Oracle® Hierarchical Storage Manager and StorageTek QFS Software

유지 관리 및 관리 설명서

릴리스 6.1

E56774-03

2016년 3월

ORACLE

Oracle® Hierarchical Storage Manager and StorageTek QFS Software
유지 관리 및 관리 설명서

E56774-03

차례

머리말

이 문서는 Oracle Hierarchical Storage Manager(이전 StorageTek Storage Archive Manager) 및 Oracle StorageTek QFS Software를 사용하여 파일 시스템 및 아카이빙 솔루션을 모니터링, 관리 및 유지 관리하는 업무를 담당하는 시스템 관리자, 스토리지 및 네트워크 관리자, 서비스 엔지니어의 요구에 맞게 작성되었습니다.

설명서 접근성

오라클의 접근성 개선 노력에 대한 자세한 내용은 http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc에서 Oracle Accessibility Program 웹 사이트를 방문하십시오.

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

이 문서 사용을 위한 필요 조건

이 문서에서는 사용자가 이미 Oracle Solaris 운영체제, 디스크 및 테이프 스토리지 시스템과 LAN 및 SAN 모두의 관리에 익숙하다고 가정합니다. 관련 작업, 명령 및 절차에 대한 정보는 Solaris 설명서 및 매뉴얼 페이지와 스토리지 하드웨어 설명서를 참조하십시오.

규약

이 문서에서는 다음 텍스트 규약이 사용되었습니다.

- 기울임꼴 유형은 책 제목 및 강조를 나타냅니다.
- 고정폭 유형은 터미널 창에 표시되는 명령 및 텍스트와 구성 파일, 셸 스크립트 및 소스 코드 파일의 내용을 나타냅니다.
- **고정폭 굵게** 유형은 사용자 입력과 명령줄 출력, 터미널 표시 또는 파일 내용에 대한 중요 변경사항을 나타냅니다. 파일 또는 표시의 관련 부분을 특히 강조하는 데 사용될 수도 있습니다.
- **고정폭 굵게 *기울임꼴*** 유형은 터미널 표시 또는 파일에서 변수 입력 및 출력을 나타냅니다.
- 고정폭 *기울임꼴* 유형은 터미널 표시 또는 파일에서 기타 변수를 나타냅니다.
- . . . (3개의 점으로 구성된 말줄임표)는 예와 관련이 없어 간결성 또는 명확성을 위해 생략된 파일 내용이나 명령을 나타냅니다.
- 예에서 라인 끝에 있는 \(백슬래시)는 다음 라인이 동일 명령의 일부가 되도록 줄바꿈을 이스케이프합니다.

- *[-]*(하이픈으로 구분된 값을 둘러싸는 대괄호)는 값 범위를 구분합니다.
- 명령 구문의 *[ ]*(대괄호)는 선택적 매개변수를 나타냅니다.
- *root@solaris:~#* 및 *[hostname]:root@solaris:~#* 은 Solaris 명령 셸 프롬프트를 나타냅니다.
- *[root@linux ~]#* 은 Linux 명령 셸 프롬프트를 나타냅니다.

사용 가능한 설명서

Oracle Hierarchical Storage Manager and StorageTek QFS Software 유지 관리 및 관리 설명서에서는 *http://docs.oracle.com/en/storage/#sw*에서 사용 가능하며 여러 권으로 구성된 *Oracle HSM* 고객 설명서 라이브러리의 일부입니다.

Oracle Solaris 운영체제 설명서는 *http://docs.oracle.com/en/operating-systems/*에서 제공됩니다.

시스템 요구 사항, 새로운 기능 및 버그 수정에 대한 자세한 내용은 다운로드 ZIP 파일 또는 파일 시스템 서버 */opt/SUNWsamfs/doc/README.txt*에서 릴리스 노트와 *README.txt*를 참조하십시오.

1

1장. Oracle HSM 솔루션 유지 관리

계획 및 배치 프로세스 전 과정에 걸쳐 QFS 파일 시스템 및 Oracle Hierarchical Storage Manager 소프트웨어는 설계 시 성능 최적화, 데이터 보호, 아카이빙의 복잡성을 간단한 UNIX 파일 시스템 인터페이스 뒤에 숨겨야 합니다. 사용자, 응용 프로그램 및 대부분의 관리자는 단일 로컬 디스크에 일반 UFS 파일 시스템을 구현하듯이, 디스크 어레이와 테이프 라이브러리가 혼합된 환경에서 완전히 최적화된 Oracle HSM 아카이빙 시스템을 구현할 수 있어야 합니다. 일단 설치 및 구성되었으면 Oracle HSM 소프트웨어는 가능한 가장 효율적이고 안정적인 방법으로 운영자 개입을 최소화하여 데이터 및 스토리지 리소스를 자동으로 관리해야 합니다.

그렇긴 하지만, 여느 UNIX 파일 시스템과 마찬가지로 여전히 모니터링과 주기적 관리 작업을 수행해야 합니다. 이 책에서는 이러한 작업을 간략히 설명합니다.

2

2장. 파일 시스템 작업 모니터링

올바르게 구성된 Oracle HSM 파일 시스템은 일상적인 관리에 사용자 개입이 거의 필요 없습니다. 그러나 비정상적인 상황이 발생하는지 각 시스템을 모니터해야 합니다. 일반적으로 가용성과 사용률의 두 가지 측면을 모니터하게 됩니다.

가용성은 개념상 간단하며 모니터하기 쉽습니다. 호스트 시스템, 네트워크 인터페이스, 파일 시스템, 스토리지 부속 시스템과 같은 주요 구성 요소를 사용할 수 없게 되면 핵심 기능이 갑자기 손실되거나 저하되면서 관리 인터페이스 및 로그에 경보가 표시됩니다.

사용률은 좀더 미묘한 사안이며 사용자의 판단이 요구됩니다. 과다 사용이 발생할 경우 스토리지 매체와 같은 리소스 부족으로 인해, 시스템이 정상적으로 작동하는 경우에도 마치 구성 요소 고장과 같이 아카이빙 프로세스가 중지될 수 있습니다. 한 조직에서 고장 임박을 예고하는 사용률 수준이 다른 조직에서는 다년간 일관되게 고장 없이 작동될 수도 있습니다. 따라서 사용률을 모니터할 때는 추세와 속도를 인식하는 것이 매우 중요합니다. 80% 사용된 리소스는 사용률이 연간 1% 증가하는 경우 양호 상태이지만, 매주 1% 이상 상승하는 경우 위기 상황입니다.

Oracle HSM는 세 가지 모니터링 인터페이스를 제공합니다.

- Oracle HSM Manager
- **`samu`**
- 로그 및 추적 파일

사용자의 작업 스타일과 습관에 따라 각자 장점이 있습니다.

Oracle HSM Manager

Oracle HSM Manager는 관리자가 파일 시스템 작업의 모든 측면을 모니터하고 제어할 수 있는 브라우저 기반의 그래픽 사용자 인터페이스입니다. Oracle HSM Manager 브라우저 인터페이스 페이지는 다음 세 영역으로 구분됩니다.

- 배너
- 탐색 트리
- 컨텐츠 창

배너에는 응용 프로그램 이름과 상태 정보가 표시됩니다. 브라우저 인터페이스에서 데이터가 마지막 업데이트된 시간, 현재 로그인한 사용자의 이름과 역할, Oracle HSM Manager

소프트웨어를 호스트하는 관리 스테이션의 이름, 현재 확인되지 않은 결함 개수와 유형 등이 포함됩니다.

인터페이스 왼쪽의 탐색 트리에는 서버 메뉴가 표시되고 사용 가능한 디스플레이가 계층적으로 나열됩니다. 탐색 트리에서 링크를 누르면 해당하는 디스플레이가 컨텐츠 창에 나타납니다.

탐색 트리의 Monitoring 노드가 주된 모니터링 리소스입니다. 모니터되는 파일 시스템 및 장비에서 감지된 모든 결함을 나열하고 정렬할 수 있습니다. 자동 전자 메일 경보를 구성할 수 있습니다. 현재 실행 중인 모든 작업을 나열할 수 있습니다. 또한 Monitoring 노드는 모니터링 대시보드(사용자에게 문제를 알려주는 팝업 창)에 대한 링크를 포함하며, 다음 관련 분야에 대한 요약 정보로 가는 빠른 링크를 제공합니다.

- 데몬
- 파일 시스템
- 아카이브 매체 사용률
- 테이프 라이브러리
- 라이브러리 드라이브
- 로드 요청 보류 중 볼륨
- 사용할 수 없는 볼륨
- 아카이빙 복사본 대기열
- 스테이징 대기열

탐색 트리의 Metrics and Reports 노드는 포괄적 범위의 상태 및 사용률 보고서와 System Details에 대한 링크를 제공합니다. System Details 페이지에서 Oracle HSM 구성을 빠르게 검토하고 로그 및 추적 파일에 편리하게 액세스할 수 있습니다.

종합 도움말 시스템에 Oracle HSM Manager 사용법이 자세히 설명되어 있습니다.

samu

samu 운영자 유틸리티는 명령줄에서 실행할 수 있는 텍스트 기반의 메뉴 방식 구성 및 관리 인터페이스입니다. Oracle HSM 장치, 파일 시스템 작동 및 오류 메시지를 모니터하는 간편한 방법입니다.

samu 유틸리티는 어떤 측면에서 UNIX *vi* 편집기와 비슷합니다. 비슷한 제어 키 시퀀스를 사용하여 디스플레이를 선택하고, 디스플레이 옵션을 설정하고, 디스플레이 안팎을 이동하고, 명령을 입력하고, 디스플레이를 새로 고치고, 유틸리티를 종료합니다. 각 디스플레이 창의 마지막 라인에는 오류 메시지가 표시됩니다. 오류가 발생하지 않는 한 디스플레이가 자동으로 새로 고쳐집니다. 만일 오류가 발생할 경우 운영자가 조치를 취할 때까지 디스플레이가 중지됩니다. 원하는 경우 나중에 참조할 수 있도록 디스플레이 창의 스냅샷을 생성할 수 있습니다.

h 명령은 모든 키보드 단축키, 명령, 매개변수를 나열하는 도움말 화면을 엽니다. 추가 정보는 *samu* 매뉴얼 페이지와 *Oracle HSM* 고객 설명서 라이브러리(*http://docs.oracle*

.com/en/storage/#sw)의 *Oracle Hierarchical Storage Manager and StorageTek QFS samu* 명령 참조를 참조할 수 있습니다.

다음은 일반적인 *samu* 모니터링 디스플레이입니다.

```
Archiver status                  samu      5.4     12:24:10 Mar 19 2014

sam-archiverd: Waiting for resources

sam-arfind:  samma1 mounted at /samma1
Files waiting to start 0    schedule 70,524    archiving 0
Monitoring file system activity.

sam-arfind:  DISKVOL1 mounted at /diskvols/DISKVOL1
Files waiting to start 0    schedule 0    archiving 0
Monitoring file system activity.

                            samu on samqfshost1
```

로그 및 추적 파일

Oracle HSM 소프트웨어는 포괄적인 로깅 및 (구성된 경우) 추적을 수행합니다. 따라서 특별히 문제가 발생할 때 다음 파일을 모니터할 수 있습니다.

- */var/adm/messages*
- */var/adm/sam-log*
- */var/opt/SUNWsamfs/trace/* (데몬 및 프로세스를 위한 추적 파일 보유)
- */var/opt/SUNWsamfs/devlog/* (*/etc/opt/SUNWsamfs/mcf* 파일에 구성된 장치를 위한 로그 보유)
- */var/opt/SUNWsamfs/archiver.log*
- */var/opt/SUNWsamfs/stager.log*
- */var/opt/SUNWsamfs/recycler.log*
- (구성된 경우) 파일 시스템과 관련된 추가 아카이빙 로그

로깅 및 추적 구성에 대한 자세한 내용은 *Oracle HSM* 고객 설명서 라이브러리(*http://docs.oracle.com/en/storage/#sw*)의 Oracle Hierarchical Storage Manager and StorageTek QFS 설치 및 구성 설명서를 참조하십시오.

3

3장. Oracle HSM 파일 시스템 관리

이 장에서는 파일 시스템 유지 관리 및 재구성 작업을 다룹니다. 첫번째 절인 Oracle HSM 파일 시스템 관리에서는 모든 Oracle HSM 파일 시스템(아카이빙 및 비아카이빙, 공유 및 비공유(독립형))의 유지 관리를 다룹니다. 두번째 절, Oracle HSM 공유 파일 시스템 관리에서는 공유 파일 시스템에 영향을 주는 특수한 고려 사항을 다룹니다.

Oracle HSM 파일 시스템 관리

이 절에서는 다음 작업을 설명합니다.

- 파일 시스템 쿼터 관리
- 아카이빙 및 스테이징 작업 제어
- 파일 시스템 이름 바꾸기
- 파일 시스템 복구
- 파일 시스템에 장치 추가
- 파일 시스템에서 데이터 장치 제거

파일 시스템 쿼터 관리

파일 시스템 쿼터를 설정하여 주어진 사용자 또는 사용자 모음이 파일 시스템 내에서 소비할 수 있는 온라인/전체 스토리지 공간을 제어할 수 있습니다. 사용자 ID, 그룹 ID 또는 관리자가 정의한 관리 세트 *ID*(특정 프로젝트에서 참여도와 같은 공통 특성으로 사용자를 묶은 것)를 기준으로 쿼터를 설정할 수 있습니다. 관리 세트 ID는 특별히 프로젝트가 여러 그룹의 사용자를 포함하고 여러 디렉토리 및 파일에 걸쳐 있을 때 유용합니다.

파일 시스템을 `quota` 마운트 옵션(기본적으로 설정됨)으로 마운트하여 쿼터를 사용으로 설정하고, `noquota` 마운트 옵션으로 마운트하여 쿼터를 사용 안함으로 설정합니다. 하나 이상의 쿼터 파일인 `.quota_u`, `.quota_g` 및 `.quota_a`를 파일 시스템 루트 디렉토리에 배치하여 쿼터를 정의합니다. 이러한 쿼터 파일은 각각 사용자, 그룹 및 관리 세트에 대한 쿼터를 설정합니다. 각 파일의 첫번째 레코드인 레코드 `0`은 기본값을 설정합니다. 이후 레코드들은 특정 사용자, 그룹, 관리 세트와 관련된 값을 설정합니다.

쿼터는 단순히 스토리지 공간이 아니라, 사용 가능한 파일 시스템 공간을 할당합니다. 따라서 매체에 할당된 512바이트 블록 수와 파일 시스템에 할당된 inode 수에 대한 상한을 설정할 수 있습니다. 블록 계수는 그 자체로 스토리지 공간을 측정합니다. inode 계수는 해당 스토리지에 액세스하기 위해 사용 가능한 리소스를 측정합니다. 따라서 스토리지 공간 블록이 매우

많이 사용되었지만 inode가 하나만 사용된 단일 파일은 많은 수의 inode를 차지하고 블록을 차지하지 않는 길이가 0인이고 매우 많은 수의 비어 있는 파일과 동일한 파일 시스템 공간을 차지할 수 있습니다.

각 쿼터에는 소프트 한계와 하드 한계를 포함할 수 있습니다. 하드 한계는 주어진 소유자의 파일이 일시적으로 사용할 수 있는 최대 파일 시스템 리소스를 정의합니다. 소프트 한계는 주어진 소유자의 파일이 무기한으로 사용할 수 있는 최대 파일 시스템 리소스를 정의합니다. 리소스 사용량은 쿼터에서 유예 기간에 정의된 대로, 짧은 간격 동안 소프트 한계와 하드 한계 사이에 놓인 양까지만 증가할 수 있습니다.

이 절에서는 다음 관리 작업을 설명합니다.

- 사용자, 그룹 및 조직 단위의 스토리지 요구 사항 특징짓기
- 다중 그룹에서 사용되는 프로젝트 및 디렉토리에 대한 관리 세트 만들기
- 새 파일 시스템에서 쿼터를 사용하도록 구성
- 기존 파일 시스템에서 쿼터를 사용하도록 구성
- 그룹, 프로젝트, 디렉토리 및 사용자에 대한 쿼터 설정
- 불일치 쿼터 복구
- 쿼터 검사
- 유예 기간 일시적 연장 또는 취소
- 새 리소스 할당 중지.

사용자, 그룹 및 조직 단위의 스토리지 요구 사항 특징짓기

지속 가능한 쿼터를 설정하려면 관리 가능하고 확장 가능한 방법으로 사용자 요구 사항을 수용하는 한계를 설정해야 합니다. 따라서 쿼터를 설정하기 전에 사용자의 스토리지 요구 사항을 추정합니다. 프로세스를 관리 가능하도록 유지하려면 최소한의 관리 노력으로 최대한의 요구 사항을 처리할 수 있도록 가능한 넓은 범위에서 사용자 요구 사항을 분류하는 것부터 시작합니다. 그런 다음 넓은 범주에 맞지 않는 소수의 사용자 요구 사항을 구체적으로 평가합니다. 결과에는 사용자가 설정한 쿼터와 한계 유형이 개략적으로 제공됩니다.

아래 설명된 접근 방법은 액세스 제어 그룹(대부분의 조직에 이미 그룹이 정의되어 있음)의 파일 시스템 요구 사항을 식별하는 것부터 시작합니다. 그런 다음 표준 그룹의 요구 사항과 잘 맞지 않는 특수한 사용자 세트를 정의합니다. 그런 다음 마지막으로 개별 사용자에 고유한 요구 사항을 처리하기 시작합니다. 다음과 같이 하십시오.

1. 기존 액세스 제어 그룹이 이미 비슷한 리소스 요구 사항을 가진 사용자를 함께 수집하므로, 파일 시스템을 사용할 그룹의 평균 스토리지 요구 사항을 정의하는 것부터 시작합니다. 사용된 평균 스토리지 공간량(512킬로바이트 블록)과 저장된 평균 파일 수(사용된 평균 inode 수에 해당)를 추정합니다.

   그룹 구성원은 일반적으로 직무 역할과 업무 책임이 비슷하므로 동일한 디렉토리 및 파일에 자주 액세스하고 일반적으로 스토리지 요구도 비슷합니다. 예제에서는 */hsm/hsmfs1* 파일 시스템을 사용할 세 그룹으로 *dev*(제품 개발), *cit*(기업 정보 기술),

pgmt(프로그램 관리)를 식별합니다. 간단한 스프레드시트에 그룹, 각각의 구성원 수, 평균 개인 및 그룹 요구 사항을 나열합니다.

그룹	사용자	사용자당 평균 블록	사용자당 평균 파일	평균 블록/그룹	평균 파일/그룹
dev	30	67108864	500	2013265920	15000
cit	15	10485760	50	157286400	750
pmgt	6	20971520	200	125829120	1200
총 블록/ 파일 (평균)					

2. 그 다음, 주어진 시간에 그룹 구성원이 저장할 최대 스토리지 공간량 및 최대 파일 수에 대해 동일한 계산을 수행합니다. 결과를 기록합니다.

   예제에서는 새 스프레드시트에 결과를 기록합니다.

그룹	사용자	사용자당 최대 블록	사용자당 최대 파일	최대 블록/그룹	최대 파일/그룹
dev	30	100663296	1000	3019898880	30000
cit	15	15728640	100	235929600	1500
pmgt	6	31457280	400	188743680	2400
총 블록/ 파일 (최대)					

3. 이제 서로 다른 그룹에 속하지만 그룹 멤버쉽을 기초로 처리할 수 없는 개별 스토리지 요구 사항을 공유하는 사용자 세트를 식별합니다. 각 액세스 제어 그룹에 했던 것처럼 각 식별된 조직에 대해 동일한 예상치를 만들고 동일한 계산을 수행합니다.

   예제에서는 스토리지 할당이 필요한 두 회사 프로젝트로 코드명 *portal* 및 *lockbox*를 식별합니다. 엔지니어링, 마케팅, 준수, 테스트, 문서화 그룹의 구성원은 이 프로젝트를 함께 진행하면서 동일한 디렉토리와 수많은 동일 파일을 사용하게 됩니다. 따라서 이것을 요구 사항 스프레드시트에 추가합니다.

그룹	사용자	사용자당 평균 블록	사용자당 평균 파일	평균 블록/그룹	평균 파일/그룹
dev	30	67108864	500	2013265920	15000
cit	15	10485760	50	157286400	750
pmgt	6	20971520	200	125829120	1200
portal	10	31457280	400	314572800	4000
lockbox	12	31457280	500	377487360	6000
총 블록/ 파일 (평균)					

그룹	사용자	사용자당 최대 블록	사용자당 최대 파일	최대 블록/그룹	최대 파일/그룹
dev	30	100663296	1000	3019898880	30000

그룹	사용자	사용자당 최대 블록	사용자당 최대 파일	최대 블록/그룹	최대 파일/그룹
cit	15	15728640	100	235929600	1500
pmgt	6	31457280	400	188743680	2400
portal	10	37748736	700	377487360	7000
lockbox	12	45613056	600	547356672	7200
총 블록/ 파일 (최대)					

4. 이제 요구 사항이 아직 처리되지 않은 개별 사용자를 식별합니다. 각 액세스 제어 그룹과 비그룹 조직에 했던 것처럼 각 사용자에 대해 동일한 예상치를 만들고 동일한 계산을 수행합니다.

   가능한 경우, 균일한 정책이 적용되고 관리 오버헤드가 최소화되도록 사용자 요구 사항을 총체적으로 처리합니다. 그러나 개인 요구 사항이 고유하다면 개별적으로 처리해야 합니다. 예제에서는 특별한 스토리지 할당이 필요한 특수 책임을 가진 사용자로 *pgmt* 그룹의 *jr23547*을 식별합니다. 따라서 이것을 요구 사항 스프레드시트에 추가합니다.

그룹	세트당 사용자	사용자당 평균 블록	사용자당 평균 파일	평균 블록	평균 파일
dev	30	67108864	500	2013265920	15000
cit	15	10485760	50	157286400	750
pmgt	6	20971520	200	125829120	1200
portal	10	31457280	400	314572800	4000
lockbox	12	31457280	500	377487360	6000
jr23547	1	10485760	600	10485760	600
총 블록/ 파일 (평균)					

그룹	사용자	사용자당 최대 블록	사용자당 최대 파일	최대 블록/그룹	최대 파일/그룹
dev	30	100663296	1000	3019898880	30000
cit	15	15728640	100	235929600	1500
pmgt	6	31457280	400	188743680	2400
portal	10	37748736	700	377487360	7000
lockbox	12	45613056	600	547356672	7200
jr23547	1	100663296	2000	100663296	2000
총 블록/ 파일 (최대)					

5. 마지막으로, 모든 사용자가 필요한 평균/최대 블록 및 파일을 계산합니다.

그룹	사용자	사용자당 평균 블록	사용자당 평균 파일	평균 블록/그룹	평균 파일/그룹
dev	30	67108864	500	2013265920	15000
cit	15	10485760	50	157286400	750
pmgt	6	20971520	200	125829120	1200
portal	10	31457280	400	314572800	4000
lockbox	12	31457280	500	377487360	6000
jr23547	1	10485760	600	10485760	600
총 블록/ 파일 (평균)				2998927360	27550

그룹	사용자	사용자당 최대 블록	사용자당 최대 파일	최대 블록/그룹	최대 파일/그룹
dev	30	100663296	1000	3019898880	30000
cit	15	15728640	100	235929600	1500
pmgt	6	31457280	400	188743680	2400
portal	10	37748736	700	377487360	7000
lockbox	12	45613056	600	547356672	7200
jr23547	1	100663296	2000	100663296	2000
총 블록/ 파일 (평균)				4470079488	50100

6. 프로젝트 기반 쿼터나 기타 액세스 제어 그룹 및 사용자 ID로 정의할 수 없는 쿼터를 관리해야 하는 경우 다중 그룹에서 사용되는 프로젝트 및 디렉토리에 대한 관리 세트 만들기를 수행합니다.
7. 새로 만든 빈 파일 시스템에 쿼터를 설정하는 경우 “새 파일 시스템에서 쿼터를 사용하도록 구성”으로 이동합니다.
8. 이미 파일을 보유한 파일 시스템에 쿼터를 설정하는 경우 “기존 파일 시스템에서 쿼터를 사용하도록 구성”으로 이동합니다.

다중 그룹에서 사용되는 프로젝트 및 디렉토리에 대한 관리 세트 만들기

관리 세트란, 쿼터 목적상 관리 세트 *ID*로 식별되는 디렉토리 계층 또는 개별 디렉토리나 파일입니다. 지정된 관리 세트 ID로 만들거나 지정된 관리 세트 ID로 디렉토리에 저장된 파일은 실제로 파일을 소유한 사용자나 그룹 ID에 관계없이 모두 동일한 쿼터를 갖습니다. 관리 세트를 정의하려면 다음과 같이 하십시오.

