  55.1G    32K  /export
rpool/export/home          31K  55.1G    31K  /export/home
rpool/swap               2.06G  55.2G  2.00G  -
```

ZFS BE 정보를 검토하십시오. 예를 들면 다음과 같습니다.

```
# beadm list
BE      Active Mountpoint Space Policy Created
--      ------ ---------- ----- ------ -------
solaris NR     /          3.75G static 2012-07-20 12:10
```

위 출력에서 `Active` 필드는 BE가 현재 활성 상태인지(`N`으로 표시됨), 재부트 시 활성화되는지(`R`로 표시됨) 또는 둘 다(`NR`로 표시됨)인지를 나타냅니다.

▼ ZFS 부트 환경 업데이트 방법

기본 ZFS BE(부트 환경)의 이름은 기본적으로 `solaris`입니다. `beadm list` 명령을 사용하여 BE를 식별할 수 있습니다. 예를 들면 다음과 같습니다.

```
# beadm list
BE      Active Mountpoint Space Policy Created
--      ------ ---------- ----- ------ -------
solaris NR     /          3.82G static 2012-07-19 13:44
```

위 출력에서 NR은 BE가 현재 활성 상태이며 재부트 시 활성 BE가 됨을 의미합니다.

pkg update 명령을 사용하여 ZFS 부트 환경을 업데이트할 수 있습니다. pkg update 명령을 사용하여 ZFS BE를 업데이트할 경우 기존 BE에 대한 업데이트가 매우 사소한 업데이트가 아닌 한 새 BE가 만들어지고 자동으로 활성화됩니다.

1 ZFS BE를 업데이트합니다.

```
# pkg update


DOWNLOAD                                  PKGS       FILES    XFER (MB)
Completed                                707/707 10529/10529  194.9/194.9
.
.
.
```

새 BE solaris-1이 자동으로 만들어지고 활성화됩니다.

업데이트 프로세스 외부에서 백업 BE를 만들고 활성화할 수도 있습니다.

```
# beadm create solaris-1
# beadm activate solaris-1
```

2 BE 활성화를 완료하려면 시스템을 재부트합니다. 그런 후 BE 상태를 확인합니다.

```
# init 6
.
.
.
# beadm list
BE        Active Mountpoint Space  Policy Created
--        ------ ---------- -----  ------ -------
solaris   -      -          46.95M static 2012-07-20 10:25
solaris-1 NR     /          3.82G  static 2012-07-19 14:45
```

3 새 BE를 부트할 때 오류가 발생하는 경우 이전 BE를 활성화하고 재부트합니다.

```
# beadm activate solaris
# init 6
```

▼ 대체 BE를 마운트하는 방법

복구 용도로 다른 BE의 파일을 복사하거나 액세스해야 할 수도 있습니다.

1 관리자로 전환합니다.

2 대체 BE를 마운트합니다.

```
# beadm mount solaris-1 /mnt
```

3 **BE에 액세스합니다.**

```
# ls /mnt
bin         export      media       pkg         rpool       tmp
boot        home        mine        platform    sbin        usr
dev         import      mnt         proc        scde        var
devices     java        net         project     shared
doe         kernel      nfs4        re          src
etc         lib         opt         root        system
```

4 **작업을 마쳤으면 대체 BE를 마운트 해제합니다.**

```
# beadm umount solaris-1
```

▼ 미러링된 루트 풀을 구성하는 방법(SPARC 또는 x86/VTOC)

자동 설치 도중 미러링된 루트 풀을 구성하지 않는 경우 설치 후에 미러링된 루트 풀을 쉽게 구성할 수 있습니다.

루트 풀의 디스크 교체에 대한 자세한 내용은 115 페이지 “ZFS 루트 풀의 디스크 교체 방법(SPARC 또는 x86/VTOC)”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
          c2t0d0s0  ONLINE       0     0     0

errors: No known data errors
```

2 **필요한 경우 루트 풀에 연결할 두번째 디스크를 준비합니다.**

- SPARC: 디스크의 디스크 레이블이 SMI(VTOC), 슬라이스가 0인지 확인합니다. 디스크 레이블을 재지정하고 슬라이스 0을 만들어야 할 경우 **Oracle Solaris 11.1 관리: 장치 및 파일 시스템**의 “ZFS 루트 파일 시스템에 사용할 디스크 슬라이스를 만드는 방법”을 참조하십시오.
- x86: 디스크의 분할 영역이 `fdisk`, 디스크 레이블이 SMI, 슬라이스가 0인지 확인합니다. 디스크의 분할 영역을 재지정하고 슬라이스 0을 만들어야 할 경우 **Oracle Solaris 11.1 관리: 장치 및 파일 시스템**의 “ZFS 루트 파일 시스템에 사용할 디스크 준비”를 참조하십시오.

3 **두번째 디스크를 연결하여 미러링된 루트 풀을 구성합니다.**

```
# zpool attach rpool c2t0d0s0 c2t1d0s0
Make sure to wait until resilver is done before rebooting.
```

올바른 디스크 레이블 지정 및 부트 블록은 자동으로 적용됩니다.

4 루트 풀 상태를 통해 리실버링이 완료되었는지 확인합니다.

```
# zpool status rpool
# zpool status rpool
  pool: rpool
 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.
        Run 'zpool status -v' to see device specific details.
  scan: resilver in progress since Fri Jul 20 13:39:53 2012
    938M scanned out of 11.7G at 46.9M/s, 0h3m to go
    938M resilvered, 7.86% done
config:

        NAME          STATE     READ WRITE CKSUM
        rpool         DEGRADED     0     0     0
          mirror-0    DEGRADED     0     0     0
            c2t0d0s0  ONLINE       0     0     0
            c2t1d0s0  DEGRADED     0     0     0  (resilvering)
```

위의 출력 결과에서는 리실버링 프로세스가 완료되지 않았습니다. 다음과 유사한 메시지가 표시되면 리실버링이 완료된 것입니다.

```
resilvered 11.6G in 0h5m with 0 errors on Fri Jul 20 13:57:25 2012
```

5 더 큰 디스크를 연결 중인 경우 풀의 autoexpand 등록 정보를 설정하여 풀 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 성공적으로 부트할 수 있는지 확인합니다.

▼ 미러링된 루트 풀을 구성하는 방법(x86/EFI(GPT))

Oracle Solaris 11.1 릴리스는 대부분의 경우 기본적으로 x86 기반 시스템에서 EFI(GPT) 레이블을 설치합니다.

자동 설치 도중 미러링된 루트 풀을 구성하지 않는 경우 설치 후에 미러링된 루트 풀을 쉽게 구성할 수 있습니다.

루트 풀의 디스크 교체에 대한 자세한 내용은 115 페이지 “ZFS 루트 풀의 디스크 교체 방법(SPARC 또는 x86/VTOC)”을 참조하십시오.

1 현재 루트 풀 상태를 표시합니다.

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

        NAME      STATE     READ WRITE CKSUM
        rpool     ONLINE       0     0     0
          c2t0d0  ONLINE       0     0     0

errors: No known data errors
```

2 두번째 디스크를 연결하여 미러링된 루트 풀을 구성합니다.

```
# zpool attach rpool c2t0d0 c2t1d0
Make sure to wait until resilver is done before rebooting.
```

올바른 디스크 레이블 지정 및 부트 블록은 자동으로 적용됩니다.

루트 풀 디스크에서 분할 영역을 사용자 정의한 경우 다음과 비슷한 구문이 필요할 수 있습니다.

```
# zpool attach rpool c2t0d0s0 c2t1d0
```

3 루트 풀 상태를 통해 리실버링이 완료되었는지 확인합니다.

```
# zpool status rpool
  pool: rpool
 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.
        Run ’zpool status -v’ to see device specific details.
  scan: resilver in progress since Fri Jul 20 13:52:05 2012
    809M scanned out of 11.6G at 44.9M/s, 0h4m to go
    776M resilvered, 6.82% done
config:

        NAME        STATE     READ WRITE CKSUM
        rpool       DEGRADED     0     0     0
          mirror-0  DEGRADED     0     0     0
            c8t0d0  ONLINE       0     0     0
            c8t1d0  DEGRADED     0     0     0  (resilvering)

errors: No known data errors
```

위의 출력 결과에서는 리실버링 프로세스가 완료되지 않았습니다. 다음과 유사한 메시지가 표시되면 리실버링이 완료된 것입니다.

```
resilvered 11.6G in 0h5m with 0 errors on Fri Jul 20 13:57:25 2012
```

4 **더 큰 디스크를 연결 중인 경우 풀의 autoexpand 등록 정보를 설정하여 풀 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  -
```

5 **새 디스크에서 성공적으로 부트할 수 있는지 확인합니다.**

▼ ZFS 루트 풀의 디스크 교체 방법(SPARC 또는 x86/VTOC)

다음과 같은 경우 루트 풀의 디스크를 교체해야 할 수 있습니다.

- 루트 풀이 너무 작은 상태에서 디스크를 더 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: CD 또는 네트워크)에서 부트해야 합니다.
- zpool replace 명령을 사용하여 루트 풀 디스크에서 디스크를 교체하는 경우 부트 블록을 수동으로 적용해야 합니다.

미러링된 루트 풀 구성에서 대체 매체로부터 부트하지 않고도 디스크 교체를 시도할 수 있습니다. zpool replace 명령을 사용하여 실패한 디스크를 교체하거나, 추가 디스크가 있는 경우 zpool attach 명령을 사용할 수 있습니다. 추가 디스크를 연결하고 루트 풀 디스크를 분리하는 예는 아래 단계를 참조하십시오.

SATA 디스크가 있는 시스템의 경우 장애가 발생한 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
<Confirm that the new disk has an SMI label and a slice 0>
# zpool online rpool c1t0d0s0
# zpool replace rpool c1t0d0s0
# zpool status rpool
```

```
<Let disk resilver before installing the boot blocks>
# bootadm install-bootloader
```

교체 디스크를 넣은 후 해당 디스크를 온라인 상태로 만들거나 재구성할 필요가 없는 하드웨어도 있습니다.

1 교체 디스크를 물리적으로 연결합니다.

2 필요한 경우 루트 풀에 연결할 두번째 디스크를 준비합니다.

- SPARC: 교체(새) 디스크의 레이블이 SMI(VTOC), 슬라이스가 0인지 확인합니다. 루트 풀에 사용할 디스크의 레이블 재지정에 대한 자세한 내용은 **Oracle Solaris 11.1 관리: 장치 및 파일 시스템**의 "디스크에 레이블을 지정하는 방법"을 참조하십시오.
- x86: 디스크의 분할 영역이 `fdisk`, 디스크 레이블이 SMI, 슬라이스가 0인지 확인합니다. 디스크 분할 영역을 다시 지정하고 슬라이스 0을 만들어야 할 경우 **Oracle Solaris 11.1 관리: 장치 및 파일 시스템**의 "ZFS 루트 파일 시스템에 사용할 디스크 설정 방법"을 참조하십시오.

3 새 디스크를 루트 풀에 연결합니다.

예를 들면 다음과 같습니다.

```
# zpool attach rpool c2t0d0s0 c2t1d0s0
Make sure to wait until resilver is done before rebooting.
```

올바른 디스크 레이블 지정 및 부트 블록은 자동으로 적용됩니다.

4 루트 풀 상태를 확인합니다.

예를 들면 다음과 같습니다.

```
# zpool status rpool
  pool: rpool
 state: ONLINE
 scan: resilvered 11.7G in 0h5m with 0 errors on Fri Jul 20 13:45:37 2012
config:

        NAME          STATE     READ WRITE CKSUM
        rpool         ONLINE       0     0     0
          mirror-0    ONLINE       0     0     0
            c2t0d0s0  ONLINE       0     0     0
            c2t1d0s0  ONLINE       0     0     0

errors: No known data errors
```

5 리실버링이 완료된 후 새 디스크에서 부트할 수 있는지 확인합니다.

예를 들어, SPARC 기반 시스템에서는 다음을 수행합니다.

```
ok boot /pci@1f,700000/scsi@2/disk@1,0
```

교체 디스크에서의 부트를 테스트하고 교체 디스크가 실패할 경우 기존 디스크에서 수동으로 부트할 수 있도록 현재 디스크와 새 디스크의 부트 장치 경로 이름을 식별합니다. 아래 예에서 현재 루트 풀 디스크(c2t0d0s0)는 다음과 같습니다.

```
/pci@1f,700000/scsi@2/disk@0,0
```

아래 예에서 교체 부트 디스크(c2t1d0s0)는 다음과 같습니다.

```
boot /pci@1f,700000/scsi@2/disk@1,0
```

6 **시스템이 새 디스크에서 부트되면 이전 디스크를 분리합니다.**

예를 들면 다음과 같습니다.

```
# zpool detach rpool c2t0d0s0
```

7 **더 큰 디스크를 연결 중인 경우 풀의 autoexpand 등록 정보를 설정하여 풀 크기를 확장하십시오.**

```
# zpool set autoexpand=on rpool
```

또는 장치 확장:

```
# zpool online -e c2t1d0s0
```

8 **새 디스크에서 자동으로 부트되도록 시스템을 설정합니다.**

- SPARC: eeprom 명령 또는 부트 PROM의 setenv 명령을 사용하여 새 디스크에서 자동으로 부트되도록 시스템을 설정합니다.
- x86: 시스템 BIOS를 재구성합니다.

▼ ZFS 루트 풀의 디스크 교체 방법(SPARC 또는 x86/EFI(GPT))

Oracle Solaris 11.1 릴리스는 대부분의 경우 기본적으로 x86 기반 시스템에서 EFI(GPT) 레이블을 설치합니다.

다음과 같은 경우 루트 풀의 디스크를 교체해야 할 수 있습니다.

- 루트 풀이 너무 작은 상태에서 디스크를 더 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: CD 또는 네트워크)에서 부트해야 합니다.
- zpool replace 명령을 사용하여 루트 풀 디스크에서 디스크를 교체하는 경우 부트 블록을 수동으로 적용해야 합니다.

미러링된 루트 풀 구성에서 대체 매체로부터 부트하지 않고도 디스크 교체를 시도할 수 있습니다. zpool replace 명령을 사용하여 실패한 디스크를 교체하거나, 추가 디스크가

있는 경우 zpool attach 명령을 사용할 수 있습니다. 추가 디스크를 연결하고 루트 풀 디스크를 분리하는 예는 아래 단계를 참조하십시오.

SATA 디스크가 있는 시스템의 경우 장애가 발생한 디스크를 교체하기 위해 zpool replace 작업을 시도하기 전에 디스크를 오프라인 상태로 만들고 구성을 해제해야 합니다. 예를 들면 다음과 같습니다.

```
# zpool offline rpool c1t0d0
# cfgadm -c unconfigure c1::dsk/c1t0d0
<Physically remove failed disk c1t0d0>
<Physically insert replacement disk c1t0d0>
# cfgadm -c configure c1::dsk/c1t0d0
# zpool online rpool c1t0d0
# zpool replace rpool c1t0d0
# zpool status rpool
<Let disk resilver before installing the boot blocks>
x86# bootadm install-bootloader
```

교체 디스크를 넣은 후 해당 디스크를 온라인 상태로 만들거나 재구성할 필요가 없는 하드웨어도 있습니다.

1 교체 디스크를 물리적으로 연결합니다.

2 새 디스크를 루트 풀에 연결합니다.

예를 들면 다음과 같습니다.

```
# zpool attach rpool c2t0d0 c2t1d0
Make sure to wait until resilver is done before rebooting.
```

올바른 디스크 레이블 지정 및 부트 블록은 자동으로 적용됩니다.

3 루트 풀 상태를 확인합니다.

예를 들면 다음과 같습니다.

```
# zpool status rpool
  pool: rpool
 state: ONLINE
  scan: resilvered 11.6G in 0h5m with 0 errors on Fri Jul 20 12:06:07 2012
config:

        NAME        STATE     READ WRITE CKSUM
        rpool       ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c2t0d0  ONLINE       0     0     0
            c2t1d0  ONLINE       0     0     0

errors: No known data errors
```

4 리실버링이 완료된 후 새 디스크에서 부트할 수 있는지 확인합니다.

5 **시스템이 새 디스크에서 부트되면 이전 디스크를 분리합니다.**

예를 들면 다음과 같습니다.

```
# zpool detach rpool c2t0d0
```

6 **더 큰 디스크를 연결 중인 경우 풀의 autoexpand 등록 정보를 설정하여 풀 크기를 확장하십시오.**

```
# zpool set autoexpand=on rpool
```

또는 장치 확장:

```
# zpool online -e c2t1d0
```

7 **새 디스크에서 자동으로 부트되도록 시스템을 설정합니다.**

시스템 BIOS를 재구성합니다.

▼ 다른 루트 풀에 BE를 만드는 방법(SPARC 또는 x86/VTOC)

다른 루트 풀에 기존 BE를 재생성하려는 경우 아래 단계를 수행합니다. 독립적인 스왑 및 덤프 장치가 있는 유사한 BE가 포함된 두 개의 루트 풀을 만들 것인지 또는 스왑 및 덤프 장치를 공유하는 다른 루트 풀에 BE를 만들 것인지에 따라 단계를 수정할 수 있습니다.

두번째 루트 풀의 새 BE를 활성화하고 부트하면 첫번째 루트 풀의 이전 BE에 대한 정보가 없습니다. 원래 BE로 재부트하려는 경우 원래 루트 풀의 부트 디스크에서 수동으로 시스템을 부트해야 합니다.

1 **SMI(VTOC) 레이블이 지정된 디스크를 사용하여 두번째 루트 풀을 만듭니다. 예를 들면 다음과 같습니다.**

```
# zpool create rpool2 c4t2d0s0
```

2 **두번째 루트 풀에 새 BE를 만듭니다. 예를 들면 다음과 같습니다.**

```
# beadm create -p rpool2 solaris2
```

3 **두번째 루트 풀에서 bootfs 등록 정보를 설정합니다. 예를 들면 다음과 같습니다.**

```
# zpool set bootfs=rpool2/ROOT/solaris2 rpool2
```

4 **새 BE를 활성화합니다. 예를 들면 다음과 같습니다.**

```
# beadm activate solaris2
```

5 **새 BE에서 부트하지만 구체적으로 두번째 루트 풀의 부트 장치에서 부트해야 합니다.**

```
ok boot disk2
```

시스템이 새 BE로 실행됩니다.

6 **스왑 볼륨을 다시 만듭니다. 예를 들면 다음과 같습니다.**

```
# zfs create -V 4g rpool2/swap
```

7 **새 스왑 장치에 대한 /etc/vfstab 항목을 업데이트합니다. 예를 들면 다음과 같습니다.**

```
/dev/zvol/dsk/rpool2/swap       -               -               swap -     no      -
```

8 **덤프 볼륨을 다시 만듭니다. 예를 들면 다음과 같습니다.**

```
# zfs create -V 4g rpool2/dump
```

9 **덤프 장치를 재설정합니다. 예를 들면 다음과 같습니다.**

```
# dumpadm -d /dev/zvol/dsk/rpool2/dump
```

10 **두번째 루트 풀의 부트 디스크에서 부트되도록 기본 부트 장치를 재설정합니다.**

- SPARC - eeprom 명령 또는 부트 PROM의 setenv 명령을 사용하여 새 디스크에서 자동으로 부트되도록 시스템을 설정합니다.
- x86 - 시스템 BIOS를 재구성합니다.

11 **재부트하여 원래 루트 풀의 스왑 및 덤프 장치를 지웁니다.**

```
# init 6
```

▼ 다른 루트 풀에 BE를 만드는 방법(SPARC 또는 x86/EFI(GPT))

Oracle Solaris 11.1 릴리스는 대부분의 경우 기본적으로 x86 기반 시스템에서 EFI(GPT) 레이블을 설치합니다.

다른 루트 풀에 기존 BE를 재생성하려는 경우 아래 단계를 수행합니다. 독립적인 스왑 및 덤프 장치가 있는 유사한 BE가 포함된 두 개의 루트 풀을 만들 것인지 또는 스왑 및 덤프 장치를 공유하는 다른 루트 풀에 BE를 만들 것인지에 따라 단계를 수정할 수 있습니다.

두번째 루트 풀의 새 BE를 활성화하고 부트하면 첫번째 루트 풀의 이전 BE에 대한 정보가 없습니다. 원래 BE로 재부트하려는 경우 원래 루트 풀의 부트 디스크에서 수동으로 시스템을 부트해야 합니다.

1 **다른 루트 풀을 만듭니다.**

```
# zpool create -B rpool2 c2t2d0
```

또는 미러링된 다른 루트 풀을 만듭니다. 예를 들면 다음과 같습니다.

```
# zpool create -B rpool2 mirror c2t2d0 c2t3d0
```

2 **두번째 루트 풀에 새 BE를 만듭니다. 예를 들면 다음과 같습니다.**

```
# beadm create -p rpool2 solaris2
```

3 **부트 정보를 두번째 루트 풀에 적용합니다. 예를 들면 다음과 같습니다.**

```
# bootadm install-bootloader -P rpool2
```

4 **두번째 루트 풀에서 bootfs 등록 정보를 설정합니다. 예를 들면 다음과 같습니다.**

```
# zpool set bootfs=rpool2/ROOT/solaris2 rpool2
```

5 **새 BE를 활성화합니다. 예를 들면 다음과 같습니다.**

```
# beadm activate solaris2
```

6 **새 BE로부터 부트합니다.**

- SPARC - eeprom 명령 또는 부트 PROM의 setenv 명령을 사용하여 새 디스크에서 자동으로 부트되도록 시스템을 설정합니다.
- x86 - 시스템 BIOS를 재구성합니다.

시스템이 새 BE로 실행됩니다.

7 **스왑 볼륨을 다시 만듭니다. 예를 들면 다음과 같습니다.**

```
# zfs create -V 4g rpool2/swap
```

8 **새 스왑 장치에 대한 /etc/vfstab 항목을 업데이트합니다. 예를 들면 다음과 같습니다.**

```
/dev/zvol/dsk/rpool2/swap       -               -               swap -     no       -
```

9 **덤프 볼륨을 다시 만듭니다. 예를 들면 다음과 같습니다.**

```
# zfs create -V 4g rpool2/dump
```

10 **덤프 장치를 재설정합니다. 예를 들면 다음과 같습니다.**

```
# dumpadm -d /dev/zvol/dsk/rpool2/dump
```

11 **재부트하여 원래 루트 풀의 스왑 및 덤프 장치를 지웁니다.**

```
# init 6
```

ZFS 스왑 및 덤프 장치 관리

설치 프로세스 중에 ZFS 루트 풀의 ZFS 볼륨에 스왑 영역이 만들어집니다. 예를 들면 다음과 같습니다.

```
# swap -l
swapfile                  dev    swaplo   blocks     free
/dev/zvol/dsk/rpool/swap 145,2        16 16646128 16646128
```

설치 프로세스 중에 ZFS 루트 풀의 ZFS 볼륨에 덤프 영역이 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/
  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 프로그램에서 만드는 스왑 및 덤프 볼륨 크기에 대한 자세한 내용은 108 페이지 "ZFS 루트 풀 요구 사항"을 참조하십시오.

설치 후 스왑 볼륨 크기와 덤프 볼륨 크기를 모두 조정할 수 있습니다. 자세한 내용은 122 페이지 "ZFS 스왑 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.
- 시스템이 설치된 후에 스왑 영역이나 덤프 장치를 변경해야 하는 경우 이전 Solaris 릴리스처럼 `swap` 및 `dumpadm` 명령을 사용합니다. 자세한 내용은 **Oracle Solaris 11.1 관리: 장치 및 파일 시스템**의 16 장, "추가 스왑 공간 구성(작업)" 및 **Oracle Solaris 11.1의 일반적인 문제 해결**의 1 장, "시스템 충돌 정보 관리(작업)"를 참조하십시오.

ZFS 스왑 및 덤프 장치의 크기 조정

설치 후 스왑 및 덤프 장치의 크기를 조정해야 하거나, 스왑 및 덤프 볼륨을 다시 만들어야 할 수 있습니다.

- 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          -
  ```

- 스왑 볼륨 크기를 조정할 수 있지만 늘어난 스왑 크기를 보려면 시스템을 재부트해야 합니다. 예를 들면 다음과 같습니다.

  ```
  # swap -d /dev/zvol/dsk/rpool/swap
  # zfs set volsize=2G rpool/swap
  # swap -a /dev/zvol/dsk/rpool/swap
  # init 6
  ```

  활성 시스템에서의 스왑 장치 제거에 대한 자세한 내용은 **Oracle Solaris 11.1 관리: 장치 및 파일 시스템**의 "Oracle Solaris ZFS 루트 환경에서 스왑 공간을 추가하는 방법"을 참조하십시오.

- 이미 설치된 시스템에 추가 스왑 공간이 필요한데 스왑 장치가 사용 중인 경우 다른 스왑 볼륨을 추가합니다. 예를 들면 다음과 같습니다.

```
# zfs create -V 2G rpool/swap2
```

- 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
```

- 두번째 스왑 볼륨에 대한 항목을 /etc/vfstab 파일에 추가합니다. 예를 들면 다음과 같습니다.

```
/dev/zvol/dsk/rpool/swap2      -               -               swap    -    no    -
```

ZFS 덤프 장치 문제 해결

시스템 충돌 덤프를 캡처하거나 덤프 장치의 크기를 조정할 때 문제가 발생하면 다음 항목을 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 장치가 만들어집니다. 대부분의 경우에는 기본 덤프 장치 크기가 너무 작은 경우에만 덤프 장치의 크기를 조정해야 합니다. 예를 들어, 메모리가 큰 시스템에서 덤프 장치 크기를 다음과 같이 40GB로 늘립니다.

```
# zfs set volsize=40G rpool/dump
```

  큰 덤프 장치의 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/
  Savecore enabled: yes
   Save compressed: on
```

- 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 **Oracle Solaris 11.1 관리: 장치 및 파일 시스템**의 "스왑 공간 계획"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 사용하여 부트됩니다. ZFS 루트 파일 시스템에서 부트되면 부트를 위해 선택된 루트 파일 시스템에서 부트 아카이브와 커널 파일의 경로 이름이 확인됩니다.

ZFS 파일 시스템에서 부트하는 것과 UFS 파일 시스템에서 부트하는 것은 다릅니다. ZFS를 사용하는 경우 장치 지정자가 단일 루트 파일 시스템이 아닌 저장소 풀을 식별하기 때문입니다. 저장소 풀에는 여러 개의 부트 가능 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 파일 시스템은 풀의 `bootfs` 등록 정보로 식별된 파일 시스템입니다. 이 기본 선택 사항은 SPARC 시스템에서 `boot -Z` 명령에 포함된 대체 부트 가능 파일 시스템을 지정하여 대체하거나 x86 기반 시스템의 BIOS에서 대체 부트 장치를 선택하여 대체할 수 있습니다.

미러링된 ZFS 루트 풀의 대체 디스크에서 부트

설치 후 미러링된 ZFS 부트 풀을 만들기 위해 디스크를 연결할 수 있습니다. 미러링된 루트 풀 만들기에 대한 자세한 내용은 112 페이지 "미러링된 루트 풀을 구성하는 방법(SPARC 또는 x86/VTOC)"을 참조하십시오.

미러링된 ZFS 루트 풀에 대해 알려진 다음 문제를 검토하십시오.

- 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, 이 풀의 디스크(`c1t0d0s0` 또는 `c1t1d0s0`)에서 부트할 수 있습니다.

```
# 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
```

SPARC 기반 시스템의 경우 ok 프롬프트에서 대체 디스크를 입력합니다.

```
ok boot /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@1
```

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@1,0:a'
```

x86 기반 시스템의 경우 다음과 비슷한 구문을 사용합니다.

```
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'
```

- x86 기반 시스템의 경우 적합한 BIOS 메뉴에서 미러링된 ZFS 루트 풀의 대체 디스크를 선택합니다.
- **SPARC 또는 x86**: zpool replace 명령을 사용하여 루트 풀 디스크를 교체할 경우 bootadm 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. bootadm 구문은 다음과 같습니다.