1. 파일 시스템 서버에 *root*로 로그인합니다.

   예제에서는 서버 이름이 *server1*로 지정됩니다.

   ```
   [server1]root@solaris:~#
   ```

2. 관리 세트를 사용하여 새 프로젝트나 팀에 대한 스토리지 쿼터를 구성하는 경우, 파일 시스템 내의 아무 곳에나 이 프로젝트나 팀에 대한 새 디렉토리를 만듭니다.

   예제에서는 */hsm/hsmfs1* 파일 시스템에 디렉토리를 만들고 동일한 이름의 프로젝트에 대해 *portalproject/* 이름을 지정합니다.

   ```
   [server1]root@solaris:~# mkdir /hsm/hsmfs1/portalproject
   ```

3. 쿼터를 설정할 디렉토리나 파일에 관리 세트 ID를 지정합니다. *samchaid [-fhR] admin-set-id directory-or-file-name* 명령을 사용합니다. 설명:
   - *-f*는 강제로 지정하고 오류를 보고하지 않습니다.
   - *-h*는 관리 세트 ID를 심볼릭 링크에 지정합니다. 이 옵션이 없으면 심볼릭 링크로 참조된 파일의 그룹이 변경됩니다.
   - *-R*은 관리 세트 ID를 하위 디렉토리 및 파일에 재귀적으로 지정합니다.
   - *admin-set-id*는 고유한 정수값입니다.
   - *directory-or-file-name*은 관리 세트 ID를 지정하려는 디렉토리나 파일의 이름입니다.

   예제에서는 관리 ID *1*을 */hsm/hsmfs1/portalproject/* 디렉토리와 하위 디렉토리의 모든 파일에 지정합니다.

   ```
   [server1]root@solaris:~# samchaid -R 1 /hsm/hsmfs1/portalproject/
   ```

4. 원하는 경우 지정을 확인할 수 있습니다. *sls -D directory-path* 명령을 사용합니다. 여기서 *-D*는 *directory-path*의 파일 및 디렉토리에 대한 상세한 Oracle HSM 디렉토리 목록을 지정합니다.

   ```
   [server1]root@solaris:~# sls -D /hsm/hsmfs1/
   /portalproject:
     mode: drwxr-xr-x    links:   2  owner: root        group: root
     length:      4096  admin id:    1  inode:     1047.1
     project: user.root(1)
     access:      Feb 24 12:49  modification: Feb 24 12:44
     changed:     Feb 24 12:49  attributes:   Feb 24 12:44
     creation:    Feb 24 12:44  residence:    Feb 24 12:44
   ```

5. 새로 만든 빈 파일 시스템에 쿼터를 설정하는 경우 "새 파일 시스템에서 쿼터를 사용하도록 구성"으로 이동합니다.
6. 이미 파일을 보유한 파일 시스템에 쿼터를 설정하는 경우 “기존 파일 시스템에서 쿼터를 사용하도록 구성”으로 이동합니다.

새 파일 시스템에서 쿼터를 사용하도록 구성

새 파일 시스템을 만들고 현재 파일 시스템에 상주하는 파일이 없는 경우 이 절차를 사용하십시오.

1. 파일 시스템 서버에 *root*로 로그인합니다.

   예제에서 서버 이름은 *server2*로 지정됩니다.

   ```
   [server2]root@solaris:~#
   ```

2. 새 파일 시스템이 현재 마운트되지 않은 경우 계속하기 전에 마운트합니다.
3. 그룹에 대한 쿼터를 설정해야 하는 경우 파일 시스템 루트 디렉토리에 그룹 쿼터 파일 *.quota_g*를 만듭니다. Solaris 명령 *dd if=/dev/zero of=mountpoint/.quota_g bs=4096 count=number-blocks*를 사용합니다. 설명:
   - *if=/dev/zero*는 UNIX 특수 파일 */dev/zero*에서 입력으로 널 문자를 지정합니다.
   - *of=mountpoint/.quota_g*는 출력 파일을 지정합니다. 여기서 *mountpoint*는 파일 시스템의 마운트 지점 디렉토리입니다.
   - *bs=4096*은 쓰기용 블록 크기를 *4096* 바이트로 설정합니다.
   - *count=number-blocks*는 쓰기용 블록 수를 지정합니다. 이 값은 파일이 보유할 레코드 수에 따라 다릅니다. 각 지정된 쿼터마다 하나의 128바이트 레코드가 있으므로 한 블록이 32개 레코드를 수용할 수 있습니다.

   예제에서는 */newsamfs*에 마운트된 파일 시스템 *newsamfs*에 대해 그룹 쿼터 파일을 만듭니다. 요구 사항 수집 단계 동안, 파일 시스템에서 쿼터가 필요한 세 그룹으로 *dev*, *cit*, *pgmt*를 식별했습니다. 다른 그룹 쿼터는 추가하지 않을 것이므로 파일 크기를 한 블록으로 지정합니다.

   ```
   [server2]root@solaris:~# dd if=/dev/zero of=/newsamfs/.quota_g bs=4096 count=1
   ```

4. 관리 세트에 대한 쿼터를 설정해야 하는 경우 파일 시스템 루트 디렉토리에 관리 세트 쿼터 파일 *.quota_a*를 만듭니다. Solaris 명령 *dd if=/dev/zero of=mountpoint/.quota_a bs=4096*을 사용합니다. 설명:
   - *mountpoint*는 파일 시스템에 대한 마운트 지점 디렉토리입니다.
   - *.quota_a*는 출력 파일의 이름입니다.
   - *4096*은 기록할 블록 크기(바이트)입니다.
   - *number-blocks*는 기록할 블록 수입니다.

   예제에서는 */newsamfs*에 마운트된 파일 시스템 *newsamfs*에 대해 관리 세트 쿼터 파일을 만듭니다. 요구 사항 수집 단계 동안, 파일 시스템에서 쿼터가 필요한 두 프로젝트로 *portal*(관리 세트 ID *1*) 및 *lockbox*(관리 세트 ID *2*)를 식별했습니다. 다른 관리 세트 쿼터는 추가하지 않을 것이므로 파일 크기를 한 블록으로 지정합니다.

   ```
   [server2]root@solaris:~# dd if=/dev/zero of=/newsamfs/.quota_a bs=4096 count=1
   ```

5. 사용자에 대한 쿼터를 설정해야 할 경우 파일 시스템 루트 디렉토리에 사용자 쿼터 파일인 *.quota_u*를 만듭니다. Solaris 명령 *dd if=/dev/zero of=mountpoint/.quota_u bs=4096 count=number-blocks*를 사용합니다. 설명:
   - *mountpoint*는 파일 시스템에 대한 마운트 지점 디렉토리입니다.
   - *.quota_u*는 출력 파일의 이름입니다.
   - *4096*은 기록할 블록 크기(바이트)입니다.
   - *number-blocks*는 기록할 블록 수입니다.

   예제에서는 */newsamfs*에 마운트된 파일 시스템 *newsamfs*에 대해 사용자 쿼터 파일을 만듭니다. 요구 사항 수집 단계 동안, 파일 시스템에서 특정 쿼터가 필요한 사용자로 *jr23547*을 식별했습니다. 다른 개별 사용자 쿼터는 추가하지 않을 것이므로 파일 크기를 한 블록으로 지정합니다.

   ```
   [server2]root@solaris:~# dd if=/dev/zero of=/newsamfs/.quota_u bs=4096 count=1
   ```

6. 파일 시스템을 마운트 해제합니다.

   파일 시스템을 마운트 해제해야 다시 마운트하고 쿼터 파일을 사용으로 설정할 수 있습니다.

   ```
   [server2]root@solaris:~# umount /newsamfs
   ```

7. 파일 시스템 검사를 수행합니다.

   ```
   [server2]root@solaris:~# samfsck -F newsamfs
   ```

8. 파일 시스템을 다시 마운트합니다.

   시스템이 파일 시스템의 루트 디렉토리에서 하나 이상의 쿼터 파일을 감지할 때 쿼터가 사용으로 설정됩니다.

   파일 시스템은 기본적으로 쿼터가 사용으로 설정된 채 마운트되기 때문에 */etc/vfstab* 또는 *samfs.cmd* 파일에 *quota* 마운트 옵션을 포함할 필요가 없습니다.

   ```
   [server2]root@solaris:~# mount /newsamfs
   ```

9. 그 다음, 필요에 따라 쿼터를 설정하거나 업데이트합니다. “그룹, 프로젝트, 디렉토리 및 사용자에 대한 쿼터 설정”을 참조하십시오.

기존 파일 시스템에서 쿼터를 사용하도록 구성

이미 파일을 보유한 파일 시스템에 대한 쿼터를 만드는 경우 이 절차를 사용하십시오.

1. 파일 시스템 서버에 *root*로 로그인합니다.

예제에서는 서버 이름이 *server1*로 지정됩니다.

```
[server1]root@solaris:~#
```

2. 텍스트 편집기에서 */etc/vfstab* 파일을 열고 *noquota* 마운트 옵션이 설정되지 않았는지 확인합니다.

   예제에서는 *vi* 텍스트 편집기에서 파일을 엽니다. *noquota* 마운트 옵션이 설정되었습니다.

```
[server1]root@solaris:~# vi /etc/vfstab
#File
#Device        Device   Mount             System  fsck  Mount    Mount
#to Mount      to fsck  Point             Type    Pass  at Boot  Options
#------------  -------  ---------------   ------  ----  -------  ------------
/devices       -        /devices          devfs   -     no       -
/proc          -        /proc             proc    -     no       -
...
hsmfs1        -        /hsm/hsmfs1          samfs   -     no       noquota
```

3. */etc/vfstab* 파일에 *noquota* 마운트 옵션이 설정된 경우 삭제하고 파일을 저장합니다.

```
[server1]root@solaris:~# vi /etc/vfstab
#File
#Device        Device   Mount             System  fsck  Mount    Mount
#to Mount      to fsck  Point             Type    Pass  at Boot  Options
#------------  -------  ---------------   ------  ----  -------  ------------
/devices       -        /devices          devfs   -     no       -
/proc          -        /proc             proc    -     no       -
...
hsmfs1        -        /hsm/hsmfs1          samfs   -     no       -
:wq
[server1]root@solaris:~#
```

4. 텍스트 편집기에서 */etc/opt/SUNWsamfs/samfs.cmd* 파일을 열고 *noquota* 마운트 옵션이 설정되지 않았는지 확인합니다.

   예제에서는 *vi* 텍스트 편집기에서 파일을 엽니다. *noquota* 마운트 옵션이 설정되지 않았습니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/samfs.cmd
# These are the defaults.  To change the default behavior, uncomment the
# appropriate line (remove the '#' character from the beginning of the line)
```

```
# and change the value.
#
#inodes = 0
#fs = hsmfs1
#  forcedirectio (default no forcedirectio)
#  high = 80
#  low = 70
#  weight_size = 1.
#  weight_age = 1.
#  readahead = 128
...
#  dio_wr_ill_min = 0
#  dio_wr_consec = 3
#  qwrite        (ma filesystem, default no qwrite)
#  shared_writer (ma filesystem, default no shared_writer)
#  shared_reader (ma filesystem, default no shared_reader)
```

5. */etc/opt/SUNWsamfs/samfs.cmd* 파일에 *noquota* 마운트 옵션이 설정된 경우 삭제하고 파일을 저장합니다.
6. */etc/vfstab* 파일 및/또는 */etc/opt/SUNWsamfs/samfs.cmd* 파일에서 *noquota* 마운트 옵션을 삭제했으면 파일 시스템을 마운트 해제합니다.

   noquota 마운트 옵션을 제거할 때 파일 시스템을 마운트 해제해야 쿼터가 사용으로 설정된 채 다시 마운트할 수 있습니다.

   ```
   [server1]root@solaris:~# umount /hsm/hsmfs1
   ```

7. 파일 시스템이 현재 마운트되지 않은 경우 지금 마운트합니다.

   파일 시스템을 마운트해야 쿼터를 사용으로 설정할 수 있습니다.

   ```
   [server1]root@solaris:~# mount /hsm/hsmfs1
   ```

8. 파일 시스템의 루트 디렉토리로 변경하고 기존 쿼터 파일이 있는지 확인합니다. Solaris 명령 *ls -a*를 사용하고 *.quota_g*, *.quota_a* 및/또는 *.quota_u* 파일을 찾습니다.

   예제에서는 현재 쿼터 파일이 존재하지 않습니다.

   ```
   [server1]root@solaris:~# cd /hsm/hsmfs1
   [server1]root@solaris:~# ls -a /hsm/hsmfs1
   .              .archive       .fuid          .stage         portalproject
   ..             .domain        .inodes        lost+found
   ```

9. 쿼터 파일이 존재하면 수정하지 마십시오.

10. 그룹에 대한 쿼터를 설정해야 하는데 파일 시스템 루트 디렉토리에 그룹 쿼터 파일 *.quota_g*가 아직 없으면 지금 파일을 만듭니다. Solaris 명령 *dd if=/dev/zero of=mountpoint/.quota_g bs=4096 count=number-blocks*를 사용합니다. 설명:
    - *if=/dev/zero*는 UNIX 특수 파일 */dev/zero*에서 입력으로 널 문자를 지정합니다.
    - *of=mountpoint/.quota_g*는 출력 파일을 지정합니다. 여기서 *mountpoint*는 파일 시스템의 마운트 지점 디렉토리입니다.
    - *bs=4096*은 쓰기용 블록 크기를 *4096*바이트로 설정합니다.
    - *count=number-blocks*는 쓰기용 블록 수를 지정합니다. 이 값은 파일이 보유할 레코드 수에 따라 다릅니다. 각 지정된 쿼터마다 하나의 128바이트 레코드가 있으므로 한 블록이 32개 레코드를 수용할 수 있습니다.

    예제에서는 */hsm/hsmfs1*에 마운트된 파일 시스템 */hsm/hsmfs1*에 대해 그룹 쿼터 파일을 만듭니다. 요구 사항 수집 단계 동안, 파일 시스템에서 쿼터가 필요한 세 그룹으로 *dev*, *cit*, *pgmt*를 식별했습니다. 다른 그룹 쿼터는 추가하지 않을 것이므로 파일 크기를 한 블록으로 지정합니다.

    ```
    [server1]root@solaris:~# dd if=/dev/zero of=/hsm/hsmfs1/.quota_g bs=4096 count=1
    ```

11. 관리 세트에 대한 쿼터를 설정해야 하는데 파일 시스템 루트 디렉토리에 관리 세트 쿼터 파일 *.quota_a*가 아직 없으면 지금 파일을 만듭니다. Solaris 명령 *dd if=/dev/zero of=mountpoint/.quota_a bs=4096 count=number-blocks*를 사용합니다. 설명:
    - *mountpoint*는 파일 시스템에 대한 마운트 지점 디렉토리입니다.
    - *.quota_a*는 출력 파일의 이름입니다.
    - *4096*은 기록할 블록 크기(바이트)입니다.
    - *number-blocks*는 기록할 블록 수입니다.

    예제에서는 */hsm/hsmfs1*에 마운트된 파일 시스템 */hsm/hsmfs1*에 대해 관리 세트 쿼터 파일을 만듭니다. 요구 사항 수집 단계 동안, 파일 시스템에서 쿼터가 필요한 두 프로젝트로 *portal*(관리 세트 ID *1*) 및 *lockbox*(관리 세트 ID *2*)를 식별했습니다. 다른 관리 세트 쿼터는 추가하지 않을 것이므로 파일 크기를 한 블록으로 지정합니다.

    ```
    [server1]root@solaris:~# dd if=/dev/zero of=/hsm/hsmfs1/.quota_a bs=4096 count=1
    ```

12. 사용자에 대한 쿼터를 설정해야 하는데 파일 시스템 루트 디렉토리에 사용자 쿼터 파일 *.quota_u*가 아직 없으면 지금 파일을 만듭니다. Solaris 명령 *dd if=/dev/zero of=mountpoint/.quota_u bs=4096 count=number-blocks*를 사용합니다. 설명:
    - *mountpoint*는 파일 시스템에 대한 마운트 지점 디렉토리입니다.
    - *.quota_u*는 출력 파일의 이름입니다.
    - *4096*은 기록할 블록 크기(바이트)입니다.
    - *number-blocks*는 기록할 블록 수입니다.

    예제에서는 */hsm/hsmfs1*에 마운트된 파일 시스템 */hsm/hsmfs1*에 대해 사용자 쿼터 파일을 만듭니다. 요구 사항 수집 단계 동안, 파일 시스템에서 특정 쿼터가 필요한 사용

자로 *jr23547*을 식별했습니다. 다른 개별 사용자 쿼터는 추가하지 않을 것이므로 파일 크기를 한 블록으로 지정합니다.

```
[server1]root@solaris:~# dd if=/dev/zero of=/hsm/hsmfs1/.quota_u bs=4096 count=1
```

13. 파일 시스템을 마운트 해제합니다.

    파일 시스템을 마운트 해제해야 다시 마운트하고 쿼터 파일을 사용으로 설정할 수 있습니다.

```
[server1]root@solaris:~# umount /hsm/hsmfs1
```

14. 파일 시스템 검사를 수행합니다.

```
[server1]root@solaris:~# samfsck -F /hsm/hsmfs1
```

15. 파일 시스템을 다시 마운트합니다.

    시스템이 파일 시스템의 루트 디렉토리에서 하나 이상의 쿼터 파일을 감지할 때 쿼터가 사용으로 설정됩니다.

    파일 시스템은 기본적으로 쿼터가 사용으로 설정된 채 마운트되기 때문에 */etc/vfstab* 또는 *samfs.cmd* 파일에 *quota* 마운트 옵션을 포함할 필요가 없습니다.

```
[server1]root@solaris:~# mount /hsm/hsmfs1
```

16. 그런 다음 그룹, 프로젝트, 디렉토리 및 사용자에 대한 쿼터 설정을 수행합니다.

그룹, 프로젝트, 디렉토리 및 사용자에 대한 쿼터 설정

samquota 명령을 사용하여 새 쿼터를 설정하고 기존 쿼터를 조정합니다. 아래 절차를 따르십시오.

1. 일단 스토리지 요구 사항을 특징지었으면 각 그룹, 사용자 및 비그룹 조직에 대한 적절한 쿼터를 결정합니다. 다음 요인을 고려하여 필요에 따라 조정하십시오.
   - 모든 사용자가 필요한 평균/최대 블록 수와 비교한 파일 시스템의 크기
   - 모든 사용자가 필요한 평균/최대 inode 수와 비교한 파일 시스템의 inode 수
   - 주어진 시간에 최대 요구 사항에 가까운 사용자의 개수와 유형.
2. 파일 시스템 서버에 *root*로 로그인합니다.

   예제에서는 서버 이름이 *server1*로 지정됩니다.

```
[server1]root@solaris:~#
```

3. 각 그룹에 대해 한계를 설정합니다. *samquota -b number-blocks:type[:scope] -f number-files:type[:scope] -t interval[:scope] -G groupID [directory-or-file]* 명령을 사용합니다. 설명:
   - *-b number-blocks*는 파일 시스템에 저장할 수 있는 512킬로바이트 블록의 최대 개수를 정수값 *number-blocks*로 설정합니다. 대체 크기 지정 방법은 *samquota* 매뉴얼 페이지를 참조하십시오. *0*(제로) 값은 무제한 블록 수를 지정합니다.
   - *:*은 필드 구분자입니다.
   - *type*은 하드 한계 *h* 또는 소프트 한계 *s*로 한계 종류를 지정합니다.
   - *scope*(선택사항)은 한계가 적용되는 스토리지의 유형을 식별합니다. 온라인(디스크 캐시) 전용 스토리지의 경우 *o*, 전체 스토리지의 경우 *t*(기본값, 디스크 캐시와 아카이브 스토리지 모두 포함)가 될 수 있습니다.
   - *-f number-files*는 파일 시스템에 저장할 수 있는 최대 파일 수를 정수 값인 *number-files*로 설정합니다. *0*(제로) 값은 무제한 파일 수를 지정합니다.
   - *-t number-seconds*는 유예 기간(소프트 한계를 초과할 수 있는 시간)을 초 수를 나타내는 정수 *number-seconds*로 설정합니다. 대체 시간 지정 방법은 *samquota* 매뉴얼 페이지를 참조하십시오.
   - *-G groupID*는 그룹 이름 또는 그룹의 정수 식별자를 지정합니다. *0*(제로) 값은 모든 그룹에 대해 기본 한계를 설정합니다.
   - *directory-or-file*(선택사항)은 특정 파일 시스템의 마운트 지점 디렉토리이거나 쿼터를 설정할 특정 디렉토리나 파일입니다.

   예제에서는 요구 사항 수집 단계의 예상치를 사용하여 */hsm/hsmfs1* 파일 시스템에서 *dev* 그룹이 사용할 수 있는 스토리지 공간량과 저장할 수 있는 파일 수에 대해 하드 및 소프트 한계를 설정합니다. 온라인 전용 스토리지에 대해 유예 기간을 *43200*초(12시간)로 설정합니다. 아래의 명령은 단일 행으로 입력됩니다. 줄바꿈은 백슬래시 문자로 이스케이프합니다.

```
[server1]root@solaris:~# samquota -b 3019898880:h:t -f 30000:h:t /
-G dev /hsm/hsmfs1
[server1]root@solaris:~# samquota -b 2013265920:s:t -f 15000:s:t -t 43200:o /
-G dev /hsm/hsmfs1
[server1]root@solaris:~#
```

4. 각 관리 세트에 대해 한계를 설정합니다. *samquota -b number-blocks:type[:scope] -f number-files:type[:scope] -t interval[:scope] -A adminsetID [directory-or-file]* 명령을 사용합니다. 여기서 *-A adminsetID*는 관리 세트를 고유하게 식별하는 정수값입니다.

   *adminsetID*를 *0*(제로)으로 설정하면 모든 관리 세트에 대해 기본 한계가 설정됩니다.

   예제에서는 요구 사항 수집 단계의 예상치를 사용하여 */hsm/hsmfs1* 파일 시스템에서 *portal* 프로젝트(관리 세트 ID *1*)가 사용할 수 있는 스토리지 공간량과 저장할 수 있는 파일 수에 대해 하드 및 소프트 한계를 설정합니다. 사용된 전체 스토리지(기본 범위)에

대해 유예 기간을 *43200*초(12시간)로 설정합니다. 아래의 명령은 단일 라인으로 입력됩니다. 줄바꿈은 백슬래시 문자로 이스케이프됩니다.

```
[server1]root@solaris:~# samquota -b 377487360:h:t -f 7000:h:t -A 1 /hsm/hsmfs1
[server1]root@solaris:~# samquota -b 314572800:s:t -f 4000:s:t -t 43200 /
-A 1 /hsm/hsmfs1
[server1]root@solaris:~#
```

5. 각 개별 사용자에 대해 한계를 설정합니다. *samquota -b number-blocks:type[:scope] -f number-files:type[:scope] -t interval[:scope] -U userID [directory-or-file]* 명령을 사용합니다. 여기서 *-U userID*는 사용자 이름 또는 사용자의 정수 식별자입니다.

   *userID*를 *0*(제로)으로 설정하면 모든 사용자에 대해 기본 한계가 설정됩니다.

   예제에서는 요구 사항 수집 단계의 예상치를 사용하여 */hsm/hsmfs1* 파일 시스템에서 *jr23547* 사용자가 사용할 수 있는 스토리지 공간량과 *jr23547*이 저장할 수 있는 파일 수에 대해 하드 및 소프트 한계를 설정합니다. 사용된 전체 스토리지(기본 범위)에 대해 유예 기간을 *1209600*초(2주)로 설정합니다. 아래의 명령은 단일 라인으로 입력됩니다. 줄바꿈은 백슬래시 문자로 이스케이프됩니다.

```
[server1]root@solaris:~# samquota -b 100663296:h:t -f 600:h:t /
-U jr23547 /hsm/hsmfs1
[server1]root@solaris:~# samquota -b 10485760:s:t -f 2000:s:t -t 1209600 /
-U jr23547 /hsm/hsmfs1
[server1]root@solaris:~#
```

6. 여기서 중지합니다.

불일치 쿼터 복구

루트 디렉토리에 쿼터 파일이 있을 때 Oracle HSM 파일 시스템을 *noquota* 마운트 옵션으로 마운트하는 경우, 블록이나 파일을 할당하거나 비울 때 쿼터 레코드가 불일치해집니다. 이 상황에서 다음과 같이 하십시오.

1. 파일 시스템 서버에 *root*로 로그인합니다.

   예제에서는 서버 이름이 *server1*로 지정됩니다.