  ```
  # bootadm install-bootloader
  ```

  다른 루트 풀에 부트 로더를 설치하려는 경우 -P(풀) 옵션을 사용합니다.

  ```
  # bootadm install-bootloader -P rpool2
  ```

  GRUB 레거시 부트 로더를 설치하려는 경우 레거시 installgrub 명령을 사용합니다.

  ```
  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, 이 풀의 디스크(c1t0d0s0 또는 c1t1d0s0)에서 부트할 수 있습니다.

  ```
  # 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
  ```

  SPARC 기반 시스템의 경우 ok 프롬프트에서 대체 디스크를 입력합니다.

  ```
  ok boot /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@1
  ```

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@1,0:a'
```

x86 기반 시스템의 경우 다음과 비슷한 구문을 사용합니다.

```
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'
```

- x86 기반 시스템의 경우 적합한 BIOS 메뉴에서 미러링된 ZFS 루트 풀의 대체 디스크를 선택합니다.

SPARC 기반 시스템의 ZFS 루트 파일 시스템에서 부트

ZFS BE가 여러 개인 SPARC 기반 시스템에서 `beadm activate` 명령을 사용하여 BE에서 부트할 수 있습니다.

설치 및 `beadm` 활성화 프로세스 중에 ZFS 루트 파일 시스템은 `bootfs` 등록 정보를 통해 자동으로 지정됩니다.

하나의 풀에 부트 가능 파일 시스템이 여러 개 있을 수 있습니다. 기본적으로 */pool-name*`/boot/menu.lst` 파일의 부트 가능 파일 시스템 항목은 풀의 `bootfs` 등록 정보로 식별되지만, `menu.lst` 항목에 풀의 대체 파일 시스템을 지정하는 `bootfs` 명령이 포함될 수 있습니다. 따라서 `menu.lst` 파일에 풀 내 여러 루트 파일 시스템에 대한 항목이 포함될 수 있습니다.

시스템에 ZFS 루트 파일 시스템이 설치되었으면 다음과 비슷한 항목이 `menu.lst` 파일에 추가됩니다.

```
title Oracle Solaris 11.1 SPARC
bootfs rpool/ROOT/solaris
```

새 BE가 만들어지면 `menu.lst` 파일이 자동으로 업데이트됩니다.

```
title Oracle Solaris 11.1 SPARC
bootfs rpool/ROOT/solaris
title solaris
bootfs rpool/ROOT/solaris2
```

SPARC 기반 시스템에서는 다음과 같이 부트할 BE를 선택할 수 있습니다.

- ZFS BE가 활성화된 후에는 `boot -L` 명령을 사용하여 ZFS 풀 내에 있는 부트 가능 파일 시스템 목록을 표시할 수 있습니다. 그런 다음 목록에서 부트 가능 파일 시스템 중 하나를 선택할 수 있습니다. 해당 파일 시스템을 부트하는 방법에 대한 자세한 지침이 표시됩니다. 지침에 따라 선택한 파일 시스템을 부트할 수 있습니다.
- 부트 `-Z` *file system* 명령을 사용하여 특정 ZFS 파일 시스템을 부트합니다.

이 부트 방법은 BE를 자동으로 활성화하지 않습니다. `boot -L` 및 `-Z` 구문으로 BE를 부트한 후 자동으로 부트를 계속하려면 이 BE를 활성화해야 합니다.

예 4–1 특정 ZFS 부트 환경에서 부트

시스템 부트 장치의 ZFS 저장소 풀에 ZFS BE가 여러 개 있을 경우 `beadm activate` 명령을 사용하여 기본 BE를 지정할 수 있습니다.

예를 들어, 다음 ZFS BE는 `beadm` 출력에 설명된 대로 사용 가능합니다.

```
# beadm list
BE        Active Mountpoint Space Policy Created
--        ------ ---------- ----- ------ -------
solaris   NR     /          3.80G static 2012-07-20 10:25
solaris-2 -      -          7.68M static 2012-07-19 13:44
```

SPARC 기반 시스템에 ZFS BE가 여러 개 있는 경우 `boot -L` 명령을 사용할 수 있습니다. 예를 들면 다음과 같습니다.

```
ok boot -L
Boot device: /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0,0:a  File and args: -L
1 Oracle Solaris 11.1 SPARC
2 solaris
Select environment to boot: [ 1 - 2 ]: 1

To boot the selected entry, invoke:
boot [<root-device>] -Z rpool/ROOT/solaris-2

Program terminated
ok boot -Z rpool/ROOT/solaris-2
```

위 명령으로 부트된 BE는 다음 재부트 시 활성화되지 않습니다. `boot -Z` 작업 도중 선택된 BE에서 계속 자동으로 부트하려는 경우 해당 BE를 활성화해야 합니다.

x86 기반 시스템의 ZFS 루트 파일 시스템에서 부트

Oracle Solaris 11에서 x86 시스템은 레거시 GRUB로 설치되며, 설치 프로세스 중 또는 ZFS를 자동으로 부트하기 위한 `beadm activate` 작업 중 다음 항목이 */pool-name* `/boot/grub/menu.lst` 파일에 추가됩니다.

```
title solaris
bootfs rpool/ROOT/solaris
kernel$ /platform/i86pc/kernel/amd64/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/amd64/boot_archive
title solaris-1
bootfs rpool/ROOT/solaris-1
kernel$ /platform/i86pc/kernel/amd64/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/amd64/boot_archive
```

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 파일 시스템일 경우 다음 옵션이 추가됩니다.

```
-B $ZFS-BOOTFS
```

Oracle Solaris 11.1에서 x86 기반 시스템은 GRUB2로 설치됩니다. `menu.lst` 파일이 `/rpool/boot/grub/grub.cfg` 파일로 바뀌었지만 이 파일을 수동으로 편집하지 않아야 합니다. `bootadm` 하위 명령을 사용하여 메뉴 항목을 추가, 변경 및 제거합니다.

GRUB 메뉴 항목 수정에 대한 자세한 내용은 **Oracle Solaris 11.1 시스템 부트 및 종료**를 참조하십시오.

예 4–2 x86: ZFS 파일 시스템 부트

GRUB2 시스템에서 ZFS 루트 파일 시스템으로 부트할 때는 루트 장치가 다음과 같이 지정됩니다.

```
# bootadm list-menu
the location of the boot loader configuration files is: /rpool/boot/grub
default 0
console text
timeout 30
0 Oracle Solaris 11.1
```

레거시 GRUB 시스템에서 ZFS 루트 파일 시스템으로 부트할 때는 루트 장치가 boot `-B $ZFS-BOOTFS` 매개변수로 지정됩니다. 예를 들면 다음과 같습니다.

```
title solaris
bootfs rpool/ROOT/solaris
kernel$ /platform/i86pc/kernel/amd64/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/amd64/boot_archive
title solaris-1
bootfs rpool/ROOT/solaris-1
kernel$ /platform/i86pc/kernel/amd64/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/amd64/boot_archive
```

예 4–3 x86: ZFS 루트 파일 시스템의 빠른 재부트

빠른 재부트 기능을 사용하면 x86 기반 시스템에서 몇 초 내에 재부트할 수 있습니다. 빠른 재부트 기능을 사용하면 BIOS 및 부트 로더에 의한 긴 지연 시간을 경험하지 않고도 새 커널로 재부트할 수 있습니다. 시스템을 빠르게 재부트하는 기능은 작동 중지 시간을 크게 줄이고 효율성을 향상시켜 줍니다.

`beadm activate` 명령을 통해 다른 BE로 전환하는 경우에도 `init 6` 명령을 사용해야 합니다. `reboot` 명령이 적합한 다른 시스템 작업의 경우 `reboot -f` 명령을 사용할 수 있습니다. 예를 들면 다음과 같습니다.

```
# reboot -f
```

복구를 위해 ZFS 루트 환경에서 부트

root 암호 분실 또는 유사한 문제를 복구할 수 있도록 시스템을 부트해야 할 경우 다음 절차를 사용하십시오.

▼ 복구를 위한 시스템 부트 방법

다음 절차에 따라 menu.lst 문제 또는 루트 암호 문제와 관련된 문제를 해결할 수 있습니다. 루트 풀에서 디스크를 교체해야 할 경우 115 페이지 "ZFS 루트 풀의 디스크 교체 방법(SPARC 또는 x86/VTOC)"을 참조하십시오. 전체 시스템(베어 메탈) 복구를 수행해야 할 경우 11 장, "스냅샷 아카이브 및 루트 풀 복구"를 참조하십시오.

1 적합한 부트 방법을 선택합니다.

- x86: 라이브 매체 - 설치 매체에서 부트한 다음 복구 절차에 GNOME 터미널을 사용합니다.
- SPARC: 텍스트 설치 – 설치 매체 또는 네트워크에서 부트한 다음 텍스트 설치 화면에서 3 Shell 옵션을 선택합니다.
- x86: 텍스트 설치 – GRUB 메뉴에서 Text Installer command line 부트 항목을 선택한 다음 텍스트 설치 화면에서 3 Shell 옵션을 선택합니다.
- SPARC: 자동 설치 – 다음 명령을 사용하여 셸로 종료할 수 있는 설치 메뉴에서 직접 부트합니다.

  ```
  ok boot net:dhcp
  ```

- x86: 자동 설치 – 네트워크의 설치 서버에서 부트하려면 PXE 부트가 필요합니다. GRUB 메뉴에서 Text Installer and command line 항목을 선택합니다. 그런 다음 텍스트 설치 화면에서 3 Shell 옵션을 선택합니다.

예를 들어 시스템이 부트되면 3 Shell 옵션을 선택합니다.

```
        1  Install Oracle Solaris
        2  Install Additional Drivers
        3  Shell
        4  Terminal type (currently xterm)
        5  Reboot

Please enter a number [1]: 3
To return to the main menu, exit the shell
#
```

2 부트 복구 문제를 선택합니다.

- 시스템을 단일 사용자 모드로 부트하고 /etc/passwd 파일에서 셸 항목을 수정하여 잘못된 루트 셸을 해결합니다.

  x86 시스템의 경우 선택된 부트 항목을 편집하고 -s 옵션을 추가합니다.

  예를 들어, SPARC 시스템의 경우 시스템을 종료하고 단일 모드로 부트합니다. 루트로 로그인한 후 /etc/passwd 파일을 편집하고 루트 셸 항목을 수정합니다.

  ```
  # init 0
  ok boot -s
  Boot device: /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0,0:a ...
  SunOS Release 5.11 Version 11.1 64-bit
  Copyright (c) 1983, 2012, Oracle and/or its affiliates. All rights reserved.
  Booting to milestone "milestone/single-user:default".
  ```

```
Hostname: tardis.central
Requesting System Maintenance Mode
SINGLE USER MODE

Enter user name for system maintenance (control-d to bypass): root
Enter root password (control-d to bypass): xxxx
single-user privilege assigned to root on /dev/console.
Entering System Maintenance Mode

Aug  3 15:46:21 su: 'su root' succeeded for root on /dev/console
Oracle Corporation      SunOS 5.11      11.1    October 2012
su: No shell /usr/bin/mybash.  Trying fallback shell /sbin/sh.
root@tardis.central:~# TERM =vt100; export TERM
root@tardis.central:~# vi /etc/passwd
root@tardis.central:~# <Press control-d>
logout
svc.startd: Returning to milestone all.
```

- x86 기반 시스템의 부트 오류 문제를 해결합니다.

  먼저 1단계에 나열된 부트 방법 중 하나를 사용하여 매체 또는 네트워크에서 부트해야 합니다. 그런 다음 루트 풀을 가져오고 GRUB 항목을 수정하는 등의 작업을 수행합니다.

  `bootadm list-menu` 명령을 사용하여 GRUB2 항목을 나열하고 수정할 수 있습니다. 또한 `set-menu` 하위 명령을 사용하여 부트 항목을 변경할 수도 있습니다. 자세한 내용은 bootadm(1M)을 참조하십시오.

```
x86# zpool import -f rpool
x86# bootadm list-menu
x86# bootadm set-menu default=1
x86# exit
        1  Install Oracle Solaris
        2  Install Additional Drivers
        3  Shell
        4  Terminal type (currently sun-color)
        5  Reboot

Please enter a number [1]: 5
```

  시스템이 성공적으로 부트되는지 확인합니다.

- 시스템에 로그인할 수 없게 하는 알 수 없는 루트 암호를 해결합니다.

  먼저 1단계에 나열된 부트 방법 중 하나를 사용하여 매체 또는 네트워크에서 부트해야 합니다. 그런 다음 루트 풀(`rpool`)을 가져오고 BE를 마운트하여 루트 암호 항목을 제거합니다. 이 프로세스는 SPARC 플랫폼과 x86 플랫폼에서 동일합니다.

```
# zpool import -f rpool
# beadm list
be_find_current_be: failed to find current BE name
be_find_current_be: failed to find current BE name
BE        Active Mountpoint Space  Policy Created
--        ------ ---------- -----  ------ -------
solaris   -      -          46.95M static 2012-07-20 10:25
solaris-2 R      -          3.81G  static 2012-07-19 13:44
# mkdir /a
# beadm mount solaris-2 /a
```

```
# TERM=vt100
# export TERM
# cd /a/etc
# vi shadow
<Carefully remove the unknown password>
# cd /
# beadm umount solaris-2
# halt
```

다음 단계로 이동하여 루트 암호를 설정합니다.

3 단일 사용자 모드로 부트하고 암호를 설정하여 루트 암호를 설정합니다.

이 단계에서는 이전 단계에서 알 수 없는 루트 암호를 제거한 것으로 간주합니다.

x86 기반 시스템의 경우 선택된 부트 항목을 편집하고 -s 옵션을 추가합니다.

SPARC 기반 시스템에서는 시스템을 단일 사용자 모드로 부트하고 루트로 로그인한 다음 루트 암호를 설정합니다. 예를 들면 다음과 같습니다.

```
ok boot -s
Boot device: /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0,0:a ...
SunOS Release 5.11 Version 11.1 64-bit
Copyright (c) 1983, 2012, Oracle and/or its affiliates. All rights reserved
Booting to milestone "milestone/single-user:default".

Enter user name for system maintenance (control-d to bypass): root
Enter root password (control-d to bypass): <Press return>
single-user privilege assigned to root on /dev/console.
Entering System Maintenance Mode

Jul 20 14:09:59 su: 'su root' succeeded for root on /dev/console
Oracle Corporation      SunOS 5.11      11.1    October 2012
root@tardis.central:~# passwd -r files root
New Password: xxxxxx
Re-enter new Password: xxxxxx
passwd: password successfully changed for root
root@tardis.central:~# <Press control-d>
logout
svc.startd: Returning to milestone all.
```

◆ ◆ ◆ 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.

- 133 페이지 "ZFS 파일 시스템 관리(개요)"
- 134 페이지 "ZFS 파일 시스템 만들기, 삭제 및 이름 바꾸기"
- 137 페이지 "ZFS 등록 정보 소개"
- 157 페이지 "ZFS 파일 시스템 정보 질의"
- 160 페이지 "ZFS 등록 정보 관리"
- 165 페이지 "ZFS 파일 시스템 마운트"
- 170 페이지 "ZFS 파일 시스템 공유 및 공유 해제"
- 179 페이지 "ZFS 쿼터 및 예약 설정"
- 185 페이지 "ZFS 파일 시스템 암호화"
- 192 페이지 "ZFS 파일 시스템 마이그레이션"
- 195 페이지 "ZFS 파일 시스템 업그레이드"

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 257 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.

- 134 페이지 "ZFS 파일 시스템 만들기"
- 135 페이지 "ZFS 파일 시스템 삭제"
- 136 페이지 "ZFS 파일 시스템 이름 바꾸기"

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 31 페이지 "ZFS 구성 요소 명명 요구 사항"의 조건을 충족해야 합니다.

파일 시스템을 만들 때 ZFS 파일 시스템의 암호화를 사용으로 설정해야 합니다. ZFS 파일 시스템 암호화에 대한 자세한 내용은 185 페이지 "ZFS 파일 시스템 암호화"를 참조하십시오.

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 165 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 137 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 166 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 179 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 200 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.

- 149 페이지 "ZFS 읽기 전용 고유 등록 정보"
- 150 페이지 "설정 가능한 ZFS 고유 등록 정보"
- 156 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 156 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`	160 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 156 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 223 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 223 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 32 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. noauto 옵션이 설정된 경우 파일 시스템을 명시적으로만 마운트 및 마운트 해제할 수 있습니다. 파일 시스템을 만들거나 가져올 때 파일 시스템이 자동으로 마운트되지 않으며, zfs mount -a 명령을 사용하여 마운트되거나 zfs unmount -a 명령을 사용하여 마운트 해제되지 않습니다. 자세한 내용은 152 페이지 "canmount 등록 정보"를 참조하십시오.

표 5–1 ZFS 고유 등록 정보 설명 *(계속)*

등록 정보 이름	유형	기본값	설명
casesensitivity	문자열	mixed	이 등록 정보는 파일 시스템에서 사용되는 파일 이름 일치 알고리즘이 casesensitive, caseinsensitive 또는 두 일치 스타일의 조합 허용(mixed)인지를 나타냅니다. 일반적으로 UNIX 및 POSIX 파일 시스템은 대소문자 구분 파일 이름을 사용합니다. 이 등록 정보 값이 mixed인 경우 파일 시스템에서 대소문자 구분 및 대소문자 무시 일치 동작 요청을 모두 지원할 수 있습니다. 현재 혼합 동작을 지원하는 파일 시스템의 대소문자 무시 일치 동작은 Oracle Solaris SMB 서버 제품으로 제한됩니다. mixed 값 사용에 대한 자세한 내용은 152 페이지 “casesensitivity 등록 정보”를 참조하십시오. casesensitivity 등록 정보 설정에 관계없이 파일 시스템은 파일을 만들기 위해 지정된 이름의 대소문자를 유지합니다. 파일 시스템이 생성된 후에는 이 등록 정보를 변경할 수 없습니다.
체크섬	문자열	on	데이터 무결성을 확인하기 위해 사용되는 체크섬을 제어합니다. 기본값은 적합한 알고리즘(현재까지 fletcher4)을 자동으로 선택하는 on입니다. 값은 on, off, fletcher2, fletcher4, sha256 및 sha256+mac입니다. 값이 off면 사용자 데이터에 대한 무결성 검사가 사용 안함으로 설정됩니다. off 값은 권장되지 않습니다.
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.

표 5–1 ZFS 고유 등록 정보 설명 (계속)

등록 정보 이름	유형	기본값	설명
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.
dedup	문자열	off	ZFS 파일 시스템에서 중복 데이터를 제거하는 기능을 제어합니다. 가능한 값은 on, off, verify 및 sha256[,verify]입니다. 중복 제거의 기본 체크섬은 sha256입니다. 자세한 내용은 154 페이지 “dedup 등록 정보”를 참조하십시오.
devices	부울	on	파일 시스템의 장치 파일을 열 수 있는지 여부를 제어합니다.
encryption	부울	off	파일 시스템의 암호화 여부를 제어합니다. 암호화된 파일 시스템은 데이터가 인코딩되며 파일 시스템 소유자가 데이터에 액세스하기 위해 키가 필요함을 의미합니다.
exec	부울	on	파일 시스템의 프로그램 실행을 허용할지 여부를 제어합니다. 또한 off로 설정된 경우, PROT_EXEC를 사용한 mmap(2) 호출은 허용되지 않습니다.
keychangedate	문자열	none	지정된 파일 시스템에서 zfs key -c 작업으로 래핑 키가 마지막으로 변경된 날짜를 식별합니다. 키 변경 작업이 수행되지 않았으면 이 읽기 전용 등록 정보의 값이 파일 시스템을 만든 날짜와 동일합니다.
keysource	문자열	none	파일 시스템 키를 래핑하는 키의 형식과 위치를 식별합니다. 유효한 등록 정보 값은 raw, hex, passphrase,prompt 또는 *file*입니다. zfs key - l 명령을 사용하여 파일 시스템을 만들고, 마운트 또는 로드할 때 키가 있어야 합니다. 새 파일 시스템에 대해 암호화가 사용으로 설정된 경우 기본 keysource는 passphrase,prompt입니다.

표 5–1 ZFS 고유 등록 정보 설명 *(계속)*

등록 정보 이름	유형	기본값	설명
keystatus	문자열	none	파일 시스템의 암호 키 상태를 식별하는 읽기 전용 등록 정보입니다. 파일 시스템의 키 가용성은 available 또는 unavailable로 표시됩니다. 암호화가 사용으로 설정되지 않은 파일 시스템의 경우 none이 표시됩니다.
logbias	문자열	latency	ZFS에서 이 파일 시스템에 대한 동기식 요청을 최적화하는 방법을 제어합니다. logbias가 latency로 설정되어 있으면 ZFS가 풀의 별도 로그 장치(있는 경우)를 사용하여 짧은 대기 시간으로 요청을 처리합니다. logbias를 throughput으로 설정하면 ZFS는 풀에 있는 별도의 로그 장치를 사용하지 않습니다. 대신 ZFS는 전역 풀 처리량 및 효율적인 리소스 사용을 위해 동기식 작업을 최적화합니다. 기본값은 latency입니다.
mlslabel	문자열	없음	다중 레벨 파일 시스템에서 mlslabel 등록 정보의 동작에 대한 설명은 multilevel 등록 정보를 참조하십시오. 다음 mlslabel 설명은 비다중 레벨 파일 시스템에 적용됩니다. Trusted Extensions 영역에 파일 시스템을 마운트할 수 있는지 여부를 결정하는 민감도 레이블을 제공합니다. 레이블 있는 파일 시스템이 레이블 있는 영역과 일치하는 경우 레이블 있는 영역에서 파일 시스템을 마운트하고 액세스할 수 있습니다. 기본값은 none입니다. 이 등록 정보는 Trusted Extensions가 사용으로 설정되고 적절한 권한이 있어야만 수정할 수 있습니다.
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.
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 165 페이지 “ZFS 마운트 지점 관리”를 참조하십시오.

표 5–1 ZFS 고유 등록 정보 설명 (계속)

등록 정보 이름	유형	기본값	설명
multilevel	부울	off	이 등록 정보는 Trusted Extensions가 사용으로 설정된 시스템에서만 사용할 수 있습니다. 기본값은 off입니다. 다중 레벨 파일 시스템의 객체는 자동으로 생성되는 명시적인 민감도 레이블 속성으로 레이블이 개별적으로 지정됩니다. 객체는 이 레이블 속성을 변경하거나, setlabel을 사용하거나, setflabel 인터페이스를 사용하여 해당 위치에서 레이블을 다시 지정할 수 있습니다. 루트 파일 시스템, Oracle Solaris 영역 파일 시스템 또는 패키지화된 Solaris 코드를 포함하는 파일 시스템은 다중 레벨이 아니어야 합니다. 다중 레벨 파일 시스템의 mlslabel 등록 정보에는 차이가 있습니다. mlslabel 값은 파일 시스템에서 객체에 대해 가능한 한 가장 높은 레이블을 정의합니다. mlslabel 값보다 높은 레이블에서는 파일을 만들거나 파일의 레이블을 다시 지정할 수 없습니다. mlslabel 값 기반의 마운트 정책은 다중 레벨 파일 시스템에 적용되지 않습니다. 다중 레벨 파일 시스템의 경우 파일 시스템을 만들 때 mlslabel 등록 정보를 명시적으로 설정할 수 있습니다. 그렇지 않으면 ADMIN_HIGH의 기본 mlslabel 등록 정보가 자동으로 만들어집니다. 다중 레벨 파일 시스템을 만든 후 mlslabel 등록 정보를 변경할 수 있지만 하위 레이블로 설정하거나, none으로 설정하거나, 제거할 수는 없습니다.
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.

표 5–1 ZFS 고유 등록 정보 설명 *(계속)*

등록 정보 이름	유형	기본값	설명
nbmand	부울	off	nbmand(비블록화 필수) 잠금을 사용하여 파일 시스템을 마운트해야 하는지 여부를 제어합니다. 이 등록 정보는 SMB 클라이언트에만 해당합니다. 이 등록 정보의 변경 사항은 파일 시스템의 마운트를 해제하고 재마운트하는 경우에만 적용됩니다.
normalization	문자열	없음	이 등록 정보는 파일 시스템에서 두 파일 이름을 비교할 때마다 파일 이름의 유니코드 정규화를 수행할지 여부 및 사용해야 하는 정규화 알고리즘을 나타냅니다. 파일 이름은 항상 수정되지 않은 상태로 저장되며, 비교 프로세스의 일부로 이름이 정규화됩니다. 이 등록 정보를 none이 아닌 올바른 값으로 설정하고 utf8only 등록 정보를 지정하지 않은 경우 utf8only 등록 정보가 자동으로 on으로 설정됩니다. normalization 등록 정보의 기본값은 none입니다. 파일 시스템이 생성된 후에는 이 등록 정보를 변경할 수 없습니다.
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 180 페이지 “ZFS 파일 시스템에 대한 쿼터 설정”을 참조하십시오.
rekeydate	문자열	해당 없음	이 파일 시스템의 zfs key -K 또는 zfs clone -K 작업에서 마지막 데이터 암호 키의 변경 날짜를 나타내는 읽기 전용 등록 정보입니다. rekey 작업을 수행하지 않은 경우 이 등록 정보 값은 creation 날짜와 같습니다.

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 155 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 183 페이지 "ZFS 파일 시스템에 대한 예약 설정"을 참조하십시오.
rstchown	부울	On	파일 시스템 소유자가 파일 소유권 변경을 허가할 수 있는지 여부를 나타냅니다. 기본값은 chown 작업을 제한하는 것입니다. rstchown이 off로 설정된 경우 사용자가 chown 작업에 대해 PRIV_FILE_CHOWN_SELF 권한을 갖습니다.
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.
shadow	문자열	None	ZFS 파일 시스템을 파일 시스템의 **그림자**로 식별합니다(파일 시스템이 *URI*로 기술됨). URI로 식별되는 파일 시스템에서 설정된 이 등록 정보를 사용하여 데이터가 그림자 파일 시스템으로 마이그레이션됩니다. 마이그레이션을 완료하기 위해서는 마이그레이션할 파일 시스템이 읽기 전용이어야 합니다.

표 5–1 ZFS 고유 등록 정보 설명 (계속)

등록 정보 이름	유형	기본값	설명
share.nfs	문자열	off	ZFS 파일 시스템의 NFS 공유를 만들고 게시할지 여부 및 사용할 옵션을 제어합니다. zfs share 및 zfs unshare 명령을 사용하여 NFS 공유를 게시 및 게시 해제할 수도 있습니다. zfs share 명령을 사용하여 NFS 공유를 게시하려면 NFS 공유 등록 정보도 설정되어 있어야 합니다. NFS 공유 등록 정보 설정에 대한 자세한 내용은 170 페이지 “ZFS 파일 시스템 공유 및 공유 해제”를 참조하십시오. ZFS 파일 시스템 공유에 대한 자세한 내용은 170 페이지 “ZFS 파일 시스템 공유 및 공유 해제”를 참조하십시오.
share.smb	문자열	off	ZFS 파일 시스템의 SMB 공유를 만들고 게시할지 여부 및 사용할 옵션을 제어합니다. zfs share 및 zfs unshare 명령을 사용하여 SMB 공유를 게시 및 게시 해제할 수도 있습니다. zfs share 명령을 사용하여 SMB 공유를 게시하려면 SMB 공유 등록 정보도 설정되어 있어야 합니다. SMB 공유 등록 정보 설정에 대한 자세한 내용은 170 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 197 페이지 “ZFS 스냅샷 개요”를 참조하십시오.

표 5–1 ZFS 고유 등록 정보 설명 *(계속)*

등록 정보 이름	유형	기본값	설명
sync	문자열	standard	파일 시스템 트랜잭션의 동기식 동작을 결정합니다. 가능한 값은 다음과 같습니다. ■ 기본값인 standard는 fsync, O_DSYNC, O_SYNC 등의 동기식 파일 시스템 트랜잭션이 계획 로그에 기록됨을 의미합니다. ■ always는 **각** 파일 시스템 트랜잭션이 반환되는 시스템 호출에 의해 안정된 저장소에 기록되고 비워지도록 합니다. 이 값을 설정하면 성능이 저하됩니다. ■ disabled는 동기식 요청이 사용 안함으로 설정됨을 의미합니다. 파일 시스템 트랜잭션은 몇 초 후일 수 있는 다음 트랜잭션 그룹 커밋 시에만 안정된 저장소로 커밋됩니다. 이 값을 설정하면 성능이 최적화되며 풀이 손상될 위험이 없습니다. **주의 –** 이 disabled 값은 ZFS에서 데이터베이스 또는 NFS 작업과 같은 응용 프로그램의 동기식 트랜잭션 요구를 무시하기 때문에 매우 위험합니다. 현재 활성 루트나 /var 파일 시스템에 이 값을 설정하면 예기치 않은 동작, 응용 프로그램 데이터 손실 또는 재생 공격에 대한 취약성 증가가 초래될 수 있습니다. 연관된 모든 위험을 완전히 이해하는 경우에만 이 값을 사용해야 합니다.
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 150 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.

표 5–1 ZFS 고유 등록 정보 설명 (계속)

등록 정보 이름	유형	기본값	설명
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.
utf8only	부울	Off	이 등록 정보는 파일 시스템에서 UTF-8 문자 코드 세트에 없는 문자가 포함된 파일 이름을 거부할지 여부를 나타냅니다. 이 등록 정보가 명시적으로 off로 설정된 경우 normalization 등록 정보를 명시적으로 설정하지 않거나 none으로 설정해야 합니다. utf8only 등록 정보의 기본값은 off입니다. 파일 시스템이 생성된 후에는 이 등록 정보를 변경할 수 없습니다.
volsize	숫자	해당 없음	볼륨에 대해 볼륨의 논리적 크기를 지정합니다. 자세한 내용은 156 페이지 “volsize 등록 정보”를 참조하십시오.
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.

표 5–1 ZFS 고유 등록 정보 설명 *(계속)*

등록 정보 이름	유형	기본값	설명
vscan	부울	Off	파일을 열고 닫을 때 정규 파일에서 바이러스를 검사할지 여부를 제어합니다. 타사 바이러스 검사 소프트웨어가 있는 경우 바이러스를 검사하려면 이 등록 정보를 사용으로 설정하는 것은 물론 바이러스 검사 서비스도 사용으로 설정해야 합니다. 기본값은 off입니다.
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 260 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
- keystatus
- mounted
- origin
- referenced
- rekeydate
- type
- used

  자세한 내용은 150 페이지 “used 등록 정보”를 참조하십시오.
- usedbychildren

- usedbydataset
- usedbyrefreservation
- usedbysnapshots

used, referenced 및 available 등록 정보를 포함한 디스크 공간 계산에 대한 자세한 내용은 32 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 33 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 160 페이지 "ZFS 등록 정보 설정"에 설명된 대로 zfs set 명령을 사용하거나 134 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 179 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 223 페이지 "ACL 등록 정보"를 참조하십시오.
- atime
- canmount
- casesensitivity
- **체크섬**
- compression
- copies
- devices
- dedup
- encryption
- exec
- keysource
- logbias
- mlslabel
- mountpoint
- nbmand
- normalization
- primarycache
- quota
- readonly
- recordsize
  자세한 내용은 155 페이지 "recordsize 등록 정보"를 참조하십시오.
- refquota
- refreservation
- reservation
- rstchown
- secondarycache
- share.smb
- share.nfs

- setuid
- snapdir
- version
- vscan
- utf8only
- volsize

  자세한 내용은 156 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
```

canmount 등록 정보를 noauto로 설정하면 파일 시스템을 자동이 아닌 명시적인 방식으로만 마운트할 수 있습니다.

casesensitivity 등록 정보

이 등록 정보는 파일 시스템에서 사용되는 파일 이름 일치 알고리즘이 casesensitive, caseinsensitive 또는 두 일치 스타일의 조합 허용(mixed)인지를 나타냅니다.

대소문자 무시 일치 요청이 **혼합된** 민감도 파일 시스템으로 구성된 경우 동작은 일반적으로 대소문자 무시 파일 시스템에 예상되는 동작과 같습니다. 차이점은 혼합된 민감도 파일 시스템에는 대소문자 구분 관점에서 고유하지만 대소문자 무시 관점에서는 고유하지 않은 여러 이름이 있는 디렉토리가 포함될 수 있다는 것입니다.

예를 들어, 디렉토리에 `foo`, `Foo` 및 `FOO` 파일이 포함될 수 있습니다. `foo`의 가능한 모든 형태에 대해 대소문자를 무시하고 일치시키도록 요청할 경우(예: `foo`, `FOO`, `Foo`, `fOo` 등) 일치 알고리즘에서 기존 파일 3개 중 하나를 일치 항목으로 선택합니다. 알고리즘이 일치 항목으로 선택하는 파일이 정확하게 보장되지는 않지만 `foo` 형태의 일치 항목으로 항상 동일한 파일이 선택됩니다. `foo`, `FOO`, `foO`, `Foo` 등의 대소문자 무시 일치 항목으로 선택되는 파일은 디렉토리를 변경하지 않는 한 항상 동일합니다.

`utf8only`, `normalization` 및 `casesensitivity` 등록 정보는 ZFS 위임 관리를 사용하여 권한 없는 사용자에게 지정할 수 있는 새 권한도 제공합니다. 자세한 내용은 246 페이지 “ZFS 권한 위임”을 참조하십시오.

`copies` 등록 정보

안정성을 확보하기 위해 ZFS 파일 시스템 메타 데이터는 가능한 한 여러 디스크에 여러 번 자동으로 저장됩니다. 이 기능은 **복제 블록***(ditto blocks)*으로 알려져 있습니다.

이 릴리스에서는 사용자 데이터도 `zfs set copies` 명령을 사용하여 파일 시스템당 여러 개의 사본을 저장할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs set copies=2 users/home
# zfs get copies users/home
NAME        PROPERTY  VALUE       SOURCE
users/home  copies    2           local
```

사용 가능한 값은 1, 2 또는 3입니다. 기본값은 1입니다. 이러한 사본은 미러링된 구성 또는 RAID-Z 구성에서와 같이 모든 풀 레벨의 중복성에 대해 추가로 설정됩니다.

ZFS 사용자 데이터의 사본을 여러 개 저장했을 때 얻을 수 있는 이점은 다음과 같습니다.

- 모든 ZFS 구성에서 매체 결함(**비트 로트***(bit rot)*라고도 부름)과 같은 복구할 수 없는 블록 읽기 결함을 복구할 수 있도록 지원하여 데이터 보존 기능이 강화됩니다.
- 단일 디스크만 사용할 수 있는 경우에도 데이터 보호를 제공합니다.
- 저장소 풀의 기능을 넘어 파일 시스템 단위로 데이터 보호 정책을 선택할 수 있습니다.

주 – 저장소 풀의 복제 블록(ditto block) 할당에 따라 여러 복사본이 단일 디스크에 배치될 수도 있습니다. 이후 전체 디스크 오류가 발생하면 모든 복제 블록(ditto block)을 사용하지 못하게 될 수 있습니다.

복제 블록(ditto block)은 중복되지 않은 풀을 실수로 만드는 경우 그리고 데이터 보존 정책을 설정해야 할 경우에 사용할 수 있습니다.

dedup 등록 정보

dedup 등록 정보는 파일 시스템에서 중복 데이터를 제거할지 여부를 제어합니다. 파일 시스템의 dedup 등록 정보가 사용으로 설정된 경우 중복 데이터 블록이 동기적으로 제거됩니다. 그 결과, 고유한 데이터만 저장되고 공통 구성 요소는 파일 간에 공유됩니다.

다음 고려 사항을 검토할 때까지 운용 시스템에 있는 파일 시스템에서 dedup 등록 정보를 사용으로 설정하지 마십시오.

1. 중복 제거를 통해 공간 절약이 가능한 데이터인지 확인합니다. 데이터가 중복 제거 가능이 아닌 경우 중복 제거를 사용으로 설정해도 아무 의미가 없습니다. 예를 들면 다음과 같습니다.

```
# zdb -S tank
Simulated DDT histogram:
bucket              allocated                       referenced
______   ______________________________   ______________________________
refcnt   blocks   LSIZE   PSIZE   DSIZE   blocks   LSIZE   PSIZE   DSIZE
------   ------   -----   -----   -----   ------   -----   -----   -----
     1    2.27M    239G    188G    194G    2.27M    239G    188G    194G
     2     327K   34.3G   27.8G   28.1G     698K   73.3G   59.2G   59.9G
     4    30.1K   2.91G   2.10G   2.11G     152K   14.9G   10.6G   10.6G
     8    7.73K    691M    529M    529M    74.5K   6.25G   4.79G   4.80G
    16      673   43.7M   25.8M   25.9M    13.1K    822M    492M    494M
    32      197   12.3M   7.02M   7.03M    7.66K    480M    269M    270M
    64       47   1.27M    626K    626K    3.86K    103M   51.2M   51.2M
   128       22    908K    250K    251K    3.71K    150M   40.3M   40.3M
   256        7    302K     48K   53.7K    2.27K   88.6M   17.3M   19.5M
   512        4    131K   7.50K   7.75K    2.74K    102M   5.62M   5.79M
    2K        1      2K      2K      2K    3.23K   6.47M   6.47M   6.47M
    8K        1    128K      5K      5K    13.9K   1.74G   69.5M   69.5M
 Total    2.63M    277G    218G    225G    3.22M    337G    263G    270G

dedup = 1.20, compress = 1.28, copies = 1.03, dedup * compress / copies = 1.50
```

   예상 중복 제거 비율이 2보다 클 경우 중복 제거를 통한 공간 절약 효과가 있습니다.

   위 예에서는 중복 제거 비율이 2보다 작기 때문에 중복 제거를 사용으로 설정하지 않는 것이 좋습니다.

2. 시스템 메모리가 중복 제거를 지원할 정도로 충분한지 확인합니다.
   - 코어 내 중복 제거 테이블 항목은 각각 약 320바이트입니다.
   - 할당된 블록 수에 320을 곱합니다. 예를 들면 다음과 같습니다.

```
in-core DDT size = 2.63M x 320 = 841.60M
```

3. 중복 제거 테이블이 메모리에 적합할 때 중복 제거 성능이 가장 좋습니다. 중복 제거 테이블을 디스크에 기록해야 할 경우 성능이 저하됩니다. 예를 들어, 중복 제거가 사용으로 설정된 대형 파일 시스템을 제거할 경우 시스템이 위에 설명된 메모리 요구 사항을 충족하지 않으면 시스템 성능이 크게 저하됩니다.

dedup가 사용으로 설정된 경우 dedup 체크섬 알고리즘이 checksum 등록 정보를 대체합니다. 등록 정보 값을 verify로 설정하는 것은 sha256,verify를 지정하는 것과

같습니다. 등록 정보가 verify로 설정되고 두 블록에 동일한 서명이 있는 경우 ZFS는 기존 블록과 바이트 단위 비교를 수행하여 컨텐츠가 동일한지 확인합니다.

파일 시스템별로 이 등록 정보를 사용으로 설정할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs set dedup=on tank/home
```

zfs get 명령을 사용하여 dedup 등록 정보가 설정되었는지 확인할 수 있습니다.

중복 제거는 파일 시스템 등록 정보로 설정되지만 범위가 풀 전체입니다. 예를 들어, 중복 제거 비율을 식별할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool list tank
NAME    SIZE  ALLOC   FREE    CAP  DEDUP  HEALTH  ALTROOT
rpool   136G  55.2G  80.8G    40%  2.30x  ONLINE  -
```

DEDUP 열은 발생한 중복 제거 양을 나타냅니다. dedup 등록 정보가 어떤 파일 시스템에서도 사용으로 설정되지 않았거나 dedup 등록 정보가 파일 시스템에서 방금 사용으로 설정된 경우 DEDUP 비율은 1.00x입니다.

zpool get 명령을 사용하여 dedupratio 등록 정보 값을 확인할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zpool get dedupratio export
NAME   PROPERTY    VALUE  SOURCE
rpool  dedupratio  3.00x  -
```

이 풀 등록 정보는 이 풀에서 발생한 데이터 중복 제거 양을 보여줍니다.

encryption 등록 정보

encryption 등록 정보를 사용하여 ZFS 파일 시스템을 암호화할 수 있습니다. 자세한 내용은 185 페이지 “ZFS 파일 시스템 암호화”를 참조하십시오.

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.

share.smb 등록 정보

이 등록 정보는 Oracle Solaris SMB 서비스와 ZFS 파일 시스템 공유를 사용으로 설정하고 사용할 옵션을 식별합니다.

등록 정보가 off에서 on으로 변경되면 이 등록 정보를 상속하는 모든 공유가 현재 옵션으로 다시 공유됩니다. 등록 정보가 off로 설정된 경우 이 등록 정보를 상속하는 공유가 해제됩니다.share.smb 등록 정보 사용 예를 보려면 170 페이지 "ZFS 파일 시스템 공유 및 공유 해제"를 참조하십시오.

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 -V로 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 volsize에 대한 변경 사항이 예약에 반영되지 않습니다.

볼륨 사용에 대한 자세한 내용은 257 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 137 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 집합 이름을 제공하여 특정 데이터 집합을 표시할 수도 있습니다. `-r` 옵션을 사용하면 해당 데이터 집합의 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 137 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 예에서는 `zfs list`를 사용하여 `share.nfs` 및 `mountpoint` 등록 정보 값과 함께 데이터 세트 이름을 표시합니다.

```
# zfs list -r -o name,share.nfs,mountpoint users/home
NAME                          NFS       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
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.

- 160 페이지 “ZFS 등록 정보 설정”
- 161 페이지 “ZFS 등록 정보 상속”
- 162 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 150 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
```

비숫자 등록 정보 값은 대소문자를 구분하며 mountpoint를 제외하고 소문자여야 합니다. 이 등록 정보 값은 대소문자가 혼합될 수 있습니다.

zfs set 명령에 대한 자세한 내용은 zfs(1M)을 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 157 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 집합 등록 정보를 검색할 수 있습니다. 다음 예제에서는 데이터 집합에서 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 set`를 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 168 페이지 "임시 마운트 등록 정보 사용"을 참조하십시오.
- (none)	이 등록 정보는 읽기 전용입니다. 해당 값은 ZFS에 의해 만들어집니다.

특수 키워드인 `all`을 사용하여 모든 데이터 세트 등록 정보 값을 검색할 수 있습니다. 다음 예제에서는 `all` 키워드가 사용됩니다.

```
# zfs get all tank/home
NAME       PROPERTY              VALUE                  SOURCE
tank/home  aclinherit            restricted             default
tank/home  aclmode               discard                default
tank/home  atime                 on                     default
tank/home  available             66.9G                  -
tank/home  canmount              on                     default
tank/home  casesensitivity       mixed                  -
tank/home  checksum              on                     default
```

```
tank/home  compression          off                  default
tank/home  compressratio        1.00x                -
tank/home  copies               1                    default
tank/home  creation             Fri May 11 10:58 2012  -
tank/home  dedup                off                  default
tank/home  devices              on                   default
tank/home  encryption           off                  -
tank/home  exec                 on                   default
tank/home  keysource            none                 default
tank/home  keystatus            none                 -
tank/home  logbias              latency              default
tank/home  mlslabel             none                 -
tank/home  mounted              yes                  -
tank/home  mountpoint           /tank/home           default
tank/home  multilevel           off                  -
tank/home  nbmand               off                  default
tank/home  normalization        none                 -
tank/home  primarycache         all                  default
tank/home  quota                none                 default
tank/home  readonly             off                  default
tank/home  recordsize           128K                 default
tank/home  referenced           43K                  -
tank/home  refquota             none                 default
tank/home  refreservation       none                 default
tank/home  rekeydate            -                    default
tank/home  reservation          none                 default
tank/home  rstchown             on                   default
tank/home  secondarycache       all                  default
tank/home  setuid               on                   default
tank/home  shadow               none                 -
tank/home  share.*              ...                  local
tank/home  snapdir              hidden               default
tank/home  sync                 standard             default
tank/home  type                 filesystem           -
tank/home  used                 8.54M                -
tank/home  usedbychildren       8.49M                -
tank/home  usedbydataset        43K                  -
tank/home  usedbyrefreservation 0                    -
tank/home  usedbysnapshots      0                    -
tank/home  utf8only             off                  -
tank/home  version              6                    -
tank/home  vscan                off                  default
tank/home  xattr                on                   default
tank/home  zoned                off                  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)을 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 137 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.

- 165 페이지 “ZFS 마운트 지점 관리”
- 167 페이지 “ZFS 파일 시스템 마운트”
- 168 페이지 “임시 마운트 등록 정보 사용”
- 169 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 50 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 152 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 166 페이지 “레거시 마운트 지점”을 참조하십시오.

자동 마운트 지점

- mountpoint 등록 정보를 legacy 또는 none에서 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 legacy가 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 27 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
xattr	xattr/noaxttr

마운트 옵션 nosuid는 nodevices,nosetuid에 대한 별칭입니다.

NFSv4 미러 마운트 기능을 사용하여 NFS 마운트된 ZFS 홈 디렉토리를 효율적으로 관리할 수 있습니다.

NFS 서버에 파일 시스템을 만들면 NFS 클라이언트가 상위 파일 시스템의 기존 마운트 내에서 새로 만든 파일 시스템을 자동으로 검색할 수 있습니다.

예를 들어, 서버 neo에서 이미 tank 파일 시스템을 공유하고 클라이언트 zee에 마운트된 파일 시스템이 있는 경우 서버에 파일 시스템을 만들면 /tank/baz가 클라이언트에 자동으로 표시됩니다.

```
zee# mount neo:/tank /mnt
zee# ls /mnt
baa    bar

neo# zfs create tank/baz

zee% ls /mnt
baa    bar    baz
zee% ls /mnt/baz
file1    file2
```

임시 마운트 등록 정보 사용

앞의 섹션에서 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 포인트에 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

Oracle Solaris 11.1 릴리스는 ZFS 등록 정보 상속을 활용하여 ZFS 공유 관리를 간소화합니다. 새로운 공유 구문은 풀 버전 34를 실행하는 풀에서 사용으로 설정됩니다.

파일 시스템별로 다중 공유를 정의할 수 있습니다. 공유 이름은 각 공유를 고유하게 식별합니다. 파일 시스템에서 특정 경로를 공유하는 데 사용되는 등록 정보를 정의할 수 있습니다. 기본적으로 모든 파일 시스템이 공유 해제됩니다. 일반적으로 공유를 만들기 전까지는 NFS 서버 서비스가 시작되지 않습니다. 유효한 공유를 만들면 NFS 서비스가 자동으로 시작됩니다. ZFS 파일 시스템의 `mountpoint` 등록 정보가 legacy로 설정된 경우 레거시 `share` 명령을 사용해서만 파일 시스템을 공유할 수 있습니다.

- `share.nfs` 등록 정보는 이전 릴리스의 `sharenfs` 등록 정보를 대신해서 NFS 공유를 정의하고 게시합니다.
- `share.smb` 등록 정보는 이전 릴리스의 `sharesmb` 등록 정보를 대신해서 SMB 공유를 정의하고 게시합니다.
- `sharenfs` 등록 정보와 `sharesmb` 등록 정보는 모두 `share.nfs` 등록 정보 및 `sharenfs` 등록 정보에 대한 별칭입니다.
- `/etc/dfs/dfstab` 파일은 더 이상 부트 시 파일 시스템을 공유하는 데 사용되지 않습니다. 이 등록 정보를 설정하면 파일 시스템이 자동으로 공유됩니다. SMF는 시스템을 재부트할 때 파일 시스템이 자동으로 공유되도록 ZFS 또는 UFS 공유 정보를 관리합니다. 이 기능은 `sharenfs` 또는 `sharesmb` 등록 정보가 off로 설정되지 않은 모든 파일 시스템이 부트 시 공유되도록 합니다.
- `sharemgr` 인터페이스는 더 이상 사용할 수 없습니다. 레거시 `share` 명령은 레거시 공유를 만드는 데 계속 사용할 수 있습니다. 다음 예제를 참조하십시오.
- `share -a` 명령은 파일 시스템 공유가 지속되도록 이전 `share -ap` 명령과 유사하게 작동합니다. `share -p` 옵션은 더 이상 사용할 수 없습니다.

예를 들어, `tank/home` 파일 시스템을 공유할 경우 다음과 비슷한 구문을 사용합니다.

```
# zfs set share.nfs=on tank/home
```

추가 등록 정보 값을 지정하거나 기존 파일 시스템 공유에서 기존 등록 정보 값을 수정할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs set share.nfs.nosuid=on tank/home/userA
```

`share.nfs` 등록 정보가 `tank/home` 파일 시스템으로 설정된 이전 예에서 `share.nfs` 등록 정보 값은 모든 종속 파일 시스템으로 상속됩니다. 예를 들면 다음과 같습니다.

```
# zfs create tank/home/userA
# zfs create tank/home/userB
```

레거시 ZFS 공유 구문

Oracle Solaris 11 구문도 계속 지원되므로 두 단계로 파일 시스템을 공유할 수 있습니다. 이 구문은 모든 풀 버전에서 지원됩니다.

- 먼저 zfs set share 명령을 사용하여 ZFS 파일 시스템의 NFS 또는 SMB 공유를 만듭니다.

```
# zfs create rpool/fs1
# zfs set share=name=fs1,path=/rpool/fs1,prot=nfs rpool/fs1
name=fs1,path=/rpool/fs1,prot=nfs
```

- 그런 후 sharenfs 또는 sharesmb 등록 정보를 on으로 설정하여 공유를 게시합니다. 예를 들면 다음과 같습니다.

```
# zfs set sharenfs=on rpool/fs1
# grep fs1 /etc/dfs/sharetab
/rpool/fs1      fs1     nfs     sec=sys,rw
```

파일 시스템 공유는 레거시 zfs get share 명령을 사용하여 표시할 수 있습니다.

```
# zfs get share rpool/fs1
NAME       PROPERTY  VALUE  SOURCE
rpool/fs1  share     name=fs1,path=/rpool/fs1,prot=nfs  local
```

또한 Oracle Solaris 10 릴리스의 구문과 비슷한 파일 시스템을 공유하는 share 명령도 파일 시스템 내의 모든 디렉토리를 공유하도록 지원됩니다. 예를 들어 ZFS 파일 시스템을 공유하려면 다음 구문을 사용합니다.

```
# share -F nfs /tank/zfsfs
# grep zfsfs /etc/dfs/sharetab
/tank/zfsfs     tank_zfsfs      nfs     sec=sys,rw
```

위 구문은 UFS 파일 시스템 공유 구문과 동일합니다.

```
# share -F nfs /ufsfs
# grep ufsfs /etc/dfs/sharetab
/ufsfs          -               nfs     rw
/tank/zfsfs     tank_zfsfs      nfs     rw
```

새 ZFS 공유 구문

zfs set 명령은 NFS 또는 SMB 프로토콜을 통해 ZFS 파일 시스템을 공유하고 게시하는 데 사용됩니다. 또는 파일 시스템을 만들 때 share.nfs 또는 share.smb 등록 정보를 설정할 수 있습니다.

예를 들어 tank/sales 파일 시스템이 생성되고 공유되었습니다. 기본 공유 권한은 모든 사용자에 대해 읽기-쓰기입니다. share.nfs 등록 정보가 종속된 파일 시스템으로 상속되고 tank/sales/log 파일 시스템이 읽기 전용 액세스로 설정되기 때문에 종속된 tank/sales/logs 파일 시스템도 자동으로 공유됩니다.

```
# zfs create -o share.nfs=on tank/sales
# zfs create -o share.nfs.ro=\* tank/sales/logs
# zfs get -r share.nfs tank/sales
NAME             PROPERTY   VALUE  SOURCE
tank/sales       share.nfs  on     local
tank/sales%      share.nfs  on     inherited from tank/sales
tank/sales/log   share.nfs  on     inherited from tank/sales
tank/sales/log%  share.nfs  on     inherited from tank/sales
```

다음과 같이 공유 파일 시스템의 특정 시스템에 루트 액세스 권한을 제공할 수 있습니다.

```
# zfs set share.nfs=on tank/home/data
# zfs set share.nfs.sec.default.root=neo tank/home/data
```

등록 정보별 상속을 사용하여 ZFS 공유

최신 풀 버전 34로 업그레이드된 풀에서는 공유 유지 관리를 돕기 위해 ZFS 등록 정보 상속을 활용하는 새로운 공유 구문을 사용할 수 있습니다. 각 공유 특성은 개별 share 등록 정보가 됩니다. share 등록 정보는 share. 접두어로 시작하는 이름으로 식별됩니다. share 등록 정보 예로는 share.desc, share.nfs.nosuid 및 share.smb.guestok가 있습니다.

share.nfs 등록 정보는 NFS 공유가 사용으로 설정되었는지 여부를 제어합니다. share.smb 등록 정보는 SMB 공유가 사용으로 설정되었는지 여부를 제어합니다. 새 풀에서 sharenfs는 share.nfs의 별칭이고 sharesmb는 share.smb의 별칭이므로 레거시 sharenfs 및 sharesmb 등록 정보 이름도 계속 사용할 수 있습니다. tank/home 파일 시스템을 공유하려는 경우 다음과 비슷한 구문을 사용합니다.

```
# zfs set share.nfs=on tank/home
```

이 예에서 share.nfs 등록 정보 값은 모든 종속된 파일 시스템에 상속됩니다. 예를 들면 다음과 같습니다.

```
# zfs create tank/home/userA
# zfs create tank/home/userB
# grep tank/home /etc/dfs/sharetab
/tank/home      tank_home       nfs     sec=sys,rw
/tank/home/userA        tank_home_userA nfs     sec=sys,rw
/tank/home/userB        tank_home_userB nfs     sec=sys,rw
```

이전 풀의 ZFS 공유 상속

이전 풀에서는 sharenfs 및 sharesmb 등록 정보만 종속된 파일 시스템에서 상속됩니다. 기타 공유 특성은 각 공유에 대해 .zfs/shares 파일에 저장되며 상속되지 않습니다.