```
[server1]root@solaris:~#
```

2. 영향을 받는 파일 시스템을 마운트 해제합니다.

   예제에서는 *samfs2* 파일 시스템을 마운트 해제합니다.

```
[server1]root@solaris:~# umount samfs2
```

```
[server1]root@solaris:~#
```

3. 텍스트 편집기에서 */etc/vfstab* 파일을 열고 *noquota* 마운트 옵션이 설정되지 않았는지 확인합니다.

   예제에서는 *vi* 텍스트 편집기에서 파일을 엽니다. *noquota* 마운트 옵션이 설정되었습니다.

```
[server1]root@solaris:~# vi /etc/vfstab
#File
#Device        Device   Mount             System  fsck  Mount    Mount
#to Mount      to fsck  Point             Type    Pass  at Boot  Options
#------------  -------  ---------------   ------  ----  -------  ------------
/devices       -        /devices          devfs   -     no       -
/proc          -        /proc             proc    -     no       -
...
samfs2         -        /samfs2           samfs   -     no       noquota
```

4. */etc/vfstab* 파일에 *noquota* 마운트 옵션이 설정된 경우 삭제하고 파일을 저장합니다.

```
[server1]root@solaris:~# vi /etc/vfstab
#File
#Device        Device   Mount             System  fsck  Mount    Mount
#to Mount      to fsck  Point             Type    Pass  at Boot  Options
#------------  -------  ---------------   ------  ----  -------  ------------
/devices       -        /devices          devfs   -     no       -
/proc          -        /proc             proc    -     no       -
...
samfs2         -        /samfs2           samfs   -     no       -
:wq
[server1]root@solaris:~#
```

5. 텍스트 편집기에서 */etc/opt/SUNWsamfs/samfs.cmd* 파일을 열고 *noquota* 마운트 옵션이 설정되지 않았는지 확인합니다.

   예제에서는 *vi* 텍스트 편집기에서 파일을 엽니다. *noquota* 마운트 옵션이 설정되지 않았습니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/samfs.cmd
# These are the defaults.  To change the default behavior, uncomment the
# appropriate line (remove the '#' character from the beginning of the line)
# and change the value.
#
#inodes = 0
```

```
#fs = hsmfs1
#  forcedirectio (default no forcedirectio)
#  high = 80
#  low = 70
#  weight_size = 1.
#  weight_age = 1.
#  readahead = 128
...
#  dio_wr_ill_min = 0
#  dio_wr_consec = 3
#  qwrite        (ma filesystem, default no qwrite)
#  shared_writer (ma filesystem, default no shared_writer)
#  shared_reader (ma filesystem, default no shared_reader)
```

6. */etc/opt/SUNWsamfs/samfs.cmd* 파일에 *noquota* 마운트 옵션이 설정된 경우 삭제하고 파일을 저장합니다.
7. 불일치 쿼터 레코드를 복구합니다. *samfsck -F family-set-name* 명령을 사용합니다. 여기서 *family-set-name*은 */etc/opt/SUNWsamfs/mcf* 파일에서 파일 시스템의 패밀리 세트 이름입니다.

   ```
   [server1]root@solaris:~# samfsck -F samfs2
   ```

8. 파일 시스템을 다시 마운트합니다.

   시스템이 파일 시스템의 루트 디렉토리에서 하나 이상의 쿼터 파일을 감지할 때 쿼터가 사용으로 설정됩니다.

   파일 시스템은 기본적으로 쿼터가 사용으로 설정된 채 마운트되기 때문에 */etc/vfstab* 또는 *samfs.cmd* 파일에 *quota* 마운트 옵션을 포함할 필요가 없습니다.

   ```
   [server1]root@solaris:~# mount /samfs2
   [server1]root@solaris:~#
   ```

9. 여기서 중지합니다.

쿼터 검사

관리자와 사용자 모두 쿼터 및 리소스 사용량을 모니터할 수 있습니다. *root* 사용자는 *samquota* 명령을 사용하여 사용자, 그룹, 관리 세트에 대한 쿼터 보고서를 생성할 수 있습니다. 파일 시스템 사용자는 *squota* 명령을 사용하여 자신이 소유한 쿼터를 검사할 수 있습니다.

아래 절차를 참조하십시오.

- 파일 시스템 관리자로서 쿼터 모니터링
- 소유한 사용자 쿼터 모니터링

파일 시스템 관리자로서 쿼터 모니터링

1. 파일 시스템 서버에 *root*로 로그인합니다.

   예제에서는 서버 이름이 *server1*로 지정됩니다.

   ```
   [server1]root@solaris:~#
   ```

2. 모든 그룹에 대한 쿼터 통계를 표시하려면 *samquota -g [directory-or-file]* 명령을 사용합니다. 여기서 선택적 *directory-or-file* 매개변수는 보고서의 범위를 지정된 디렉토리에 마운트된 파일 시스템, 지정된 디렉토리 자체 또는 지정된 파일로 제한합니다.

   예제에서는 */hsm/hsmfs1*에 마운트되는 *hsmfs1* 파일 시스템에 대한 보고서를 요청합니다.

   ```
   [server1]root@solaris:~# samquota -g /hsm/hsmfs1
   ```

3. 모든 관리 세트에 대한 쿼터 통계를 표시하려면 *samquota -a [directory-or-file]* 명령을 사용합니다. 여기서 선택적 *directory-or-file* 매개변수는 보고서의 범위를 지정된 디렉토리에 마운트된 파일 시스템, 지정된 디렉토리 자체 또는 지정된 파일로 제한합니다.

   예제에서는 */hsm/hsmfs1*에 마운트되는 *hsmfs1* 파일 시스템에 대한 보고서를 요청합니다.

   ```
   [server1]root@solaris:~# samquota -a /hsm/hsmfs1
   ```

4. 모든 사용자에 대한 쿼터 통계를 표시하려면 *samquota -u [directory-or-file]* 명령을 사용합니다. 여기서 선택적 *directory-or-file* 매개변수는 보고서의 범위를 지정된 디렉토리에 마운트된 파일 시스템, 지정된 디렉토리 자체 또는 지정된 파일로 제한합니다.

   예제에서는 */hsm/hsmfs1*에 마운트되는 *hsmfs1* 파일 시스템에 대한 보고서를 요청합니다.

   ```
   [server1]root@solaris:~# samquota -u /hsm/hsmfs1
   ```

5. 특정 그룹에 대한 쿼터 통계를 표시하려면 *samquota -G groupID [directory-or-file]* 명령을 사용합니다. 여기서 *groupID*는 그룹에 대한 그룹 이름 또는 정수 식별자를 지정하고 선택적 *directory-or-file* 매개변수는 보고서의 범위를 지정된 디렉토리에 마운트된 파일 시스템, 지정된 디렉토리 자체 또는 지정된 파일로 제한합니다.

   예제에서는 */hsm/hsmfs1*에 마운트되는 *hsmfs1* 파일 시스템에서 *dev* 그룹의 쿼터에 대한 보고서를 요청합니다.

```
[server1]root@solaris:~# samquota -G dev /hsm/hsmfs1
```

6. 특정 관리 세트에 대한 쿼터 통계를 표시하려면 *samquota -A adminsetID [directory-or-file]* 명령을 사용합니다. 여기서 *adminsetID*는 관리 세트에 대한 정수 식별자를 지정하고 선택적 *directory-or-file* 매개변수는 보고서의 범위를 지정된 디렉토리에 마운트된 파일 시스템, 지정된 디렉토리 자체 또는 지정된 파일로 제한합니다.

   예제에서는 */hsm/hsmfs1*에 마운트되는 *hsmfs1* 파일 시스템에서 관리 세트 *1*의 쿼터에 대한 보고서를 요청합니다.

```
[server1]root@solaris:~# samquota -A 1 /hsm/hsmfs1
```

7. 특정 사용자에 대한 쿼터 통계를 표시하려면 *samquota -U userID [directory-or-file]* 명령을 사용합니다. 여기서 *userID*는 사용자에 대한 사용자 이름 또는 정수 식별자를 지정하고 선택적 *directory-or-file* 매개변수는 보고서의 범위를 지정된 디렉토리에 마운트된 파일 시스템, 지정된 디렉토리 자체 또는 지정된 파일로 제한합니다.

   예제에서는 */hsm/hsmfs1*에 마운트되는 *hsmfs1* 파일 시스템에서 *jr23547* 사용자의 쿼터에 대한 보고서를 요청합니다.

```
[server1]root@solaris:~# samquota -U jr23547 /hsm/hsmfs1
```

8. 여기서 중지합니다.

소유한 사용자 쿼터 모니터링

1. 사용자 ID를 사용하여 파일 시스템 호스트에 로그인합니다.

   예제에서는 *server1* 호스트에 *od447* 사용자로 로그인합니다.

```
[server1]od447@solaris:~#
```

2. 모든 그룹에 대한 쿼터 통계를 표시하려면 *squota [directory-or-file]* 명령을 사용합니다. 여기서 선택적 *directory-or-file* 매개변수는 보고서의 범위를 지정된 디렉토리에 마운트된 파일 시스템, 지정된 디렉토리 자체 또는 지정된 파일로 제한합니다.

   예제에서는 모든 파일 시스템에 대한 보고서를 요청합니다.

```
[server1]od447@solaris:~# squota
                                        Limits
            Type    ID    In Use     Soft     Hard
    /hsm/hsmfs1
    Files  group   101         1     1000     1200
```

```
        Blocks group   101         8    20000    30000
        Grace period                    25920
        No user quota entry.
    [server1]od447@solaris:~#
```

3. 여기서 중지합니다.

유예 기간 일시적 연장 또는 취소

유예 기간을 일시적으로 연장하거나 유예 기간을 단축해야 하는 경우 이를 수행할 수 있습니다.

- 지정된 양만큼 유예 기간 연장
- 유예 기간 다시 시작
- 유예 기간 조기 종료.

지정된 양만큼 유예 기간 연장

그룹, 사용자, 관리 세트가 쿼터에 지정된 소프트 한계를 초과한 경우, 일시적이지만 현재 유예 기간이 허용하는 것보다 오랫동안 소프트 한계 이상을 유지하려면 다음과 같이 연장 권한을 부여할 수 있습니다.

1. 파일 시스템 서버에 *root*로 로그인합니다.

   예제에서는 *server1* 호스트에 로그인합니다.

   ```
   [server1]root@solaris:~#
   ```

2. 연장이 필요한 쿼터를 확인합니다. *samquota -quota-type ID [directory-or-file]* 명령을 사용합니다. 설명:
   - *quota-type ID*는 *G* 더하기 그룹 이름 또는 ID 번호, *A* 더하기 관리 세트 ID 번호, *U* 더하기 사용자 이름 또는 ID 번호입니다.
   - *directory-or-file*(선택사항)은 유예 기간을 연장해야 하는 특정 파일 시스템의 마운트 지점 디렉토리이거나 특정 디렉토리 또는 파일입니다.

   예제에서는 *dev* 그룹이 소프트 한계를 크게 초과하며 남은 유예 기간은 2시간 뿐입니다.

   ```
   [server1]root@solaris:~# samquota -G dev /hsm/hsmfs1
                                  Online Limits                    Total Limits
           Type   ID    In Use       Soft      Hard       In Use     Soft  Hard
   /hsm/hsmfs1
   Files  group  101       323      15000      30000         323    15000  30000
   Blocks group  101 3109330961 2013265920 3019898880 3109330961 2013265920
    3019898880
   Grace period                4320                           4320
   ```

```
---> Warning:  soft limits to be enforced in 2h21m16s
[server1]root@solaris:~#
```

3. 보증된 경우 유예 기간을 연장합니다. *samquota -quota-type ID -x number-seconds [directory-or-file]* 명령을 사용합니다. 설명:
   - *quota-type ID*는 *G* 더하기 그룹 이름 또는 ID 번호, *A* 더하기 관리 세트 ID 번호, *U* 더하기 사용자 이름 또는 ID 번호입니다.
   - *directory-or-file*(선택사항)은 유예 기간을 연장해야 하는 특정 파일 시스템의 마운트 지점 디렉토리이거나 특정 디렉토리 또는 파일입니다.
   - *number-seconds*는 연장 초 수를 나타내는 정수입니다(대체 시간 지정 방법은 *samquota* 매뉴얼 페이지 참조).

   계속하려면 프롬프트가 표시될 때 *y*(예)를 입력합니다.

   예제에서는 *hsmfs1* 파일 시스템의 파일에 대해 *dev* 그룹의 유예 기간을 *2678400*초(31일)로 연장합니다.

```
[server1]root@solaris:~# samquota -G dev -x 2678400 /hsm/hsmfs1
Setting Grace Timer:  continue? y
```

   dev 그룹 쿼터를 다시 검사하면 유예 기간이 연장되었습니다.

```
[server1]root@solaris:~# samquota -G dev /hsm/hsmfs1
                                Online Limits                    Total Limits
        Type    ID    In Use       Soft      Hard   In Use       Soft   Hard
/hsm/hsmfs1
Files  group   101       323      15000      30000     323      15000   30000
Blocks group   101     43208 2013265920 3019898880   43208 2013265920 3019898880
Grace period                2678400                      2678400
---> Warning:  soft limits to be enforced in 31d
[server1]root@solaris:~#
```

4. 그룹, 관리 세트, 사용자에 정기적으로 연장이 필요한 경우 스토리지 요구 사항을 다시 평가하고 유예 기간을 영구적으로 늘리는 것을 고려하십시오. "그룹, 프로젝트, 디렉토리 및 사용자에 대한 쿼터 설정" 절차를 사용하십시오.
5. 여기서 중지합니다.

유예 기간 다시 시작

그룹, 사용자, 관리 세트가 쿼터에 지정된 소프트 한계를 초과한 경우, 현재 유예 기간이 만료되기 전에 소프트 한계 아래로 떨어뜨릴 공간을 재빨리 확보할 수 없으면 유예 기간을 다시 시작할 수 있습니다. 다음과 같이 하십시오.

1. 파일 시스템 서버에 *root*로 로그인합니다.

예제에서는 *server1* 호스트에 로그인합니다.

```
[server1]root@solaris:~#
```

2. 연장이 필요한 쿼터를 확인합니다. *samquota -quota-type ID [directory-or-file]* 명령을 사용합니다. 설명:
   - *quota-type ID*는 *G* 더하기 그룹 이름 또는 ID 번호, *A* 더하기 관리 세트 ID 번호, *U* 더하기 사용자 이름 또는 ID 번호입니다.
   - *directory-or-file*(선택사항)은 유예 기간을 연장해야 하는 특정 파일 시스템의 마운트 지점 디렉토리이거나 특정 디렉토리 또는 파일입니다.

   예제에서는 *cit* 그룹이 *hsmfs1* 파일 시스템에 대한 소프트 한계를 초과하며 남은 유예 기간은 1시간 조금 넘습니다.

```
[server1]root@solaris:~# samquota -G cit /hsm/hsmfs1
                                Online Limits                    Total Limits
        Type   ID    In Use        Soft       Hard     In Use     Soft  Hard
/hsm/hsmfs1
Files  group  119       762         750       1500        762      750  1500
Blocks group  119 3109330961 2013265920 3019898880  120096782  157286400
 235929600
Grace period                4320                             4320
---> Warning:  soft limits to be enforced in 1h11m23s
[server1]root@solaris:~#
```

3. 다음에 파일이나 블록을 할당할 때 유예 기간을 전체 시작 크기로 재설정하려면 유예 기간 타이머를 지웁니다. *samquota -quota-type ID -x clear [directory-or-file]* 명령을 사용합니다. 설명:
   - *quota-type ID*는 *G* 더하기 그룹 이름 또는 ID 번호, *A* 더하기 관리 세트 ID 번호, *U* 더하기 사용자 이름 또는 ID 번호입니다.
   - *directory-or-file*(선택사항)은 유예 기간을 연장해야 하는 특정 파일 시스템의 마운트 지점 디렉토리이거나 특정 디렉토리 또는 파일입니다.

   계속하려면 프롬프트가 표시될 때 *y*(예)를 입력합니다.

   예제에서는 *hsmfs1* 파일 시스템에서 *cit* 그룹의 쿼터에 대한 유예 기간 타이머를 지웁니다.

```
[server1]root@solaris:~# samquota -G cit -x clear /hsm/hsmfs1
Setting Grace Timer:  continue? y
[server1]root@solaris:~#
```

cit 그룹 쿼터를 다시 검사하면 파일이 할당되었고 유예 기간이 *12h*, 즉 12시간(*4320*초)으로 재설정되었습니다.

```
[server1]root@solaris:~# samquota -G cit /hsm/hsmfs1
                                Online Limits                    Total Limits
        Type   ID    In Use        Soft       Hard     In Use     Soft  Hard
/hsm/hsmfs1
Files  group  119       763         750       1500        763      750  1500
Blocks group  119 3109330961 2013265920 3019898880  120096782  157286400
 235929600
Grace period                 4320                              4320
---> Warning:  soft limits to be enforced in 12h
[server1]root@solaris:~#
```

4. 다른 방법으로, 즉각적으로 유예 기간을 전체 시작 크기로 재설정하려면 유예 기간 타이머를 재설정합니다. *samquota -quota-type ID -x reset [directory-or-file]* 명령을 사용합니다.
   - *quota-type ID*는 *G* 더하기 그룹 이름 또는 ID 번호, *A* 더하기 관리 세트 ID 번호, *U* 더하기 사용자 이름 또는 ID 번호입니다.
   - *directory-or-file*(선택사항)은 유예 기간을 연장해야 하는 특정 파일 시스템의 마운트 지점 디렉토리이거나 특정 디렉토리 또는 파일입니다.

   계속하려면 프롬프트가 표시될 때 *y*(예)를 입력합니다.

   예제에서는 *hsmfs1* 파일 시스템에서 *cit* 그룹의 쿼터에 대한 유예 기간 타이머를 지웁니다.

```
[server1]root@solaris:~# samquota -G cit -x reset /hsm/hsmfs1
Setting Grace Timer:  continue? y
[server1]root@solaris:~#
```

cit 그룹 쿼터를 다시 검사하면 유예 기간이 *12h*, 즉 12시간(*4320*초)으로 재설정되었습니다.

```
[server1]root@solaris:~# samquota -G cit /hsm/hsmfs1
                                Online Limits                    Total Limits
        Type   ID    In Use        Soft       Hard     In Use     Soft  Hard
/hsm/hsmfs1
Files  group  119       762         750       1500        762      750  1500
Blocks group  119 3109330961 2013265920 3019898880  120096782  157286400
 235929600
Grace period                 4320                              4320
---> Warning:  soft limits to be enforced in 12h
```

```
[server1]root@solaris:~#
```

5. 여기서 중지합니다.

유예 기간 조기 종료

1. 파일 시스템 서버에 *root*로 로그인합니다.

   예제에서는 *server1* 호스트에 로그인합니다.

```
[server1]root@solaris:~#
```

2. 단축할 유예 기간을 확인합니다. *samquota -quota-type ID [directory-or-file]* 명령을 사용합니다. 설명:
   - *quota-type ID*는 *G* 더하기 그룹 이름 또는 ID 번호, *A* 더하기 관리 세트 ID 번호, *U* 더하기 사용자 이름 또는 ID 번호입니다.
   - *directory-or-file*(선택사항)은 유예 기간을 연장해야 하는 특정 파일 시스템의 마운트 지점 디렉토리이거나 특정 디렉토리 또는 파일입니다.

   예제에서는 *cit* 그룹이 소프트 한계를 초과하며 유예 기간이 11시간 남았지만, 유예 기간을 조기에 종료하려고 합니다.

```
[server1]root@solaris:~# samquota -G cit /hsm/hsmfs1
                                Online Limits                    Total Limits
        Type   ID    In Use        Soft       Hard     In Use       Soft
 Hard
/hsm/hsmfs1
Files  group  119       822         750       1500        822        750
 1500
Blocks group  119 3109330961 2013265920 3019898880  120096782  157286400
 235929600
Grace period                4320                              4320
---> Warning:  soft limits to be enforced in 11h
[server1]root@solaris:~#
```

3. 유예 기간을 만료합니다. *samquota -quota-type ID -x expire [directory-or-file]* 명령을 사용합니다. 설명:
   - *quota-type ID*는 *G* 더하기 그룹 이름 또는 ID 번호, *A* 더하기 관리 세트 ID 번호, *U* 더하기 사용자 이름 또는 ID 번호입니다.
   - *directory-or-file*(선택사항)은 유예 기간을 연장해야 하는 특정 파일 시스템의 마운트 지점 디렉토리이거나 특정 디렉토리 또는 파일입니다.

   예제에서는 *cit* 그룹에 대한 유예 기간을 만료합니다.

```
root@solaris:~# samquota -G cit -x expire /hsm/hsmfs1
Setting Grace Timer:  continue? y
```

쿼터를 다시 검사하면 *cit* 그룹의 소프트 한계가 하드 한계로 강제 적용되고 있습니다.

```
[server1]root@solaris:~# samquota -G cit /hsm/hsmfs1
                                Online Limits                    Total Limits
        Type   ID    In Use        Soft       Hard     In Use     Soft  Hard
/hsm/hsmfs1
Files  group  119       762         750       1500        762      750  1500
Blocks group  119 3109330961 2013265920 3019898880  120096782  157286400
 235929600
Grace period                 4320                             4320
---> Online soft limits under enforcement (since 6s ago)
[server1]root@solaris:~#
```

4. 여기서 중지합니다.

새 리소스 할당 중지

불일치 쿼터 값을 만들어 파일 시스템 리소스 할당을 금지할 수 있습니다. 파일 시스템이 사용자, 그룹, 관리 세트에 대한 쿼터 값 불일치를 감지하면 해당 사용자, 그룹, 관리 세트에서 더 이상 시스템 리소스를 사용하지 못하게 됩니다. 따라서 쿼터의 하드 한계를 해당하는 소프트 한계보다 낮게 설정하면 추가 할당이 중지됩니다. 이 기법을 사용하려면 다음과 같이 하십시오.

1. 파일 시스템 서버에 *root*로 로그인합니다.

   예제에서는 *server1* 호스트에 로그인합니다.

   ```
   [server1]root@solaris:~#
   ```

2. 나중에 복원할 수 있도록 쿼터를 백업합니다. 현재 구성을 내보내고 해당 정보를 파일로 재지정합니다. *samquota -quota-type ID [directory-or-file]* > *file* 명령을 사용합니다. 설명:
   - *quota-type ID*는 *G* 더하기 그룹 이름 또는 ID 번호, *A* 더하기 관리 세트 ID 번호, *U* 더하기 사용자 이름 또는 ID 번호입니다.
   - *directory-or-file*(선택사항)은 유예 기간을 연장해야 하는 특정 파일 시스템의 마운트 지점 디렉토리이거나 특정 디렉토리 또는 파일입니다.
   - *file*은 출력 파일의 이름입니다.

   예제에서는 *cit* 그룹의 쿼터를 *root* 사용자의 홈 디렉토리에 있는 *restore.hsmfs1 .quota_g.cit* 파일로 내보냅니다. 아래의 명령은 단일 라인으로 입력됩니다. 줄바꿈은 백슬래시 문자로 이스케이프됩니다.

```
[server1]root@solaris:~# samquota -G cit -e /hsm/hsmfs1 > /
/root/restore.hsmfs1.quota_g.cit
[server1]root@solaris:~#
```

3. 출력을 확인합니다. Solaris 명령 *more* < *file*을 사용합니다. 여기서 *file*은 출력 파일의 이름입니다.

```
[server1]root@solaris:~# more < /root/restore.hsmfs1.quota_g.cit
# Type  ID
#                  Online Limits                          Total   Limits
#              soft            hard                    soft       hard
# Files
# Blocks
# Grace Periods
samquota -G 119 /
    -f       750:s:o  -f      1500:h:o        -f       750:s:t -f      1500:h:t /
    -b 157286400:s:o  -b 235929600:h:o        -b 157286400:s:t -b 235929600:h:t /
            -t 4320:o                               -t 4320:t
[server1]root@solaris:~#
```

4. 쿼터의 하드 한계를 *0*(제로)으로 설정하고 소프트 한계를 *1*(또는 0이 아닌 아무 값)로 설정합니다. *samquota -quota-type ID -f 1:s -f 0:h -b 1:s -b 0:h [directory-or-file]* 명령을 사용합니다.
   - *quota-type ID*는 *G* 더하기 그룹 이름 또는 ID 번호, *A* 더하기 관리 세트 ID 번호, *U* 더하기 사용자 이름 또는 ID 번호입니다.
   - *directory-or-file*(선택사항)은 특정 파일 시스템의 마운트 지점 디렉토리이거나 유예 기간을 연장할 특정 디렉토리나 파일입니다.