특별 규칙은 상위 항목으로부터 sharenfs 또는 sharesmb를 상속하는 새 파일 시스템을 만들 때마다 해당 파일 시스템에 대해 sharenfs 또는 sharesmb 값으로부터 기본 공유를 만드는 규칙입니다. sharenfs가 단순히 설정된 경우 종속 파일 시스템에 생성된 기본 공유는 기본 NFS 특성만 갖습니다. 예를 들면 다음과 같습니다.

```
# zpool get version tank
NAME  PROPERTY  VALUE  SOURCE
tank  version   33     default
# zfs create -o sharenfs=on tank/home
# zfs create tank/home/userA
# grep tank/home /etc/dfs/sharetab
/tank/home      tank_home       nfs     sec=sys,rw
/tank/home/userA        tank_home_userA nfs     sec=sys,r
```

ZFS 이름이 지정된 공유

SMB 환경에서 권한 및 등록 정보를 설정하는 데 보다 많은 유연성을 제공하는 **이름이 지정된** 공유를 만들 수도 있습니다. 예를 들면 다음과 같습니다.

```
# zfs share -o share.smb=on tank/workspace%myshare
```

이전 예에서 `zfs share` 명령은 `tank/workspace` 파일 시스템의 `myshare`라는 SMB 공유를 만듭니다. 파일 시스템의 `.zfs/shares` 디렉토리를 통해 이 SMB 공유에 액세스하고 특정 권한 또는 ACL을 표시하거나 설정할 수 있습니다. 각 SMB 공유는 개별 `.zfs/shares` 파일로 표시됩니다. 예를 들면 다음과 같습니다.

```
# ls -lv /tank/workspace/.zfs/shares
-rwxrwxrwx+  1 root     root           0 May 15 10:31 myshare
     0:everyone@:read_data/write_data/append_data/read_xattr/write_xattr
         /execute/delete_child/read_attributes/write_attributes/delete
         /read_acl/write_acl/write_owner/synchronize:allow
```

이름이 지정된 공유는 상위 파일 시스템에서 공유 등록 정보를 상속합니다. 이전 예에서 `share.smb.guestok` 등록 정보를 상위 파일 시스템에 추가하면 이 등록 정보가 이름이 지정된 공유에 상속됩니다. 예를 들면 다음과 같습니다.

```
# zfs get -r share.smb.guestok tank/workspace
NAME                   PROPERTY           VALUE  SOURCE
tank/workspace         share.smb.guestok  on     inherited from tank
tank/workspace%myshare  share.smb.guestok  on     inherited from tank
```

이름이 지정된 공유는 NFS 환경에서 파일 시스템의 하위 디렉토리에 대한 공유를 정의할 때 유용할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs create -o share.nfs=on -o share.nfs.anon=99 -o share.auto=off tank/home
# mkdir /tank/home/userA
# mkdir /tank/home/userB
# zfs share -o share.path=/tank/home/userA tank/home%userA
# zfs share -o share.path=/tank/home/userB tank/home%userB
# grep tank/home /etc/dfs/sharetab
/tank/home/userA        userA   nfs     anon=99,sec=sys,rw
/tank/home/userB        userB   nfs     anon=99,sec=sys,rw
```

위 예에서는 파일 시스템에 대해 `share.auto`를 off로 설정하여 다른 모든 등록 정보 상속을 그대로 유지하면서 해당 파일 시스템에 대한 자동 공유를 해제하는 방법도 보여줍니다. 대부분의 다른 공유 등록 정보와 달리 `share.auto` 등록 정보는 상속할 수 없습니다.

이름이 지정된 공유는 공용 NFS 공유를 만들 때도 사용됩니다. 공용 공유는 이름이 지정된 NFS 공유에서만 만들 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs create -o mountpoint=/pub tank/public
# zfs share -o share.nfs=on -o share.nfs.public=on tank/public%pubshare
# grep pub /etc/dfs/sharetab
/pub    pubshare        nfs     public,sec=sys,rw
```

NFS 및 SMB 공유 등록 정보에 대한 자세한 내용은 share_nfs(1M) 및 share_smb(1M)을 참조하십시오.

ZFS 자동 공유

자동 공유를 만들면 파일 시스템 이름에서 고유한 리소스 이름이 생성됩니다. 생성된 이름은 리소스 이름에 부적합한 파일 시스템 이름의 문자가 밑줄(_) 문자로 바뀌는 점을 제외하고 파일 시스템 이름의 복사본입니다. 예를 들어, data/home/john의 리소스 이름은 data_home_john입니다.

share.autoname 등록 정보 이름을 설정하면 자동 공유를 만들 때 파일 시스템 이름을 특정 이름으로 바꿀 수 있습니다. 이러한 특정 이름은 상속의 경우 접두어 파일 시스템 이름을 대신하는 데에도 사용됩니다. 예를 들면 다음과 같습니다.

```
# zfs create -o share.smb=on -o share.autoname=john data/home/john
# zfs create data/home/john/backups
# grep john /etc/dfs/sharetab
/data/home/john john    smb
/data/home/john/backups john_backups    smb
```

레거시 share 명령 또는 zfs set share 명령을 아직 공유되지 않은 파일 시스템에서 사용하면 해당 share.auto 값이 자동으로 off로 설정됩니다. 레거시 명령은 항상 이름이 지정된 공유를 만듭니다. 이 특별 규칙에 따라 자동 공유는 생성 중인 이름이 지정된 공유를 간섭하지 않습니다.

ZFS 공유 정보 표시

zfs get 명령을 사용하여 파일 공유 등록 정보의 값을 표시합니다. 다음 예에서는 단일 파일 시스템에 대해 share.nfs 등록 정보를 표시하는 방법을 보여줍니다.

```
# zfs get share.nfs tank/sales
NAME        PROPERTY   VALUE  SOURCE
tank/sales  share.nfs  on     local
```

다음 예에서는 종속된 파일 시스템에 대해 share.nfs 등록 정보를 표시하는 방법을 보여줍니다.

```
# zfs get -r share.nfs tank/sales
NAME             PROPERTY   VALUE  SOURCE
tank/sales       share.nfs  on     local
tank/sales%      share.nfs  on     inherited from tank/sales
tank/sales/log   share.nfs  on     inherited from tank/sales
tank/sales/log%  share.nfs  on     inherited from tank/sales
```

zfs get all 명령 구문에서는 확장 공유 등록 정보를 사용할 수 없습니다.

다음 구문을 사용하여 NFS 또는 SMB 공유 정보에 대한 특정 세부 정보를 표시할 수 있습니다.

```
# zfs get share.nfs.all tank/sales
NAME        PROPERTY             VALUE  SOURCE
tank/sales  share.nfs.aclok      off    default
tank/sales  share.nfs.anon               default
tank/sales  share.nfs.charset.*  ...    default
tank/sales  share.nfs.cksum              default
tank/sales  share.nfs.index              default
tank/sales  share.nfs.log                default
tank/sales  share.nfs.noaclfab   off    default
tank/sales  share.nfs.nosub      off    default
tank/sales  share.nfs.nosuid     off    default
tank/sales  share.nfs.public     -      -
tank/sales  share.nfs.sec                default
tank/sales  share.nfs.sec.*      ...    default
```

공유 등록 정보가 여러 개 있기 때문에 비기본값을 사용하여 등록 정보를 표시하는 것이 좋습니다. 예를 들면 다음과 같습니다.

```
# zfs get -e -s local,received,inherited share.all tank/home
NAME       PROPERTY           VALUE  SOURCE
tank/home  share.auto         off    local
tank/home  share.nfs          on     local
tank/home  share.nfs.anon     99     local
tank/home  share.protocols    nfs    local
tank/home  share.smb.guestok  on     inherited from tank
```

ZFS 공유 등록 정보 값 변경

파일 시스템 공유에서 신규 또는 수정된 등록 정보를 지정하여 공유 등록 정보 값을 변경할 수 있습니다. 예를 들어, 파일 시스템을 만들 때 읽기 전용 등록 정보가 설정된 경우 등록 정보를 off로 설정할 수 있습니다.

```
# zfs create -o share.nfs.ro=\* tank/data
# zfs get share.nfs.ro tank/data
NAME       PROPERTY             VALUE  SOURCE
tank/data  share.nfs.sec.sys.ro  on     local
# zfs set share.nfs.ro=none tank/data
# zfs get share.nfs.ro tank/data
NAME       PROPERTY             VALUE  SOURCE
tank/data  share.nfs.sec.sys.ro  off    local
```

SMB 공유를 만들 때 NFS 공유 프로토콜을 추가할 수도 있습니다. 예를 들면 다음과 같습니다.

```
# zfs set share.smb=on tank/multifs
# zfs set share.nfs=on tank/multifs
# grep multifs /etc/dfs/sharetab
/tank/multifs   tank_multifs    nfs     sec=sys,rw
/tank/multifs   tank_multifs    smb     -
```

SMB 프로토콜을 제거합니다.

```
# zfs set share.smb=off tank/multifs
# grep multifs /etc/dfs/sharetab
/tank/multifs   tank_multifs    nfs     sec=sys,rw
```

이름이 지정된 공유의 이름을 바꿀 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs share -o share.smb=on tank/home/abc%abcshare
# grep abc /etc/dfs/sharetab
/tank/home/abc  abcshare        smb     -
# zfs rename tank/home/abc%abcshare tank/home/abc%a1share
# grep abc /etc/dfs/sharetab
/tank/home/abc  a1share        smb     -
```

ZFS 공유 게시 및 게시 해제

이름이 지정된 공유를 삭제하는 대신 zfs unshare 명령을 사용하여 일시적으로 공유 해제할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs unshare tank/home/abc%a1share
# grep abc /etc/dfs/sharetab
#
# zfs share tank/home/abc%a1share
# grep abc /etc/dfs/sharetab
/tank/home/abc  a1share smb     -
```

zfs unshare 명령을 실행하면 모든 파일 시스템 공유가 해제됩니다. 이러한 공유는 파일 시스템에 대해 zfs share 명령을 실행하거나 share.nfs 또는 share.smb 등록 정보가 파일 시스템에 대해 설정될 때까지 해제된 상태로 유지됩니다.

정의된 공유는 zfs unshare 명령을 실행할 때 제거되지 않으며, 다음에 파일 시스템에 대해 zfs share 명령을 실행할 때 또는 파일 시스템에 대해 share.nfs 또는 share.smb 등록 정보를 설정할 때 다시 공유됩니다.

ZFS 공유 제거

share.nfs 또는 share.smb 등록 정보를 off로 설정하면 파일 시스템 공유를 해제할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs set share.nfs=off tank/multifs
# grep multifs /etc/dfs/sharetab
#
```

zfs destroy 명령을 사용하면 이름이 지정된 공유를 영구적으로 제거할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs destroy tank/home/abc%a1share
```

비전역 영역 내의 ZFS 파일 공유

Oracle Solaris 11부터는 Oracle Solaris 비전역 영역에서 NFS 공유를 만들고 게시할 수 있습니다.

- ZFS 파일 시스템이 비전역 영역에 마운트되고 사용 가능한 경우 해당 영역에서 공유될 수 있습니다.
- 비전역 영역에 위임되지 않았고 비전역 영역에 마운트되지 않은 경우 전역 영역에서 파일 시스템을 공유할 수 있습니다. 파일 시스템이 비전역 영역에 추가된 경우 레거시 `share` 명령을 사용해서만 파일 시스템을 공유할 수 있습니다.

예를 들어, `/export/home/data` 및 `/export/home/data1` 파일 시스템은 `zfszone`에서 사용할 수 있습니다.

```
zfszone# share -F nfs /export/home/data
zfszone#  cat /etc/dfs/sharetab

zfszone# zfs set share.nfs=on tank/zones/export/home/data1
zfszone# cat /etc/dfs/sharetab
```

ZFS 공유 마이그레이션/전환 문제

이 절에서는 전환 문제에 대해 설명합니다.

- **이전 공유 등록 정보를 사용하여 파일 시스템 가져오기** - 풀을 가져오거나 Oracle Solaris 11 전에 만든 파일 시스템 스트림을 수신할 때 `sharenfs` 및 `sharesmb` 등록 정보는 등록 정보 값에 직접 모든 공유 등록 정보를 포함합니다. 대부분의 경우에 이러한 레거시 공유 등록 정보는 각 파일 시스템이 공유되는 즉시 해당하는 이름이 지정된 공유 세트로 변환됩니다. 가져오기 작업은 대부분의 경우 마운트 및 공유를 트리거하므로 이름이 지정된 공유에 대한 변환은 가져오기 프로세스 중에 직접 발생합니다.
- **Oracle Solaris 11에서 업그레이드** - 버전 34로의 풀 업그레이드 이후 첫번째 파일 시스템 공유는 이름이 지정된 공유를 새 형식으로 변환하기 때문에 시간이 오래 걸릴 수 있습니다. 업그레이드 프로세스로 생성된 이름이 지정된 공유는 올바르지만 공유 등록 정보 상속을 활용할 수 없습니다.
  - 공유 등록 정보 값 표시:

    ```
    # zfs get share.nfs filesystem
    # zfs get share.smb filesystem
    ```
  - 이전 BE로 재부트하는 경우 `sharenfs` 및 `sharesmb` 등록 정보를 원래 값으로 재설정합니다.
- **Oracle Solaris 11 Express에서 업그레이드** - Oracle Solaris 11 및 11.1에서 `sharenfs` 및 `sharesmb` 등록 정보는 `off` 및 `on` 값만 포함할 수 있습니다. 이러한 등록 정보는 공유 특성을 정의하는 데 더 이상 사용되지 않습니다.

/etc/dfs/dfstab 파일은 더 이상 부트 시 파일 시스템을 공유하는 데 사용되지 않습니다. 부트 시, 사용으로 설정된 파일 시스템 공유를 포함하는 모든 마운트된 ZFS 파일 시스템이 자동으로 공유됩니다. 공유는 sharenfs 또는 sharesmb를 on으로 설정할 때 사용으로 설정됩니다.

sharemgr 인터페이스는 더 이상 사용할 수 없습니다. 레거시 share 명령은 레거시 공유를 만드는 데 계속 사용할 수 있습니다. share -a 명령은 파일 시스템 공유가 지속되도록 이전 share -ap 명령과 유사하게 작동합니다. share -p 옵션은 더 이상 사용할 수 없습니다.

- **시스템 업그레이드** – 이 릴리스에서 등록 정보 변경으로 인해 Oracle Solaris 11 Express BE로 재부트할 경우 ZFS 공유가 잘못될 수 있습니다. 비ZFS 공유는 영향을 받지 않습니다. 이전 BE로 재부트하려는 경우 ZFS 데이터 세트의 공유 구성을 복원할 수 있으려면 pkg update 작업 전에 기존 공유 구성의 복사본을 먼저 저장해야 합니다.

  이전 BE에서 sharemgr show -vp 명령을 사용하여 모든 공유 및 해당 구성을 나열합니다.

  다음 명령을 사용하여 공유 등록 정보 값을 표시합니다.

  ```
  # zfs get sharenfs filesystem
  # zfs get sharesmb filesystem
  ```

  이전 BE로 돌아가면 sharenfs 및 sharesmb 등록 정보와 sharemgr로 정의된 모든 공유를 원래 값으로 재설정합니다.

- **레거시 공유 해제 동작** – unshare -a 명령 또는 unshareall 명령을 사용하면 파일 시스템이 공유 해제되지만, SMF 공유 저장소가 업데이트되지는 않습니다. 따라서 기존 공유를 다시 공유하려고 하면 공유 저장소가 충돌하는지 검사하여 오류가 표시됩니다.

ZFS 파일 시스템 공유 문제 해결

다음 공유 동작 시나리오 및 고려 사항을 검토하십시오.

- 공유 등록 정보 및 .zfs/shares 파일은 zfs clone 및 zfs send 작업에서 서로 다르게 처리됩니다. .zfs/shares 파일은 스냅샷에 포함되며 zfs clone 및 zfs send 작업에서 보존됩니다. 이름이 지정된 공유를 포함하는 공유 등록 정보는 스냅샷에 포함되지 않습니다. zfs send 및 zfs receive 작업을 수행하는 동안 등록 정보의 동작에 대한 설명은 211 페이지 “ZFS 스냅샷 스트림에 다른 등록 정보 값 적용”을 참조하십시오. 복제 작업 후 모든 파일은 사전 복제 스냅샷에서 가져오며 등록 정보는 ZFS 파일 시스템 계층에 있는 복제본의 새 위치에서 상속됩니다.
- 특정 레거시 공유 작업은 자동 공유를 자동으로 해제하거나 기존 자동 공유를 해당하는 이름이 지정된 공유로 변환합니다. 파일 시스템이 예상한 대로 공유되지 않으면 해당 share.auto 값이 off로 설정되었는지 확인합니다.
- 이 공유가 자동 공유와 충돌되어 이름이 지정된 공유를 만드는 요청이 실패할 경우, 작업을 계속하려면 자동 공유를 해제해야 할 수 있습니다.

- 풀을 읽기 전용으로 가져오면 해당 등록 정보 또는 파일을 수정할 수 없습니다. 이 경우 새 공유를 시작할 수 없습니다. 풀을 내보내기 전에 공유가 이미 설정된 경우 기존 공유 특성이 사용됩니다(가능한 경우).

다음 표에서는 공유 상태를 확인하고 필요에 따라 이를 해결하는 방법을 설명합니다.

공유 상태	설명	해결 방법
INVALID	내부적으로 일관성이 없거나 다른 공유와 충돌하기 때문에 공유가 잘못되었습니다.	다음 명령을 사용하여 잘못된 공유를 다시 공유하도록 시도합니다. **# zfs share FS%share** 이 명령을 사용하면 검증에 실패한 공유 특성에 대한 오류 메시지가 표시됩니다. 이를 해결하고 공유를 다시 시도하십시오.
SHARED	공유가 설정되었습니다.	필요한 작업이 없습니다.
UNSHARED	공유가 유효하지만 해제되어 있습니다.	zfs share 명령을 사용하여 개별 공유 또는 상위 파일 시스템을 다시 공유합니다.
UNVALIDATED	공유가 아직 검증되지 않았습니다. 해당 공유를 포함하는 파일 시스템이 공유 가능한 상태가 아닐 수 있습니다. 예를 들어, 마운트되지 않았거나 현재 영역이 아닌 다른 영역에 위임되어 있습니다. 또는 원하는 공유를 나타내는 ZFS 등록 정보가 만들어졌지만 아직 합법적인 공유로 검증되지 않았습니다.	zfs share 명령을 사용하여 개별 공유 또는 상위 파일 시스템을 다시 공유합니다. 파일 시스템 자체가 공유 가능한 경우 다시 공유하려는 시도가 성공하고 상태가 공유됨으로 변환되거나 실패하고 상태가 잘못됨으로 변환됩니다. 또는 share -A 명령을 사용하여 마운트된 모든 파일 시스템에서 모든 공유를 나열합니다. 이렇게 하면 마운트된 파일 시스템의 모든 공유가 공유 해제됨(유효하지만 아직 공유되지 않음) 또는 잘못됨으로 확인됩니다.

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/home에 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 180 페이지 “ZFS 파일 시스템에 대한 쿼터 설정” 및 183 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 암호화

암호화는 개인 정보 보호를 위해 데이터가 인코딩되는 프로세스이며 데이터 소유자가 인코딩된 데이터에 액세스하려면 키가 필요합니다. ZFS 암호화 사용의 이점은 다음과 같습니다.

- ZFS 암호화는 ZFS 명령 세트와 통합됩니다. 다른 ZFS 작업과 마찬가지로 키 변경 및 rekey와 같은 암호화 작업은 온라인으로 수행됩니다.
- 기존 저장소 풀은 업그레이드하여 사용할 수 있습니다. 특정 파일 시스템을 암호화할 수도 있습니다.
- ZFS 암호화는 종속 파일 시스템에 상속될 수 있으며, ZFS 위임 관리를 통해 키 관리를 위임할 수 있습니다.
- CCM 및 GCM 작업 모드에서 키 길이 128, 192 및 256으로 AES(Advanced Encryption Standard)를 사용하여 데이터가 암호화됩니다.
- ZFS 암호화는 암호화 알고리즘의 사용 가능한 하드웨어 가속 또는 최적화된 소프트웨어 구현에 자동으로 액세스할 수 있게 하는 Oracle Solaris Cryptographic Framework를 사용합니다.
- 현재까지는 개별 파일 시스템이더라도 `/var` 디렉토리와 같은 다른 OS 구성 요소 또는 ZFS 루트 파일 시스템을 암호화할 수 없습니다.

ZFS 파일 시스템을 만들 때 암호화 정책을 설정할 수 있지만 정책을 변경할 수는 없습니다. 예를 들어, `tank/home/darren` 파일 시스템은 암호화 등록 정보를 사용으로 설정하여 생성됩니다. 기본 암호화 정책은 길이가 최소 8자 이상이어야 하는 문장암호를 프롬프트하는 것입니다.

```
# zfs create -o encryption=on tank/home/darren
Enter passphrase for 'tank/home/darren': xxxxxxx
Enter again: xxxxxxxx
```

파일 시스템에 암호화가 사용으로 설정되었는지 확인합니다. 예를 들면 다음과 같습니다.

```
# zfs get encryption tank/home/darren
NAME              PROPERTY    VALUE        SOURCE
tank/home/darren  encryption  on           local
```

파일 시스템의 암호화 값이 on일 경우 기본 암호화 알고리즘은 aes-128-ccm입니다.

래핑 키는 실제 데이터 암호 키를 암호화하는 데 사용됩니다. 래핑 키는 위 예와 같이 암호화된 파일 시스템을 만들 때 zfs 명령에서 커널로 전달됩니다. 래핑 키는 파일에 raw 또는 hex 형식으로 포함되어 있거나 문장암호에서 파생됩니다.

래핑 키의 형식과 위치는 다음과 같이 keysource 등록 정보에 지정됩니다.

```
keysource=format,location
```

- 형식은 다음 중 하나입니다.
  - raw – 원시 키 바이트입니다.
  - hex – 16진수 키 문자열입니다.
  - passphrase – 키를 생성하는 문자열입니다.
- 위치는 다음 중 하나입니다.
  - prompt – 파일 시스템을 만들거나 마운트할 때 키 또는 문장암호를 묻는 메시지가 표시됩니다.
  - file:///*filename* – 파일 시스템의 키 또는 문장암호 파일 위치입니다.
  - pkcs11 – PKCS#11 토큰에서 키 또는 문장암호의 위치를 설명하는 URI입니다.
  - https://*location* – 보안 서버의 키 또는 문장암호 파일 위치입니다. 이 방법을 사용하여 키 정보를 일반 텍스트 상태로 전송하는 것은 권장되지 않습니다. URL에서 GET은 keysource 등록 정보의 형식으로 요청된 사항에 따라 키 값 또는 문장암호를 반환합니다.

    keysource에 대해 https:// 로케이터를 사용할 때 서버가 제공하는 인증서는 libcurl 및 OpenSSL로 신뢰되는 인증서여야 합니다. 고유한 신뢰 앵커 또는 자체 서명된 인증서를 /etc/openssl/certs의 인증서 저장소에 추가합니다. PEM 형식 인증서를 /etc/certs/CA 디렉토리에 두고 다음 명령을 실행합니다.

    ```
    # svcadm refresh ca-certificates
    ```

keysource 형식이 *passphrase*인 경우 래핑 키가 문장암호에서 파생됩니다. 그렇지 않으면 keysource 등록 정보 값이 원시 바이트나 16진수 형식으로 실제 래핑 키를 가리킵니다. 문장암호를 파일에 저장하거나 확인 메시지가 표시되는 원시 바이트 스트림에 저장하도록 지정할 수 있습니다. 후자의 경우 대체로 스크립팅에만 적합합니다.

파일 시스템의 keysource 등록 정보 값이 passphrase를 식별하는 경우 래핑 키는 PKCS#5 PBKD2 및 파일 시스템별 모든 생성된 salt를 사용하여 문장암호에서 파생됩니다. 즉, 종속 파일 시스템에서 사용할 경우 동일한 문장암호가 다른 래핑 키를 생성합니다.

파일 시스템의 암호화 정책은 종속 파일 시스템에 상속되며 제거할 수 없습니다. 예를 들면 다음과 같습니다.

```
# zfs snapshot tank/home/darren@now
# zfs clone tank/home/darren@now tank/home/darren-new
Enter passphrase for 'tank/home/darren-new': xxxxxxx
Enter again: xxxxxxxx
# zfs set encryption=off tank/home/darren-new
cannot set property for 'tank/home/darren-new': 'encryption' is readonly
```

암호화 또는 암호화 해제된 ZFS 파일 시스템을 복사하거나 마이그레이션해야 하는 경우 다음 사항을 고려해 보십시오.

- 현재 수신 풀의 데이터 세트에 암호화가 사용으로 설정된 경우에도 암호화 해제된 데이터 세트 스트림을 보내고 암호화된 스트림으로 받을 수 없습니다.
- 다음 명령을 사용하여 암호화 해제된 데이터를 암호화가 사용으로 설정된 풀/파일 시스템에 마이그레이션할 수 있습니다.
  - cp -r
  - find | cpio
  - tar
  - rsync
- 복제된 암호화된 파일 시스템 스트림을 암호화된 파일 시스템으로 수신할 수 있으며 데이터가 암호화된 상태로 유지됩니다. 자세한 내용은 예 5–4를 참조하십시오.

암호화된 ZFS 파일 시스템의 키 변경

zfs key -c 명령을 사용하여 암호화된 파일 시스템의 래핑 키를 변경할 수 있습니다. 부트 시 또는 파일 시스템 키를 명시적으로 로드(zfs key -l)하거나 파일 시스템을 마운트(zfs mount *filesystem*)하여 기존 래핑 키가 먼저 로드되어 있어야 합니다. 예:

```
# zfs key -c tank/home/darren
Enter new passphrase for 'tank/home/darren': xxxxxxxx
Enter again: xxxxxxxx
```

다음 예에서는 래핑 키를 변경하고 keysource 등록 정보 값을 변경하여 래핑 키가 파일에서 제공되도록 지정합니다.

```
# zfs key -c -o keysource=raw,file:///media/stick/key tank/home/darren
```

암호화된 파일 시스템의 데이터 암호 키는 zfs key -K 명령을 사용하여 변경할 수 있지만 새 암호 키는 새로 작성된 데이터에만 사용됩니다. 이 기능을 사용하여 데이터 암호 키의 시간 제한에 대한 NIST 800-57 지침을 준수할 수 있습니다. 예를 들면 다음과 같습니다.

```
# zfs key -K tank/home/darren
```

위 예에서는 데이터 암호 키가 표시되지 않으며 직접 관리되지도 않습니다. 또한 키 변경 작업을 수행하려면 keychange 위임이 필요합니다.

다음 암호화 알고리즘을 사용할 수 있습니다.

- aes-128-ccm, aes-192-ccm, aes-256-ccm
- aes-128-gcm, aes-192-gcm, aes-256-gcm

ZFS keysource 등록 정보는 파일 시스템의 데이터 암호 키를 래핑하는 키의 형식과 위치를 식별합니다. 예를 들면 다음과 같습니다.

```
# zfs get keysource tank/home/darren
NAME              PROPERTY   VALUE              SOURCE
tank/home/darren  keysource  passphrase,prompt  local
```

ZFS rekeydate 등록 정보는 마지막 zfs key -K 작업의 날짜를 식별합니다. 예를 들면 다음과 같습니다.

```
# zfs get rekeydate tank/home/darren
NAME              PROPERTY   VALUE                  SOURCE
tank/home/darren  rekeydate  Wed Jul 25 16:54 2012  local
```

암호화된 파일 시스템의 creation 및 rekeydate 등록 정보 값이 같은 경우 파일 시스템이 zfs key -K 작업에 의해 rekey된 적이 없습니다.

ZFS 암호화 키 관리

ZFS 암호화 키는 중앙 위치가 필요할 경우 로컬 시스템 또는 원격으로 필요에 따라 다른 방식으로 관리할 수 있습니다.

- **로컬** – 위 예에서는 래핑 키가 로컬 시스템의 파일에 저장된 문장암호 프롬프트 또는 원시 키일 수 있음을 보여줍니다.
- **원격** – Oracle Key Manager와 같은 중앙화된 키 관리 시스템을 사용하거나 http 또는 https URI에서 단순 GET 요청을 지원하는 웹 서비스를 사용하여 키 정보를 원격으로 저장할 수 있습니다. Oracle Key Manager 키 정보는 PKCS#11 토큰을 사용하여 Oracle Solaris 시스템에 액세스할 수 있습니다.

ZFS 암호화 키 관리에 대한 자세한 내용은 다음 항목을 참조하십시오.

http://www.oracle.com/technetwork/articles/servers-storage-admin/manage-zfs-encryption-1715034.html

Oracle Key Manager를 사용하여 키 정보를 관리하는 방법은 다음 항목을 참조하십시오.

http://docs.oracle.com/cd/E24472_02/

ZFS 키 작업 권한 위임

키 작업 위임에 대한 다음 권한 설명을 검토합니다.

- zfs key -l 및 zfs key -u 명령을 사용하여 파일 시스템 키를 로드하거나 언로드하려면 key 권한이 필요합니다. 대부분의 경우 mount 권한도 필요합니다.
- zfs key -c 및 zfs key -K 명령을 사용하여 파일 시스템 키를 변경하려면 keychange 권한이 필요합니다.

키 사용(로드 또는 언로드) 및 키 변경에 대해 별도의 권한을 위임하는 것이 좋습니다. 이렇게 하면 두 사용자 키 작업 모델을 사용할 수 있습니다. 예를 들어, 키를 사용할 수 있는 사용자와 키를 변경할 수 있는 사용자를 결정합니다. 또는 키를 변경하려면 두 사용자가 모두 있어야 합니다. 이 모델에서는 키 위탁 시스템도 구축할 수 있습니다.