   예제에서는 */hsm/hsmfs1* 파일 시스템의 *cit* 그룹에 대한 쿼터 설정을 불일치로 만듭니다. 그러면 새 리소스 할당이 중지됩니다.

```
[server1]root@solaris:~# samquota -G cit -f 1:s -f 0:h -b 1:s -b 0:h /hsm/hsmfs1
[server1]root@solaris:~#
```

   cit 그룹에 대한 쿼터를 검사하면 제로 쿼터가 발효됩니다. 느낌표 문자(**!**)는 모든 현재 사용량이 쿼터를 초과해서 더 이상 할당되지 않음을 보여줍니다.

```
[server1]root@solaris:~# samquota -G cit /hsm/hsmfs1
Online Limits                Total Limits
Type   ID              In Use    Soft     Hard         In Use     Soft     Hard
/sam6
Files  group 119        822!       1        0         822!       1        0
Blocks group 119 3109330961!       1        0  3109330961!       1        0
```

```
Grace period                        4320                              4320
---> Quota values inconsistent; zero quotas in effect.
[server1]root@solaris:~#
```

5. 수정된 쿼터를 원래 상태로 복원하여 정상적 할당을 재개할 준비가 되었으면 셸 스크립트로 만든 백업 파일을 실행합니다. Solaris 명령 *sh file*을 사용합니다. 여기서 *file*은 백업 파일의 이름입니다.

   예제에서는 */root/restore.hsmfs1.quota_g.cit* 파일을 실행하여 *cit* 그룹에 대한 쿼터를 복원합니다.

```
[server1]root@solaris:~# sh /root/restore.hsmfs1.quota_g.cit
Setting Grace Timer:  continue? y
Setting Grace Timer:  continue? y
[server1]root@solaris:~#
```

   쿼터를 검사하면 정상적 한계가 복원되었고 더 이상 할당이 차단되지 않습니다.

```
[server1]root@solaris:~# samquota -G cit /hsm/hsmfs1
                                Online Limits                    Total Limits
        Type    ID    In Use        Soft       Hard     In Use     Soft   Hard
/hsm/hsmfs1
Files  group  119       822         750       1500        822     750   1500
Blocks group  119 3109330961 2013265920 3019898880  120096782  157286400
 235929600
Grace period                        4320                              4320
---> Warning:  soft limits to be enforced in 11h
[server1]root@solaris:~#
```

6. 여기서 중지합니다.

파일 시스템에 대한 쿼터 제거

파일 시스템에 대한 쿼터를 제거하거나 사용 안함으로 설정하려면 마운트 프로세스에서 쿼터를 사용 안함으로 설정합니다.

1. 파일 시스템 서버에 *root*로 로그인합니다.

   예제에서는 *server1* 호스트에 로그인합니다.

```
[server1]root@solaris:~#
```

2. 텍스트 편집기에서 */etc/vfstab* 파일을 열고 파일 시스템 행의 Mount Options 열에 *noquota* 마운트 옵션을 추가하고 파일을 저장합니다.

예제에서는 *vi* 텍스트 편집기에서 파일을 열고 *hsmfs1* 파일 시스템에 대해 *noquota* 마운트 옵션을 설정합니다.

```
[server1]root@solaris:~# vi /etc/vfstab
#File
#Device        Device   Mount             System  fsck  Mount    Mount
#to Mount      to fsck  Point             Type    Pass  at Boot  Options
#------------  -------  ---------------   ------  ----  -------  ------------
/devices       -        /devices          devfs   -     no       -
/proc          -        /proc             proc    -     no       -
...
hsmfs1         -        /hsm/hsmfs1          samfs   -     no       noquota
:wq
[server1]root@solaris:~#
```

3. 파일 시스템이 마운트된 경우 마운트 해제합니다.

   파일 시스템을 마운트 해제한 후 다시 마운트해야 합니다. 그래야 운영체제가 */etc/vfstab* 파일을 다시 로드하고 지정된 변경을 수행합니다. 예제에서는 *hsmfs1* 파일 시스템을 마운트 해제합니다.

   ```
   [server1]root@solaris:~# umount hsmfs1
   [server1]root@solaris:~#
   ```

4. 파일 시스템을 마운트합니다.

   예제에서는 *hsmfs1* 파일 시스템을 마운트합니다.

   ```
   [server1]root@solaris:~# mount hsmfs1
   [server1]root@solaris:~#
   ```

5. 나중에 쿼터를 복구하려면 쿼터 파일을 제자리에 둡니다.

   쿼터를 복구할 준비가 되었으면 간단히 파일 시스템을 마운트 해제하고, 파일 시스템에서 *samfsck -F* 명령을 실행하고, *noquota* 마운트 옵션을 제거하고, 파일 시스템을 다시 마운트할 수 있습니다.
6. 쿼터를 복구하지 않으려는 경우나 쿼터 파일에서 소비한 공간을 회수해야 하는 경우 Solaris 명령 *rm*을 사용하여 파일 시스템의 루트 디렉토리에서 *.quota_g*, *.quota_a*, *.quota_u* 파일을 삭제합니다.

   예제에서는 */hsm/hsmfs1* 파일 시스템 루트 디렉토리에서 모든 쿼터 파일을 제거합니다.

   ```
   [server1]root@solaris:~# rm /hsm/hsmfs1/.quota_g
   [server1]root@solaris:~# rm /hsm/hsmfs1/.quota_a
   ```

```
[server1]root@solaris:~# rm /hsm/hsmfs1/.quota_u
[server1]root@solaris:~#
```

7. 여기서 중지합니다.

아카이빙 및 스테이징 작업 제어

일반적으로 아카이빙 파일 시스템은 비아카이빙 파일 시스템과 마찬가지 방법으로 관리합니다. 그러나 대부분의 파일 시스템 관리 작업을 수행하기 전에 아카이빙 프로세스를 중지해야 합니다. 활성일 때 아카이빙 프로세스는 파일 시스템의 주요 디스크 캐시를 변경합니다. 따라서 디스크 캐시에 유지 관리 작업을 수행하기 전에 이러한 프로세스를 중지해야 합니다. 이 절에서는 다음 작업을 다룹니다.

- 아카이빙 및 스테이징 프로세스 유휴 설정
- 아카이빙 및 스테이징 프로세스 중지
- 아카이빙 및 스테이징 프로세스 다시 시작.

아카이빙 및 스테이징 프로세스 유휴 설정

1. 파일 시스템 호스트에 *root*로 로그인합니다.

   예제에서는 *server1* 호스트에 로그인합니다.

   ```
   [server1]root@solaris:~#
   ```

2. 모든 아카이빙 프로세스를 유휴 설정합니다. *samcmd aridle* 명령을 사용합니다.

   이 명령은 현재 아카이빙 및 스테이징 작업이 완료되도록 허용하지만, 새로운 작업을 시작하지는 않습니다.

   ```
   [server1]root@solaris:~# samcmd aridle
   [server1]root@solaris:~#
   ```

3. 모든 스테이징 프로세스를 유휴 설정합니다. *samcmd stidle* 명령을 사용합니다.

   이 명령은 현재 아카이빙 및 스테이징 작업이 완료되도록 허용하지만, 새로운 작업을 시작하지는 않습니다.

   ```
   [server1]root@solaris:~# samcmd stidle
   [server1]root@solaris:~#
   ```

4. 활성 아카이빙 작업이 완료될 때까지 기다립니다. *samcmd a* 명령을 사용하여 아카이빙 프로세스의 상태를 확인합니다.

   아카이빙 프로세스가 *Waiting for :arrun*이면 아카이빙 프로세스가 유휴 상태입니다.

```
[server1]root@solaris:~# samcmd a
Archiver status samcmd     5.4 10:20:34 May 20 2014
samcmd on samfs-mds
sam-archiverd:  Waiting for :arrun
sam-arfind: ...
Waiting for :arrun
```

5. 활성 스테이징 작업이 완료될 때까지 기다립니다. *samcmd u* 명령을 사용하여 스테이징 프로세스의 상태를 확인합니다.

   스테이징 프로세스가 *Waiting for :strun*이면 스테이징 프로세스가 유휴 상태임을 나타냅니다.

```
[server1]root@solaris:~# samcmd u
Staging queue samcmd      5.4  10:20:34 May 20 2014
samcmd on solaris.demo.lan
Staging queue by media type: all
sam-stagerd:  Waiting for :strun
root@solaris:~#
```

6. 시스템을 완전히 중지하려면 아카이빙 및 스테이징 프로세스 중지도 수행합니다.

아카이빙 및 스테이징 프로세스 중지

1. 아직 완료하지 않았으면 아카이빙 및 스테이징 프로세스 유휴 설정을 수행합니다.
2. 아직 하지 않았으면 파일 시스템 호스트에 *root*로 로그인합니다.

   예제에서는 *server1* 호스트에 로그인합니다.

```
[server1]root@solaris:~#
```

3. 더 진행하기 전에 모든 이동식 매체 드라이브를 유휴 설정합니다. 각 드라이브에 대해 *samcmd equipment-number idle* 명령을 사용합니다. 여기서 *equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 드라이브에 지정된 장비 순서 번호입니다.

   이 명령은 드라이브를 *off*로 설정하기 전에 현재 아카이빙 및 스테이징 작업이 완료되도록 허용하지만, 새로운 작업을 시작하지는 않습니다. 예제에서는 순서 번호 *801*, *802*, *803*, *804*를 가진 4개 드라이브를 유휴 설정합니다.

```
[server1]root@solaris:~# samcmd 801 idle
[server1]root@solaris:~# samcmd 802 idle
[server1]root@solaris:~# samcmd 803 idle
[server1]root@solaris:~# samcmd 804 idle
[server1]root@solaris:~#
```

4. 실행 중인 작업이 완료될 때까지 기다립니다.

   samcmd r 명령을 사용하여 드라이브의 상태를 확인할 수 있습니다. 모든 드라이브가 *notrdy* 및 *empty*이면 진행할 준비가 된 것입니다.

```
[server1]root@solaris:~# samcmd r
Removable media samcmd     5.4 18:37:09 Feb 17 2014
samcmd on hsmfs1host
ty   eq   status      act  use  state  vsn
li  801   ---------p    0   0%  notrdy
          empty
li  802   ---------p    0   0%  notrdy
          empty
li  803   ---------p    0   0%  notrdy
          empty
li  804   ---------p    0   0%  notrdy
          empty
[server1]root@solaris:~#
```

5. 아카이버 및 스테이저 프로세스가 유휴 상태이고 테이프 드라이버가 모두 *notrdy*이면 라이브러리 제어 데몬을 중지합니다. *samd stop* 명령을 사용합니다.

```
[server1]root@solaris:~# samd stop
[server1]root@solaris:~#
```

6. 파일 시스템 유지 관리를 계속 수행합니다.
7. 유지 관리가 완료되면 아카이빙 및 스테이징 프로세스 다시 시작을 수행합니다.

   작업을 다시 시작하면 보류 중인 스테이지가 재실행되고 아카이빙이 재개됩니다.
8. 여기서 중지합니다.

아카이빙 및 스테이징 프로세스 다시 시작

준비가 되었으면 정상적 자동 작업을 재개하고 다음과 같이 하십시오.

1. 파일 시스템 호스트에 *root*로 로그인합니다.

   예제에서는 *server1* 호스트에 로그인합니다.

```
[server1]root@solaris:~#
```

2. Oracle HSM 라이브러리 제어 데몬을 다시 시작합니다. *samd start* 명령을 사용합니다.

```
[server1]root@solaris:~# samd start
[server1]root@solaris:~#
```

3. 여기서 중지합니다.

파일 시스템 이름 바꾸기

파일 시스템 이름 바꾸기는 2단계 프로세스입니다. 먼저 */etc/opt/SUNWsamfs/mcf* 파일을 편집하여 파일 시스템의 패밀리 세트 이름을 변경합니다. 그런 다음 *samfsck -R -F* 명령으로 새 이름을 읽고 해당 디스크 장치의 수퍼 블록을 업데이트합니다. 파일 시스템 이름을 바꾸려면 아래 절차를 사용하십시오.

파일 시스템 이름 바꾸기

1. 파일 시스템 서버에 *root*로 로그인합니다.

   예제에서는 *server1* 호스트에 로그인합니다.

   ```
   [server1]root@solaris:~#
   ```

2. 아카이빙 파일 시스템을 복구하려면 더 진행하기 전에 "아카이빙 및 스테이징 프로세스 유휴 설정" 절차를 수행합니다.
3. 이름을 바꾸려는 파일 시스템을 마운트 해제합니다.

   예제에서는 *hsmfs1* 파일 시스템을 마운트 해제합니다.

   ```
   [server1]root@solaris:~# umount hsmfs1
   ```

4. 텍스트 편집기에서 */etc/opt/SUNWsamfs/mcf* 파일을 열고 이름을 바꾸려는 파일 시스템을 찾습니다.

   예제에서는 *vi* 편집기를 사용합니다. *hsmfs1* 파일 시스템의 이름을 변경해야 합니다.

   ```
   [server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
   # Equipment          Equipment  Equipment  Family        Device   Additional
   # Identifier         Ordinal    Type       Set           State    Parameters
   #-----------------   ---------  ---------  ------------  ------   ----------
   hsmfs1              100        ms         hsmfs1        on
    /dev/dsk/c1t3d0s3    101        md         hsmfs1        on
    /dev/dsk/c1t4d0s5    102        md         hsmfs1        on
   ```

5. 파일의 네번째 열에서 파일 시스템의 패밀리 세트 이름을 새 값으로 변경합니다. 첫번째 열에서 파일 시스템 장비 식별자를 변경해야 할 수 있지만 그 외 항목은 변경하지 마십시오. 파일을 저장하고 편집기를 닫습니다.

   예제에서는 파일 시스템의 장비 식별자 및 패밀리 세트 이름을 모두 *hsmfs1*에서 *samqfs-hpcc*로 변경합니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment          Equipment  Equipment  Family        Device   Additional
# Identifier         Ordinal    Type       Set           State    Parameters
#-----------------   ---------  ---------  ------------  ------   ----------
samqfs-hpcc          100        ms         samqfs-hpcc   on
 /dev/dsk/c1t3d0s3    101        md          samqfs-hpcc   on
 /dev/dsk/c1t4d0s5    102        md          samqfs-hpcc   on
:wq
root@solaris:~#
```

6. 새 패밀리 세트 이름이 반영되도록 파일 시스템 수퍼 블록을 재작성합니다. *samfsck -R -F family-set-name* 명령을 사용합니다. 여기서 *family-set-name*은 */etc/opt/SUNWsamfs/mcf* 파일에 지정한 패밀리 세트 이름입니다.

   -R 및 *-F* 옵션과 함께 실행할 때 *samfsck* 명령은 */etc/opt/SUNWsamfs/mcf* 파일에서 새 패밀리 세트 이름과 해당하는 디스크 스토리지 장비 식별자를 읽습니다. 그런 다음 지정된 디스크 장치의 수퍼 블록을 새 패밀리 세트 이름으로 재작성합니다. 예제에서는 새 패밀리 세트 이름 *samqfs-hpcc*로 명령을 실행합니다.

```
[server1]root@solaris:~# samfsck -R -F samqfs-hpcc
```

7. 텍스트 편집기에서 */etc/vfstab* 파일을 열고 이름을 바꾸려는 파일 시스템 항목을 찾습니다.

   예제에서는 *vi* 텍스트 편집기에서 파일을 엽니다. *hsmfs1* 파일 시스템 항목이 새 이름을 사용하도록 변경해야 합니다.

```
[server1]root@solaris:~# vi /etc/vfstab
#File
#Device        Device   Mount             System  fsck  Mount    Mount
#to Mount      to fsck  Point             Type    Pass  at Boot  Options
#------------  -------  ---------------   ------  ----  -------  ------------
/devices       -        /devices          devfs   -     no       -
/proc          -        /proc             proc    -     no       -
...
hsmfs1        -        /hsm/hsmfs1          samfs   -     no       -
```

8. 이름을 바꾼 파일 시스템에 대한 */etc/vfstab* 항목의 첫번째 열에서 파일 시스템 이름을 변경하고 세번째 열(필요한 경우)에서 마운트 지점 디렉토리 이름을 변경하고, 파일을 저장합니다.

   예제에서는 *hsmfs1* 파일 시스템의 이름을 *samqfs-hpcc*로 변경하고 마운트 지점이 일치하도록 변경합니다.

```
[server1]root@solaris:~# vi /etc/vfstab
#File
#Device        Device   Mount             System  fsck  Mount    Mount
#to Mount      to fsck  Point             Type    Pass  at Boot  Options
#------------  -------  ---------------   ------  ----  -------  ------------
/devices       -        /devices          devfs   -     no       -
/proc          -        /proc             proc    -     no       -
...
samqfs-hpcc    -        /samqfs-hpcc      samfs   -     no       -
:wq
[server1]root@solaris:~#
```

9. 필요한 경우 새 파일 시스템에 대한 새 마운트 지점 디렉토리를 만들고 마운트 지점에 대한 액세스 권한을 설정합니다.

   사용자가 마운트 지점 디렉토리를 변경하고 마운트된 파일 시스템의 파일에 액세스하려면 실행(*x*) 권한이 있어야 합니다. 예제에서는 */samqfs-hpcc* 마운트 지점 디렉토리를 만들고 권한을 *755*(*-rwxr-xr-x*)로 설정합니다.

   ```
   [server1]root@solaris:~# mkdir /samqfs-hpcc
   [server1]root@solaris:~# chmod 755 /samqfs-hpcc
   [server1]root@solaris:~#
   ```

10. *sam-fsd* 명령을 실행하여 *mcf* 파일에 오류가 있는지 확인하고, 오류가 감지되면 수정합니다.

    *sam-fsd*는 Oracle HSM 구성 파일을 읽는 초기화 명령입니다. 오류가 발견되면 실행을 중지합니다.

    ```
    [server1]root@solaris:~# sam-fsd
    ```

11. Oracle HSM 소프트웨어에 *mcf* 파일을 다시 읽고 그에 따라 재구성하도록 지시합니다. *samd config* 명령을 사용합니다.

    ```
    [server1]root@solaris:~# samd config
    ```

12. *samd config*에서 오류가 보고되면 이를 수정하고 오류가 발견되지 않을 때까지 명령을 재실행합니다.
13. 파일 시스템을 마운트합니다.

    예제에서는 새 마운트 지점 디렉토리를 사용합니다.

    ```
    [server1]root@solaris:~# mount /samqfs-hpcc
    ```

14. 여기서 중지합니다.

파일 시스템 복구

파일 시스템이 *samu*, Oracle HSM Manager 또는 */var/adm/sam-log* 파일을 통해 오류를 보고하면 아래 절차를 따르십시오.

파일 시스템 복구

1. 파일 시스템 서버에 *root*로 로그인합니다.

   예제에서는 *server1* 호스트에 로그인합니다.

   ```
   [server1]root@solaris:~#
   ```

2. 아카이빙 파일 시스템을 복구하려면 더 진행하기 전에 "아카이빙 및 스테이징 프로세스 유휴 설정" 절차를 수행합니다.
3. 영향을 받는 파일 시스템을 마운트 해제합니다.

   아카이빙이 중지되기를 기다리는 경우 여러 번 시도해야 할 수도 있습니다. 예제에서는 *hsmfs1* 파일 시스템을 마운트 해제합니다.

   ```
   [server1]root@solaris:~# umount hsmfs1
   samfs umount: /hsm/hsmfs1: is busy
   [server1]root@solaris:~# umount hsmfs1
   [server1]root@solaris:~#
   ```

4. 파일 시스템을 복구합니다. *samfsck -F -V family-set-name* 명령을 사용합니다. 여기서 *family-set-name*은 */etc/opt/SUNWsamfs/mcf* 파일의 파일 시스템에 지정된 패밀리 세트 이름입니다.

   필요에 따라 나중에 참조하고 진단 목적으로 사용할 수 있도록 복구 결과를 날짜 기록 파일에 저장하는 것이 좋을 수 있습니다. 따라서 이 예제에서는 *tee /var/tmp/samfsck-FV.family-set-name.`date '+%Y%m%d.%H%M%S'`* 명령에 *samfsck* 출력을 파이핑하여 결과를 저장합니다. 아래의 명령은 단일 라인으로 입력됩니다. 줄바꿈은 백슬래시 문자로 이스케이프됩니다.

   ```
   [server1]root@solaris:~# samfsck -F -V hsmfs1 | tee /
   /var/tmp/samfsck-FV.hsmfs1. `date '+%Y%m%d.%H%M%S'`
   name:     /hsm/hsmfs1       version:     2A
   First pass
   Second pass
   Third pass
   NOTICE: ino 2.2,  Repaired link count from 8 to 14
   Inodes processed: 123392
   total data kilobytes       = 1965952
   ```

```
total data kilobytes free  = 1047680
total meta kilobytes       = 131040
total meta kilobytes free  = 65568
INFO:  FS samma1 repaired:
        start:  May 19, 2014 10:57:13 AM MDT
        finish: May 19, 2014 10:57:37 AM MDT
NOTICE: Reclaimed 70057984 bytes
NOTICE: Reclaimed 9519104 meta bytes
[server1]root@solaris:~#
```

5. 파일 시스템을 다시 마운트합니다.

```
[server1]root@solaris:~# mount /hsm/hsmfs1
[server1]root@solaris:~#
```

6. 여기서 중지합니다.

파일 시스템에 장치 추가

기존 파일 시스템에 장치를 추가하기 전에 요구 사항과 대안을 고려해야 합니다. 기존 파일 시스템을 확대하는 것이 증가하는 용량 요구 사항을 충족하는 최선의 방법인지 확인합니다. 새 프로젝트나 사용자 커뮤니티를 수용할 물리적 스토리지 공간이 더 필요한 경우 하나 이상의 새로운 Oracle HSM 파일 시스템을 만드는 것이 더 좋은 선택일 수도 있습니다. 여러 개의 소형 파일 시스템이 일반적으로 하나의 대형 파일 시스템보다 훨씬 좋은 성능을 제공하며, 소형 파일 시스템을 만들고 유지 관리하기가 더 쉽습니다.

일단 파일 시스템을 확대하기로 결정했으면 다음 접근 방법 중 하나를 선택합니다.

- 마운트된 파일 시스템에 장치 추가(권장)
- 마운트 해제된 파일 시스템에 장치 추가

마운트된 파일 시스템에 장치 추가

다음과 같이 하십시오.

1. 파일 시스템 서버에 *root*로 로그인합니다.

   예제에서는 *server1* 호스트에 로그인합니다.

```
[server1]root@solaris:~#
```

2. 텍스트 편집기에서 */etc/opt/SUNWsamfs/mcf* 파일을 열고 확대하려는 파일 시스템을 찾습니다.

   예제에서는 *vi* 편집기를 사용합니다. 두 개의 파일 시스템인 범용 *samqfsms* 파일 시스템과 고성능 *samqfs2ma* 파일 시스템을 확장해야 합니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment         Equipment  Equipment  Family     Device   Additional
# Identifier        Ordinal    Type       Set        State    Parameters
#-----------------  ---------  ---------  ---------  ------   ---------------
samqfsms            100        ms         samqfsms   on
 /dev/dsk/c1t3d0s3   101        md         samqfsms   on
 /dev/dsk/c1t4d0s5   102        md         samqfsms   on
samqfs2ma           200        ma         samqfs2ma  on
 /dev/dsk/c1t3d0s3   201        mm         samqfs2ma  on
 /dev/dsk/c1t3d0s5   202        md         samqfs2ma  on
 /dev/dsk/c1t4d0s5   203        md         samqfs2ma  on
```

3. 범용 *ms* 파일 시스템에 장치를 추가하는 경우 *mcf* 파일에서 파일 시스템 정의 끝에 추가 데이터/메타데이터 장치를 추가합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   최대 252개의 논리 장치를 추가할 수 있습니다. 예제에서는 *samqfsms* 파일 시스템에 두 개의 장치 *103* 및 *104*를 추가합니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment         Equipment  Equipment  Family     Device   Additional
# Identifier        Ordinal    Type       Set        State    Parameters
#-----------------  ---------  ---------  ---------  ------   ---------------
samqfsms            100        ms         samqfsms    on
 /dev/dsk/c1t3d0s3   101        md         samqfsms    on
 /dev/dsk/c1t4d0s5   102        md         samqfsms    on
 /dev/dsk/c1t3d0s7   103        md         samqfsms    on
 /dev/dsk/c1t4d0s7   104        md         samqfsms    on
:wq
[server1]root@solaris:~#
```

4. 고성능 *ma* 파일 시스템에 장치를 추가하는 경우 *mcf* 파일에서 파일 시스템 정의 끝에 데이터 장치와 하나 이상의 *mm* 디스크 장치를 추가합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   항상 기존 장치 목록의 끝에 새 장치를 추가하십시오. 데이터 장치를 추가하는 것에 비례해서 최대 252개의 메타데이터 장치를 추가할 수 있습니다. 예제에서는 *samqfs2ma* 파일 시스템에 하나의 *mm* 메타데이터 장치 *204*와 두 개의 *md* 데이터 장치 *205* 및 *206*을 추가합니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment         Equipment  Equipment  Family     Device   Additional
# Identifier        Ordinal    Type       Set        State    Parameters
#-----------------  ---------  ---------  ---------  ------   ---------------
...
```

```
samqfs2ma           200        ma         samqfs2ma  on
 /dev/dsk/c1t3d0s3   201        mm         samqfs2ma  on
 /dev/dsk/c1t3d0s5   202        md         samqfs2ma  on
 /dev/dsk/c1t4d0s5   203        md         samqfs2ma  on
 /dev/dsk/c1t5d0s6   204        mm         samqfs2ma  on
 /dev/dsk/c1t3d0s7   205        md         samqfs2ma  on
 /dev/dsk/c1t4d0s7   206        md         samqfs2ma  on
:wq
[server1]root@solaris:~#
```

5. *sam-fsd* 명령을 실행하여 *mcf* 파일에 오류가 있는지 확인하고, 오류가 감지되면 수정합니다.