암호화된 ZFS 파일 시스템 마운트

암호화된 ZFS 파일 시스템을 마운트하려는 경우 다음 고려 사항을 검토합니다.

- 부트 시 암호화된 파일 시스템 키를 사용할 수 없는 경우 파일 시스템이 자동으로 마운트되지 않습니다. 예를 들어, 암호화 정책이 passphrase,prompt로 설정된 파일 시스템은 문장암호를 묻는 메시지를 표시하기 위해 부트 프로세스가 중단되지 않으므로 부트 시 마운트되지 않습니다.
- 부트 시 암호화 정책이 passphrase,prompt로 설정된 파일 시스템을 마운트하려는 경우 zfs mount 명령을 사용하여 명시적으로 마운트하고 문장암호를 지정하거나 zfs key -l 명령을 사용하여 시스템 부트 후 키를 묻는 메시지를 표시해야 합니다.

  예를 들면 다음과 같습니다.

  ```
  # zfs mount -a
  Enter passphrase for 'tank/home/darren': xxxxxxxx
  Enter passphrase for 'tank/home/ws': xxxxxxxx
  Enter passphrase for 'tank/home/mark': xxxxxxxx
  ```
- 암호화된 파일 시스템의 keysource 등록 정보가 다른 파일 시스템의 파일을 가리키는 경우 특히 파일이 이동식 매체에 있으면 파일 시스템의 마운트 순서가 부트 시 암호화된 파일 시스템의 마운트 여부에 영향을 줄 수 있습니다.

암호화된 ZFS 파일 시스템 업그레이드

Solaris 11 시스템을 Solaris 11.1로 업그레이드하려면 먼저 암호화된 파일 시스템이 마운트되어 있는지 확인하십시오. 암호화된 파일 시스템을 마운트하고 메시지가 표시되면 문장암호를 제공하십시오.

```
# zfs mount -a
Enter passphrase for 'pond/amy': xxxxxxxx
Enter passphrase for 'pond/rory': xxxxxxxx
# zfs mount | grep pond
pond                            /pond
pond/amy                        /pond/amy
pond/rory                       /pond/rory
```

그런 다음 암호화된 파일 시스템을 업그레이드하십시오.

```
# zfs upgrade -a
```

마운트 해제된 암호화된 ZFS 파일 시스템을 업그레이드하려고 시도하면 다음과 유사한 메시지가 표시됩니다.

```
# zfs upgrade -a
cannot set property for 'pond/amy': key not present
```

또한 zpool status 출력이 손상된 데이터를 표시할 수 있습니다.

```
# zpool status -v pond
.
.
.
        pond/amy:<0x1>
        pond/rory:<0x1>
```

위 오류가 발생하는 경우 위에서 설명한 대로 암호화된 파일 시스템을 다시 마운트하십시오. 그런 다음 풀 오류를 스크럽하고 지우십시오.

```
# zpool scrub pond
# zpool clear pond
```

파일 시스템 업그레이드에 대한 자세한 내용은 195 페이지 "ZFS 파일 시스템 업그레이드"를 참조하십시오.

ZFS 압축, 중복 제거 및 암호화 등록 정보 간의 상호 작용

ZFS 압축, 중복 제거 및 암호화 등록 정보를 사용하는 경우 다음 고려 사항을 검토합니다.

- 파일을 작성할 때 데이터가 압축 및 암호화되고 체크섬이 확인됩니다. 가능한 경우 데이터의 중복이 제거됩니다.
- 파일을 읽을 때 체크섬이 확인되고 데이터의 암호가 해독됩니다. 필요한 경우 데이터의 압축이 해제됩니다.
- 복제된 암호화된 파일 시스템에서 dedup 등록 정보를 사용으로 설정하고 zfs key -K 또는 zfs clone -K 명령을 복제본에 사용하지 않으면 가능한 경우 모든 복제본의 데이터에서 중복이 제거됩니다.

ZFS 파일 시스템 암호화의 예

예 5–1 원시 키를 사용하여 ZFS 파일 시스템 암호화

다음 예에서 aes-256-ccm 암호화 키는 pktool 명령을 사용해서 생성되며 /cindykey.file 파일에 기록됩니다.

```
# pktool genkey keystore=file outkey=/cindykey.file keytype=aes keylen=256
```

예 5–1 원시 키를 사용하여 ZFS 파일 시스템 암호화 *(계속)*

그런 다음 tank/home/cindy 파일 시스템을 만들 때 /cindykey.file을 지정합니다.

```
# zfs create -o encryption=aes-256-ccm -o keysource=raw,file:///cindykey.file
tank/home/cindy
```

예 5–2 다른 암호화 알고리즘을 사용하여 ZFS 파일 시스템 암호화

ZFS 저장소 풀을 만들고 저장소 풀의 모든 파일 시스템이 암호화 알고리즘을 상속하도록 할 수 있습니다. 이 예에서는 users 풀을 만들며 users/home 파일 시스템을 만들고 문장암호를 사용하여 암호화합니다. 기본 암호화 알고리즘은 aes-128-ccm입니다.

그런 다음 aes-256-ccm 암호화 알고리즘을 사용하여 users/home/mark 파일 시스템을 만들고 암호화합니다.

```
# zpool create -O encryption=on users mirror c0t1d0 c1t1d0 mirror c2t1d0 c3t1d0
Enter passphrase for 'users': xxxxxxxx
Enter again: xxxxxxxx
# zfs create users/home
# zfs get encryption users/home
NAME        PROPERTY    VALUE        SOURCE
users/home  encryption  on           inherited from users
# zfs create -o encryption=aes-256-ccm users/home/mark
# zfs get encryption users/home/mark
NAME                PROPERTY    VALUE        SOURCE
users/home/mark     encryption  aes-256-ccm  local
```

예 5–3 암호화된 ZFS 파일 시스템 복제

복제 파일 시스템이 원래 스냅샷과 동일한 파일 시스템에서 keysource 등록 정보를 상속하는 경우 새 keysource가 필요하지 않으며 keysource=passphrase,prompt이면 새 문장암호를 묻는 메시지가 표시되지 않습니다. 동일한 keysource가 복제본에 사용됩니다. 예를 들면 다음과 같습니다.

기본적으로 암호화된 파일 시스템의 종속 항목을 복제할 때는 키를 묻는 메시지가 표시되지 않습니다.

```
# zfs create -o encryption=on tank/ws
Enter passphrase for 'tank/ws': xxxxxxxx
Enter again: xxxxxxxx
# zfs create tank/ws/fs1
# zfs snapshot tank/ws/fs1@snap1
# zfs clone tank/ws/fs1@snap1 tank/ws/fs1clone
```

복제 파일 시스템에 대한 새 키를 만들려는 경우 zfs clone -K 명령을 사용합니다.

암호화된 종속 파일 시스템 대신 암호화된 파일 시스템을 복제하는 경우 새 키를 제공하라는 메시지가 표시됩니다. 예를 들면 다음과 같습니다.

```
# zfs create -o encryption=on tank/ws
Enter passphrase for 'tank/ws': xxxxxxxx
```

예 5–3 암호화된 ZFS 파일 시스템 복제 (*계속*)

```
Enter again: xxxxxxxx
# zfs snapshot tank/ws@1
# zfs clone tank/ws@1 tank/ws1clone
Enter passphrase for 'tank/ws1clone': xxxxxxxx
Enter again: xxxxxxxx
```

예 5–4 암호화된 ZFS 파일 시스템 보내기 및 받기

다음 예에서 `tank/home/darren@snap1` 스냅샷은 암호화된 `/tank/home/darren` 파일 시스템에서 생성됩니다. 결과로 수신된 데이터가 암호화되도록 encryption 등록 정보를 사용으로 설정하면 스냅샷이 `bpool/snaps`로 전송됩니다. 하지만 보내는 동안 `tank/home/darren@snap1` 스트림은 암호화되지 않습니다.

```
# zfs get encryption tank/home/darren
NAME              PROPERTY    VALUE        SOURCE
tank/home/darren  encryption  on           local
# zfs snapshot tank/home/darren@snap1
# zfs get encryption bpool/snaps
NAME         PROPERTY    VALUE        SOURCE
bpool/snaps  encryption  on           inherited from bpool
# zfs send tank/home/darren@snap1 | zfs receive bpool/snaps/darren1012
# zfs get encryption bpool/snaps/darren1012
NAME                     PROPERTY    VALUE        SOURCE
bpool/snaps/darren1012  encryption  on           inherited from bpool
```

이 경우 수신된 암호화된 파일 시스템에 대한 새 키가 자동으로 생성됩니다.

ZFS 파일 시스템 마이그레이션

섀도우 마이그레이션 기능을 사용하면 파일 시스템을 다음과 같이 마이그레이션할 수 있습니다.

- 로컬 또는 원격 ZFS 파일 시스템을 대상 ZFS 파일 시스템으로
- 로컬 또는 원격 UFS 파일 시스템을 대상 ZFS 파일 시스템으로

섀도우 마이그레이션은 마이그레이션할 데이터를 가져오는 프로세스입니다.

- 빈 ZFS 파일 시스템을 만듭니다.
- 대상(또는 그림자) 파일 시스템인 빈 ZFS 파일 시스템의 `shadow` 등록 정보를 마이그레이션할 파일 시스템을 가리키도록 설정합니다.
- 마이그레이션할 파일 시스템의 데이터가 그림자 파일 시스템으로 복사됩니다.

shadow 등록 정보 URI를 사용하여 다음 두 가지 방법으로 마이그레이션할 파일 시스템을 식별할 수 있습니다.

- `shadow=file:///`*path* – 로컬 파일 시스템을 마이그레이션하려면 이 구문을 사용합니다.

- `shadow=nfs://`*host*`:`*path* – NFS 파일 시스템을 마이그레이션하려면 이 구문을 사용합니다.

파일 시스템 마이그레이션 시 다음 고려 사항을 검토하십시오.

- 마이그레이션할 파일 시스템을 읽기 전용으로 설정해야 합니다. 파일 시스템이 읽기 전용으로 설정되지 않을 경우, 진행 중인 변경 사항이 마이그레이션되지 않습니다.
- 대상 파일 시스템이 완전히 비어 있어야 합니다.
- 마이그레이션 도중 시스템을 재부트하면 시스템이 부트된 후 마이그레이션이 계속됩니다.
- 전체 컨텐츠가 마이그레이션될 때까지 완전히 마이그레이션되지 않은 디렉토리 컨텐츠에 대한 액세스 또는 완전히 마이그레이션되지 않은 파일 컨텐츠에 대한 액세스가 차단됩니다.
- NFS 마이그레이션 중 UID, GID 및 ACL 정보를 섀도우 파일 시스템으로 마이그레이션하려는 경우, 로컬 시스템과 원격 시스템 간에 이름 지정 서비스 정보에 액세스할 수 있는지 확인합니다. NFS를 통해 대규모 데이터 마이그레이션을 완료하기 전에 테스트 마이그레이션을 위해 마이그레이션할 파일 시스템 데이터의 일부를 복사하여 모든 정보가 제대로 마이그레이션되는지 확인할 수 있습니다.
- 네트워크 대역폭에 따라 NFS를 통한 파일 시스템 데이터 마이그레이션이 느릴 수 있습니다. 잠시 기다려 주십시오.
- `shadowstat` 명령을 사용하여 다음 데이터를 제공하는 파일 시스템 마이그레이션을 모니터할 수 있습니다.
  - `BYTES XFRD` 열은 그림자 파일 시스템에 전송된 바이트 수를 식별합니다.
  - `BYTES LEFT` 열은 마이그레이션이 거의 완료될 때까지 계속 변동됩니다. ZFS는 마이그레이션을 시작할 때 마이그레이션해야 하는 데이터 양을 식별하지 않습니다. 이 프로세스에 많은 시간이 걸릴 수 있기 때문입니다.
  - `BYTES XFRD` 및 `ELAPSED TIME` 정보를 사용하여 마이그레이션 프로세스의 길이를 예상합니다.

▼ 파일 시스템을 ZFS 파일 시스템으로 마이그레이션하는 방법

1 원격 NFS 서버에서 데이터를 마이그레이션할 경우 두 시스템 모두에서 이름 서비스 정보에 액세스할 수 있는지 확인합니다.

NFS를 사용하는 대규모 마이그레이션의 경우 일부 데이터에 대해 테스트 마이그레이션을 수행하여 UID, GUID 및 ACL 정보가 올바르게 마이그레이션되는지 확인할 수 있습니다.

2 필요한 경우 데이터를 마이그레이션할 시스템에 섀도우 마이그레이션 패키지를 설치하고 shadowd 서비스가 마이그레이션 프로세스를 지원할 수 있도록 설정합니다.

```
# pkg install shadow-migration

# svcadm enable shadowd
```

shadowd 프로세스를 사용으로 설정하지 않는 경우 마이그레이션 프로세스가 완료될 때 shadow 등록 정보를 none으로 재설정해야 합니다.

3 마이그레이션할 로컬 또는 원격 파일 시스템을 읽기 전용으로 설정합니다.

로컬 ZFS 파일 시스템을 마이그레이션하는 경우 읽기 전용으로 설정합니다. 예를 들면 다음과 같습니다.

```
# zfs set readonly=on tank/home/data
```

원격 파일 시스템을 마이그레이션하는 경우 읽기 전용으로 공유합니다. 예를 들면 다음과 같습니다.

```
# share -F nfs -o ro /export/home/ufsdata
# share
-               /export/home/ufsdata   ro   ""
```

4 shadow 등록 정보를 마이그레이션할 파일 시스템으로 설정하여 새 ZFS 파일 시스템을 만듭니다.

예를 들어, 로컬 ZFS 파일 시스템 rpool/old를 새 ZFS 파일 시스템인 users/home/shadow로 마이그레이션하는 경우 users/home/shadow 파일 시스템을 만들 때 shadow 등록 정보를 rpool/old로 설정합니다.

```
# zfs create -o shadow=file:///rpool/old users/home/shadow
```

예를 들어, 원격 서버에서 /export/home/ufsdata를 마이그레이션하려면 ZFS 파일 시스템을 만들 때 shadow 등록 정보를 설정합니다.

```
# zfs create -o shadow=nfs://neo/export/home/ufsdata users/home/shadow2
```

5 마이그레이션 진행률을 확인합니다.

예를 들면 다음과 같습니다.

```
# shadowstat
                                  EST
                          BYTES   BYTES           ELAPSED
DATASET                   XFRD    LEFT    ERRORS  TIME
users/home/shadow         45.5M   2.75M   -       00:02:31
users/home/shadow         55.8M   -       -       00:02:41
users/home/shadow         69.7M   -       -       00:02:51
No migrations in progress
```

마이그레이션이 완료되면 shadow 등록 정보가 none으로 설정됩니다.

```
# zfs get -r shadow users/home/shadow*
NAME                PROPERTY  VALUE   SOURCE
users/home/shadow   shadow    none    -
users/home/shadow2  shadow    none    -
```

ZFS 파일 시스템 마이그레이션 문제 해결

ZFS 마이그레이션 문제를 해결할 때는 다음 사항을 검토하십시오.

- 마이그레이션할 파일 시스템을 읽기 전용으로 설정하지 않으면 일부 데이터가 마이그레이션되지 않습니다.
- shadow 등록 정보를 설정할 때 대상 파일 시스템이 비어 있지 않으면 데이터 마이그레이션이 시작되지 않습니다.
- 마이그레이션이 진행 중일 때 마이그레이션할 파일 시스템에서 데이터를 추가하거나 제거하면 해당 변경 사항이 마이그레이션되지 않을 수 있습니다.
- 마이그레이션이 진행 중일 때 그림자 파일 시스템의 마운트를 변경하려고 하면 다음 메시지가 표시됩니다.

```
# zfs set mountpoint=/users/home/data users/home/shadow3
cannot unmount '/users/home/shadow3': Device busy
```

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.
```

암호화된 파일 시스템 업그레이드에 대한 자세한 내용은 189 페이지 “암호화된 ZFS 파일 시스템 업그레이드”를 참조하십시오.

◆ ◆ ◆ 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.

- 197 페이지 "ZFS 스냅샷 개요"
- 198 페이지 "ZFS 스냅샷 만들기 및 삭제"
- 201 페이지 "ZFS 스냅샷 표시 및 액세스"
- 202 페이지 "ZFS 스냅샷 롤백"
- 204 페이지 "ZFS 복제본 개요"
- 205 페이지 "ZFS 복제본 만들기"
- 205 페이지 "ZFS 복제본 삭제"
- 205 페이지 "ZFS 복제본으로 ZFS 파일 시스템 대체"
- 206 페이지 "ZFS 데이터 전송 및 수신"

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 206 페이지 "ZFS 데이터 전송 및 수신"을 참조하십시오.

- 198 페이지 "ZFS 스냅샷 만들기 및 삭제"
- 201 페이지 "ZFS 스냅샷 표시 및 액세스"
- 202 페이지 "ZFS 스냅샷 롤백"

ZFS 스냅샷 만들기 및 삭제

스냅샷은 `zfs snapshot` 또는 `zfs snap` 명령으로 생성되는데, 이 명령은 만들 스냅샷의 이름만 인수로 사용합니다. 스냅샷은 이름은 다음과 같이 지정됩니다.

filesystem@snapname
volume@snapname

스냅샷 이름은 31 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 135 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

기본적으로 스냅샷은 zfs list 출력에 더 이상 표시되지 않습니다. zfs list -t snapshot 명령을 사용하여 스냅샷 정보를 표시해야 합니다. 또는 listsnapshots 풀 등록 정보를 사용으로 설정하십시오. 예를 들면 다음과 같습니다.

```
# zpool get listsnapshots tank
NAME  PROPERTY       VALUE      SOURCE
tank  listsnapshots  off        default
# zpool set listsnapshots=on tank
# zpool get listsnapshots tank
NAME  PROPERTY       VALUE      SOURCE
tank  listsnapshots  on         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                     124G  9.57G         0    302K              0      9.57G
rpool/ROOT                124G  3.38G         0     31K              0      3.38G
rpool/ROOT/solaris        124G  20.5K         0       0              0      20.5K
rpool/ROOT/solaris/var    124G  20.5K         0   20.5K              0          0
rpool/ROOT/solaris-1      124G  3.38G      66.3M   3.14G              0       184M
rpool/ROOT/solaris-1/var  124G   184M      49.9M    134M              0          0
rpool/VARSHARE            124G  39.5K         0   39.5K              0          0
rpool/dump                124G  4.12G         0   4.00G           129M          0
rpool/export              124G    63K         0     32K              0        31K
rpool/export/home         124G    31K         0     31K              0          0
rpool/swap                124G  2.06G         0   2.00G          64.7M          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 `M`은 디렉토리가 수정되었음을 나타냅니다. `+`는 `fileB`가 나중 스냅샷에 존재함을 나타냅니다.

다음 출력 결과에서 `M`은 스냅샷에 포함된 파일의 이름이 변경되었음을 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 160 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.

- 205 페이지 “ZFS 복제본 만들기”
- 205 페이지 “ZFS 복제본 삭제”
- 205 페이지 “ZFS 복제본으로 ZFS 파일 시스템 대체”

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 share.nfs=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 205 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.

- 207 페이지 “다른 백업 제품으로 ZFS 데이터 저장”
- 209 페이지 “ZFS 스냅샷 전송”
- 210 페이지 “ZFS 스냅샷 수신”
- 211 페이지 “ZFS 스냅샷 스트림에 다른 등록 정보 값 적용”
- 213 페이지 “복잡한 ZFS 스냅샷 스트림 전송 및 수신”
- 216 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 197 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 Oracle Solaris 11 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 비슷한 구문을 사용하십시오.

```
# zfs send tank/dana@snap1 | zfs recv spool/ds01
```

zfs recv를 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 207 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.

- 217 페이지 “Solaris ACL 모델”
- 224 페이지 “ZFS 파일에 ACL 설정”
- 226 페이지 “Verbose 형식으로 ZFS 파일에서 ACL 설정 및 표시”
- 236 페이지 “Compact 형식으로 ZFS 파일에서 ACL 설정 및 표시”
- 242 페이지 “ZFS 파일에 특수 속성 적용”

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 222 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 207 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

ZFS ACL 세트

다음 ACL 조합은 개별 권한을 별도로 설정하는 대신 *ACL* **세트**로 적용할 수 있습니다. 다음과 같은 ACL 세트를 사용할 수 있습니다.

ACL 세트 이름	포함된 ACL 권한
full_set	모든 권한
modify_set	write_acl 및 write_owner를 제외한 모든 권한
read_set	read_data, read_attributes, read_xattr 및 read_acl
write_set	write_data, append_data, write_attributes 및 write_xattr

이러한 ACL 세트는 미리 정의되며 수정할 수 없습니다.

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.

표 7–4 ACL 상속 플래그 (계속)

상속 플래그	Compact 상속 플래그	설명
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.
현재 다음 플래그는 SMB 클라이언트나 서버에만 적용할 수 있습니다.		
successful_access	S	액세스에 성공할 경우 알람 또는 감사 레코드를 시작할지 여부를 나타냅니다. 이 플래그는 감사 또는 알람 ACE 유형과 함께 사용됩니다.
failed_access	F	액세스에 실패할 경우 알람 또는 감사 레코드를 시작할지 여부를 나타냅니다. 이 플래그는 감사 또는 알람 ACE 유형과 함께 사용됩니다.
inherited	I	ACE가 상속되었음을 나타냅니다.

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 7–14를 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 수퍼 유저에 대해 어떤 액세스 요청이 거부되는지 결정합니다. 이 방식 덕분에 파일 소유자가 자신의 파일이 잠기는 것을 방지하고 수퍼 유저가 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 231 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_acl/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 218 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 236 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.

다음 예에서 단순 ACL이 `file.1`에 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.1`에 대한 권한이 644로 다시 설정됩니다.

```
# chmod 644 file.1
# ls -v file.1
```

예 7–1 ZFS 파일의 단순 ACL 수정 *(계속)*

```
-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 `gozer`에 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
         /read_acl/synchronize:allow
     3:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

다음 예에서 사용자 `gozer`에 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.2`에 존재합니다.

```
# ls -v file.2
-rw-r--r--   1 root     root        2693 Jul 20 14:26 file.2
```

예 7–3 ZFS 파일의 권한과 ACL 상호 작용 *(계속)*

```
     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.

```
# 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
```

예 7–3 ZFS 파일의 권한과 ACL 상호 작용 *(계속)*

```
        /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.dir`에 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
     4:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
         /read_acl/synchronize:allow
```

다음 예에서 사용자 `gozer` 및 `lp`에 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
```

예 7–5 ZFS 파일에 ACL 세트 적용

ACL 권한을 개별적으로 적용할 필요가 없도록 ACL 세트를 사용할 수 있습니다. ACL 세트에 대한 자세한 내용은 222 페이지 “ZFS ACL 세트”를 참조하십시오.

예를 들어, 다음과 같이 `read_set`를 적용할 수 있습니다.

```
# chmod A+user:otto:read_set:allow file.1
# ls -v file.1
-r--r--r--+  1 root     root      206695 Jul 20 13:43 file.1
     0:user:otto:read_data/read_xattr/read_attributes/read_acl:allow
```

예 7–5 ZFS 파일에 ACL 세트 적용 *(계속)*

```
     1:owner@:read_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
```

다음과 같이 `write_set` 및 `read_set`를 적용할 수 있습니다.

```
# chmod A+user:otto:read_set/write_set:allow file.2
# ls -v file.2
-rw-r--r--+  1 root     root        2693 Jul 20 14:26 file.2
     0:user:otto:read_data/write_data/append_data/read_xattr/write_xattr
         /read_attributes/write_attributes/read_acl: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
```

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 `aclinherit`는 `restricted`로 설정됩니다.

자세한 내용은 222 페이지 “ACL 상속”을 참조하십시오.

예 7–6 기본 ACL 상속 부여

기본적으로 ACL은 디렉토리 구조를 통해 전파되지 않습니다.

다음 예에서 `read_data/write_data/execute`의 복잡한 ACE가 `test.dir`의 사용자 `gozer`에 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
```

예 7–6 기본 ACL 상속 부여 *(계속)*

```
        /read_acl/synchronize:allow
     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 7–7 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
```

예 7–7 파일 및 디렉토리에 ACL 상속 부여 *(계속)*

```
     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
```

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 Jul 20 14:57 test2.dir/subdir.2
     0:user:gozer:list_directory/read_data/add_file/write_data:file_inherit
         /inherit_only/inherited: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 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
```

예 7–7 파일 및 디렉토리에 ACL 상속 부여 *(계속)*

아래 출력에서 inherited 텍스트는 ACE가 상속되었음을 나타내는 정보 메시지입니다.

```
# touch test3.dir/file.3
# ls -v test3.dir/file.3
-rw-r--r--+  1 root     root           0 Jul 20 15:01 test3.dir/file.3
     0:user:gozer:read_data:inherited: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
```

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 Jul 20 15:09 test4.dir/file.4
     0:user:gozer:read_data:inherited: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 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.5
# ls -v test4.dir/file.5
-rw-r--r--+  1 root     root           0 Jul 20 14:16 test4.dir/file.5
     0:user:gozer:read_data/write_data/execute:inherited: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 7–9 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
         /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 7–10 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 Jul 20 14:23 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      206695 Jul 20 14:27 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 226 페이지 “Verbose 형식으로 ZFS 파일에서 ACL 설정 및 표시”를 참조하십시오.

예 7–11 Compact 형식으로 ACL 설정 및 표시

다음 예에서 단순 ACL이 file.1에 존재합니다.

```
# ls -V file.1
-rw-r--r--   1 root     root      206695 Jul 20 14:27 file.1
                owner@:rw-p--aARWcCos:-------:allow
                group@:r-----a-R-c--s:-------:allow
             everyone@:r-----a-R-c--s:-------:allow
```

이 예에서 file.1의 사용자 gozer에 대해 read_data/execute 권한이 추가됩니다.

```
# chmod A+user:gozer:rx:allow file.1
# ls -V file.1
-rw-r--r--+  1 root     root      206695 Jul 20 14:27 file.1
            user:gozer:r-x-----------:-------:allow
                owner@:rw-p--aARWcCos:-------:allow
                group@:r-----a-R-c--s:-------:allow
```

예 7–11 Compact 형식으로 ACL 설정 및 표시 *(계속)*

```
             everyone@:r-----a-R-c--s:-------:allow
```

다음 예에서 사용자 gozer에 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 Jul 20 14:33 dir.2
             user:gozer:rwx-----------:fd-----:allow
                 owner@:rwxp-DaARWcCos:-------: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 gozer에 대한 dir.2의 권한 및 상속 플래그를 dir.2의 사용자 cindy로 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 Jul 20 14:33 dir.2
             user:cindy:rwx-----------:fd-----:allow
             user:gozer:rwx-----------:fd-----:allow
                 owner@:rwxp-DaARWcCos:-------:allow
                 group@:r-x---a-R-c--s:-------:allow
              everyone@:r-x---a-R-c--s:-------:allow
```

예 7–12 ACL 상속 모드를 Pass Through로 설정한 채 ACL 상속

aclinherit 등록 정보가 passthrough로 설정된 파일 시스템은 상속 당시 ACL 항목에 어떤 수정도 없이 모든 상속 가능한 ACL 항목을 상속받습니다. 이 등록 정보가 passthrough로 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 passthrough로 설정한 채 권한 비트를 상속하는 방법을 보여줍니다.

이 예에서 ACL은 상속을 강제하도록 test1.dir에 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
```

예 7–12 ACL 상속 모드를 Pass Through로 설정한 채 ACL 상속 *(계속)*

```
# chmod A=owner@:rwxpcCosRrWaAdD:fd:allow,group@:rwxp:fd:allow,
everyone@::fd:allow test1.dir
# ls -Vd test1.dir
drwxrwx---+  2 root     root           2 Jul 20 14:42 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 Jul 20 14:44 file.1
                 owner@:rwxpdDaARWcCos:------I:allow
                 group@:rwxp----------:------I:allow
              everyone@:--------------:------I: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 Jul 20 14:45 subdir.1
                 owner@:rwxpdDaARWcCos:fd----I:allow
                 group@:rwxp----------:fd----I:allow
              everyone@:--------------:fd----I:allow
```

`fd----I` 항목은 상속을 전파하기 위한 것이며 액세스 제어 중 고려되지 않습니다.

다음 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 Jul 20 14:48 file.2
                 owner@:rw-p--aARWcCos:-------:allow
                 group@:r-----a-R-c--s:-------:allow
              everyone@:r-----a-R-c--s:-------:allow
```

예 7–13 ACL 상속 모드를 Pass Through-X로 설정한 채 ACL 상속

`aclinherit=passthrough-x`가 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).

예 7–13 ACL 상속 모드를 Pass Through-X로 설정한 채 ACL 상속 *(계속)*

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 Jul 20 14:50 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 Jul 20 14:52 test1.dir/file1
                owner@:rw-pdDaARWcCos:------I:allow
                group@:rw-p----------:------I:allow
             everyone@:----------:------I: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:------I:allow
                group@:rwxp----------:------I:allow
             everyone@:----------:------I:allow
```

결과 권한은 0770입니다. cc가 권한 0777을 요청했기 때문이며, 이에 따라 owner@, group@, everyone@ 항목에서 실행 권한이 상속됩니다.

예 7–14 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.

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.

예 7–14 ZFS 파일에서 chmod 작업과의 ACL 상호 작용 (계속)

```
# 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.

```
$ 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
```

예 7–14 ZFS 파일에서 chmod 작업과의 ACL 상호 작용 *(계속)*

```
                owner@:rwxp--aARWcCos:-------:allow
                group@:rwxp--aARWc--s:-------:allow
              everyone@:------a-R-c--s:-------:allow
```

ZFS 파일에 특수 속성 적용

다음 예에서는 변경 불가능 또는 읽기 전용 액세스와 같은 특수 속성을 ZFS 파일에 적용하고 표시하는 방법을 보여줍니다.

특수 속성 표시 및 적용에 대한 자세한 내용은 ls(1) 및 chmod(1)을 참조하십시오.

예 7–15 ZFS 파일에 변경 불가능 적용

파일을 변경할 수 없게 하려면 다음 구문을 사용합니다.

```
# chmod S+ci file.1
# echo this >>file.1
-bash: file.1: Not owner
# rm file.1
rm: cannot remove ‘file.1’: Not owner
```

다음 구문을 사용하여 ZFS 파일의 특수 속성을 표시할 수 있습니다.

```
# ls -l/c file.1
-rw-r--r--+  1 root     root      206695 Jul 20 14:27 file.1
                {A-----im----}
```

파일 변경 불가능을 제거하려면 다음 구문을 사용합니다.

```
# chmod S-ci file.1
# ls -l/c file.1
-rw-r--r--+  1 root     root      206695 Jul 20 14:27 file.1
                {A------m----}
# rm file.1
```

예 7–16 ZFS 파일에 읽기 전용 액세스 적용

다음 예에서는 ZFS 파일에 읽기 전용 액세스를 적용하는 방법을 보여줍니다.

```
# chmod S+cR file.2
# echo this >>file.2
-bash: file.2: Not owner
```

예 7–17 ZFS 파일 속성 표시 및 변경

다음 구문을 사용하여 특수 속성을 표시하고 설정할 수 있습니다.

예 7–17 ZFS 파일 속성 표시 및 변경 *(계속)*

```
# ls -l/v file.3
-r--r--r--   1 root     root      206695 Jul 20 14:59 file.3
                {archive,nohidden,noreadonly,nosystem,noappendonly,nonodump,
noimmutable,av modified,noav_quarantined,nonounlink,nooffline,nosparse}
# chmod S+cR file.3
# ls -l/v file.3
-r--r--r--   1 root     root      206695 Jul 20 14:59 file.3
                {archive,nohidden,readonly,nosystem,noappendonly,nonodump,noimmutable,
av_modified,noav_quarantined,nonounlink,nooffline,nosparse}
```

이러한 속성 중 일부는 Oracle Solaris SMB 환경에서만 적용됩니다.

파일의 모든 속성을 지울 수 있습니다. 예를 들면 다음과 같습니다.

```
# chmod S-a file.3
# ls -l/v file.3
-r--r--r--   1 root     root      206695 Jul 20 14:59 file.3
                {noarchive,nohidden,noreadonly,nosystem,noappendonly,nonodump,
noimmutable,noav_modified,noav_quarantined,nonounlink,nooffline,nosparse}
```

◆ ◆ ◆ 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.

- 245 페이지 “ZFS 위임 관리 개요”
- 246 페이지 “ZFS 권한 위임”
- 254 페이지 “ZFS 위임 권한 표시(예)”
- 250 페이지 “ZFS 권한 위임(예)”
- 255 페이지 “ZFS 위임 권한 제거(예)”

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 사용한 일반적인 Oracle Solaris 관리 작업 관리에 대한 자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 제III부, "역할, 권한 프로파일 및 권한"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 share.nfs를 모두 포함해야 합니다. SMB 공유를 만들려면 share 및 share.smb를 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
- key
- keychange
- userprop
- userquota
- userused

또한 다음과 같은 ZFS 등록 정보 관리를 비루트 사용자에게 위임할 수 있습니다.

- aclinherit
- aclmode
- atime
- canmount
- casesensitivity
- **체크섬**
- compression
- copies
- dedup
- devices
- encryption
- exec
- keysource
- logbias
- mountpoint
- nbmand
- normalization
- primarycache
- quota
- readonly
- recordsize
- refquota
- refreservation
- reservation
- rstchown
- secondarycache
- setuid
- shadow
- share.nfs
- share.smb
- 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 137 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
```

예 8–2 그룹에 create 및 destroy 권한 위임 *(계속)*

```
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
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.

```
# 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 `mark`는 `staff` 그룹에 속하므로 `tank/home`에서 파일 시스템을 만들 수 있습니다. 또한 사용자 `mark`는 관련 권한을 가지고 있으므로 `tank/home/mark2`의 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 `mark`는 관련 권한을 가지고 있지 않으므로 `tank/home/mark`에서 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 `mark`는 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
```

예 8–4 복잡한 위임 권한 정의 및 사용 *(계속)*

```
# 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
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 cindy가 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 tom에게 @eng 권한 세트가 위임되고 사용자 joe에게 로컬 파일 시스템에 대한 create, destroy 및 mount 권한이 부여됩니다.
- 사용자 fred에게 로컬 및 종속 파일 시스템에 대한 @basic 권한 세트와 share 및 rename 권한이 위임됩니다.
- 사용자 barney와 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.

- 257 페이지 “ZFS 볼륨”
- 260 페이지 “영역이 설치된 Solaris 시스템에서 ZFS 사용”
- 265 페이지 “ZFS 대체 루트 풀 사용”

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 273 페이지 "ZFS 관련 문제 해결"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 262 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 122 페이지 "ZFS 스왑 및 덤프 장치의 크기 조정"을 참조하십시오.

ZFS 볼륨을 iSCSI LUN으로 사용

COMSTAR(Common Multiprotocol SCSI Target) 소프트웨어 프레임워크를 사용하면 개시자 호스트가 저장소 네트워크를 통해 액세스할 수 있는 SCSI 대상 장치로 Oracle Solaris 호스트를 변환할 수 있습니다. ZFS 볼륨을 만들고 iSCSI LUN(논리 장치)으로 공유되도록 구성할 수 있습니다.

먼저 COMSTAR 패키지를 설치합니다.

```
# pkg install group/feature/storage-server
```

iSCSI 대상으로 사용할 ZFS 볼륨을 만든 다음 SCSI 블록 장치 기반 LUN을 만듭니다. 예를 들면 다음과 같습니다.

```
# zfs create -V 2g tank/volumes/v2
# sbdadm create-lu /dev/zvol/rdsk/tank/volumes/v2
Created the following LU:

              GUID                    DATA SIZE           SOURCE
--------------------------------  -------------------  ----------------
600144f000144f1dafaa4c0faff20001  2147483648           /dev/zvol/rdsk/tank/volumes/v2
# sbdadm list-lu
Found 1 LU(s)

              GUID                    DATA SIZE           SOURCE
--------------------------------  -------------------  ----------------
600144f000144f1dafaa4c0faff20001  2147483648           /dev/zvol/rdsk/tank/volumes/v2
```

LUN 뷰를 모든 클라이언트나 선택한 클라이언트에 노출할 수 있습니다. LUN GUID를 식별하고 LUN 뷰를 공유합니다. 다음 예에서 LUN 뷰는 모든 클라이언트에 공유됩니다.

```
# stmfadm list-lu
LU Name: 600144F000144F1DAFAA4C0FAFF20001
# stmfadm add-view 600144F000144F1DAFAA4C0FAFF20001
# stmfadm list-view -l 600144F000144F1DAFAA4C0FAFF20001
View Entry: 0
    Host group   : All
    Target group : All
    LUN          : 0
```

다음 단계에서는 iSCSI 대상을 만듭니다. iSCSI 대상 만들기에 대한 자세한 내용은 **Oracle Solaris 11.1 관리: 장치 및 파일 시스템**의 11 장, “COMSTAR를 사용하여 저장 장치 구성(작업)”을 참조하십시오.

iSCSI 대상인 ZFS 볼륨은 데이터 세트의 이름을 바꾸거나, 볼륨 스냅샷을 롤백하거나, ZFS 볼륨이 iSCSI LUN으로 공유되는 동안 풀을 내보낼 수 없다는 점을 제외하고 다른 ZFS 데이터 세트와 동일하게 관리됩니다. 다음과 유사한 메시지가 표시됩니다.

```
# zfs rename tank/volumes/v2 tank/volumes/v1
cannot rename 'tank/volumes/v2': dataset is busy
# zpool export tank
cannot export 'tank': pool is busy
```

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.

- 261 페이지 "비전역 영역에 ZFS 파일 시스템 추가"
- 261 페이지 "비전역 영역에 데이터 세트 위임"
- 262 페이지 "비전역 영역에 ZFS 볼륨 추가"
- 263 페이지 "영역 내에서 ZFS 저장소 풀 사용"
- 263 페이지 "영역 내에서 ZFS 등록 정보 관리"
- 264 페이지 "zoned 등록 정보 이해"

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.
- 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 **Oracle Solaris 관리: Oracle Solaris Zones, Oracle Solaris 10 Zones 및 리소스 관리**의 제II부, "Oracle Solaris Zones"을 참조하십시오.

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 zonecfg를 사용한 리소스 유형 구성에 대한 자세한 내용은 **Oracle Solaris 관리: Oracle Solaris Zones, Oracle Solaris 10 Zones 및 리소스 관리**의 제II부, "Oracle Solaris Zones"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> set alias=tank
zonecfg:zion:dataset> end
```

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 절 및 dev(7FS)를 참조하십시오.

비전역 영역에 ZFS 볼륨 추가

다음과 같은 방법으로 비전역 영역에 ZFS 볼륨을 추가 또는 생성하거나 비전역 영역의 볼륨 데이터에 대한 액세스를 추가할 수 있습니다.

- 권한 있는 영역 관리자는 비전역 영역에 ZFS 볼륨을 이전에 위임된 파일 시스템의 종속 항목으로 만들 수 있습니다. 예를 들면 다음과 같습니다.

  ```
  # zfs create -V 2g tank/zone/zion/vol1
  ```

  위 구문은 영역 관리자가 비전역 영역의 볼륨 등록 정보와 데이터를 관리할 수 있음을 의미합니다.

- 전역 영역에서 zonecfg add dataset 하위 명령을 사용하고 비전역 영역에 추가할 ZFS 볼륨을 지정합니다. 예를 들면 다음과 같습니다.

  ```
  # zonecfg -z zion
  zonecfg:zion> add dataset
  zonecfg:zion:dataset> set name=tank/volumes/vol1
  zonecfg:zion:dataset> end
  ```

  위 구문은 영역 관리자가 비전역 영역의 볼륨 등록 정보와 데이터를 관리할 수 있음을 의미합니다.

- 전역 영역에서 zonecfg add device 하위 명령을 사용하고 비전역 영역에서 해당 데이터에 액세스할 수 있는 ZFS 볼륨을 지정합니다. 예를 들면 다음과 같습니다.

  ```
  # zonecfg -z zion
  zonecfg:zion> add device
  zonecfg:zion:device> set match=/dev/zvol/dsk/tank/volumes/vol2
  zonecfg:zion:device> end
  ```

  위 구문은 비전역 영역에서 볼륨 데이터만 액세스할 수 있음을 의미합니다.

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/zion`이 기본 `zion` 별칭으로 영역에 추가된 경우 각 데이터 세트는 다음 등록 정보를 갖게 됩니다.

데이터 세트	표시 가능	쓰기 가능	변경할 수 없는 등록 정보
`tank`	아니오	-	-
`tank/home`	아니오	-	-
`tank/data`	아니오	-	-
`tank/data/zion`	예	예	`zoned`, `quota`, `reservation`
`tank/data/zion/home`	예	예	`zoned`

`tank/zone/zion`의 모든 부모는 보이지 않으며 모든 자손은 쓰기 가능합니다. 영역 관리자는 `zoned` 등록 정보를 변경할 수 없습니다. 그러면 다음 절에 설명된 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 세트가 비전역 영역에 위임된 후에는 전역 영역 관리자가 share.nfs 및 mountpoint 등록 정보를 변경할 수 없습니다.

zoned 등록 정보 이해

데이터 세트가 비전역 영역에 위임될 때 특정 등록 정보가 전역 영역의 컨텍스트에서 해석되지 않도록 데이터 세트를 특별히 표시해야 합니다. 데이터 세트가 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 세트가 비전역 영역에 위임되었는지 나타냅니다.

zoned 등록 정보는 ZFS 데이터 세트를 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 세트를 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.

다른 시스템에 영역 복사

하나 이상의 영역을 다른 시스템으로 마이그레이션해야 하는 경우 `zfs send` 및 `zfs receive` 명령을 사용합니다. 시나리오에 따라 복제 스트림 또는 순환적 스트림을 사용하는 것이 좋습니다.

이 절의 예에서는 시스템 간에 영역 데이터를 복사하는 방법에 대해 설명합니다. 각 영역의 구성을 전송하고 각 영역을 새 시스템에 연결하려면 추가 단계를 수행해야 합니다. 자세한 내용은 **Oracle Solaris 11.1 관리: Oracle Solaris 영역, Oracle Solaris 10 영역 및 리소스 관리**의 제II부, "Oracle Solaris 영역"을 참조하십시오.

한 시스템의 모든 영역을 다른 시스템으로 이동해야 하는 경우 스냅샷과 복제가 유지되므로 복제 스트림을 사용합니다. 스냅샷과 복제는 `pkg update`, `beadm create` 및 `zoneadm clone` 명령에서 광범위하게 사용됩니다.

다음 예에서 `sysA`의 영역은 `rpool/zones` 파일 시스템에 설치되며 `sys`의 `tank/zones` 파일 시스템에 복사해야 합니다. 다음 명령은 스냅샷을 만들고 복제 스트림을 사용하여 데이터를 `sysB`에 복사합니다.

```
sysA# zfs snapshot -r rpool/zones@send-to-sysB
sysA# zfs send -R rpool/zones@send-to-sysB | ssh sysB zfs receive -d tank
```

다음 예에서 여러 영역 중 하나는 `sysC` 에서 `sysD`로 복사됩니다. `ssh` 명령을 사용할 수 없지만 NFS 서버 인스턴스를 사용할 수 있다고 가정합니다. 다음 명령을 사용하면 영역이 다른 영역의 복제본인지 여부에 관계없이 순환적 `zfs send` 스트림을 생성할 수 있습니다.

```
sysC# zfs snapshot -r rpool/zones/zone1@send-to-nfs
sysC# zfs send -rc rpool/zones/zone1@send-to-nfs > /net/nfssrv/export/scratch/zone1.zfs
sysD# zfs create tank/zones
sysD# zfs receive -d tank/zones < /net/nfssrv/export/scratch/zone1.zfs
```

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.

- 267 페이지 "ZFS 공간 문제 해결"
- 269 페이지 "ZFS 오류 식별"
- 270 페이지 "ZFS 파일 시스템 무결성 검사"
- 273 페이지 "ZFS 관련 문제 해결"
- 278 페이지 "손상된 ZFS 구성 복구"
- 278 페이지 "누락된 장치 해결"
- 282 페이지 "손상된 장치 교체 또는 복구"
- 291 페이지 "손상된 데이터 복구"
- 296 페이지 "부트할 수 없는 시스템 복구"

완전한 루트 풀 복구에 대한 자세한 내용은 11 장, "스냅샷 아카이브 및 루트 풀 복구"를 참조하십시오.

ZFS 공간 문제 해결

ZFS의 파일 시스템 및 풀 공간 계산 보고 방법이 확실하지 않은 경우 다음 절을 참조하십시오. 또한 32 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 ALLOC에 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
```

ZFS 오류 식별

ZFS는 결합된 파일 시스템 및 볼륨 관리자로서 여러 가지 오류가 발생할 수 있습니다. 이 장에서는 먼저 다양한 오류에 대해 간략히 설명한 다음 실행 중인 시스템에서 이러한 오류를 식별하는 방법에 대해 설명합니다. 끝으로, 문제를 복구하는 방법에 대해 설명합니다. ZFS에서는 다음 세 가지 유형의 오류가 발생할 수 있습니다.

- 269 페이지 “ZFS 저장소 풀에서 장치 누락”
- 270 페이지 “ZFS 저장소 풀에서 장치 손상”
- 270 페이지 “손상된 ZFS 데이터”

하나의 풀에서 이 세 가지 오류가 모두 발생할 수 있으므로, 전체 복구 절차는 하나의 오류를 찾아 해결한 다음 그 다음 오류로 진행하는 방식으로 진행됩니다.

ZFS 저장소 풀에서 장치 누락

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

ZFS 저장소 풀에서 장치 손상

"손상"이라는 용어에는 발생 가능한 다양한 오류가 포함됩니다. 예를 들면 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.

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 파일 시스템 무결성 검사

`fsck`에 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 294 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
 scan: scrub in progress since Mon Jun  7 12:07:52 2010
    201M scanned out of 222M at 9.55M/s, 0h0m to go
    0 repaired, 90.44% done
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 83 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 289 페이지 “리실버링 상태 보기”를 참조하십시오.

ZFS 관련 문제 해결

다음 절에서는 ZFS 파일 시스템 또는 저장소 풀 관련 문제를 식별하고 해결하는 방법에 대해 설명합니다.

- 274 페이지 “ZFS 저장소 풀에 문제가 있는지 확인”
- 274 페이지 “zpool status 출력 결과 검토”
- 277 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':
  2010-07-15.12:06:50 zpool create tank mirror c0t1d0 c0t2d0 c0t3d0
  2010-07-15.12:06:58 zfs create tank/eric
  2010-07-15.12:07:01 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 83 페이지 “ZFS 저장소 풀 상태 질의”를 참조하십시오.

zpool status 출력 결과 검토

전체 zpool status 출력 결과는 다음과 비슷합니다.

```
# zpool status pond
  pool: pond
 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: Determine if the device needs to be replaced, and clear the errors
        using 'zpool clear' or 'fmadm repaired', or replace the device
        with 'zpool replace'.
        Run 'zpool status -v' to see device specific details.
  scan: scrub repaired 0 in 0h0m with 0 errors on Wed Jun 20 13:16:09 2012
config:

        NAME                       STATE     READ WRITE CKSUM
        pond                       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

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.

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 282 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 289 페이지 "리실버링 상태 보기"를 참조하십시오.

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.

다음 `zpool status` 스크러빙 상태 메시지가 제공됩니다.

- 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
```

- 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 완료 메시지는 시스템 재부트 후에도 보존됩니다.

데이터 스크러빙 및 이 정보를 해석하는 방법에 대한 자세한 내용은 270 페이지 "ZFS 파일 시스템 무결성 검사"를 참조하십시오.

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 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://support.oracle.com/msg/ZFS-8000-8A
  scan: scrub repaired 0 in 0h0m with 2 errors on Fri Jun 29 16:58:58 2012
config:

        NAME            STATE     READ WRITE CKSUM
        tank            ONLINE       2     0     0
          c8t0d0        ONLINE       0     0     0
          c8t1d0        ONLINE       2     0     0

errors: Permanent errors have been detected in the following files:

        /tank/file.1
```

`fmd`에 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 292 페이지 “데이터 손상 유형 식별”을 참조하십시오.

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.

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 95 페이지 “ZFS 저장소 풀 마이그레이션”을 참조하십시오.

누락된 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** – 124 페이지 “미러링된 ZFS 루트 풀의 대체 디스크에서 부트”
- **루트 풀에서 디스크 교체**
  - 115 페이지 “ZFS 루트 풀의 디스크 교체 방법(SPARC 또는 x86/VTOC)”
  - 117 페이지 “ZFS 루트 풀의 디스크 교체 방법(SPARC 또는 x86/EFI(GPT))”
- **완전한 루트 풀 재해 복구** – 11 장, “스냅샷 아카이브 및 루트 풀 복구”

예를 들어, 장치 오류 후 `fmd`에서 다음과 비슷한 메시지를 표시할 수 있습니다.

```
SUNW-MSG-ID: ZFS-8000-QJ, TYPE: Fault, VER: 1, SEVERITY: Minor
EVENT-TIME: Wed Jun 20 13:09:55 MDT 2012
PLATFORM: ORCL,SPARC-T3-4, CSN: 1120BDRCCD, HOSTNAME: tardis
SOURCE: zfs-diagnosis, REV: 1.0
EVENT-ID: e13312e0-be0a-439b-d7d3-cddaefe717b0
DESC: Outstanding dtls on ZFS device 'id1,sd@n5000c500335dc60f/a' in pool 'pond'.
AUTO-RESPONSE: No automated response will occur.
IMPACT: None at this time.
REC-ACTION: Use 'fmadm faulty' to provide a more detailed view of this event.
Run 'zpool status -lx' for more information. Please refer to the associated
reference document at http://support.oracle.com/msg/ZFS-8000-QJ for the latest
service procedures and policies regarding this diagnosis.
```

장치 문제 및 해결 방법에 대한 자세한 정보를 보려면 `zpool status -v` 명령을 사용하십시오. 예를 들면 다음과 같습니다.

```
# zpool status -v
  pool: pond
 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: Determine if the device needs to be replaced, and clear the errors
        using 'zpool clear' or 'fmadm repaired', or replace the device
        with 'zpool replace'.
  scan: scrub repaired 0 in 0h0m with 0 errors on Wed Jun 20 13:16:09 2012
config:

        NAME                      STATE     READ WRITE CKSUM
        pond                      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

device details:

        c0t5000C500335DC60Fd0    UNAVAIL          cannot open
        status: ZFS detected errors on this device.
                The device was missing.
           see: http://support.oracle.com/msg/ZFS-8000-LR for recovery
```

이 출력 결과에서는 `c0t5000C500335DC60Fd0` 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.

`fmadm faulty` 출력에서 장치 오류가 식별되는 경우 장치가 교체되었음을 FMA에 알립니다. 예를 들면 다음과 같습니다.

```
# fmadm faulty
--------------- ------------------------------------  -------------- ---------
```

```
TIME            EVENT-ID                              MSG-ID         SEVERITY
--------------- ------------------------------------  -------------- ---------
Jun 20 13:15:41 3745f745-371c-c2d3-d940-93acbb881bd8  ZFS-8000-LR    Major

Problem Status    : solved
Diag Engine       : zfs-diagnosis / 1.0
System
    Manufacturer  : unknown
    Name          : ORCL,SPARC-T3-4
    Part_Number   : unknown
    Serial_Number : 1120BDRCCD
    Host_ID       : 84a02d28

----------------------------------------
Suspect 1 of 1 :
   Fault class : fault.fs.zfs.open_failed
   Certainty   : 100%
   Affects     : zfs://pool=86124fa573cad84e/vdev=25d36cd46e0a7f49/pool_name=pond/vdev_
name=id1,sd@n5000c500335dc60f/a
   Status      : faulted and taken out of service

   FRU
     Name             : "zfs://pool=86124fa573cad84e/vdev=25d36cd46e0a7f49/pool_name=pond/vdev_
name=id1,sd@n5000c500335dc60f/a"
        Status        : faulty

Description : ZFS device 'id1,sd@n5000c500335dc60f/a' in pool 'pond' failed to
              open.

Response    : An attempt will be made to activate a hot spare if available.

Impact      : Fault tolerance of the pool may be compromised.

Action      : Use 'fmadm faulty' to provide a more detailed view of this event.
              Run 'zpool status -lx' for more information. Please refer to the
              associated reference document at
              http://support.oracle.com/msg/ZFS-8000-LR for the latest service
              procedures and policies regarding this diagnosis.
```

`fmadm faulty` 출력 결과에서 `Affects:` 절의 문자열을 추출하고 이를 다음 명령에 포함하여 FMA에 해당 장치가 교체되었음을 알립니다.

```
# fmadm repaired zfs://pool=86124fa573cad84e/vdev=25d36cd46e0a7f49/pool_name=pond/vdev_
name=id1,sd@n5000c500335dc60f/a
fmadm: recorded repair to of zfs://pool=86124fa573cad84e/vdev=25d36cd46e0a7f49/pool_name=pond/vdev_
name=id1,sd@n5000c500335dc60f/a
```

끝으로, 교체된 장치를 포함하는 풀이 정상적으로 작동하는지 확인하십시오. 예를 들면 다음과 같습니다.

```
# zpool status -x tank
pool 'tank' is healthy
```

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. 282 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'.
        Run 'zpool status -v' to see device specific details.
   see: http://support.oracle.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 71 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://support.oracle.com/msg/ZFS-8000-8A
config:

        NAME             STATE     READ WRITE CKSUM
        tank             ONLINE       2     0     0
          c8t0d0         ONLINE       0     0     0
          c8t0d0         ONLINE       2     0     0

errors: Permanent errors have been detected in the following files:

        /tank/file.1
```

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 291 페이지 “손상된 데이터 복구”를 참조하십시오.

일시적인 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 72 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"
- 289 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             UNAVAIL
```

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.

```
    raidz1              UNAVAIL
    c1t0d0              ONLINE
    c2t0d0              UNAVAIL
    c3t0d0              UNAVAIL
    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 294 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 장치를 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 73 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 73 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.
- 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 스패어로 교체된 경우 실패한 디스크가 교체된 후 핫 스패어를 분리해야 합니다. 예를 들어 실패한 디스크를 교체한 후에도 여전히 c2t4d0가 활성 핫 스패어인 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
```

예 10–1 ZFS 저장소 풀의 SATA 디스크 교체 *(계속)*

```
<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
            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
```

예 10–1 ZFS 저장소 풀의 SATA 디스크 교체 *(계속)*

```
            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
 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.

예 10–2 실패한 로그 장치 교체 **(계속)**

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.

다음 `zpool status` 리실버링 상태 메시지가 제공됩니다.

- 리실버링 진행 중 보고입니다. 예를 들면 다음과 같습니다.

```
scan: resilver in progress since Mon Jun  7 09:17:27 2010
   13.3G scanned out of 16.2G at 18.5M/s, 0h2m to go
   13.3G resilvered, 82.34% done
```

- 리실버링 완료 메시지입니다. 예를 들면 다음과 같습니다.

```
resilvered 16.2G in 0h16m with 0 errors on Mon Jun  7 09:34:21 2010
```

리실버링 완료 메시지는 시스템 재부트 후에도 보존됩니다.

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: 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.
 scan: resilver in progress since Mon Jun  7 10:49:20 2010
    54.6M scanned out of 222M at 5.46M/s, 0h0m to go
    54.5M resilvered, 24.64% done
config:

        NAME             STATE     READ WRITE CKSUM
        tank             ONLINE       0     0     0
          mirror-0       ONLINE       0     0     0
            replacing-0  ONLINE       0     0     0
              c1t0d0     ONLINE       0     0     0
              c2t0d0     ONLINE       0     0     0  (resilvering)
            c1t1d0       ONLINE       0     0     0
```

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.