   *sam-fsd*는 Oracle HSM 구성 파일을 읽는 초기화 명령입니다. 오류가 발견되면 실행을 중지합니다.

   ```
   [server1]root@solaris:~# sam-fsd
   ```

6. *sam-fsd* 명령이 *mcf* 파일에서 오류를 찾을 경우 파일을 편집하여 오류를 해결하고 이전 단계에 설명된 대로 다시 검사합니다.

   아래의 예에서는 *sam-fsd*가 장치에서 지정되지 않은 문제를 보고합니다.

   ```
   [server1]root@solaris:~# sam-fsd
   Problem in mcf file /etc/opt/SUNWsamfs/mcf for filesystem samqfsms
   sam-fsd: Problem with file system devices.
   ```

   대개 이러한 오류는 부주의한 타이핑 실수의 결과입니다. 여기서 *mcf* 파일을 편집기에서 열면 두번째 신규 *md* 장치인 *104* 장치의 장비 이름에 0 대신 문자 *o*를 입력했음을 알 수 있습니다.

   ```
   samqfsms                   100        ms         samqfsms    on
    /dev/dsk/c1t3d0s3         101        md         samqfsms    on
    /dev/dsk/c1t4d0s5         102        md         samqfsms    on
    /dev/dsk/c1t3d0s7         103        md         samqfsms    on
    /dev/dsk/c1t4dos7         104        md         samqfsms    on
                   ^
   ```

7. *sam-fsd* 명령이 오류 없이 실행되면 *mcf* 파일이 올바른 것입니다. 다음 단계로 진행하십시오.

   이 예는 오류가 없는 출력의 일부입니다.

   ```
   [server1]root@solaris:~# sam-fsd
   Trace file controls:
   sam-amld      /var/opt/SUNWsamfs/trace/sam-amld
   ```

```
                cust err fatal ipc misc proc date
...
Would start sam-archiverd()
Would start sam-stagealld()
Would start sam-stagerd()
Would start sam-amld()
[server1]root@solaris:~#
```

8. Oracle HSM 소프트웨어에 *mcf* 파일을 다시 읽고 그에 따라 재구성하도록 지시합니다. *samd config* 명령을 사용합니다.

```
[server1]root@solaris:~# samd config
Configuring SAM-FS
[server1]root@solaris:~#
```

9. *samd config*로 Oracle HSM 파일 시스템 구성이 새 장치를 포함하도록 업데이트되었는지 확인합니다. *samcmd f* 명령을 사용합니다.

   장치는 *off* 상태여야 합니다. 예제에서 *samcmd f*는 새 장치 *103* 및 *104*를 보여주며 둘 다 *off* 상태입니다.

```
[server1]root@solaris:~# samcmd f
File systems samcmd     5.4 16:57:35 Feb 27 2014
samcmd on server1
ty      eq   state device_name       status      high low mountpoint server
ms     100   on    samqfsms          m----2-----  80% 70% /samqfsms
 md     101   on   /dev/dsk/c1t3d0s3
 md     102   on   /dev/dsk/c1t4d0s5
 md     103   off  /dev/dsk/c1t3d0s7
 md     104   off  /dev/dsk/c1t4d0s7
[server1]root@solaris:~#
```

10. 새로 추가된 장치를 사용으로 설정합니다. 각 장치에 대해 *samcmd add equipment-number* 명령을 사용합니다. 여기서 *equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 장치에 지정된 장비 순서 번호입니다.

    예제에서는 새 장치 *103* 및 *104*를 사용으로 설정합니다.

```
[server1]root@solaris:~# samcmd add 103
[server1]root@solaris:~# samcmd add 104
```

11. 공유 파일 시스템에 장치를 추가하는 경우 "공유 파일 시스템에 추가된 새 장치 구성 마치기"로 이동합니다.
12. 비공유 독립형 파일 시스템에 장치를 추가하는 경우 장치가 추가되었고 파일 시스템에서 사용할 준비가 되었는지 확인합니다. *samcmd m* 명령을 사용하고 결과를 확인합니다.

장치가 *on* 상태일 때 성공적으로 추가되었고 사용할 준비가 되었습니다. 예제에서는 *103* 및 *104* 장치를 성공적으로 추가했습니다.

```
[server1]root@solaris:~# samcmd f
Mass storage status samcmd     5.4 17:17:08 Feb 27 2014
samcmd on server1
ty   eq  status       use state  ord  capacity      free  ra  part high low
ms   100  m----2----- 13% on           3.840G    3.588G  1M    16  80% 70%
 md   101             31% on       0  959.938M  834.250M
 md   102             13% on       1  959.938M  834.250M
 md   103              0% on       2  959.938M  959.938M
 md   104              0% on       3  959.938M  959.938M
[server1]root@solaris:~#
```

13. 여기서 중지합니다.

공유 파일 시스템에 추가된 새 장치 구성 마치기

공유 파일 시스템에 장치를 추가할 때 모든 파일 시스템 호스트에 장치가 구성되기 전에 몇 가지 추가 단계를 수행해야 합니다. 다음과 같이 하십시오.

1. 파일 시스템 메타데이터 서버 호스트에 *root*로 로그인합니다.

   예제에서 메타데이터 서버 호스트는 이름이 *metadata-server*로 지정됩니다.

   ```
   [metadata-server]root@solaris:~#
   ```

2. 새 장치가 메타데이터 서버에 추가되었는지 확인합니다. *samcmd m* 명령을 사용합니다.

   장치가 *unavail* 상태일 때 성공적으로 추가되었지만 아직 사용할 준비가 되지 않았습니다. 예제에서는 *103* 및 *104* 장치를 성공적으로 추가했습니다.

   ```
   [metadata-server]root@solaris:~# samcmd f
   Mass storage status samcmd     5.4 17:17:08 Feb 27 2014
   samcmd on metadata-server
   ty      eq  status       use state  ord   capacity     free    ra  part high low
   ms     100  m----2-----  13% on           3.840G    3.588G    1M    16  80% 70%
    md     101              31% on       0  959.938M  834.250M
    md     102              13% on       1  959.938M  834.250M
    md     103               0% unavail  2  959.938M  959.938M
    md     104               0% unavail  3  959.938M  959.938M
   [metadata-server]root@solaris:~#
   ```

3. 각 파일 시스템 클라이언트 호스트에 *root*로 로그인합니다.

잠재적 메타데이터 서버를 넣는 것을 잊지 마십시오. 이들도 클라이언트입니다. 예제에서는 이름이 *potential-metadata-server*인 잠재적인 메타데이터 서버와 두 개의 클라이언트 *client1* 및 *client2Linux*에 로그인해야 합니다. 따라서 3개의 터미널 창을 열고 보안 셸(*ssh*)을 사용합니다.

```
[metadata-server]root@solaris:~# ssh root@potential-metadata-server
Password:
[potential-metadata-server]root@solaris:~#
[metadata-server]root@solaris:~# ssh root@client1
Password:
[client1]root@solaris:~#
[metadata-server]root@solaris:~# ssh root@client2Linux
Password:
[client2Linux]:[root@linux ~]#
```

4. Linux 클라이언트의 경우 공유 파일 시스템을 마운트 해제합니다.

```
[client2Linux]:[root@linux ~]# umount /samqfsms
```

5. 각 클라이언트마다 텍스트 편집기에서 */etc/opt/SUNWsamfs/mcf* 파일을 열고, 서버에 했던 것처럼 파일 시스템 정의 끝에 새 장치를 추가합니다.

   예제에서는 *client1*의 *mcf* 파일에 *103* 및 *104* 장치를 추가합니다.

```
[client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment          Equipment  Equipment  Family     Device   Additional
# Identifier         Ordinal    Type       Set        State    Parameters
#-----------------   ---------  ---------  ---------  ------   ----------
samqfsms             100        ms         samqfsms    on      shared
 /dev/dsk/c1t3d0s3    101        md         samqfsms    on
 /dev/dsk/c1t4d0s5    102        md         samqfsms    on
 /dev/dsk/c1t3d0s7    103        md         samqfsms    on
 /dev/dsk/c1t4d0s7    104        md         samqfsms    on
:wq
[metadata-server]root@solaris:~#
```

6. 각 클라이언트마다 *sam-fsd* 명령을 실행하여 *mcf* 파일에 오류가 있는지 확인하고, 오류가 감지되면 수정합니다.

```
[metadata-server]root@solaris:~# sam-fsd
```

7. 각 클라이언트에서 Oracle HSM 소프트웨어에 *mcf* 파일을 다시 읽고 그에 따라 재구성하도록 지시합니다.

```
[metadata-server]root@solaris:~# samd config
```

8. Linux 클라이언트의 경우 공유 파일 시스템을 마운트합니다.

```
[client2Linux]:[root@linux ~]# mount /samqfsms
```

9. 모든 클라이언트가 구성되었으면 메타데이터 서버로 돌아가서 새 장치에 스토리지 할당을 사용으로 설정합니다. 각 장치에 대해 *samcmd alloc equipment-number* 명령을 사용합니다. 여기서 *equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 장치에 지정된 장비 순서 번호입니다.

   예제에서는 *103* 및 *104* 장치에 스토리지 할당을 사용으로 설정합니다.

```
[metadata-server]root@solaris:~# samcmd alloc 103
[metadata-server]root@solaris:~# samcmd alloc 104
```

10. 마지막으로, 파일 시스템에서 장치를 사용할 준비가 되었는지 확인합니다. *samcmd m* 명령을 사용하고 결과를 확인합니다.

    장치가 *on* 상태일 때 성공적으로 추가되었고 사용할 준비가 되었습니다. 예제에서는 *103* 및 *104* 장치를 성공적으로 추가했습니다.

```
[metadata-server]root@solaris:~# samcmd f
Mass storage status samcmd     5.4 17:17:08 Feb 27 2014
samcmd on metadata-server
ty      eq  status       use state  ord   capacity     free    ra  part high low
ms     100  m----2-----  13% on            3.840G    3.588G    1M    16  80% 70%
 md     101               31% on       0  959.938M  834.250M
 md     102               13% on       1  959.938M  834.250M
 md     103                0% on       2  959.938M  959.938M
 md     104                0% on       3  959.938M  959.938M
[metadata-server]root@solaris:~#
```

11. 여기서 중지합니다.

마운트 해제된 파일 시스템에 장치 추가

다음과 같이 하십시오.

1. 파일 시스템 서버 호스트에 *root*로 로그인합니다.

   예제에서 메타데이터 서버 호스트는 이름이 *server1*로 지정됩니다.

```
[server1]root@solaris:~#
```

2. 아카이빙 파일 시스템을 마운트 해제하기 전에 "아카이빙 및 스테이징 프로세스 유휴 설정" 절차를 수행해야 합니다.
3. 파일 시스템을 마운트 해제합니다.

   파일 시스템을 마운트 해제할 때까지는 작업을 진행하지 마십시오. 예제에서는 *hsmfs1* 파일 시스템을 마운트 해제합니다.

```
[server1]root@solaris:~# umount hsmfs1
```

4. 텍스트 편집기에서 */etc/opt/SUNWsamfs/mcf* 파일을 열고 확대하려는 파일 시스템을 찾습니다.

   예제에서는 *vi* 편집기를 사용합니다. *hsmfs1* 파일 시스템을 확대해야 합니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment           Equipment  Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#------------------   ---------  ---------  ---------  ------   ---------
hsmfs1               100        ms         hsmfs1     on
 /dev/dsk/c1t3d0s3     101        md         hsmfs1     on
 /dev/dsk/c1t4d0s5     102        md         hsmfs1     on
```

5. 고성능 *ma* 파일 시스템에 장치를 추가하는 경우 데이터 스토리지와 함께 메타데이터 스토리지를 추가해야 합니다. 추가할 데이터 장치의 메타데이터를 저장하기에 충분한 추가 *mm* 디스크 장치를 추가합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   최대 252개의 논리 장치를 추가할 수 있습니다. 예제에서는 *samqfs2ma* 파일 시스템에 하나의 *mm* 메타데이터 장치를 추가하고 *samqfs2ma* 파일 시스템에 두 개의 데이터 장치를 추가합니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment           Equipment  Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#------------------   ---------  ---------  ---------  ------   ---------
samqfs2ma            200        ma         samqfs2ma  on
 /dev/dsk/c1t3d0s3     201        mm         samqfs2ma  on
 /dev/dsk/c1t5d0s6     204        mm         samqfs2ma  on
 /dev/dsk/c1t3d0s5     202        md         samqfs2ma  on
 /dev/dsk/c1t4d0s5     203        md         samqfs2ma  on
 /dev/dsk/c1t3d0s7     205        md         samqfs2ma  on
 /dev/dsk/c1t4dos7     206        md         samqfs2ma  on
:wq
[server1]root@solaris:~#
```

6. 범용 *ms* 파일 시스템에 장치를 추가하는 경우 *mcf* 파일에서 파일 시스템 정의에 데이터/메타데이터 장치를 추가합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   최대 252개의 논리 장치를 추가할 수 있습니다. 예제에서는 *hsmfs1* 파일 시스템에 두 개의 장치를 추가합니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment           Equipment  Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#------------------   ---------  ---------  ---------  ------   ---------
hsmfs1               100        ms         hsmfs1    on
 /dev/dsk/c1t3d0s3     101        md         hsmfs1    on
 /dev/dsk/c1t4d0s5     102        md         hsmfs1    on
 /dev/dsk/c1t3d0s7     103        md         hsmfs1    on
 /dev/dsk/c1t4dos7     104        md         hsmfs1    on
:wq
[server1]root@solaris:~#
```

7. *sam-fsd* 명령을 실행하여 *mcf* 파일에 오류가 있는지 확인하고, 오류가 감지되면 수정합니다.

   *sam-fsd*는 Oracle HSM 구성 파일을 읽는 초기화 명령입니다. 오류가 발견되면 실행을 중지합니다.

```
[server1]root@solaris:~# sam-fsd
```

8. Oracle HSM 소프트웨어에 *mcf* 파일을 다시 읽고 그에 따라 재구성하도록 지시합니다.

```
root@solaris:~# samd config
```

9. 새 장치를 파일 시스템으로 통합합니다. *samgrowfs family-set-name* 명령을 사용합니다. 여기서 *family-set-name*은 */etc/opt/SUNWsamfs/mcf* 파일에서 파일 시스템에 대해 지정된 패밀리 세트 이름입니다.

   예제에서는 *hsmfs1* 파일 시스템을 확대합니다.

```
[server1]root@solaris:~# samgrowfs hsmfs1
```

10. 파일 시스템을 다시 마운트합니다.

```
[server1]root@solaris:~# mount /hsm/hsmfs1
```

11. 아카이빙 파일 시스템에 장치를 추가했으면 Oracle HSM 라이브러리 관리 데몬을 다시 시작합니다. *samd start* 명령을 사용합니다.

```
[server1]root@solaris:~# samd start
```

12. 파일 시스템을 변경하기 전에 마운트 해제하는 것을 잊어버린 결과, 파일 시스템이 마운트하지 않을 경우 추가된 장치에 대한 참조를 삭제하여 원래 *mcf* 파일을 복원합니다. 그런 다음 *samd config*를 실행하여 구성을 복원하고 파일 시스템을 마운트 해제하고 다시 시작합니다.
13. 여기서 중지합니다.

파일 시스템에서 데이터 장치 제거

필요한 경우 마운트된 Oracle HSM 파일 시스템에서 데이터 장치를 제거할 수 있습니다. 일반적으로 고장난 부품을 교체하거나 활용되지 않는 장치를 다른 용도로 전환할 때 필요하게 됩니다. 그러나 몇 가지 제한 사항이 있습니다.

데이터 장치만 제거할 수 있습니다. 메타데이터는 파일 시스템 자체의 구조를 정의하기 때문에 메타데이터를 보유한 장치는 제거할 수 없습니다. 이는 고성능 *ma* 파일 시스템에서 *md*, *mr* 및 스트라이프 그룹 장치만 제거할 수 있음을 의미합니다. *ma* 파일 시스템에서 *mm* 메타데이터 장치는 제거할 수 없습니다. 또한 이러한 장치는 데이터와 메타데이터를 모두 저장하므로 범용 *ms* 파일 시스템에서 *md* 장치를 제거할 수도 없습니다.

장치를 제거하려면 대상 장치에 상주하는 유효한 데이터 파일을 이동할 장소도 있어야 합니다. 이는 모든 장치를 제거할 수 없음을 의미합니다. 한 장치는 항상 파일 시스템에서 사용 가능해야 하며, 제거할 장치에 상주하는 모든 파일을 보유할 충분한 여유 용량이 있어야 합니다. 따라서 스트라이프 그룹을 제거해야 하는 경우 동일한 개수의 멤버 장치로 구성된 또 하나의 스트라이프 그룹이 사용 가능해야 합니다.

장치를 제거하려면 다음과 같이 하십시오.

- 파일 시스템 메타데이터 및 데이터가 백업되었는지 확인
- 마운트된 고성능 파일 시스템에서 장치 제거.

파일 시스템 메타데이터 및 데이터가 백업되었는지 확인

다음 작업을 수행합니다.

- **samexplorer** 실행
- 파일 시스템에 대한 복구 지점 파일 만들기.

samexplorer 실행

1. 파일 시스템 서버 호스트에 *root*로 로그인합니다.

   예제에서 메타데이터 서버 호스트는 이름이 *server1*로 지정됩니다.

   ```
   [server1]root@solaris:~#
   ```

2. *samexplorer* 보고서를 만듭니다. *samexplorer path/hostname.YYYYMMDD.hhmmz.tar.gz* 명령을 사용합니다. 설명:
   - *path*는 선택한 디렉토리에 대한 경로입니다.
   - *hostname*은 Oracle HSM 파일 시스템 호스트의 이름입니다.
   - *YYYYMMDD.hhmmz*는 날짜 및 시간 기록입니다.

   기본적으로 이 파일은 */tmp/SAMreport.hostname.YYYYMMDD.hhmmz.tar.gz*입니다. 예제에서는 */zfs1/tmp/* 디렉토리를 사용합니다. 여기서 */zfs1*은 Oracle HSM 파일 시스템과 공통 구성 요소가 없는 파일 시스템입니다. 아래의 명령은 단일 라인으로 입력됩니다. 줄바꿈은 백슬래시 문자로 이스케이프됩니다.

```
[server1]root@solaris:~# samexplorer /
/zfs1/sam_config/explorer/samhost1.20140130.1659MST.tar.gz

     Report name:     /zfs1/sam_config/explorer/samhost1.20140130.1659MST.tar.gz
     Lines per file:  1000
     Output format:   tar.gz (default) Use -u for unarchived/uncompressed.

     Please wait.....................................
     Please wait.....................................
     Please wait.....................................

     The following files should now be ftp'ed to your support provider
     as ftp type binary.

     /zfs1/sam_config/explorer/samhost1.20140130.1659MST.tar.gz
```

3. 파일 시스템에 대한 복구 지점 파일 만들기를 수행합니다.

파일 시스템에 대한 복구 지점 파일 만들기

1. 파일 시스템 서버 호스트에 *root*로 로그인합니다.

   예제에서 메타데이터 서버 호스트는 이름이 *server1*로 지정됩니다.

```
[server1]root@solaris:~#
```

2. 복구 지점 파일을 저장할 위치를 선택합니다. 선택한 위치는 백업 중인 파일 시스템과 어떤 장치도 공유하지 않아야 하며 엄청나게 큰 파일을 저장할 공간이 있어야 합니다.

   제거하려는 장치에 아카이브되지 않은 파일이 포함될 수 있습니다. 이러한 파일은 단일 복사본으로만 존재하므로 적어도 일부 데이터와 메타데이터를 저장할 복구 지점 파일을 만들어야 합니다. 그러면 복구 지점 파일의 크기가 상당히 증가할 수 있습니다.

예제에서는 Oracle HSM 파일 시스템과 공통 구성 요소가 없는 파일 시스템 */zfs1*에 하위 디렉토리 *tmp/*를 만듭니다.

```
[server1]root@solaris:~# mkdir /zfs1/tmp/
[server1]root@solaris:~#
```

3. 파일 시스템의 루트 디렉토리로 변경합니다.

   예제에서는 마운트 지점 디렉토리 **/hsm/hsmfs1**로 변경합니다.

```
[server1]root@solaris:~# cd /hsm/hsmfs1
[server1]root@solaris:~#
```

4. 파일 시스템 메타데이터와 아카이브되지 않은 데이터를 백업합니다. *samfsdump -f -u recovery-point* 명령을 사용합니다. 여기서 *recovery-point*는 완성된 복구 지점 파일의 경로 및 파일 이름입니다.

   -u 옵션은 아카이브되지 않은 파일의 데이터 부분을 복구 지점에 추가합니다. 그러면 파일 크기가 크게 증가할 수 있습니다.

   예제에서는 */zfs1/tmp/* 디렉토리에 *hsmfs1-20140313.025215*라는 *hsmfs1* 파일 시스템에 대한 복구 지점 파일을 만듭니다. *ls -l* 명령을 사용하여 결과를 확인합니다. 아래의 두번째 명령은 단일 라인으로 입력됩니다. 줄바꿈은 백슬래시 문자로 이스케이프됩니다.

```
[server1]root@solaris:~# cd /hsm/hsmfs1
[server1]root@solaris:~# samfsdump -f /
/zfs1/tmp/hsm/hsmfs1-`date '+%Y%m%d.%H%M%S'` -T /hsm/hsmfs1
samfsdump statistics:
    Files:              10010
    Directories:        2
    Symbolic links:     0
    Resource files:     0
    Files as members of hard links :    0
    Files as first hard link :  0
    File segments:      0
    File archives:      10010
    Damaged files:      0
    Files with data:    0
    File  warnings:     0
    Errors:             0
    Unprocessed dirs:   0
    File data bytes:    0
[server1]root@solaris:~# ls -l /zfs1/tmp/hsmfs1*
-rw-r--r-- 1 root other 5376517 Mar 13 02:52 /zfs1/tmp/hsm/hsmfs1-20140313.025215
```

```
[server1]root@solaris:~#
```

5. 이제 마운트된 고성능 파일 시스템에서 장치 제거를 수행합니다.

마운트된 고성능 파일 시스템에서 장치 제거

한번에 하나씩 장치를 제거해야 합니다. 각 장치에 대해 다음과 같이 하십시오.

1. 파일 시스템 서버 호스트에 *root*로 로그인합니다.

   예제에서 메타데이터 서버 호스트는 이름이 *server1*로 지정됩니다.

   ```
   [server1]root@solaris:~#
   ```

2. */etc/opt/SUNWsamfs/mcf* 파일을 열고 제거해야 하는 장치에 대한 장비 순서 번호를 확인합니다.

   예제에서는 *vi* 편집기를 사용합니다. *hsmfs1* 파일 시스템의 장비 목록에서 */dev/dsk/c1t4d0s7* 장치를 제거해야 합니다. 장비 순서 번호는 *104*입니다.

   ```
   [server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
   # Equipment          Equipment  Equipment  Family     Device   Additional
   # Identifier         Ordinal    Type       Set        State    Parameters
   #-----------------   ---------  ---------  ---------  ------   --------------
   hsmfs1              100        ms         hsmfs1    on
    /dev/dsk/c1t3d0s3    101        md         hsmfs1    on
    /dev/dsk/c1t4d0s5    102        md         hsmfs1    on
    /dev/dsk/c1t3d0s7    103        md         hsmfs1    on
    /dev/dsk/c1t4d0s7    104        md         hsmfs1    on
   :q
   [server1]root@solaris:~#
   ```

3. 장치를 제거하기 전에 파일 시스템에 남은 장치들이 삭제하려는 장치에서 이동된 파일을 수용할 수 있는지 확인합니다.
   - 남은 장치에 충분한 용량이 있는지 확인합니다.
   - 장치가 스트라이프 그룹인 경우 파일 시스템에 동일 구성의 스트라이프 그룹이 또 하나 있는지 확인합니다.

     예를 들어, 제거할 스트라이프 그룹에 4개의 장비 번호가 있는 경우 또 하나의 스트라이프 그룹이 ON 상태이고 4개의 장비 번호가 있어야 합니다.
4. 수정할 파일 시스템에 버전 2A 수퍼 블록이 있는지 확인합니다. *samfsinfo filesystem-name* 명령을 사용합니다. 여기서 *filesystem-name*은 파일 시스템의 이름입니다.

   예제에서는 *hsmfs1* 파일 시스템이 *version:2A* 수퍼 블록을 사용합니다.