손상된 데이터 복구

다음 절에서는 데이터 손상 유형을 식별하고 데이터를 복구하는 방법에 대해 설명합니다.

- 292 페이지 “데이터 손상 유형 식별”
- 293 페이지 “손상된 파일 또는 디렉토리 복구”
- 294 페이지 “ZFS 저장소 풀 전반의 손상 복구”

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 270 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 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://support.oracle.com/msg/ZFS-8000-8A
config:

        NAME                        STATE     READ WRITE CKSUM
        tank                        ONLINE       4     0     0
          c0t5000C500335E106Bd0     ONLINE       0     0     0
          c0t5000C500335FC3E7d0     ONLINE       4     0     0


errors: 2 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: 13289416187275223932
 state: UNAVAIL
status: The pool metadata is corrupted.
action: The pool cannot be imported due to damaged devices or data.
   see: http://support.oracle.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 tank -v
  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://support.oracle.com/msg/ZFS-8000-8A
config:

        NAME                        STATE     READ WRITE CKSUM
        tank                        ONLINE       4     0     0
          c0t5000C500335E106Bd0     ONLINE       0     0     0
          c0t5000C500335FC3E7d0     ONLINE       4     0     0

errors: Permanent errors have been detected in the following files:
        /tank/file.1
        /tank/file.2
```

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_t의 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 스냅샷에서와 같이 여러 블록 참조가 있는 손상된 데이터가 있는 경우 zpool status -v 명령은 손상된 데이터 경로를 **모두** 표시하지는 않습니다. ZFS 스크럽 알고리즘은 풀을 순회하고 데이터의 각 블록을 한번만 방문합니다. 손상이 발생하면 **처음**에만 손상을 보고할 수 있습니다. 따라서 영향을 받는 파일에 대한 단일 경로만 생성합니다. 이는 중복이 제거된 손상된 블록에도 적용됩니다.

손상된 데이터가 있고 zpool status - v 명령이 스냅샷 데이터가 영향을 받았음을 식별하는 경우 추가로 손상된 경로가 있는지 검색해 보십시오.

```
# find mount-point -inum $inode -print
# find mount-point/.zfs/snapshot -inum $inode -print
```

첫번째 명령은 지정된 파일 시스템 및 모든 스냅샷에서 보고된 손상 데이터의 inode 번호를 검색합니다. 두번째 명령은 동일한 inode 번호가 있는 스냅샷을 검색합니다.

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: UNAVAIL
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 Fri Jun 29 17:22:49 2012
        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://support.oracle.com/msg/ZFS-8000-72
 scrub: none requested
config:
```

```
        NAME        STATE     READ WRITE CKSUM
         tpool      UNAVAIL      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 Fri Jun 29 17:22:49 2012
        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 Fri Jun 29 17:22:49 2012.
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 101 페이지 "읽기 전용 모드로 풀 가져오기"를 참조하십시오.

- zpool import -m 명령을 사용하여 누락된 로그 장치가 있는 풀을 가져올 수 있습니다. 자세한 내용은 100 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.

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.

◆ ◆ ◆ 11장

11 스냅샷 아카이브 및 루트 풀 복구

이 장에서는 시스템 고장 시 Oracle Solaris 11 시스템을 마이그레이션 또는 복원하는 데 사용할 수 있는 스냅샷을 아카이브하는 방법에 대해 설명합니다. 이러한 단계를 사용하여 중요한 기본 재해 복구 계획을 만들거나 시스템의 구성을 새 부트 장치로 마이그레이션할 수 있습니다.

이 장에서는 다음과 같은 내용을 다룹니다.

- 297 페이지 “ZFS 복구 프로세스 개요”
- 298 페이지 “복구에 사용할 ZFS 스냅샷 아카이브 만들기”
- 300 페이지 “루트 풀 다시 만들기 및 루트 풀 스냅샷 복구”

ZFS 복구 프로세스 개요

적어도 모든 파일 시스템 데이터를 정기적으로 백업하여 시스템 고장으로 인한 작동 중지 시간을 줄여야 합니다. 대규모 시스템 고장이 발생할 경우 OS를 재설치하고 시스템 구성을 다시 만드는 대신 ZFS 루트 풀 스냅샷을 복원할 수 있습니다. 그런 다음 비루트 풀 데이터를 복원합니다.

Oracle Solaris 11을 실행하는 모든 시스템이 백업 및 아카이브 가능 대상입니다. 전체 프로세스에는 다음 단계가 필요합니다.

- 루트 풀 파일 시스템 및 마이그레이션 또는 복구해야 하는 비루트 풀에 대한 ZFS 스냅샷 아카이브를 만듭니다.

  OS를 업데이트한 후 루트 풀 스냅샷을 재아카이브해야 합니다.
- USB 드라이브 등의 로컬 이동식 매체에 스냅샷 아카이브를 저장하거나 잠재적 검색을 위해 스냅샷을 원격 시스템으로 보냅니다.
- 장애가 발생한 디스크 또는 기타 시스템 구성 요소가 교체됩니다.
- 대상 시스템이 Oracle Solaris 11 설치 매체에서 부트되고 새 저장소 풀이 생성되며 파일 시스템이 복구됩니다.

- 최소 부트 구성을 수행하면 시스템을 사용할 수 있으며 아카이브 시 실행하고 있던 모든 서비스가 제공됩니다.

ZFS 풀 복구 요구 사항

- 아카이브된 시스템과 복구 시스템은 동일한 구조여야 하며 지원되는 플랫폼에 대한 Oracle Solaris 11 최소 요구 사항을 충족해야 합니다.
- 새 ZFS 저장소 풀이 포함될 교체 디스크는 용량이 적어도 아카이브된 풀에 사용된 데이터만큼 커야 합니다(아래 참조).

  Oracle Solaris 11에서 루트 풀 디스크는 SMI(VTOC) 레이블을 포함해야 합니다. Oracle Solaris 11.1에서 x86 기반 시스템의 루트 풀 디스크는 SMI(VTOC) 또는 EFI(GPT) 디스크 레이블을 포함할 수 있습니다. EFI(GPT) 레이블 디스크의 부트 지원에 대한 자세한 내용은 17 페이지 “EFI(GPT) 레이블 디스크에 대한 부트 지원”을 참조하십시오.
- 아카이브된 스냅샷과 복구 시스템이 포함된 두 시스템에서 모두 루트 액세스가 필요합니다. ssh를 사용하여 원격 시스템에 액세스하는 경우 권한 있는 액세스를 위해 구성해야 합니다.

복구에 사용할 ZFS 스냅샷 아카이브 만들기

ZFS 루트 풀 스냅샷을 만들기 전에 다음 정보를 저장하는 것이 좋습니다.

- 루트 풀 등록 정보를 캡처합니다.

```
sysA# zpool get all rpool
```

- 루트 풀 디스크의 크기 및 현재 용량을 식별합니다.

```
sysA# zpool list
NAME   SIZE  ALLOC   FREE  CAP  DEDUP  HEALTH  ALTROOT
rpool   74G  5.42G  68.6G   7%  1.00x  ONLINE  -
```

- 루트 풀 구성 요소를 식별합니다.

```
sysA# zfs list -r rpool
NAME                      USED  AVAIL  REFER  MOUNTPOINT
rpool                    13.8G  53.1G  73.5K  /rpool
rpool/ROOT               3.54G  53.1G    31K  legacy
rpool/ROOT/solaris       3.54G  53.1G  3.37G  /
rpool/ROOT/solaris/var    165M  53.1G   163M  /var
rpool/VARSHARE           37.5K  53.1G  37.5K  /var/share
rpool/dump               8.19G  53.4G  7.94G  -
rpool/export               63K  53.1G    32K  /export
rpool/export/home          31K  53.1G    31K  /export/home
rpool/swap               2.06G  53.2G  2.00G  -
```

▼ ZFS 스냅샷 아카이브를 만드는 방법

다음 단계에서는 루트 풀의 모든 파일 시스템이 포함될 루트 풀에 순환적 스냅샷을 만드는 방법에 대해 설명합니다. 다른 비루트 풀도 동일한 방법으로 아카이브할 수 있습니다.

다음 사항을 고려해 보십시오.

- 전체 시스템 복구의 경우 원격 시스템의 풀에 스냅샷을 보냅니다.
- 원격 시스템에서 NFS 공유를 만들고 필요한 경우 권한 있는 액세스도 허용하도록 `ssh`를 구성합니다.
- 순환적 루트 풀 스냅샷이 하나의 큰 스냅샷 파일로 원격 시스템에 전송되지만, 원격 시스템에 개별 스냅샷으로 저장될 순환적 스냅샷도 보낼 수 있습니다.

아래 단계에서 순환적 스냅샷의 이름은 `rpool@snap1`입니다. 복구할 로컬 시스템은 `sysA`이고 원격 시스템은 `sysB`입니다. `rpool`이 기본 루트 풀 이름이며 사용자 시스템에서는 다를 수 있습니다.

1 관리자로 전환합니다.

2 순환적 루트 풀 스냅샷을 만듭니다.

```
sysA# zfs snapshot -r rpool@rpool.snap1
```

3 원하는 경우 스왑 및 덤프 스냅샷을 제거하여 스냅샷 아카이브를 줄입니다.

```
sysA# zfs destroy rpool/dump@rpool.snap1
sysA# zfs destroy rpool/swap@rpool.snap1
```

스왑 볼륨에는 시스템 마이그레이션 또는 복구와 관련된 데이터가 포함되지 않습니다. 충돌 덤프를 유지하려는 경우 덤프 볼륨 스냅샷을 제거하지 마십시오.

4 다른 시스템의 다른 풀에 순환적 루트 풀 스냅샷을 보냅니다.

a. 스냅샷 수신을 위해 원격 시스템의 파일 시스템을 공유합니다.

다음 단계에서는 순환적 루트 스냅샷을 공유하기 위해 `/tank/snaps` 파일 시스템이 공유됩니다.

```
sysB# zfs set share.nfs=on tank/snaps
sysB# zfs set share.nfs.sec.default.root=sysA tank/snaps
```

b. 원격 시스템에 순환적 루트 풀 스냅샷을 보냅니다.

이전 단계에서 공유된 원격 파일 시스템에 순환적 스냅샷을 보냅니다.

```
sysA# zfs send -Rv rpool@rpool.snap1 | gzip > /net/sysB/tank/snaps/
rpool.snap1.gz
sending from @ to rpool@rpool.snap1
sending from @ to rpool/VARSHARE@rpool.snap1
sending from @ to rpool/export@rpool.snap1
sending from @ to rpool/export/home@rpool.snap1
```

```
sending from @ to rpool/ROOT@rpool.snap1
sending from @ to rpool/ROOT/solaris@install
sending from @install to rpool/ROOT/solaris@rpool.snap1
sending from @ to rpool/ROOT/solaris/var@install
sending from @install to rpool/ROOT/solaris/var@rpool.snap1
```

루트 풀 다시 만들기 및 루트 풀 스냅샷 복구

루트 풀을 다시 만들고 루트 풀 스냅샷을 복구해야 하는 경우 일반 단계는 다음과 같습니다.

- 교체 루트 풀 디스크 준비 및 루트 풀 다시 만들기
- 루트 풀 파일 시스템 스냅샷 복원
- 원하는 부트 환경 선택 및 활성화
- 시스템 부트

▼ 복구 시스템에서 루트 풀을 다시 만드는 방법

루트 풀을 복구하는 경우 다음 고려 사항을 검토합니다.

- 중복되지 않는 루트 풀 디스크에서 장애가 발생할 경우 설치 매체 또는 설치 서버에서 시스템을 부트하여 OS를 재설치하거나 이전에 아카이브한 루트 풀 스냅샷을 복원해야 합니다.

  시스템의 디스크 교체에 대한 자세한 내용은 하드웨어 설명서를 참조하십시오.
- 미러링된 루트 풀 디스크에서 장애가 발생할 경우 시스템이 작동하는 동안 장애가 발생한 디스크를 교체할 수 있습니다. 미러링된 루트 풀에서 결함이 있는 디스크 교체에 대한 자세한 내용은 115 페이지 “ZFS 루트 풀의 디스크 교체 방법(SPARC 또는 x86/VTOC)”을 참조하십시오.

1 장애가 발생한 루트 풀 디스크 또는 시스템 구성 요소를 식별하고 교체합니다.

이 디스크는 일반적으로 기본 부트 장치이거나, 다른 디스크를 선택하고 기본 부트 장치를 재설정할 수 있습니다.

2 다음 중 하나를 선택하여 Oracle Solaris 11 설치 매체에서 시스템을 부트합니다.

- DVD 또는 USB 설치 매체(SPARC 또는 x86) – 매체를 삽입하고 적절한 장치를 부트 장치로 선택합니다.

  텍스트 기반 매체를 사용하는 경우 텍스트 설치 프로그램 메뉴에서 Shell(셸) 옵션을 선택합니다.
- 라이브 매체(x86에만 해당) – 복구 프로시저 실행 중 GNOME 데스크탑 세션을 사용할 수 있습니다.
- 자동 설치 프로그램 또는 AI 매체의 로컬 복사본(SPARC 또는 x86) – 텍스트 설치 프로그램 메뉴에서 Shell(셸) 옵션을 선택합니다. SPARC 시스템에서 AI 매체(로컬 또는 네트워크)를 부트하고 Shell(셸) 옵션을 선택합니다.

```
ok boot net:dhcp
.
.
.Welcome to the Oracle Solaris 11 installation menu

        1  Install Oracle Solaris
        2  Install Additional Drivers
        3  Shell
        4  Terminal type (currently xterm)
        5  Reboot

Please enter a number [1]: 3
```

3 SPARC 또는 x86(VTOC): 루트 풀 디스크를 준비합니다.

a. 교체 루트 풀 디스크가 format 유틸리티에 표시되는지 확인합니다.

```
# format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
       0. c2t0d0 <FUJITSU-MAY2073RCSUN72G-0401 cyl 14087 alt 2 hd 24 sec 424>
          /pci@780/pci@0/pci@9/scsi@0/sd@0,0
       1. c2t1d0 <FUJITSU-MAY2073RCSUN72G-0401 cyl 14087 alt 2 hd 24 sec 424>
          /pci@780/pci@0/pci@9/scsi@0/sd@1,0
       2. c2t2d0 <SEAGATE-ST973402SSUN72G-0400-68.37GB>
          /pci@780/pci@0/pci@9/scsi@0/sd@2,0
       3. c2t3d0 <SEAGATE-ST973401LSUN72G-0556-68.37GB>
          /pci@780/pci@0/pci@9/scsi@0/sd@3,0
Specify disk (enter its number): 0
```

b. SPARC 또는 x86(VTOC): 루트 풀 디스크에 SMI(VTOC) 레이블과 대량 디스크 공간이 포함된 슬라이스 0이 있는지 확인합니다.

분할 영역 테이블을 검토하여 루트 풀 디스크에 SMI 레이블과 슬라이스 0이 있는지 확인합니다.

```
selecting c2t0d0
[disk formatted]
format> partition
partition> print
```

c. SPARC 또는 x86 (VTOC): 필요한 경우 SMI(VTOC) 레이블을 사용하여 디스크에 레이블을 재지정합니다.

다음 단축 명령을 사용하여 디스크에 레이블을 재지정합니다. 이러한 명령은 오류 검사를 제공하지 않으므로 올바른 디스크에 레이블을 재지정 중인지 확인합니다.

- SPARC:

  ```
  sysA# format -L vtoc -d c2t0d0
  ```

  슬라이스 0의 디스크 공간이 적절히 할당되었는지 확인합니다. 위 명령에서는 기본 분할 영역이 적용되며 이 영역은 루트 풀 슬라이스 0에 사용하기에 너무 작을 수 있습니다. 기본 분할 영역 테이블 수정에 대한 자세한 내용은 **Oracle Solaris 11.1 관리: 장치 및 파일 시스템**의 "ZFS 루트 풀 디스크(EFI(GPT))를 교체하는 방법"을 참조하십시오.

- x86:

```
sysA# fdisk -B /dev/rdsk/c2t0d0p0
sysA# format -L vtoc -d c2t0d0
```

슬라이스 0의 디스크 공간이 적절히 할당되었는지 확인합니다. 위 명령에서는 기본 분할 영역이 적용되며 이 영역은 루트 풀 슬라이스 0에 사용하기에 너무 작을 수 있습니다. 기본 분할 영역 테이블 수정에 대한 자세한 내용은 **Oracle Solaris 11.1 관리: 장치 및 파일 시스템**의 "ZFS 루트 풀 디스크(EFI(GPT))를 교체하는 방법"을 참조하십시오.

4 루트 풀을 다시 만듭니다.

SPARC 또는 x86 (VTOC) 시스템에서 다음을 수행합니다.

```
sysA# zpool create rpool c2t0d0s0
```

EFI(GPT) 레이블이 지정된 루트 풀 디스크를 포함하는 x86 기반 시스템에서 다음과 비슷한 구문을 사용합니다.

```
sysA# zpool create -B rpool c2t0d0
```

5 원격 시스템의 스냅샷이 포함된 파일 시스템을 마운트합니다.

```
sysA# mount -F nfs sysB:/tank/snaps /mnt
```

6 루트 풀 스냅샷을 복원합니다.

```
sysA# gzcat /mnt/rpool.snap1.qz | zfs receive -Fv rpool
receiving full stream of rpool@rpool.snap1 into rpool@rpool.snap1
received 92.7KB stream in 1 seconds (92.7KB/sec)
receiving full stream of rpool/export@rpool.snap1 into rpool/export@rpool.snap1
received 47.9KB stream in 1 seconds (47.9KB/sec)
.
.
.
```

7 필요한 경우 스왑 및 덤프 장치를 다시 만듭니다.

예를 들면 다음과 같습니다.

```
sysA# zfs create -V 4G rpool/swap
sysA# zfs create -V 4G rpool/dump
```

스왑 및 덤프 볼륨 크기 지정에 대한 자세한 내용은 **Oracle Solaris 11.1 관리: 장치 및 파일 시스템**의 "스왑 공간 계획"을 참조하십시오.

8 BE를 마운트합니다.

다음 단계에서는 부트 블록을 설치할 수 있도록 BE를 마운트해야 합니다.

```
sysA# beadm mount solaris /tmp/mnt
```

9 새 디스크에 부트 블록을 설치합니다.

SPARC 또는 x86 기반 시스템에서 다음 명령을 사용합니다.

```
sysA# bootadm install-bootloader -P rpool
```

10 동일한 장치를 사용하지 않거나 원래 시스템과는 다른 방식으로 장치를 구성하려는 경우 기존 장치 정보를 지웁니다. 그런 다음 시스템이 새 장치 정보를 재구성하도록 합니다.

```
# devfsadm -Cn -r /tmp/mnt
# touch /tmp/mnt/reconfigure
```

11 BE의 마운트를 해제합니다.

```
#beadm unmount solaris
```

12 필요한 경우 부트 환경을 활성화합니다.

예를 들면 다음과 같습니다.

```
sysA# beadm list
BE        Active Mountpoint Space  Policy Created
--        ------ ---------- -----  ------ -------
solaris-1 -      -          46.95M static 2012-07-20 10:25
solaris   -      -           3.83G static 2012-07-19 13:44
# beadm activate solaris
```

13 교체 루트 풀 디스크에서 성공적으로 부트할 수 있는지 확인합니다.

필요한 경우 기본 부트 장치를 재설정합니다.

- SPARC: `eeprom` 명령 또는 부트 PROM의 `setenv` 명령을 사용하여 새 디스크에서 자동으로 부트되도록 시스템을 설정합니다.
- x86: 시스템 BIOS를 재구성합니다.

◆ ◆ ◆ 12장

12 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.

- 305 페이지 "저장소 풀 권장 방법"
- 312 페이지 "파일 시스템 권장 방법"

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.

  309 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 스패어를 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.

루트 풀 생성 방법

- **SPARC (SMI (VTOC))**: `s*` 식별자를 사용하여 슬라이스로 구성된 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
  ```
- **x86(EFI (GPT))**: `d*` 식별자를 사용하여 전체 디스크로 구성된 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 c0t1d0
  ```

  또는 미러링된 루트 풀을 만듭니다. 예를 들면 다음과 같습니다.

  ```
  # zpool create rpool mirror c0t1d0 c0t2d0
  ```

- 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(데이터) 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 90% 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 90% 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 90%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.
- 하드웨어 구성 요소에 결함이 있는 것으로 진단되면 알리도록 `smtp-notify` 서비스를 구성하는 것이 좋습니다. 자세한 내용은 `smf(5)` 및 `smtp-notify(1M)`의 알림 매개변수 절을 참조하십시오.

  기본적으로 일부 알림은 루트 사용자에게 자동으로 전송되도록 설정됩니다. `/etc/aliases` 파일에서 루트로 사용자 계정에 대한 별칭을 추가하면 다음과 비슷한 전자 메일 알림을 받게 됩니다.

```
From noaccess@tardis.space.com Fri Jun 29 16:58:59 2012
Date: Fri, 29 Jun 2012 16:58:58 -0600 (MDT)
```

```
From: No Access User <noaccess@tardis.space.com>
Message-Id: <201206292258.q5TMwwFL002753@tardis.space.com>
Subject: Fault Management Event: tardis:ZFS-8000-8A
To: root@tardis.space.com
Content-Length: 771

SUNW-MSG-ID: ZFS-8000-8A, TYPE: Fault, VER: 1, SEVERITY: Critical
EVENT-TIME: Fri Jun 29 16:58:58 MDT 2012
PLATFORM: ORCL,SPARC-T3-4, CSN: 1120BDRCCD, HOSTNAME: tardis
SOURCE: zfs-diagnosis, REV: 1.0
EVENT-ID: 76c2d1d1-4631-4220-dbbc-a3574b1ee807
DESC: A file or directory in pool 'pond' could not be read due to corrupt data.
AUTO-RESPONSE: No automated response will occur.
IMPACT: The file or directory is unavailable.
REC-ACTION: Use 'fmadm faulty' to provide a more detailed view of this event.
Run 'zpool status -xv' and examine the list of damaged files to determine what
has been affected. Please refer to the associated reference document at
http://support.oracle.com/msg/ZFS-8000-8A for the latest service procedures
and policies regarding this diagnosis.
```

- 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 primarycache를 metadata로 설정하여 아카이브 풀에 아카이브 로그 파일 시스템을 만듭니다.

자세한 내용은 **Oracle Solaris 11.1 조정 가능 매개변수 참조 설명서**의 "Oracle 데이터베이스에 대한 ZFS 조정"을 참조하십시오.

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 df는 종속 파일 시스템이나 스냅샷에서 소비하는 공간을 고려하지 않으므로 사용하지 마십시오.

  자세한 내용은 267 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.

- 315 페이지 “ZFS 버전 개요”
- 315 페이지 “ZFS 풀 버전”
- 317 페이지 “ZFS 파일 시스템 버전”

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 195 페이지 “ZFS 파일 시스템 업그레이드” 및 104 페이지 “ZFS 저장소 풀 업그레이드”를 참조하십시오.

ZFS 풀 버전

다음 표에서는 Oracle Solaris 릴리스에서 제공하는 ZFS 풀 버전 목록을 보여줍니다.

버전	Oracle Solaris 11	설명
1	snv_36	초기 ZFS 버전
2	snv_38	Ditto 블록(복제된 메타 데이터)

버전	Oracle Solaris 11	설명
3	snv_42	핫 스패어 및 이중 패리티 RAID-Z
4	snv_62	`zpool history`
5	snv_62	`gzip` 압축 알고리즘
6	snv_62	`bootfs` 풀 등록 정보
7	snv_68	분리된 계획 로그 장치
8	snv_69	위임 관리
9	snv_77	`refquota` 및 `refreservation` 등록 정보
10	snv_78	캐시 장치
11	snv_94	향상된 스크러빙 성능
12	snv_96	스냅샷 등록 정보
13	snv_98	`snapused` 등록 정보
14	snv_103	`aclinherit passthrough-x` 등록 정보
15	snv_114	사용자 및 그룹 공간 계산
16	snv_116	`stmf` 등록 정보
17	snv_120	타사 RAID-Z
18	snv_121	스냅샷 사용자 유지
19	snv_125	로그 장치 제거
20	snv_128	`zle`(0 길이 인코딩) 압축 알고리즘
21	snv_128	중복 제거
22	snv_128	수신된 등록 정보
23	snv_135	Slim ZIL
24	snv_137	시스템 속성
25	snv_140	향상된 스크러빙 통계
26	snv_141	향상된 스냅샷 삭제 성능
27	snv_145	향상된 스냅샷 만들기 성능
28	snv_147	다중 vdev 대체
29	snv_148	RAID-Z/미러 하이브리드 할당자
30	snv_149	암호화

버전	Oracle Solaris 11	설명
31	snv_150	향상된 'zfs list' 성능
32	snv_151	1MB 블록 크기
33	snv_163	향상된 공유 지원
34	S11.1	상속을 사용하여 공유

ZFS 파일 시스템 버전

다음 표에서는 Oracle Solaris 릴리스에서 제공하는 ZFS 파일 시스템 버전을 보여줍니다. 특정 파일 시스템 버전에서 사용 가능한 해당 기능은 특정 풀 버전이 필요합니다.

버전	Oracle Solaris 11	설명
1	snv_36	초기 ZFS 파일 시스템 버전
2	snv_69	향상된 디렉토리 항목
3	snv_77	대소문자 비구분 및 파일 시스템 고유 식별자(FUID)
4	snv_114	`userquota` 및 `groupquota` 등록 정보
5	snv_137	시스템 속성
6	S11.1	다중 레벨 파일 시스템 지원

색인

A

ACL
  ACL 등록 정보, 223
  ACL 상속, 222
  ACL 상속 플래그, 222
  aclinherit 등록 정보, 223
  POSIX 드래프트 ACL과 차이점, 218
  ZFS 디렉토리의 ACL
    자세한 설명, 225
  ZFS 파일에 ACL 설정(Compact 모드)
    (예), 237
  ZFS 파일에서 ACL 상속 설정(Verbose 모드)
    (예), 231
  ZFS 파일에서 ACL 설정(Compact 모드)
    설명, 236
  ZFS 파일에서 ACL 설정(Verbose 모드)
    설명, 226
  ZFS 파일에서 단순 ACL 수정(상세 정보 표시 모드)
    (예), 227
  ZFS 파일에서 설정
    설명, 224
  ZFS 파일의 ACL
    자세한 설명, 225
  ZFS 파일의 단순 ACL 복원(Verbose 모드)
    (예), 230
  설명, 217
  액세스 권한, 220
  항목 유형, 220
  형식 설명, 218
ACL 등록 정보 모드
  aclinherit, 137
ACL 등록 정보 모드 *(**계속**)*
  aclmode, 138
ACL 모델, Solaris, ZFS와 기존 파일 시스템의 차이점, 34
aclinherit 등록 정보, 223
allocated 등록 정보, 설명, 81
altroot 등록 정보, 설명, 81
atime 등록 정보, 설명, 138
autoreplace 등록 정보, 설명, 81
available 등록 정보, 설명, 138

B

bootfs 등록 정보, 설명, 81

C

cachefile 등록 정보, 설명, 82
canmount 등록 정보
  설명, 138
  자세한 설명, 152
capacity 등록 정보, 설명, 82
casesensitivity 등록 정보, 설명, 139
checksum 등록 정보, 설명, 139
compression 등록 정보, 설명, 139
compressratio 등록 정보, 설명, 139
copies 등록 정보, 설명, 140
creation 등록 정보, 설명, 140

D

dedup 등록 정보, 설명, 140
dedupditto 등록 정보, 설명, 82
dedupratio 등록 정보, 설명, 82
delegation 등록 정보, 사용 안함으로 설정, 246
delegation 등록 정보, 설명, 82
devices 등록 정보, 설명, 140
dry run
  ZFS 저장소 풀 만들기(zpool create -n) (예), 59
dumpadm, 덤프 장치를 사용으로 설정, 123

E

EFI 레이블
  ZFS와 상호 작용, 45
  설명, 45
exec 등록 정보, 설명, 140

F

failmode 등록 정보, 설명, 82
free 등록 정보, 설명, 82

G

guid 등록 정보, 설명, 83

H

health 등록 정보, 설명, 83

L

listshares 등록 정보, 설명, 83
listsnapshots 등록 정보, 설명, 83
logbias 등록 정보, 설명, 141

M

mlslabel 등록 정보, 설명, 141
mounted 등록 정보, 설명, 141
mountpoint 등록 정보, 설명, 141

N

NFSv4 ACL
  ACL 등록 정보, 223
  ACL 상속, 222
  ACL 상속 플래그, 222
  POSIX 드래프트 ACL과 차이점, 218
  모델
    설명, 217
  형식 설명, 218

O

origin 등록 정보, 설명, 143

P

POSIX 드래프트 ACL, 설명, 218
primarycache 등록 정보, 설명, 142

Q

quota 등록 정보, 설명, 143

R

RAID-Z, 정의, 30
RAID-Z 구성
  (예), 52
  개념적 보기, 48
  단일 패리티, 설명, 48
  이중 패리티, 설명, 48
  중복성 기능, 48
RAID-Z 구성, 디스크 추가, (예), 62
read-only 등록 정보, 설명, 144

recordsize 등록 정보
  설명, 144
  자세한 설명, 155
referenced 등록 정보, 설명, 144
refquota 등록 정보, 설명, 144
refreservation 등록 정보, 설명, 144
reservation 등록 정보, 설명, 145

S

savecore, 충돌 덤프 저장, 123
secondarycache 등록 정보, 설명, 145
setuid 등록 정보, 설명, 145
shadow 등록 정보, 설명, 145
share.nfs 등록 정보, 설명, 146
share.smb 등록 정보, description, 146
share.smb 등록 정보, 설명, 자세히, 156
size 등록 정보, 설명, 83
snapdir 등록 정보, 설명, 146
Solaris ACL
  ACL 등록 정보, 223
  ACL 상속, 222
  ACL 상속 플래그, 222
  POSIX 드래프트 ACL과 차이점, 218
  새 모델
    설명, 217
  형식 설명, 218
sync 등록 정보, 설명, 147

T

type 등록 정보, 설명, 147

U

used 등록 정보
  설명, 147
  자세한 설명, 150
usedbychildren 등록 정보, 설명, 147
usedbydataset 등록 정보, 설명, 148
usedbyrefreservation 등록 정보, 설명, 148
usedbysnapshots 등록 정보, 설명, 148

V

version 등록 정보, 설명, 148
version 등록 정보, 설명, 83
volblocksize 등록 정보, 설명, 148
volsize 등록 정보
  설명, 148
  자세한 설명, 156

X

xattr 등록 정보, 설명, 149

Z

zfs allow
  설명, 249
  위임 권한 표시, 254
zfs create
  (예), 39
  (예제), 134
  설명, 134
zfs destroy, (예제), 135
zfs destroy -r, (예제), 135
zfs get, (예제), 162
zfs get -H -o, (예제), 164
zfs get -s, (예제), 163
zfs inherit, (예), 161
zfs list
  (예), 40
  (예제), 157
zfs list -H, (예제), 159
zfs list -r, (예), 158
zfs list -t, (예제), 159
zfs mount, (예제), 167
zfs promote, 복제 프로모션(예), 205
zfs receive, (예), 210
zfs rename, (예제), 136
zfs send, (예), 209
zfs set atime, (예제), 160
zfs set compression, (예), 40
zfs set mountpoint
  (예), 40
  (예제), 166

zfs set mountpoint=legacy, (예제), 166
zfs set quota
(예), 40
zfs set quota, (예), 161
zfs set quota
예제, 180
zfs set reservation, (예제), 183
zfs set share.nfs, (예), 40
zfs unallow, 설명, 249
zfs unmount, (예제), 169
zfs upgrade, 195
ZFS 공간 계산, ZFS와 기존 파일 시스템의
차이점, 32
ZFS 구성 요소, 명명 요구 사항, 31
ZFS 등록 정보
aclinherit, 137
aclmode, 138
atime, 138
available, 138
canmount, 138
자세한 설명, 152
casesensitivity, 139
checksum, 139
compression, 139
compressratio, 139
copies, 140
creation, 140
dedup, 140
devices, 140
exec, 140
logbias, 141
mlslabel, 141
mounted, 141
mountpoint, 141
origin, 143
quota, 143
read-only, 144
recordsize, 144
자세한 설명, 155
referenced, 144
refquota, 144
refreservation, 144
reservation, 145
secondarycache, 142, 145
ZFS 등록 정보 *(**계속**)*
setuid, 145
shadow, 145
share.nfs, 146
share.smb, 146
snapdir, 146
sync, 147
type, 147
used, 147
자세한 설명, 150
usedbychildren, 147
usedbydataset, 148
usedbyrefreservation, 148
usedbysnapshots, 148
user 등록 정보
자세한 설명, 156
version, 148
volblocksize, 148
volsize, 148
자세한 설명, 156
xattr, 149
zoned, 149
zoned 등록 정보
자세한 설명, 264
상속 가능, 설명, 137
상속 가능한 등록 정보에 대한 설명, 137
설명, 137
설정 가능, 150
영역 내의 관리
설명, 263
읽기 전용, 149
ZFS 버전
ZFS 기능 및 Solaris OS
설명, 315
ZFS 볼륨, 설명, 257
ZFS 위임 권한, 개요, 245
ZFS 저장소 풀
dry run 수행(zpool create -n)
(예), 59
RAID-Z
정의, 30
RAID-Z 구성, 설명, 48
RAID-Z 구성 만들기(zpool create)
(예), 52

ZFS 저장소 풀 *(계속)*
vdev I/O 통계
(예), 89
ZFS에 다시 연결된 장치 알림(zpool online)
(예), 281
가상 장치, 56
정의, 30
가져오기
(예), 99
가져오기를 위해 식별(zpool import -a)
(예), 97
건전성 상태 표시, 90
(예), 92
교체 가능한 장치인지 확인
설명, 284
구성 요소, 43
권한 프로파일, 35
기본 마운트 지점, 59
나열
(예), 84
내보내기
(예), 96
누락된(UNAVAIL) 장치
설명, 269
누락된 장치 교체
(예), 278
대체 디렉토리에서 가져오기(zpool import -d)
(예), 98
대체 루트 풀, 265
데이터 검증
설명, 271
데이터 복구
설명, 271
데이터 손상 유형 식별(zpool status -v)
(예), 292
데이터 스크러빙
(예), 271
설명, 271
데이터 스크러빙 및 리실버링
설명, 272
동적 스트라이프, 49
리실버링
정의, 30

ZFS 저장소 풀 *(계속)*
리실버링 프로세스 확인
(예), 290
마이그레이션
설명, 95
만들기(zpool create)
(예), 50
문제 식별
설명, 273
문제 해결을 위한 전체 풀 상태 정보
설명, 275
문제가 있는지 확인(zpool status -x)
설명, 274
미러
정의, 30
미러된 구성 만들기(zpool create)
(예), 51
미러링 구성, 설명, 48
미러링된 저장소 풀 분할(zpool split)
(예), 67
버전
설명, 315
부트할 수 없는 시스템 복구
설명, 296
삭제(zpool destroy)
(예), 60
삭제된 풀 복구
(예), 102
손상된 ZFS 구성 복구, 278
손상된 데이터
설명, 270
손상된 장치
설명, 270
손상된 파일 또는 디렉토리 복구
설명, 293
시스템 오류 메시지
설명, 277
식별된 데이터 손상(zpool status -v)
(예), 277
업그레이드
설명, 104
오류, 269
자세한 건전성 상태 표시
(예), 93

ZFS 저장소 풀 *(계속)*
장치 교체(zpool replace)
(예), 73, 285
장치 분리(zpool detach)
(예), 67
장치 연결(zpool attach)
(예), 66
장치 오류 유형 확인
설명, 282
장치 오류 지우기(zpool clear)
(예), 283
장치 오프라인 전환(zpool offline)
(예), 71
장치 온라인 및 오프라인 전환
설명, 70
장치 지우기
(예), 72
장치 추가(zpool add)
(예), 61
저장소 풀 출력 결과 스크립팅
(예), 86
전체 디스크 사용, 45
파일 사용, 46
풀
정의, 30
풀 전역 I/O 통계
(예), 88
풀 전체 손상 복구
설명, 295
ZFS 저장소 풀(zpool online)
장치 온라인으로 전환
(예), 72
ZFS 저장소 풀 마이그레이션, 설명, 95
ZFS 파일 시스템
atime 등록 정보 설정
(예제), 160
boot -L 및 boot -Z를 사용하여 ZFS BE 부트
(SPARC 예), 127
quota 등록 정보 설정
(예), 161
ZFS 디렉토리의 ACL
자세한 설명, 225
ZFS 볼륨 만들기
(예), 257
ZFS 파일 시스템 *(계속)*
ZFS 파일에서 ACL 상속 설정(Verbose 모드)
(예), 231
ZFS 파일에서 ACL 설정
설명, 224
ZFS 파일에서 ACL 설정(Compact 모드)
(예), 237
설명, 236
ZFS 파일에서 ACL 설정(Verbose 모드)
설명, 226
ZFS 파일에서 단순 ACL 수정(상세 정보 표시 모드)
(예), 227
ZFS 파일의 ACL
자세한 설명, 225
ZFS 파일의 단순 ACL 복원(Verbose 모드)
(예), 230
간소화된 관리
설명, 29
구성 요소 명명 요구 사항, 31
권한 프로파일, 35
기본 마운트 지점
(예제), 134
나열
(예제), 157
데이터 세트
정의, 29
데이터 세트 유형
설명, 159
데이터 스트림 수신(zfs receive)
(예), 210
데이터 스트림 저장(zfs send)
(예), 209
등록 정보 나열(zfs list)
(예제), 162
등록 정보 상속(zfs inherit)
(예), 161
레거시 마운트 지점 관리
설명, 165
레거시 마운트 지점 설정
(예제), 166
루트 파일 시스템 부트
설명, 124

ZFS 파일 시스템 *(계속)*
마운트
(예제), 167
마운트 지점 관리
설명, 165
마운트 지점 설정(zfs set mountpoint)
(예제), 166
마운트 해제
(예제), 169
마이그레이션, 193
만들기
(예제), 134
버전
설명, 315
복제본
설명, 204
정의, 29
파일 시스템 대체(예), 205
복제본 만들기, 205
복제본 삭제, 205
볼륨
정의, 30
비전역 영역에 ZFS 볼륨 추가
(예), 262
비전역 영역에 ZFS 파일 시스템 추가
(예), 261
비전역 영역에 데이터 세트 위임
(예), 261
삭제
(예제), 135
설명, 26, 133
소스 값별로 등록 정보 나열
(예제), 163
스냅샷
롤백, 203
만들기, 198
삭제, 199
설명, 197
액세스, 201
이름 바꾸기, 200
정의, 30
스냅샷 공간 계산, 202
스왑 및 덤프 장치
문제, 122
ZFS 파일 시스템, 스왑 및 덤프 장치 *(계속)*
설명, 121
크기 조정, 122
스크립팅을 위한 등록 정보 나열
(예제), 164
암호화, 185
업그레이드
설명, 195
영역 내의 등록 정보 관리
설명, 263
영역이 설치된 Solaris 시스템에서 사용
설명, 260
예약 설정
(예제), 183
유형 나열
(예제), 159
이름 바꾸기
(예제), 136
자동 마운트 지점 관리, 165
전송 및 수신
설명, 206
종속 항목 나열
(예), 158
종속 항목을 포함하여 삭제
(예제), 135
체크섬
정의, 29
체크섬 데이터
설명, 28
트랜잭션 개념
설명, 27
파일 시스템
정의, 30
풀 저장소
설명, 27
헤더 정보 없이 나열
(예제), 159
ZFS 파일 시스템(zfs set quota)
쿼터 설정
예제, 180
ZFS 파일 시스템 공유, share.smb 등록 정보, 156
ZFS 파일 시스템 마운트, ZFS와 기존 파일 시스템의 차이점, 34

ZFS 파일 시스템 마이그레이션
  개요, 192
  문제 해결, 195
  예, 193
ZFS 파일 시스템 암호화
  개요, 185
  예, 185, 190
  키 변경, 187
ZFS 풀 등록 정보
  allocated, 81
  alroot, 81
  autoreplace, 81
  bootfs, 81
  cachefile, 82
  capacity, 82
  dedupditto, 82
  dedupratio, 82
  delegation, 82
  failmode, 82
  free, 82
  guid, 83
  health, 83
  listsharess, 83
  listsnapshots, 83
  size, 83
  version, 83
ZFS와 기존 파일 시스템의 차이점
  ZFS 공간 계산, 32
  ZFS 파일 시스템 마운트, 34
  공간 부족 동작, 33
  기존 볼륨 관리, 34
  새로운 Solaris ACL 모델, 34
  파일 시스템 세분성, 31
ZFS의 복제 기능, 미러링 또는 RAID-Z, 47
ZFS의 사용자 등록 정보
  (예제), 156
  자세한 설명, 156
ZFS의 설정 가능한 등록 정보
  aclinherit, 137
  aclmode, 138
  atime, 138
  canmount, 138
    자세한 설명, 152
  casesensitivity, 139
ZFS의 설정 가능한 등록 정보 (**계속**)
  checksum, 139
  compression, 139
  copies, 140
  dedup, 140
  devices, 140
  exec, 140
  mountpoint, 141
  primarycache, 142
  quota, 143
  read-only, 144
  recordsize, 144
    자세한 설명, 155
  refquota, 144
  refreservation, 144
  reservation, 145
  secondarycache, 145
  setuid, 145
  shadow, 145
  share,nfs, 146
  share.smb, 146
  snapdir, 146
  sync, 147
  used
    자세한 설명, 150
  version, 148
  volblocksize, 148
  volsize, 148
    자세한 설명, 156
  xattr, 149
  zoned, 149
  설명, 150
ZFS의 읽기 전용 등록 정보
  available, 138
  compression, 139
  creation, 140
  mounted, 141
  origin, 143
  referenced, 144
  type, 147
  used, 147
  usedbychildren, 147
  usedbydataset, 148
  usedbyrefreservation, 148

ZFS의 읽기 전용 등록 정보 *(계속)*
usedbysnapshots, 148
설명, 149
ZIL(ZFS 계획 로그), 설명, 53
zoned 등록 정보
설명, 149
자세한 설명, 264
zpool add, (예), 61
zpool attach, (예), 66
zpool clear
(예), 72
설명, 72
zpool create
(예), 36, 38
RAID-Z 저장소 풀
(예), 52
기본 풀
(예), 50
미러된 저장소 풀
(예), 51
zpool create -n, dry run(예), 59
zpool destroy, (예), 60
zpool detach, (예), 67
zpool export, (예), 96
zpool import -a, (예), 97
zpool import -D, (예), 102
zpool import -d, (예), 98
zpool import **이름**, (예), 99
zpool iostat, 풀 전역(예), 88
zpool iostat -v, vdev(예), 89
zpool list
(예), 38, 84
설명, 83
zpool list -Ho name, (예), 86
zpool offline, (예), 71
zpool online, (예), 72
zpool replace, (예), 73
zpool split, (예), 67
zpool status -v, (예), 93
zpool status -x, (예), 92
zpool upgrade, 104

가
가상 장치
ZFS 저장소 풀의 구성 요소, 56
정의, 30
가져오기
ZFS 저장소 풀
(예), 99
대체 디렉토리에서 ZFS 저장소 풀(zpool import -d)
(예), 98
대체 루트 풀
(예), 266

간
간소화된 관리, 설명, 29

감
감지
사용 중인 장치
(예), 57
일치하지 않는 복제 레벨
(예), 58

개
개별 사용자에게 권한 위임, (예), 250

검
검사, ZFS 데이터 무결성, 271

공
공간 부족 동작, ZFS와 기존 파일 시스템의 차이점, 33

교
교체
누락된 장치
(예), 278
장치(zpool replace)
(예), 73, 285, 290

구
구성 요소, ZFS 저장소 풀, 43

권
권한 세트, 정의됨, 245
권한 위임, zfs allow, 249
권한 제거, zfs unallow, 249
권한 프로파일, ZFS 파일 시스템 및 저장소 풀 관리용, 35

그
그룹에 권한 위임, (예), 250

기
기존 볼륨 관리, ZFS와 기존 파일 시스템의 차이점, 34

나
나열
ZFS 등록 정보(zfs list)
(예제), 162
ZFS 저장소 풀
(예), 84
설명, 83
ZFS 파일 시스템
(예제), 157
ZFS 파일 시스템(zfs list)
(예), 40
나열 *(**계속**)*
ZFS 파일 시스템의 유형
(예제), 159
ZFS 파일 시스템의 종속 항목
(예), 158
ZFS 풀 정보, 38
소스 값별 ZFS 등록 정보
(예제), 163
스크립팅을 위한 ZFS 등록 정보
(예제), 164
헤더 정보가 없는 ZFS 파일 시스템
(예제), 159

내
내보내기
ZFS 저장소 풀
(예), 96

대
대체 루트 풀
가져오기
(예), 266
만들기
(예), 266
설명, 265

데
데이터
검증(스크러빙), 271
리실버링
설명, 272
복구, 271
손상됨, 270
스크러빙
(예), 271
식별된 손상(zpool status -v)
(예), 277
데이터 세트
설명, 134

데이터 세트 *(**계속**)*
정의, 29
데이터 세트 유형, 설명, 159
데이터 자가 치료, 설명, 49

동
동적 스트라이프
설명, 49
저장소 풀 기능, 49

디
디스크, ZFS 저장소 풀의 구성 요소, 45

롤
롤백
ZFS 스냅샷
(예), 203

리
리실버링, 정의, 30
리실버링 및 데이터 스크러빙, 설명, 272

마
마운트
ZFS 파일 시스템
(예제), 167
마운트 지점
ZFS 관리
설명, 165
ZFS 저장소 풀에 대한 기본값, 59
ZFS 파일 시스템 기본값, 134
레거시, 165
자동, 165
마운트 해제
ZFS 파일 시스템
(예제), 169

만
만들기
ZFS 복제본(예), 205
ZFS 볼륨
(예), 257
ZFS 스냅샷
(예), 198
ZFS 저장소 풀
설명, 50
ZFS 저장소 풀(`zpool create`)
(예), 36, 50
ZFS 파일 시스템, 39
(예제), 134
설명, 134
ZFS 파일 시스템 계층, 38
기본 ZFS 파일 시스템(`zpool create`)
(예), 36
단일 패리티 RAID-Z 저장소 풀(`zpool create`)
(예), 52
대체 루트 풀
(예), 266
로그 장치를 사용하여 ZFS 저장소 풀(예), 53
미러된 ZFS 저장소 풀(`zpool create`)
(예), 51
미러링된 저장소 풀을 분할하여 새 풀(`zpool split`)
(예), 67
삼중 패리티 RAID-Z 저장소 풀(`zpool create`)
(예), 52
이중 패리티 RAID-Z 저장소 풀(`zpool create`)
(예), 52
캐시 장치가 있는 ZFS 저장소 풀(예), 55

명
명명 요구 사항, ZFS 구성 요소, 31

문

문제 해결
  FS에 다시 연결된 장치 알림(zpool online)
    (예), 281
  ZFS 오류, 269
  ZFS 오류 메시지의 syslog 보고, 277
  ZFS 파일 시스템 마이그레이션, 195
  교체 가능한 장치인지 확인
    설명, 284
  누락된(UNAVAIL) 장치, 269
  누락된 장치 교체
    (예), 278
  데이터 손상 유형 확인(zpool status -v)
    (예), 292
  문제 식별, 273
  문제가 있는지 확인(zpool status -x), 274
  부트할 수 없는 시스템 복구
    설명, 296
  손상된 ZFS 구성 복구, 278
  손상된 장치, 270
  손상된 파일 또는 디렉토리 복구
    설명, 293
  식별된 데이터 손상(zpool status -v)
    (예), 277
  장치 교체(zpool replace)
    (예), 285, 290
  장치 오류 유형 확인
    설명, 282
  장치 오류 지우기(zpool clear)
    (예), 283
  전체 풀 상태 정보
    설명, 275
  풀 전체 손상 복구
    설명, 295

미

미러, 정의, 30
미러된 저장소 풀(zpool create), (예), 51
미러링 구성
  개념적 보기, 48
  설명, 48
  중복성 기능, 48
미러링된 로그 장치, ZFS 저장소 풀 만들기(예), 53
미러링된 로그 장치, 추가, (예), 63
미러링된 저장소 풀 분할
  (zpool split)
    (예), 67

별

별도의 로그 장치, 사용 시의 고려 사항, 53

복

복구
  부트할 수 없는 시스템
    설명, 296
  삭제된 ZFS 저장소 풀
    (예), 102
  손상된 ZFS 구성
    설명, 278
  손상된 파일 또는 디렉토리 복구
    설명, 293
  풀 전체 손상
    설명, 295
복원
  ZFS 파일의 단순 ACL(Verbose 모드)
    (예), 230
복제 스트림 패키지, 208
복제본
  기능, 204
  만들기(예), 205
  삭제(예), 205
  정의, 29

볼

볼륨, 정의, 30

부

부트
  boot -L 및 boot -Z를 사용하여 SPARC 시스템에서
    부트, 127

부트 *(계속)*
  루트 파일 시스템, 124
부트 블록, bootadm으로 설치, 125
부트 블록 설치
  bootadm
    (예), 125

분
분리
  ZFS 저장소 풀에 장치(zpool detach)
    (예), 67

사
사용 중인 장치
  감지
    (예), 57

삭
삭제
  ZFS 복제본(예), 205
  ZFS 스냅샷
    (예), 199
  ZFS 저장소 풀
    설명, 50
  ZFS 저장소 풀(zpool destroy)
    (예), 60
  ZFS 파일 시스템
    (예제), 135
  종속 항목이 포함된 ZFS 파일 시스템
    (예제), 135

상
상속
  ZFS 등록 정보(zfs inherit)
    설명, 161

새
새도우 마이그레이션, 개요, 192

설
설정
  compression 등록 정보
    (예), 40
  mountpoint 등록 정보, 40
  quota 등록 정보(예), 40
  share.nfs 등록 정보
    (예), 40
  ZFS atime 등록 정보
    (예제), 160
  ZFS 마운트 지점(zfs set mountpoint)
    (예제), 166
  ZFS 쿼터
    (예), 161
  ZFS 파일 시스템 예약
    (예제), 183
  ZFS 파일 시스템 쿼터(zfs set quota)
    예제, 180
  ZFS 파일의 ACL
    설명, 224
  ZFS 파일의 ACL(Compact 모드)
    (예), 237
    설명, 236
  ZFS 파일의 ACL(Verbose 모드)
    (설명, 226
  ZFS 파일의 ACL 상속(Verbose 모드)
    (예), 231
  레거시 마운트 지점
    (예제), 166

수
수신
  ZFS 파일 시스템 데이터(zfs receive)
    (예), 210
수정
  ZFS 파일의 단순 ACL(상세 정보 표시 모드)
    (예), 227

순
순환적 스트림 패키지, 208

스
스냅샷
  공간 계산, 202
  기능, 197
  롤백
    (예), 203
  만들기
    (예), 198
  삭제
    (예), 199
  액세스
    (예), 201
  이름 바꾸기
    (예), 200
  정의, 30
스왑 및 덤프 장치
  문제, 122
  설명, 121
  크기 조정, 122
스크러빙
  (예), 271
  데이터 검증, 271
스크립팅
  ZFS 저장소 풀 출력 결과
    (예), 86
스트림 패키지
  복제, 208
  순환적, 208

식
식별
  가져올 ZFS 저장소 풀(zpool import -a)
    (예), 97
  데이터 손상 유형(zpool status -v)
    (예), 292
  저장소 요구 사항, 37

알
알림
  FS에 다시 연결된 장치 알림(zpool online)
    (예), 281

액
액세스
  ZFS 스냅샷
    (예), 201

업
업그레이드
  ZFS 저장소 풀
    설명, 104
  ZFS 파일 시스템
    설명, 195

연
연결
  ZFS 저장소 풀에 장치(zpool attach)
    (예), 66

영
영역
  ZFS 파일 시스템에서 사용
    설명, 260
  zoned 등록 정보
    자세한 설명, 264
  비전역 영역에 ZFS 볼륨 추가
    (예), 262
  비전역 영역에 ZFS 파일 시스템 추가
    (예), 261
  비전역 영역에 데이터 세트 위임
    (예), 261
  영역 내의 ZFS 등록 정보 관리
    설명, 263

오
오류, 269
오류 모드
  누락된(UNAVAIL) 장치, 269
  손상된 데이터, 270
  손상된 장치, 270

용
용어
  RAID-Z, 30
  가상 장치, 30
  데이터 세트, 29
  리실버링, 30
  미러, 30
  복제본, 29
  볼륨, 30
  스냅샷, 30
  체크섬, 29
  파일 시스템, 30
  풀, 30

위
위임
  권한(예), 250
  비전역 영역에 데이터 세트
    (예), 261
위임 관리, 개요, 245

이
이름 바꾸기
  ZFS 스냅샷
    (예), 200
  ZFS 파일 시스템
    (예제), 136

일
일치하지 않는 복제 레벨
  감지
    (예), 58

장
장치 오프라인 전환(zpool offline)
  ZFS 저장소 풀
    (예), 71
장치 온라인 및 오프라인 전환
  ZFS 저장소 풀
    설명, 70
장치 온라인으로 전환
  ZFS 저장소 풀(zpool online)
    (예), 72
장치 지우기
  ZFS 저장소 풀
    (예), 72

저
저장
  ZFS 파일 시스템 데이터(zfs send)
    (예), 209
  충돌 덤프
    savecore, 123
저장소 요구 사항, 식별, 37

전
전송 및 수신
  ZFS 파일 시스템 데이터
    설명, 206
전체 디스크, ZFS 저장소 풀의 구성 요소, 45

제
제거, 캐시 장치(예), 65
제어, 데이터 검증(스크러빙), 271

조
조정, 스왑 및 덤프 장치의 크기, 122

지
지우기
  ZFS 저장소 풀의 장치(zpool clear)
    설명, 72
  장치 오류(zpool clear)
    (예), 283

체
체크섬, 정의, 29
체크섬 데이터, 설명, 28

추
추가
  RAID-Z 구성에 디스크(예), 62
  ZFS 저장소 풀에 장치(zpool add)
    (예), 61
  미러링된 로그 장치(예), 63
  비전역 영역에 ZFS 볼륨
    (예), 262
  비전역 영역에 ZFS 파일 시스템
    (예), 261
  캐시 장치(예), 65

충
충돌 덤프, 저장, 123

캐
캐시 장치
  ZFS 저장소 풀 만들기(예), 55
  사용 고려 사항, 55
캐시 장치, 제거, (예), 65
캐시 장치, 추가, (예), 65

쿼
쿼터 및 예약, 설명, 179

트
트랜잭션 개념, 설명, 27

파
파일, ZFS 저장소 풀의 구성 요소, 46
파일 시스템, 정의, 30
파일 시스템 계층, 만들기, 38
파일 시스템 세분성, ZFS와 기존 파일 시스템의
  차이점, 31

표
표시
  ZFS 오류 메시지의 syslog 보고
    설명, 277
  ZFS 저장소 풀 I/O 통계
    설명, 88
  ZFS 저장소 풀 vdev I/O 통계
    (예), 89
  ZFS 저장소 풀 건전성 상태
    (예), 92
  ZFS 저장소 풀 전역 I/O 통계
    (예), 88
  위임 권한(예), 254
  자세한 ZFS 저장소 풀 건전성 상태
    (예), 93
  저장소 풀의 건전성 상태
    설명, 90

풀
풀, 정의, 30
풀 저장소, 설명, 27

하

하드웨어 및 소프트웨어 요구 사항, 36

핫

핫 스패어
  만들기
    (예), 75
  설명
    (예), 75

확

확인
  교체 가능한 장치인지
    설명, 284
  장치 오류 유형
    설명, 282