```
[server1]root@solaris:~# /opt/SUNWsamfs/sbin/samfsinfo hsmfs1
samfsinfo: filesystem hsmfs1 is mounted.
name:     hsmfs1       version:     2A
time:     Tuesday, June 28, 2011  6:07:36 AM MDT
feature:  Aligned Maps
count:    4
...
[server1]root@solaris:~#
```

5. 파일 시스템에 버전 2A 수퍼 블록이 없는 경우 여기서 중지합니다. 이 파일 시스템을 마운트하는 동안 장치를 제거할 수 없습니다.
6. Oracle HSM 아카이빙 파일 시스템에서 장치를 제거하는 경우 모든 아카이브된 파일을 제거 중인 디스크 장치에서 해제합니다. *samcmd release equipment-number* 명령을 사용합니다. 여기서 *equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 장치를 식별하는 장비 순서 번호입니다.

   장치가 스트라이프 그룹인 경우 그룹의 아무 장치나 장비 번호를 제공합니다.

   Oracle HSM 소프트웨어는 지정된 장치에 새 파일이 저장되지 않도록 상태를 *noalloc*(할당 없음)로 변경하고, 이전에 아카이브된 파일을 해제하기 시작합니다. 장치에 아카이브되지 않은 파일이 없을 때 소프트웨어는 파일 시스템 구성에서 장치를 제거하고 상태를 *off*로 변경합니다.

   예제에서는 아카이빙 파일 시스템 *hsmfs1*의 *104* 장치에서 파일을 해제합니다.

   ```
   [server1]root@solaris:~# samcmd release 104
   ```

7. Oracle HSM 비아카이빙 파일 시스템에서 장치를 제거하는 경우 모든 남은 유효한 파일을 제거 중인 디스크 장치 밖으로 이동합니다. *samcmd remove equipment-number* 명령을 사용합니다. 여기서 *equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 장치를 식별하는 장비 순서 번호입니다.

   Oracle HSM 소프트웨어는 지정된 장치에 새 파일이 저장되지 않도록 상태를 *noalloc*(할당 없음)로 변경하고, 유효한 데이터가 포함된 파일을 파일 시스템에 남은 장치로 이동하기 시작합니다. 모든 파일이 이동되었을 때 소프트웨어는 파일 시스템 구성에서 장치를 제거하고 상태를 *off*로 변경합니다.

   예제에서는 *104* 장치의 파일을 밖으로 이동합니다.

   ```
   [server1]root@solaris:~# samcmd remove 104
   ```

8. 선택한 프로세스 *samcmd remove* 또는 *samcmd release*의 진행률을 모니터합니다. *samcmd m* 명령을 사용하고 로그 파일과 */var/opt/SUNWsamfs/trace/sam-shrink* 파일을 관찰합니다.

release 프로세스는 모든 파일이 아카이브된 경우 단순히 아카이브 매체로 복사된 파일과 연관된 공간을 해제하면 되므로 아주 빠르게 완료됩니다. *remove* 프로세스는 데이터 양과 파일 수에 따라 디스크 장치 사이에 파일을 이동해야 하므로 상당히 오래 걸립니다.

```
[server1]root@solaris:~# samcmd m
ty     eq  status       use state  ord  capacity      free    ra  part high low
ms     100 m----2-----  27% on            3.691G    2.628G    1M    16  80% 70%
 md     101              27% on       0  959.938M  703.188M
 md     102              28% on       1  899.938M  646.625M
 md     103              13% on       2  959.938M  834.250M
 md     104               0% noalloc  3  959.938M  959.938M
[server1]root@solaris:~#
```

9. *samcmd release*를 사용할 때 대상 장치가 *off* 상태로 진입하지 않으면 장치에 아카이브되지 않은 파일이 있는 것입니다. 아카이버가 실행되고 아카이빙이 완료될 때까지 기다립니다. 그런 다음 다시 *samcmd release* 명령을 사용합니다. *samcmd a* 명령을 사용하여 아카이빙의 진행률을 확인할 수 있습니다.

   아카이브되지 않은 파일이 아카이브될 때까지 *release* 프로세스는 디스크 공간을 확보할 수 없습니다.

```
[server1]root@solaris:~# samcmd a
Archiver status samcmd     5.4 14:12:14 Mar  1 2014
samcmd on server1
sam-archiverd:  Waiting for resources
sam-arfind:  hsmfs1  mounted at /hsm/hsmfs1
Files waiting to start          4   schedule          2   archiving          2
[server1]root@solaris:~#
```

10. 하나 이상의 아카이브되지 않은 파일을 아카이브할 수 없어서 *samcmd release*가 실패하면 아카이브되지 않은 파일을 다른 장치로 이동합니다. 비아카이빙 독립형 파일 시스템에서 장치를 제거할 때와 마찬가지로 *samcmd remove equipment-number* 명령을 사용합니다.

    예제에서는 *104* 장치의 파일을 밖으로 이동합니다.

```
[server1]root@solaris:~# samcmd remove 104
```

11. 장치 상태가 *off*로 변경되었으면 텍스트 편집기에서 */etc/opt/SUNWsamfs/mcf* 파일을 열고 파일 시스템을 찾아서 변경사항이 반영되도록 장비 목록을 업데이트합니다. 파일을 저장하고 편집기를 닫습니다.

예제에서 *samcmd m*은 *104*가 *off*임을 보여줍니다. 따라서 *vi* 편집기를 사용하여 *mcf* 파일을 엽니다. *hsmfs1* 파일 시스템의 장비 목록에서 *104* 장치 항목을 제거하고 변경사항을 저장합니다.

```
[server1]root@solaris:~# samcmd m
ty      eq  status       use state  ord  capacity      free    ra  part high low
ms     100  m----2-----  27% on           3.691G    2.628G    1M    16  80% 70%
 md     101              27% on       0  959.938M  703.188M
 md     102              28% on       1  899.938M  646.625M
 md     103              13% on       2  959.938M  834.250M
 md     104               0% off      3  959.938M  959.938M
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment          Equipment  Equipment  Family     Device   Additional
# Identifier         Ordinal    Type       Set        State    Parameters
#-----------------   ---------  ---------  ---------  ------   ---------
hsmfs1              100        ms         hsmfs1    on
 /dev/dsk/c1t3d0s3    101        md         hsmfs1    on
 /dev/dsk/c1t4d0s5    102        md         hsmfs1    on
 /dev/dsk/c1t3d0s7    103        md         hsmfs1    on
:wq
[server1]root@solaris:~#
```

12. *sam-fsd* 명령을 실행하여 수정된 *mcf* 파일에 오류가 있는지 확인하고, 오류가 감지되면 수정합니다.

    만일 오류가 발생하면 *sam-fsd* 명령이 중지됩니다.

    ```
    [server1]root@solaris:~# sam-fsd
    ```

13. Oracle HSM 소프트웨어에 *mcf* 파일을 다시 읽고 그에 따라 재구성하도록 지시합니다.

    ```
    [server1]root@solaris:~# samd config
    ```

14. 여기서 중지합니다.

Oracle HSM 공유 파일 시스템 관리

이 절에서는 다음 작업을 설명합니다.

- 공유 파일 시스템 마운트 및 마운트 해제
- 공유 파일 시스템의 호스트 구성 변경
- 활성 메타데이터 서버에서 잠재적 메타데이터 서버로 전환
- 비공유 파일 시스템을 공유 파일 시스템으로 변환
- 공유 파일 시스템을 비공유 파일 시스템으로 변환.

공유 파일 시스템 마운트 및 마운트 해제

공유 파일 시스템을 마운트하거나 마운트 해제할 때 메타데이터 서버와 클라이언트를 마운트/마운트 해제하는 순서가 매우 중요합니다.

페일오버 목적상, 마운트 옵션은 메타데이터 서버와 모든 잠재적 메타데이터 서버에서 동일해야 합니다. 예를 들어, 마운트 옵션을 포함하는 *samfs.cmd* 파일을 만들어서 모든 호스트에 복사할 수 있습니다.

공유 파일 시스템 마운트에 대한 자세한 내용은 *mount_samfs* 매뉴얼 페이지를 참조하십시오.

공유 파일 시스템 마운트

1. Oracle HSM 메타데이터 서버 및 클라이언트 호스트에 *root*로 로그인합니다.

   예제에서는 *sharefs* 파일 시스템 *sharefs-mds*에 대한 메타데이터 서버 호스트에 로그인합니다. 그런 다음 각 클라이언트 *sharefs-client1* 및 *sharefs-client2*에 대한 터미널 창을 엽니다. *ssh*(보안 셸)를 사용하여 로그인합니다.

   ```
   [sharefs-mds]root@solaris:~# ssh root@sharefs-client1
   Password:
   [sharefs-client1]root@solaris:~#

   [sharefs-mds]root@solaris:~# ssh root@sharefs-client2
   Password:
   [sharefs-client2]root@solaris:~#
   ```

2. 파일 시스템의 Solaris */etc/vfstab* 파일에 항목이 있는 경우 *mount mountpoint* 명령을 사용하여 메타데이터 서버 호스트에 공유 파일 시스템을 마운트합니다. 여기서 *mountpoint*는 호스트 루트 파일 시스템의 마운트 지점 디렉토리입니다.

   항상 먼저 메타데이터 서버 호스트에 파일 시스템을 마운트한 후에 클라이언트에 파일 시스템을 마운트하십시오.

   예제에서는 *sharefs* 파일 시스템의 */etc/vfstab* 파일에 다음 항목이 있습니다.

   ```
   sharefs   -         /sharefs  samfs   -     no       shared
   ```

   따라서 마운트 지점 매개변수만 제공하면 파일 시스템을 마운트할 수 있습니다.

   ```
   [sharefs-mds]root@solaris:~# mount /sharefs
   [sharefs-mds]root@solaris:~#
   ```

3. 파일 시스템의 Solaris */etc/vfstab* 파일에 항목이 없는 경우 *mount -F samfs -o shared mountpoint* 명령을 사용하여 메타데이터 서버 호스트에 공유 파일 시스템을

마운트합니다. 여기서 *mountpoint*는 호스트 루트 파일 시스템의 마운트 지점 디렉토리입니다.

항상 먼저 메타데이터 서버 호스트에 파일 시스템을 마운트한 후에 클라이언트에 파일 시스템을 마운트하십시오.

예제에서는 *sharefs* 파일 시스템의 */etc/vfstab* 파일에 항목이 없습니다.

```
[sharefs-mds]root@solaris:~# mount -F samfs -o shared /sharefs
[sharefs-mds]root@solaris:~#
```

4. 파일 시스템의 Solaris */etc/vfstab* 파일에 항목이 있는 경우 *mount mountpoint* 명령을 사용하여 각 클라이언트 호스트에 공유 파일 시스템을 마운트합니다. 여기서 *mountpoint*는 호스트 루트 파일 시스템의 마운트 지점 디렉토리입니다.

   원하는 순서로 클라이언트 호스트에 파일 시스템을 마운트할 수 있습니다.

   ```
   [sharefs-client1]root@solaris:~# mount /sharefs
   [sharefs-client1]root@solaris:~#

   [sharefs-client2]root@solaris:~# mount /sharefs
   [sharefs-client2]root@solaris:~#
   ```

5. 파일 시스템의 Solaris */etc/vfstab* 파일에 항목이 없는 경우 *mount -F samfs -o shared mountpoint* 명령을 사용하여 각 클라이언트 호스트에 공유 파일 시스템을 마운트합니다. 여기서 *mountpoint*는 호스트 루트 파일 시스템의 마운트 지점 디렉토리입니다.

   원하는 순서로 클라이언트 호스트에 파일 시스템을 마운트할 수 있습니다.

   ```
   [sharefs-client1]root@solaris:~# mount -F samfs -o shared /sharefs
   [sharefs-client1]root@solaris:~#

   [sharefs-client2]root@solaris:~# mount -F samfs -o shared /sharefs
   [sharefs-client2]root@solaris:~#
   ```

6. 여기서 중지합니다.

공유 파일 시스템 마운트 해제

1. Oracle HSM 메타데이터 서버 및 클라이언트 호스트에 *root*로 로그인합니다.

   예제에서는 *sharefs* 파일 시스템 *sharefs-mds*에 대한 메타데이터 서버 호스트에 로그인합니다. 그런 후 각 클라이언트 *sharefs-client1* 및 *sharefs-client2*에 대한 터미널 창을 열고 *ssh*(보안 셸)를 사용해서 로그인합니다.

```
[sharefs-mds]root@solaris:~# ssh root@sharefs-client1
Password:
[sharefs-client1]root@solaris:~#

[sharefs-mds]root@solaris:~# ssh root@sharefs-client2
Password:
[sharefs-client2]root@solaris:~#
```

2. 파일 시스템이 NFS 또는 SAMBA를 통해 공유되는 경우 파일 시스템을 마운트 해제하기 전에 공유를 취소합니다. 메타데이터 서버에서 *unshare mount-point* 명령을 사용합니다. 여기서 *mount-point*는 Oracle HSM 파일 시스템의 마운트 지점 디렉토리입니다.

```
[sharefs-mds]root@solaris:~# unshare /sharefs
[sharefs-mds]root@solaris:~#
```

3. 각 클라이언트에서 Oracle HSM 공유 파일 시스템을 마운트 해제합니다. *umount mount-point*명령을 사용합니다. 여기서 *mount-point*는 Oracle HSM 파일 시스템의 마운트 지점 디렉토리입니다.

   자세한 내용은 *umount_samfs* 매뉴얼 페이지를 참조하십시오. 예제에서는 두 개의 클라이언트 *sharefs-client1* 및 *sharefs-client2*에서 */sharedqfs1*을 마운트 해제합니다.

```
[sharefs-client1]root@solaris:~# umount /sharefs
[sharefs-client1]root@solaris:~# exit
[sharefs-mds]root@solaris:~#

[sharefs-client2]root@solaris:~# umount /sharefs
[sharefs-client1]root@solaris:~# exit
[sharefs-mds]root@solaris:~#
```

4. 메타데이터 서버에서 Oracle HSM 공유 파일 시스템을 마운트 해제합니다. *umount -o await_clients=interval mount-point* 명령을 사용합니다. 여기서 *mount-point*는 Oracle HSM 파일 시스템의 마운트 지점 디렉토리이고, *interval*은 *-o await_clients* 옵션으로 지정된 실행 지연 시간(초)입니다.

   Oracle HSM 공유 파일 시스템의 메타데이터 서버에서 *umount* 명령을 실행할 때 *-o await_clients* 옵션은 지정된 초 수만큼 *umount*를 대기시켜서 클라이언트가 공유 파일 시스템을 마운트 해제할 시간을 벌어줍니다. 비공유 파일 시스템을 마운트 해제하거나 Oracle HSM 클라이언트에서 명령을 실행할 경우 아무 효과가 없습니다. 자세한 내용은 *umount_samfs* 매뉴얼 페이지를 참조하십시오.

예제에서는 서버에서 */sharefs* 파일 시스템을 마운트 해제하고, 클라이언트가 *60*초 동안 마운트 해제되도록 허용합니다.

```
[sharefs-mds]root@solaris:~# umount -o await_clients=60 /sharefs
[sharefs-mds]root@solaris:~#
```

5. 여기서 중지합니다.

공유 파일 시스템의 호스트 구성 변경

이 절에서는 공유 파일 시스템의 클라이언트로서 추가 호스트를 구성하고 기존 클라이언트 구성을 해제하기 위한 지침을 제공합니다. 다음 절이 포함됩니다.

- 추가 파일 시스템 클라이언트 구성
- 공유 파일 시스템 구성에서 호스트 제거
- 분산 테이프 I/O를 위한 Datamover 클라이언트 구성
- 지속 바인딩을 사용하여 테이프 드라이브 연결.

추가 파일 시스템 클라이언트 구성

공유 파일 시스템에 클라이언트 호스트를 추가하는 과정은 다음 세 부분으로 구성됩니다.

- 먼저, 공유 파일 시스템 구성에 호스트 정보 추가를 수행합니다.
- 그 다음, 호스트 운영체제에 따라 Solaris 또는 Linux 절차를 사용하여 호스트에 공유 파일 시스템을 구성합니다.
- 마지막으로, 호스트 운영체제에 따라 Solaris 또는 Linux 절차를 사용하여 호스트에 공유 파일 시스템을 마운트합니다.

공유 파일 시스템 구성에 호스트 정보 추가

1. Oracle HSM 메타데이터 서버에 *root*로 로그인합니다.

   예제에서 Oracle HSM 공유 파일 시스템은 *sharefs*, 메타데이터 서버 호스트는 *sharefs-mds*입니다.

   ```
   [sharefs-mds]root@solaris:~#
   ```

2. */etc/opt/SUNWsamfs/hosts.filesystem* 파일을 백업합니다. 여기서 *filesystem*은 클라이언트 호스트를 추가하려는 파일 시스템의 이름입니다.

   아래의 명령은 단일 라인으로 입력됩니다. 줄바꿈은 백슬래시 문자로 이스케이프됩니다.

   ```
   [sharefs-mds]root@solaris:~# cp /etc/opt/SUNWsamfs/hosts.sharefs / /etc/opt/SUNWsamfs/
   hosts.sharefs.bak
   ```

3. 공유 파일 시스템이 마운트된 경우 활성 메타데이터 서버에서 *samsharefs filesystem* 명령을 실행하여 */etc/opt/SUNWsamfs/hosts.filesystem* 파일로 출력이 재지정되도록 합니다. 여기서 *filesystem*은 클라이언트 호스트를 추가하려는 파일 시스템의 이름입니다.

   samsharefs 명령은 Oracle HSM 공유 파일 시스템에 대한 호스트 구성을 표시합니다. 출력을 파일로 재지정하면 새 호스트 파일이 만들어집니다. 아래의 명령은 단일 라인으로 입력됩니다. 줄바꿈은 백슬래시 문자로 이스케이프됩니다.

   ```
   [sharefs-mds]root@solaris:~# samsharefs sharedqfs1 > / /etc/opt/SUNWsamfs/
   hosts.sharedqfs1
   ```

4. 공유 파일 시스템이 마운트되지 않았으면 활성 또는 잠재적 메타데이터 서버에서 *samsharefs -R filesystem* 명령을 실행하여 출력이 */etc/opt/SUNWsamfs/hosts.filesystem* 파일로 재지정되도록 합니다. 여기서 *filesystem*은 클라이언트 호스트를 추가하려는 파일 시스템의 이름입니다.

   samsharefs -R 명령은 활성 또는 잠재적 메타데이터 서버에서만 실행할 수 있습니다. 자세한 내용은 *samsharefs* 매뉴얼 페이지를 참조하십시오. *samsharefs* 명령은 Oracle HSM 공유 파일 시스템에 대한 호스트 구성을 표시합니다. 출력을 파일로 재지정하면 새 hosts 파일이 만들어집니다. 예제에서는 메타데이터 서버 *sharefs-mds*에서 명령을 실행합니다. 아래의 명령은 단일 라인으로 입력됩니다. 줄바꿈은 백슬래시 문자로 이스케이프됩니다.

   ```
   [sharefs-mds]root@solaris:~# samsharefs -R sharedqfs1 /
   > /etc/opt/SUNWsamfs/hosts.sharedqfs1
   ```

5. 새로 만든 hosts 파일을 텍스트 편집기에서 엽니다.

   예제에서는 *vi* 편집기를 사용합니다. 호스트 구성에는 활성 메타데이터 서버인 *sharefs-mds*와 또한 잠재적 메타데이터 서버인 하나의 클라이언트 *sharefs-mds_alt* 및 두 개의 다른 클라이언트인 *sharefs-client1* 및 *sharefs-client2*가 포함됩니다.

   ```
   [sharefs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sharefs
   #                                        Server   On/  Additional
   #Host Name           Network Interface   Ordinal  Off  Parameters
   #------------------  ------------------  -------  ---  ----------
   sharefs-mds          10.79.213.117       1        0    server
   sharefs-mds_alt      10.79.213.217       2        0
   sharefs-client1      10.79.213.133       0        0
   sharefs-client2      10.79.213.47        0        0
   ```

6. hosts 파일에서 새 클라이언트 호스트 행을 추가하고 파일을 저장하고 편집기를 닫습니다.

예제에서는 *sharefs-client3* 호스트 항목을 추가합니다.

```
[sharefs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sharefs
#                                          Server   On/  Additional
#Host Name            Network Interface     Ordinal  Off  Parameters
#------------------   --------------------  -------  ---  ----------
sharefs-mds           10.79.213.117         1        0    server
sharefs-mds_alt       10.79.213.217         2        0
sharefs-client1       10.79.213.133         0        0
sharefs-client2       10.79.213.47          0        0
sharefs-client3       10.79.213.49          0        0
:wq
[sharefs-mds]root@solaris:~#
```

7. 파일 시스템이 마운트된 경우 활성 메타데이터 서버에서 파일 시스템을 업데이트합니다. *samsharefs -u filesystem* 명령을 사용합니다. 여기서 *filesystem*은 클라이언트 호스트를 추가하려는 파일 시스템의 이름입니다.

   samsharefs 명령은 수정된 hosts 파일을 다시 읽고 구성을 업데이트합니다.

   ```
   [sharefs-mds]root@solaris:~# samsharefs -u sharefs1
   ```

8. 파일 시스템이 마운트되지 않았으면 활성 또는 잠재적 메타데이터 서버에서 파일 시스템을 업데이트합니다. *samsharefs -R -u filesystem* 명령을 사용합니다. 여기서 *filesystem*은 클라이언트 호스트를 추가하려는 파일 시스템의 이름입니다.

   samsharefs 명령은 수정된 hosts 파일을 다시 읽고 구성을 업데이트합니다.

   ```
   [sharefs-mds]root@solaris:~# samsharefs -R -u sharefs1
   ```

9. Solaris 호스트를 클라이언트로 추가하는 경우 "Solaris 클라이언트에서 공유 파일 시스템 구성"으로 이동합니다.
10. Linux 호스트를 클라이언트로 추가하는 경우 “Linux 클라이언트 호스트에서 공유 파일 시스템 구성”으로 이동합니다.

Solaris 클라이언트에서 공유 파일 시스템 구성

1. 공유 파일 시스템 클라이언트에서 *root*로 로그인합니다.

   예제에서 Oracle HSM 공유 파일 시스템은 *sharefs*이고, 클라이언트 호스트는 *sharefs-client1*입니다.

   ```
   [sharefs-client1]root@solaris:~#
   ```

2. 터미널 창에서 공유 파일 시스템에 대한 구성 정보를 검색합니다. *samfsconfig device-path* 명령을 사용합니다. 여기서 *device-path*는 명령이 파일 시스템 디스크 장치(예: */dev/dsk/** 또는 */dev/zvol/dsk/rpool/**) 검색을 시작해야 하는 위치입니다.

```
[sharefs-client1]root@solaris:~# samfsconfig /dev/dsk/*
```

3. 호스트가 파일 시스템의 메타데이터 장치에 액세스할 수 있는 경우 잠재적 메타데이터 서버로 사용하기에 적절하므로 *samfsconfig* 출력은 파일 시스템 메타데이터 서버에서 만든 *mcf* 파일과 아주 비슷합니다.

   예제에서 *sharefs-client1* 호스트에는 메타데이터 장치(장비 유형 *mm*)에 대한 액세스 권한이 있으므로, 이 명령은 *sharefs-mds* 서버의 *mcf* 파일에 나열된 것과 동일한 장비를 보여줍니다. 호스트 지정 장치 컨트롤러 번호만 다릅니다.

```
[sharefs-client1]root@solaris:~# samfsconfig /dev/dsk/*
# Family Set 'sharefs' Created Thu Feb 21 07:17:00 2013
# Generation 0 Eq count 4 Eq meta count 1
sharefs             300         ma         sharefs   -
/dev/dsk/c1t0d0s0    301         mm         sharefs   -
/dev/dsk/c1t3d0s0    302         mr         sharefs   -
/dev/dsk/c1t3d0s1    303         mr         sharefs   -
```

4. 호스트가 파일 시스템의 메타데이터 장치에 액세스할 수 없는 경우 *samfsconfig* 명령이 메타데이터 장치를 찾을 수 없습니다. 따라서 발견된 Oracle HSM 장치가 파일 시스템 구성에 맞지 않을 수 있습니다. 명령 출력은 *Missing Slices* 아래에 메타데이터 장치 *Ordinal 0*을 나열하고, 파일 시스템 패밀리 세트를 식별하는 라인을 포함하지 못하며, 데이터 장치의 목록을 주석 처리합니다.

   예제에서는 *sharefs-client2* 호스트가 데이터 장치에만 액세스할 수 있습니다. 따라서 *samfsconfig* 출력은 다음과 비슷합니다.

```
[sharefs-client2]root@solaris:~# samfsconfig /dev/dsk/*
# Family Set 'sharefs' Created Thu Feb 21 07:17:00 2013
# Missing slices
# Ordinal 0
# /dev/dsk/c4t3d0s0    302         mr         sharefs   -
# /dev/dsk/c4t3d0s1    303         mr         sharefs   -
```

5. *samfsconfig* 출력에서 공유 파일 시스템 항목을 복사합니다. 그리고 두번째 창에서, 텍스트 편집기에서 */etc/opt/SUNWsamfs/mcf* 파일을 열고 복사한 항목을 파일로 붙여넣습니다.

   첫번째 예제에서 *sharefs-client1* 호스트는 파일 시스템의 메타데이터 장치에 액세스할 수 있으므로 *mcf* 파일의 시작은 다음과 비슷합니다.

```
[sharefs-client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment         Equipment  Equipment  Family     Device   Additional
# Identifier        Ordinal    Type       Set        State    Parameters
#----------------   ---------  ---------  ---------  ------   ---------------
sharefs             300        ma         sharefs    -
/dev/dsk/c1t0d0s0   301        mm         sharefs    -
/dev/dsk/c1t3d0s0   302        mr         sharefs    -
/dev/dsk/c1t3d0s1   303        mr         sharefs    -
```

두번째 예제에서 *sharefs-client2* 호스트는 파일 시스템의 메타데이터 장치에 액세스할 수 없으므로 *mcf* 파일의 시작은 다음과 비슷합니다.

```
[sharefs-client2]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment         Equipment  Equipment  Family     Device   Additional
# Identifier        Ordinal    Type       Set        State    Parameters
#----------------   ---------  ---------  ---------  ------   ---------------
# /dev/dsk/c4t3d0s0   302         mr         sharefs   -
# /dev/dsk/c4t3d0s1   303         mr         sharefs   -
```

6. 호스트가 파일 시스템의 메타데이터 장치에 액세스할 수 있는 경우 공유 파일 시스템 항목의 *Additional Parameters* 필드에 *shared* 매개변수를 추가합니다.

   첫번째 예제에서 *sharefs-client1* 호스트는 메타데이터에 액세스할 수 있습니다.

```
[sharefs-client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment         Equipment  Equipment  Family     Device   Additional
# Identifier        Ordinal    Type       Set        State    Parameters
#----------------   ---------  ---------  ---------  ------   ---------------
sharefs             300        ma         sharefs    -        shared
/dev/dsk/c1t0d0s0   301        mm         sharefs    -
/dev/dsk/c1t3d0s0   302        mr         sharefs    -
/dev/dsk/c1t3d0s1   303        mr         sharefs    -
```

7. 호스트가 파일 시스템의 메타데이터 장치에 액세스할 수 없는 경우 공유 파일 시스템 행을 추가하고 *shared* 매개변수를 포함합니다.

```
[sharefs-client2]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment         Equipment  Equipment  Family     Device   Additional
# Identifier        Ordinal    Type       Set        State    Parameters
#----------------   ---------  ---------  ---------  ------   ---------------
sharefs             300        ma         sharefs    -        shared
# /dev/dsk/c4t3d0s0   302         mr         sharefs   -
# /dev/dsk/c4t3d0s1   303         mr         sharefs   -
```

8. 호스트가 파일 시스템의 메타데이터 장치에 액세스할 수 없는 경우 메타데이터 장치 행을 추가합니다. *Equipment Identifier* 필드를 *nodev*(장치 없음)로 설정하고 남은 필드를 메타데이터 서버에 지정한 것과 똑같은 값으로 설정합니다.

```
[sharefs-client2]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment         Equipment  Equipment  Family     Device   Additional
# Identifier        Ordinal    Type       Set        State    Parameters
#----------------   ---------  ---------  ---------  ------   ---------------
sharefs             300         ma        sharefs    on       shared
nodev               301         mm        sharefs    on
# /dev/dsk/c4t3d0s0   302        mr          sharefs    -
# /dev/dsk/c4t3d0s1   303        mr          sharefs    -
```

9. 호스트가 파일 시스템의 메타데이터 장치에 액세스할 수 없는 경우 데이터 장치 항목의 주석 처리를 해제합니다.

```
[sharefs-client2]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment         Equipment  Equipment  Family     Device   Additional
# Identifier        Ordinal    Type       Set        State    Parameters
#----------------   ---------  ---------  ---------  ------   ---------------
sharefs             300        ma         sharefs    on       shared
nodev               301        mm         sharefs    on
/dev/dsk/c4t3d0s0   302        mr         sharefs    -
/dev/dsk/c4t3d0s1   303        mr         sharefs    -
```

10. 모든 장치에 대해 *Device State* 필드가 *on*으로 설정되었는지 확인하고 *mcf* 파일을 저장하고 편집기를 닫습니다.

    첫번째 예제에서 *sharefs-client1* 호스트는 파일 시스템의 메타데이터 장치에 액세스할 수 있으므로 *mcf* 파일의 끝은 다음과 비슷합니다.

```
[sharefs-client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment         Equipment  Equipment  Family     Device   Additional
# Identifier        Ordinal    Type       Set        State    Parameters
#----------------   ---------  ---------  ---------  ------   ---------------
sharefs             300        ma         sharefs    on       shared
/dev/dsk/c1t0d0s0   301        mm         sharefs    on
/dev/dsk/c1t3d0s0   302        mr         sharefs    on
/dev/dsk/c1t3d0s1   303        mr         sharefs    on
:wq
[sharefs-client1]root@solaris:~#
```

두번째 예제에서 *sharefs-client2* 호스트는 파일 시스템의 메타데이터 장치에 액세스할 수 없으므로 *mcf* 파일의 끝은 다음과 비슷합니다.

```
[sharefs-client2]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment         Equipment  Equipment  Family     Device   Additional
# Identifier        Ordinal    Type       Set        State    Parameters
#----------------   ---------  ---------  ---------  ------   ---------------
sharefs               300         ma        sharefs    on       shared
nodev                 301         mm        sharefs    on
/dev/dsk/c4t3d0s0     302         mr        sharefs    on
/dev/dsk/c4t3d0s1     303         mr        sharefs    on
:wq
[sharefs-client2]root@solaris:~#
```

11. *sam-fsd* 명령을 실행하여 *mcf* 파일에 오류가 있는지 확인하고, 오류가 발견되면 수정합니다.

    *sam-fsd*는 Oracle HSM 구성 파일을 읽는 초기화 명령입니다. 만일 오류가 발생하면 중지됩니다. 예제에서는 *sharefs-client1*에서 *mcf* 파일을 확인합니다.

    ```
    [sharefs-client1]root@solaris:~# sam-fsd
    ```

12. 그런 다음 Solaris 호스트에서 공유 파일 시스템 마운트를 수행합니다.

Solaris 호스트에서 공유 파일 시스템 마운트

1. 공유 파일 시스템 호스트에서 *root*로 로그인합니다.

   예제에서 Oracle HSM 공유 파일 시스템은 *sharefs*이고, 호스트는 *sharefs-client1*라는 이름의 클라이언트입니다.

   ```
   [sharefs-client1]root@solaris:~#
   ```

2. 운영체제의 */etc/vfstab* 파일을 백업합니다.

   ```
   [sharefs-client1]root@solaris:~# cp /etc/vfstab /etc/vfstab.backup
   ```

3. 텍스트 편집기에서 */etc/vfstab* 파일을 열고 공유 파일 시스템 행을 추가합니다.

   예제에서는 *vi* 텍스트 편집기에서 파일을 열고 *sharefs* 패밀리 세트 장치 행을 추가합니다.

   ```
   [sharefs-client1]root@solaris:~# vi /etc/vfstab
   #File
   ```

```
#Device    Device   Mount     System  fsck  Mount    Mount
#to Mount  to fsck  Point     Type    Pass  at Boot  Options
#--------  -------  --------  ------  ----  -------  ------------------------
/devices   -        /devices  devfs   -     no       -
/proc      -        /proc     proc    -     no       -
...
sharefs    -        /sharefs  samfs   -     no
```

4. 클라이언트에서 파일 시스템을 공유 파일 시스템으로 마운트하려면 공유 파일 시스템에 대해 *vfstab* 항목의 *Mount Options* 열에 *shared* 옵션을 입력합니다.

   현재 클라이언트에서 공유 파일 시스템 *sharefs*를 읽기 전용으로 마운트하려면 아래 예제에 표시된 대로 *vfstab* 항목을 편집합니다.

```
#File
#Device    Device   Mount     System  fsck  Mount    Mount
#to Mount  to fsck  Point     Type    Pass  at Boot  Options
#--------  -------  --------  ------  ----  -------  ------------------------
/devices   -        /devices  devfs   -     no       -
/proc      -        /proc     proc    -     no       -
...
sharefs    -        /sharefs  samfs   -     no       shared
```

5. 콤마를 구분자로 사용하여 다른 원하는 마운트 옵션을 추가하고 */etc/vfstab* 파일에 다른 원하는 변경을 수행합니다. 그런 다음 */etc/vfstab* 파일을 저장합니다.

   예제에서는 마운트 옵션을 추가하지 않습니다.

```
#File
#Device    Device   Mount     System  fsck  Mount    Mount
#to Mount  to fsck  Point     Type    Pass  at Boot  Options
#--------  -------  --------  ------  ----  -------  -------------------------
/devices   -        /devices  devfs   -     no       -
/proc      -        /proc     proc    -     no       -
...
sharefs    -        /sharefs  samfs   -     no       shared
:wq
[sharefs-client1]root@solaris:~#
```

6. */etc/vfstab* 파일에 지정된 마운트 지점을 만들고 마운트 지점에 대한 액세스 권한을 설정합니다.

   마운트 지점 권한은 메타데이터 서버와 모든 다른 클라이언트에서 동일해야 합니다. 사용자가 마운트 지점 디렉토리를 변경하고 마운트된 파일 시스템의 파일에 액세스하려면

실행(*x*) 권한이 있어야 합니다. 예제에서는 */sharefs* 마운트 지점 디렉토리를 만들고 *755*(*-rwxr-xr-x*)로 권한을 설정합니다.

```
[sharefs-client1]root@solaris:~# mkdir /sharefs
[sharefs-client1]root@solaris:~# chmod 755 /sharefs
[sharefs-client1]root@solaris:~#
```

7. 공유 파일 시스템을 마운트합니다.

```
[sharefs-client1]root@solaris:~# mount /sharefs
[sharefs-client1]root@solaris:~#
```

8. 잠재적 메타데이터 서버 호스트를 분산 테이프 I/O datamover로 추가하는 경우 "분산 테이프 I/O를 위한 Datamover 클라이언트 구성 "으로 이동합니다.
9. 여기서 중지합니다.

Linux 클라이언트 호스트에서 공유 파일 시스템 구성

1. Linux 클라이언트에서 *root*로 로그인합니다.

   예제에서 Oracle HSM 공유 파일 시스템은 *sharefs*이고, 호스트는 *sharefs-clientL*이라는 이름의 Linux 클라이언트입니다.

```
[sharefs-clientL][root@linux ~]#
```

2. 단말기 창에서 *samfsconfig device-path* 명령을 사용하여 공유 파일 시스템에 대한 구성 정보를 검색합니다. 여기서 *device-path*는 파일 시스템 디스크 장치를 검색하기 위해 명령을 시작할 위치입니다(예: */dev/**).

   Linux 호스트는 파일 시스템의 메타데이터 장치에 대한 액세스 권한이 없으므로, *samfsconfig* 명령이 메타데이터 장치를 찾을 수 없으며, 따라서 발견된 Oracle HSM 장치가 파일 시스템 구성에 맞지 않을 수 있습니다. 명령 출력은 *Missing Slices* 아래에 메타데이터 장치 *Ordinal 0*을 나열하고, 파일 시스템 패밀리 세트를 식별하는 라인을 포함하지 못하며, 데이터 장치의 목록을 주석 처리합니다.

   예제에서 Linux 호스트 *sharefs-clientL*에 대한 *samfsconfig* 출력은 다음과 비슷합니다.

```
[sharefs-clientL][root@linux ~]# samfsconfig /dev/*
# Family Set 'sharefs' Created Thu Feb 21 07:17:00 2013
#
# Missing slices
# Ordinal 0
# /dev/sda4            302         mr         sharefs   -
# /dev/sda5            303         mr         sharefs   -
```

3. *samfsconfig* 출력에서 공유 파일 시스템 항목을 복사합니다. 그리고 두번째 창에서, 텍스트 편집기에서 */etc/opt/SUNWsamfs/mcf* 파일을 열고 복사한 항목을 파일로 붙여 넣습니다.

   예제에서 Linux 호스트 *sharefs-clientL*에 대한 *mcf* 파일의 시작은 다음과 비슷합니다.

```
[sharefs-clientL][root@linux ~]# vi /etc/opt/SUNWsamfs/mcf
# Equipment           Equipment  Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#------------------   ---------  ---------  ---------  ------   -------------
# /dev/sda4             302        mr         sharefs   -
# /dev/sda5             303        mr         sharefs   -
```

4. *mcf* 파일에서 공유 파일 시스템 행을 삽입하고 *shared* 매개변수를 포함합니다.

```
# Equipment           Equipment  Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#------------------   ---------  ---------  ---------  ------   -------------
sharefs               300        ma         sharefs    -        shared
# /dev/sda4             302         mr         sharefs   -
# /dev/sda5             303         mr         sharefs   -
```

5. *mcf* 파일에서 파일 시스템의 메타데이터 장치 행을 삽입합니다. Linux 호스트는 메타데이터 장치에 액세스할 수 없으므로 *Equipment Identifier* 필드를 *nodev*(장치 없음)로 설정하고 남은 필드를 메타데이터 서버에 지정한 것과 똑같은 값으로 설정합니다.

```
# Equipment           Equipment  Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#------------------   ---------  ---------  ---------  ------   -------------
sharefs               300         ma        sharefs    on       shared
nodev                 301         mm        sharefs    on
# /dev/sda4             302         mr         sharefs   -
# /dev/sda5             303         mr         sharefs   -
```

6. *mcf* 파일에서 Linux 데이터 장치에 대한 항목의 주석 처리를 해제합니다.

```
# Equipment           Equipment  Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#------------------   ---------  ---------  ---------  ------   -------------
sharefs               300        ma         sharefs    on       shared
nodev                 301        mm         sharefs    on
/dev/sda4             302        mr         sharefs    -
/dev/sda5             303        mr         sharefs    -
```

7. 모든 장치에 대해 *Device State* 필드가 *on*으로 설정되었는지 확인하고 *mcf* 파일을 저장합니다.

```
# Equipment           Equipment  Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#------------------   ---------  ---------  ---------  ------   -------------
sharefs               300        ma         sharefs    on       shared
nodev                 301        mm         sharefs    on
/dev/sda4             302        mr         sharefs    on
/dev/sda5             303        mr         sharefs    on
:wq
[sharefs-clientL][root@linux ~]#
```

8. *sam-fsd* 명령을 실행하여 *mcf* 파일에 오류가 있는지 확인하고, 오류가 발견되면 수정합니다.

   *sam-fsd*는 Oracle HSM 구성 파일을 읽는 초기화 명령입니다. 만일 오류가 발생하면 중지됩니다. 예제에서는 Linux 클라이언트 *sharefs-clientL*에서 *mcf* 파일을 확인합니다.

```
[sharefs-clientL][root@linux ~]# sam-fsd
```

9. 이제 Linux 호스트에서 공유 파일 시스템 마운트를 수행합니다.

Linux 클라이언트 호스트에서 공유 파일 시스템 마운트

1. Linux 클라이언트에서 *root*로 로그인합니다.

   예제에서 Oracle HSM 공유 파일 시스템은 *sharefs*이고, 호스트는 *sharefs-clientL*이라는 이름의 Linux 클라이언트입니다.

```
[sharefs-clientL][root@linux ~]#
```

2. 운영체제의 */etc/fstab* 파일을 백업합니다.

```
[sharefs-clientL][root@linux ~]# cp /etc/fstab /etc/fstab.backup
```

3. 텍스트 편집기에서 */etc/fstab* 파일을 열고 공유 파일 시스템 행을 시작합니다.

   예제에서는 *vi* 텍스트 편집기를 사용하고 *sharefs* 패밀리 세트 장치에 대한 행을 추가합니다.

```
[sharefs-clientL][root@linux ~]# vi /etc/fstab
#File
```

```
#Device    Mount    System    Mount                      Dump      Pass
#to Mount  Point    Type      Options                    Frequency Number
#--------  -------  --------  -------------------------  --------- ------
...
/proc      /proc     proc     defaults
sharefs    /sharefs  samfs
```

4. 파일의 네번째 열에서 필수 *shared* 마운트 옵션을 추가합니다.

```
[sharefs-clientL][root@linux ~]# vi /etc/fstab
#File
#Device    Mount    System    Mount                      Dump      Pass
#to Mount  Point    Type      Options                    Frequency Number
#--------  -------  --------  -------------------------  --------- ------
...
/proc      /proc     proc     defaults
sharefs    /sharefs  samfs    shared
```

5. 파일의 네번째 열에서 콤마를 구분자로 사용하여 다른 원하는 마운트 옵션을 추가합니다.

   Linux 클라이언트는 다음 추가 마운트 옵션을 지원합니다.

   - *rw*, *ro*
   - *retry*
   - *meta_timeo*
   - *rdlease*, *wrlease*, *aplease*
   - *minallocsz*, *maxallocsz*
   - *noauto*, *auto*

   예제에서는 *noauto* 옵션을 추가합니다.

```
#File
#Device    Mount    System    Mount                      Dump      Pass
#to Mount  Point    Type      Options                    Frequency Number
#--------  -------  --------  -------------------------  --------- ------
...
/proc      /proc     proc     defaults
sharefs    /sharefs  samfs    shared,noauto
```

6. 파일에 남은 두 열에 각각 제로(*0*)를 입력합니다. 그런 다음 */etc/fstab* 파일을 저장합니다.

```
#File
#Device    Mount    System    Mount                      Dump      Pass
```

```
#to Mount  Point    Type      Options                    Frequency Number
#--------  -------  --------  -------------------------  --------- ------
...
/proc      /proc     proc     defaults
sharefs    /sharefs  samfs    shared,noauto              0         0
:wq
[sharefs-clientL][root@linux ~]#
```

7. */etc/fstab* 파일에 지정된 마운트 지점을 만들고 마운트 지점에 대한 액세스 권한을 설정합니다.

   마운트 지점 권한은 메타데이터 서버와 모든 다른 클라이언트에서 동일해야 합니다. 사용자가 마운트 지점 디렉토리를 변경하고 마운트된 파일 시스템의 파일에 액세스하려면 실행(*x*) 권한이 있어야 합니다. 예제에서는 */sharefs* 마운트 지점 디렉토리를 만들고 *755*(*-rwxr-xr-x*)로 권한을 설정합니다.

```
[sharefs-clientL][root@linux ~]# mkdir /sharefs
[sharefs-clientL][root@linux ~]# chmod 755 /sharefs
```

8. 공유 파일 시스템을 마운트합니다. *mount mountpoint* 명령을 사용합니다. 여기서 *mountpoint*는 */etc/fstab* 파일에 지정된 마운트 지점 디렉토리입니다.

   예제에 표시된 대로 *mount* 명령은 경고를 생성합니다. 이는 정상적 상황이며 무시할 수 있습니다.

```
[sharefs-clientL][root@linux ~]# mount /sharefs
Warning: loading SUNWqfs will taint the kernel: SMI license
See http://www.tux.org/lkml/#export-tainted for information
about tainted modules. Module SUNWqfs loaded with warnings
```

9. 여기서 중지합니다.

공유 파일 시스템 구성에서 호스트 제거

공유 파일 시스템에서 호스트를 제거하면 아래 설명된 대로 단순히 서버 구성에서 호스트를 제거하는 것입니다. 호스트 구성을 완전히 해제하려면 소프트웨어와 구성 파일을 제거하십시오.

파일 시스템 호스트 파일에서 호스트 제거

1. Oracle HSM 메타데이터 서버에 *root*로 로그인합니다.

   예제에서 Oracle HSM 공유 파일 시스템은 *sharefs*, 메타데이터 서버 호스트는 *sharefs-mds*입니다.

```
[sharefs-mds]root@solaris:~#
```

2. 각 클라이언트에 *root*로 로그인하고 공유 파일 시스템을 마운트 해제합니다.

   잠재적 메타데이터 서버도 클라이언트임을 잊지 마십시오. 예제에서는 *sharefs-client1*, *sharefs-client2* 및 *sharefs-mds_alt*라는 잠재적 메타데이터 서버를 포함한 세 개의 클라이언트가 있습니다. 각 클라이언트에 대해 *ssh*를 사용해서 로그인하고 파일 시스템 *sharefs*를 마운트 해제하고 *ssh* 세션을 닫습니다.

```
[sharefs-mds]root@solaris:~# ssh root@sharefs-client1
Password:
[sharefs-client1]root@solaris:~# umount sharefs
[sharefs-client1]root@solaris:~# exit
[sharefs-mds]root@solaris:~# ssh root@sharefs-client2
Password:
[sharefs-client2]root@solaris:~# umount sharefs
[sharefs-client2]root@solaris:~# exit
[sharefs-mds]root@solaris:~# ssh root@sharefs-mds_alt
Password:
[sharefs-mds_alt]root@solaris:~# umount sharefs
root@solaris:~# exit
[sharefs-mds]root@solaris:~#
```

3. 메타데이터 서버에서 공유 파일 시스템을 마운트 해제합니다.

```
[sharefs-mds]root@solaris:~# umount sharefs
```

4. 메타데이터 서버에서 */etc/opt/SUNWsamfs/hosts.filesystem* 파일의 이름을 */etc/opt/SUNWsamfs/hosts.filesystem.bak*로 바꿉니다. 여기서 *filesystem*은 클라이언트 호스트를 제거하려는 파일 시스템의 이름입니다.

   아래의 명령은 단일 라인으로 입력됩니다. 줄바꿈은 백슬래시 문자로 이스케이프됩니다.

```
[sharefs-mds]root@solaris:~# mv /etc/opt/SUNWsamfs/hosts.sharefs /
/etc/opt/SUNWsamfs/hosts.sharefs.bak
```

5. 현재 공유 파일 시스템 호스트 구성을 파일에 캡처합니다. 메타데이터 서버에서 *samsharefs -R filesystem* 명령을 실행하여 */etc/opt/SUNWsamfs/hosts.filesystem* 파일로 출력이 재지정되도록 합니다. 여기서 *filesystem*은 클라이언트 호스트를 추가하려는 파일 시스템의 이름입니다.

   samsharefs 명령은 지정된 Oracle HSM 공유 파일 시스템에 대한 호스트 구성을 표시합니다. 출력을 파일로 재지정하면 새 hosts 파일이 만들어집니다. 예제에서는 메타데이터 서버 *sharefs-mds*에서 명령을 실행합니다.

```
[sharefs-mds]root@solaris:~# samsharefs -R sharedqfs1 > /
/etc/opt/SUNWsamfs/hosts.sharedqfs1
```

6. 새로 만든 hosts 파일을 텍스트 편집기에서 엽니다.

   예제에서는 *vi* 편집기를 사용합니다. *sharefs-client3* 클라이언트를 제거해야 합니다.

```
[sharefs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sharefs
#                                          Server   On/  Additional
#Host Name            Network Interface     Ordinal  Off  Parameters
#------------------   --------------------  -------  ---  ----------
sharefs-mds          10.79.213.117         1        0    server
sharefs-mds_alt      10.79.213.217         2        0
sharefs-client1      10.79.213.133         0        0
sharefs-client2      10.79.213.47          0        0
sharefs-client3      10.79.213.49          0        0
```

7. hosts 파일에서 제거할 클라이언트 호스트에 해당하는 행을 삭제합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   예제에서는 *sharefs-client3* 호스트 항목을 삭제합니다.

```
[sharefs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sharefs
#                                          Server   On/  Additional
#Host Name            Network Interface     Ordinal  Off  Parameters
#------------------   --------------------  -------  ---  ----------
sharefs-mds          10.79.213.117         1        0    server
sharefs-mds_alt      10.79.213.217         2        0
sharefs-client1      10.79.213.133         0        0
sharefs-client2      10.79.213.47          0        0
:wq
[sharefs-mds]root@solaris:~#
```

8. 수정된 hosts 파일을 사용해서 파일 시스템을 업데이트합니다. 메타데이터 서버에서 *samsharefs -R -u filesystem* 명령을 사용합니다. 여기서 *filesystem*은 클라이언트 호스트를 제거하려는 파일 시스템의 이름입니다.

```
[sharefs-mds]root@solaris:~# samsharefs -u sharefs
```

9. 메타데이터 서버 호스트에서 공유 파일 시스템을 마운트합니다.

   예제에서는 */etc/vfstab* 파일에 *sharefs* 파일 시스템 항목이 포함되므로 간단한 마운팅 구문을 사용합니다. 자세한 내용은 *mount_samfs* 매뉴얼 페이지를 참조하십시오.

```
[sharefs-mds]root@solaris:~# mount sharefs
```

10. 각 클라이언트 호스트에서 공유 파일 시스템을 마운트합니다.

    잠재적 메타데이터 서버도 클라이언트임을 잊지 마십시오. 예제에서는 *sharefs-client1*, *sharefs-client2* 및 *sharefs-mds_alt*라는 잠재적 메타데이터 서버를 포함한 세 개의 클라이언트가 있습니다. 각 클라이언트에 대해 *ssh*를 사용해서 로그인하고 파일 시스템 *sharefs*를 마운트 해제하고 *ssh* 세션을 닫습니다.

```
[sharefs-mds]root@solaris:~# ssh root@sharefs-mds_alt
Password:
[sharefs-mds_alt]root@solaris:~# mount sharefs
sharefs-mds_alt]root@solaris:~# exit
[sharefs-mds]root@solaris:~# ssh root@sharefs-client1
Password:
[sharefs-client1]root@solaris:~# mount sharefs
sharefs-client1]root@solaris:~# exit
[sharefs-mds]root@solaris:~# ssh root@sharefs-client2
Password:
[sharefs-client2]root@solaris:~# mount sharefs
sharefs-client2]root@solaris:~# exit
[sharefs-mds]root@solaris:~#
```

11. 여기서 중지합니다.

분산 테이프 I/O를 위한 Datamover 클라이언트 구성

Oracle HSM 릴리스 6.1부터 Solaris 11 이상에서 실행되는 공유 아카이빙 파일 시스템의 클라이언트는 테이프 드라이브를 연결하여 파일 시스템 대신 테이프 I/O를 수행할 수 있습니다. 이러한 *datamover* 호스트에 걸쳐 테이프 I/O를 분산시키면 서버 오버헤드가 크게 감소하고, 파일 시스템 성능이 향상되고, Oracle HSM 구현을 확장할 때 유연성이 증대됩니다. 사용자의 아카이빙 수요가 증가함에 따라 이제 Oracle HSM 메타데이터 서버를 더 강력한 시스템으로 교체하거나(수직적 확장) 더 많은 클라이언트에 걸쳐 로드를 분산시킬 수 있습니다(수평적 확장).

Datamover 클라이언트 구성

분산 테이프 I/O를 위한 클라이언트를 구성하려면 다음과 같이 하십시오.

1. 분산 I/O에 사용할 모든 장치를 클라이언트에 연결합니다.
2. 아직 수행하지 않은 경우 "지속 바인딩을 사용하여 테이프 드라이브 연결" 절차를 수행합니다. 그런 다음 여기로 돌아옵니다.
3. 공유 아카이빙 파일 시스템의 메타데이터 서버에 *root*로 로그인합니다.

   예제에서 호스트 이름은 *samsharefs-mds*입니다.

```
[samsharefs-mds]root@solaris:~#
```

4. 메타데이터 서버가 Oracle HSM Solaris 11 이상을 실행 중인지 확인합니다.

```
[samsharefs-mds]root@solaris:~# uname -r
5.11
[samsharefs-mds]root@solaris:~#
```

5. datamover로 작동하는 모든 클라이언트가 Oracle HSM Solaris 11 이상을 실행 중인지 확인합니다.

   예제에서는 각 클라이언트 호스트인 *samsharefs-client1* 및 *samsharefs-client2*에 대해 터미널 창을 열고 *ssh*를 사용해서 원격으로 로그인합니다. 로그인 배너에는 Solaris 버전이 표시됩니다.

```
[samsharefs-mds]root@solaris:~# ssh root@samsharefs-client1
...
Oracle Corporation      SunOS 5.11      11.1    September 2013
[samsharefs-client1]root@solaris:~#

[samsharefs-mds]root@solaris:~# ssh root@samsharefs-client2
...
Oracle Corporation      SunOS 5.11      11.1    September 2013
[samsharefs-client2]root@solaris:~#
```

6. 메타데이터 서버에서 텍스트 편집기로 */etc/opt/SUNWsamfs/defaults.conf*를 열고, *distio =* 라인의 주석 처리를 해제하고 값을 *on*으로 설정하여 분산 I/O를 사용으로 설정합니다.

   기본적으로 *distio*는 *off*(사용 안함)입니다.

   예제에서는 *vi* 편집기에서 파일을 열고 다음 라인을 추가합니다.

```
[samsharefs-mds]root@solaris:~#  vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults.  To change the default behavior, uncomment the
# appropriate line (remove the '#' character from the beginning of the line)
# and change the value.
...
distio = on
```

7. 그런 다음, 분산 I/O에 참여해야 하는 장치 유형을 식별합니다. 분산 I/O에 *dev* 장치 유형을 사용하려면 *defaults.conf* 파일에 *dev_distio = on* 행을 추가합니다. 분산 I/O에서 *dev* 장치 유형을 제외하려면 *dev_distio = off* 행을 추가합니다. 파일을 저장하고 편집기를 닫습니다.

기본적으로 Oracle HSM T10000 드라이브 및 LTO 드라이브는 분산 I/O 참여가 허용되며(*ti_distio = on* 및 *li_distio = on*), 그 밖의 다른 유형은 제외됩니다. 예제에서는 LTO 드라이브를 제외합니다.

```
[samsharefs-mds]root@solaris:~#  vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults.  To change the default behavior, uncomment the
# appropriate line (remove the '#' character from the beginning of the line)
# and change the value.
...
distio = on
li_distio = off
:wq
[samsharefs-mds]root@solaris:~#
```

8. datamover로 작동할 각 클라이언트에서 *defaults.conf* 파일을 편집하여 서버의 파일과 일치하도록 합니다.

   예제에서는 *samsharefs-client1* 클라이언트에서 *vi*를 사용하여 *defaults.conf* 파일을 편집하고, 파일을 저장하고, 편집기를 닫습니다.

```
[samsharefs-mds]root@solaris:~# ssh root@samsharefs-client1
Password:
[samsharefs-client1]root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults.  To change the default behavior, uncomment the
# appropriate line (remove the '#' character from the beginning of the line)
# and change the value.
...
distio = on
li_distio = off
:wq
[samsharefs-client1]root@solaris:~#
[samsharefs-mds]root@solaris:~#
```

9. datamover로 작동할 각 클라이언트에서 텍스트 편집기에서 */etc/opt/SUNWsamfs/mcf* 파일을 엽니다. 메타데이터 서버가 분산 테이프 I/O에 사용 중인 테이프 장치를 모두 추가합니다. 장치 순서 및 장비 번호가 메타데이터 서버의 *mcf* 파일과 동일한지 확인합니다.

   예제에서는 *samsharefs-client1* 클라이언트에서 *vi*를 사용하여 *mcf* 파일을 편집합니다.

```
[samsharefs-client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment              Equipment Equipment Family      Device Additional
# Identifier             Ordinal   Type      Set         State  Parameters
```

```
#----------------------- --------- --------- ----------  ------ -------------
samsharefs               800       ms        samsharefs  on
...
# Archival storage for copies:
 /dev/rmt/60cbn          901       ti                    on
 /dev/rmt/61cbn          902       ti                    on
 /dev/rmt/62cbn          903       ti                    on
 /dev/rmt/63cbn          904       ti                    on
```

10. 메타데이터 서버의 */etc/opt/SUNWsamfs/mcf* 파일에 나열된 테이프 라이브러리가 datamover로 작동할 클라이언트에 구성된 경우, 분산 테이프 I/O에 사용 중인 테이프 장치의 패밀리 세트 이름으로 라이브러리 패밀리 세트를 지정합니다. 파일을 저장합니다.

    예제에서는 라이브러리가 호스트에 구성되었으므로 테이프 장치에 패밀리 세트 이름 *library1*을 사용합니다.

```
[samsharefs-client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment              Equipment Equipment Family      Device Additional
# Identifier             Ordinal   Type      Set         State  Parameters
#----------------------- --------- --------- ----------  ------ -------------
samsharefs               800       ms        samsharefs  on
...
# Archival storage for copies:
/dev/scsi/changer/c1t0d5 900       rb        library1    on
 /dev/rmt/60cbn           901       ti        library1    on
 /dev/rmt/61cbn           902       ti        library1    on
 /dev/rmt/62cbn           903       ti        library1    on
 /dev/rmt/63cbn           904       ti        library1    on
:wq
[samsharefs-client1]root@solaris:~#
```

11. 메타데이터 서버의 */etc/opt/SUNWsamfs/mcf* 파일에 나열된 테이프 라이브러리가 datamover로 작동할 클라이언트에 구성되지 않은 경우, 분산 테이프 I/O에 사용 중인 테이프 장치의 패밀리 세트 이름으로 하이픈(-)을 사용합니다.

    예제에서는 라이브러리가 호스트에 구성되어 있지 않습니다.

```
[samsharefs-client2]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment              Equipment Equipment Family      Device Additional
# Identifier             Ordinal   Type      Set         State  Parameters
#----------------------- --------- --------- ----------  ------ -------------
samsharefs               800       ms        samsharefs  on
...
# Archival storage for copies:
/dev/rmt/60cbn           901       ti        -           on
```

```
/dev/rmt/61cbn           902       ti        -          on
/dev/rmt/62cbn           903       ti        -          on
/dev/rmt/63cbn           904       ti        -          on
:wq
[samsharefs-client2]root@solaris:~#
```

12. 특정 아카이브 세트 복사본에 대해 분산 테이프 I/O를 사용 또는 사용 안함으로 설정하려면 텍스트 편집기에서 서버의 */etc/opt/SUNWsamfs/archiver.cmd* 파일을 열고 copy 지시어에 *-distio* 매개변수를 추가합니다. 분산 I/O를 사용으로 설정하려면 *-distio on*, 사용 안함으로 설정하려면 *off*를 지정합니다. 파일을 저장하고 편집기를 닫습니다.

    예제에서는 vi 편집기를 사용하여 복사본 *1*에 대해 분산 I/O를 *off*로 설정하고 복사본 *2*에 대해 *on*으로 설정합니다.

```
[samsharefs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/archiver.cmd
# archiver.cmd
# Generated by config api Mon Nov 22 14:31:39 2013
...
#
# Copy Parameters Directives
params
allsets -sort path -offline_copy stageahead
allsets.1 -startage 10m -startsize 500M -startcount 500000 -distio off
allsets.2 -startage 24h -startsize 20G  -startcount 500000 -distio on
:wq
[samsharefs-mds]root@solaris:~#
```

13. 각 호스트에서 *sam-fsd* 명령을 실행하여 *mcf* 파일 오류를 확인하고 오류가 발견되었으면 수정합니다.

    *sam-fsd*는 Oracle HSM 구성 파일을 읽는 초기화 명령입니다. 만일 오류가 발생하면 중지됩니다. 예제에서는 Linux 클라이언트 *sharefs-clientL*에서 *mcf* 파일을 확인합니다.

```
[sharefs-clientL][root@linux ~]# sam-fsd
```

14. 서버에서 Oracle HSM 소프트웨어가 수정된 구성 파일을 읽고 그에 따라 재구성하도록 지시합니다. *samd config* 명령을 사용하고 발견된 오류를 수정합니다.

    예제에서는 *sharefs-mds* 서버에서 *samd config* 명령을 실행합니다.

```
[samsharefs-mds]root@solaris:~#  samd config
```

15. 여기서 중지합니다.

지속 바인딩을 사용하여 테이프 드라이브 연결

잠재적 메타데이터 서버 또는 분산 I/O datamover 클라이언트로 작동하는 호스트를 추가할 때는 지속 바인딩을 사용해서 이동식 매체 장치를 구성해야 합니다. Solaris 운영체제는 시작 시 장치가 검색된 순서에 따라 시스템 장치 트리에 드라이브를 연결합니다. 이 순서는 다른 파일 시스템 호스트에서 장치를 발견한 순서나 테이프 라이브러리에 장치가 물리적으로 설치된 순서를 반영할 수도 있고 그렇지 않을 수도 있습니다. 따라서 다른 호스트에 바인딩된 것과 동일한 방법 및 이동식 매체 라이브러리에 설치된 것과 동일한 순서로 새 호스트에 장치를 바인딩해야 합니다.

아래 절차는 필요한 단계를 간략히 설명합니다. 자세한 내용은 *devfsadm* 및 *devlinks* 매뉴얼 페이지와 Solaris 운영체제 버전의 관리 설명서를 참조하십시오.

- 라이브러리에서 드라이브를 이동, 추가, 제거했거나 아카이빙 Oracle HSM 공유 파일 시스템과 연관된 라이브러리를 교체하거나 재구성한 경우 변경사항이 반영되도록 지속 바인딩 업데이트를 수행합니다.
- 새 메타데이터 서버나 datamover 클라이언트를 아카이빙 Oracle HSM 공유 파일 시스템에 추가하는 경우 새 파일 시스템 호스트를 이동식 매체 장치에 지속적으로 바인드를 수행합니다.

하드웨어 구성 변경사항이 반영되도록 지속 바인딩 업데이트

1. 활성 메타데이터 서버 호스트에 *root*로 로그인합니다.

   ```
   [sharefs-mds]root@solaris:~#
   ```

2. "라이브러리에 드라이브가 설치되는 순서 결정"에 설명된 대로 새 드라이브 매핑 파일을 만듭니다.

   예제에서 *device-mappings.txt* 파일은 다음과 비슷합니다.

   ```
   [sharefs-mds]root@solaris:~# vi /root/device-mappings.txt
   LIBRARY SOLARIS          SOLARIS
   DEVICE  LOGICAL          PHYSICAL
   NUMBER  DEVICE           DEVICE
   ------- -------------    ----------------------------------------------------
      2    /dev/rmt/0cbn -> ../../devices/pci@8.../st@w500104f00093c438,0:cbn
      1    /dev/rmt/1cbn -> ../../devices/pci@8.../st@w500104f0008120fe,0:cbn
      3    /dev/rmt/2cbn -> ../../devices/pci@8.../st@w500104f000c086e1,0:cbn
      4    /dev/rmt/3cbn ->  ../../devices/pci@8.../st@w500104f000b6d98d,0:cbn
   ```

3. 텍스트 편집기에서 */etc/devlink.tab* 파일을 엽니다.

   예제에서는 *vi* 편집기를 사용합니다:

```
[sharefs-mds]root@solaris:~# vi /etc/devlink.tab
# Copyright (c) 1993, 2011, Oracle and/or its affiliates. All rights reserved.
# This is the table used by devlinks
# Each entry should have 2 fields; but may have 3.  Fields are separated
# by single tab ('/t') characters.
...
```

4. *device-mappings.txt* 파일을 길잡이로 삼아 Solaris 테이프 장치 트리의 시작 노드를 라이브러리의 첫번째 드라이브에 재매핑합니다. */etc/devlink.tab* 파일에서 *type=ddi_byte:tape; addr=device_address,0; rmt/node-number/M0* 형식의 행을 추가합니다. 여기서 *device_address*는 장치의 물리적 주소이고 *node-number*는 Solaris 장치 트리에서 위치로, Solaris에서 자동으로 구성하는 장치와 충돌하지 않도록 충분히 높은 숫자를 선택합니다(Solaris는 노드 *0*부터 시작함).

   예제에서 라이브러리의 첫번째 장치 *1*에 대한 장치 주소는 *w500104f0008120fe*입니다. 장치가 현재 *rmt/1*에서 호스트에 연결되어 있습니다.

```
[sharefs-mds] vi /root/device-mappings.txt
LIBRARY SOLARIS          SOLARIS
DEVICE  LOGICAL          PHYSICAL
NUMBER  DEVICE           DEVICE
------- -------------    ----------------------------------------------------
   2    /dev/rmt/0cbn -> ../../devices/pci@8.../st@w500104f00093c438,0:cbn
   1    /dev/rmt/1cbn -> ../../devices/pci@8.../st@w500104f0008120fe,0:cbn
   3    /dev/rmt/2cbn -> ../../devices/pci@8.../st@w500104f000c086e1,0:cbn
   4    /dev/rmt/3cbn -> ../../devices/pci@8.../st@w500104f000b6d98d,0:cbn
```

   따라서 */etc/devlink.tab*에 행을 만들어서 라이브러리의 번호 *1* 드라이브 *w500104f0008120fe*에 *rmt/60*을 재매핑합니다.

```
[sharefs-mds]root@solaris:~# vi /etc/devlink.tab
# Copyright (c) 1993, 2011, Oracle and/or its affiliates. All rights reserved.
...
type=ddi_byte:tape;addr=w500104f0008120fe,0;    rmt/60/M0
:w
```

5. Oracle HSM 아카이빙을 위해 지정된 각 테이프 장치마다 */etc/devlink.tab* 파일에 행을 계속 추가합니다. 메타데이터 서버의 장치 트리의 드라이브 순서가 라이브러리의 설치 순서와 일치하도록 합니다. 파일을 저장하고 편집기를 닫습니다.

   예제에서는 세 개의 남은 장치의 순서와 주소로 라이브러리 드라이브 *2*가 *w500104f00093c438*에, 라이브러리 드라이브 *3*이 *w500104f000c086e1*에, 라이브러리 드라이브 *4*가 *w500104f000c086e1*에 있습니다.

```
[sharefs-mds]root@solaris:~# vi /root/device-mappings.txt
...
   2    /dev/rmt/0cbn -> ../../devices/pci@8/.../st@w500104f00093c438,0:cbn
   1    /dev/rmt/1cbn -> ../../devices/pci@8/.../st@w500104f0008120fe,0:cbn
   3    /dev/rmt/2cbn -> ../../devices/pci@8/.../st@w500104f000c086e1,0:cbn
   4    /dev/rmt/3cbn -> ../../devices/pci@8/.../st@w500104f000b6d98d,0:cbn
```

그리고 다음 세 개의 Solaris 장치 노드에 장치 주소를 매핑하여 라이브러리와 동일한 순서가 유지되도록 합니다.

```
[sharefs-mds]root@solaris:~# vi /etc/devlink.tab
...
type=ddi_byte:tape;addr=w500104f0008120fe,0;    rmt/60/M0
type=ddi_byte:tape;addr=w500104f00093c438,0;    rmt/61/M0
type=ddi_byte:tape;addr=w500104f000c086e1,0;    rmt/62/M0
type=ddi_byte:tape;addr=w500104f000b6d98d,0;    rmt/63/M0
:wq
[sharefs-mds]root@solaris:~#
```

6. */dev/rmt*의 테이프 장치에 대한 기존 링크를 모두 삭제합니다.

   ```
   [sharefs-mds]root@solaris:~# rm /dev/rmt/*
   ```

7. */etc/devlink.tab* 파일의 항목으로부터 새로운 지속 테이프 장치 링크를 만듭니다. *devfsadm -c tape* 명령을 사용합니다.

   devfsadm 명령을 실행할 때마다 */etc/devlink.tab* 파일에 지정된 구성을 사용하여 파일에 지정된 장치에 대해 새 테이프 장치 링크를 만듭니다. *-c tape* 옵션은 테이프 종류의 장치에만 새 링크를 만들도록 명령을 제한합니다.

   ```
   [sharefs-mds]root@solaris:~# devfsadm -c tape
   ```

8. 공유 파일 시스템 구성에 있는 각 잠재적 메타데이터 서버 및 datamover에서 작업을 반복합니다. 각각의 경우에 */etc/devlink.tab* 파일에 동일한 라인을 추가하고, */dev/rmt*에서 링크를 삭제하고, *devfsadm -c tape*를 실행합니다.

   예제에서는 *ssh*를 사용해서 각 호스트에 교대로 로그인하고, 4개의 동일한 논리적 장치인 *rmt/60/M0*, *rmt/61/M0*, *rmt/62/M0* 및 *rmt/63/M0*을 구성합니다.

   ```
   [sharefs-mds]root@solaris:~# ssh root@sharefs-mds_alt
   Password:
   [sharefs-mds_alt]root@solaris:~# vi /etc/devlink.tab
   ...
   type=ddi_byte:tape;addr=w500104f0008120fe,0;    rmt/60/M0
   ```

```
type=ddi_byte:tape;addr=w500104f00093c438,0;    rmt/61/M0
type=ddi_byte:tape;addr=w500104f000c086e1,0;    rmt/62/M0
type=ddi_byte:tape;addr=w500104f000b6d98d,0;    rmt/63/M0
:wq
[sharefs-mds_alt]root@solaris:~# rm /dev/rmt/*
[sharefs-mds_alt]root@solaris:~# devfsadm -c tape
[sharefs-mds_alt]root@solaris:~# exit
sharefs-mds]root@solaris:~# ssh root@sharefs-client1
Password:
[sharefs-client1]root@solaris:~# vi /etc/devlink.tab
...
type=ddi_byte:tape;addr=w500104f0008120fe,0;    rmt/60/M0
type=ddi_byte:tape;addr=w500104f00093c438,0;    rmt/61/M0
type=ddi_byte:tape;addr=w500104f000c086e1,0;    rmt/62/M0
type=ddi_byte:tape;addr=w500104f000b6d98d,0;    rmt/63/M0
:wq
[sharefs-client1]root@solaris:~# rm /dev/rmt/*
[sharefs-client1]root@solaris:~# devfsadm -c tape
[sharefs-client1]root@solaris:~# exit
[sharefs-mds]root@solaris:~#
```

9. "분산 테이프 I/O를 위한 Datamover 클라이언트 구성 " 또는 "추가 파일 시스템 클라이언트 구성"으로 돌아갑니다.

새 파일 시스템 호스트를 이동식 매체 장치에 지속적으로 바인드

1. 호스트에 *root*로 로그인합니다.

```
[sharefs-mds]root@solaris:~#
```

2. 기존 파일 시스템 호스트가 구성된 이후 매체 라이브러리에서 드라이브의 물리적 순서가 변경된 경우 "라이브러리에 드라이브가 설치되는 순서 결정"에 설명된 대로 새 매핑 파일을 만듭니다.

   예제에서 *device-mappings.txt* 파일은 다음과 비슷합니다.

```
[sharefs-mds]root@solaris:~# vi /root/device-mappings.txt
LIBRARY SOLARIS          SOLARIS
DEVICE  LOGICAL          PHYSICAL
NUMBER  DEVICE           DEVICE
------- -------------    ------------------------------------------------------
   2    /dev/rmt/0cbn -> ../../devices/pci@8.../st@w500104f00093c438,0:cbn
   1    /dev/rmt/1cbn -> ../../devices/pci@8.../st@w500104f0008120fe,0:cbn
   3    /dev/rmt/2cbn -> ../../devices/pci@8.../st@w500104f000c086e1,0:cbn
```

```
   4    /dev/rmt/3cbn -> ../../devices/pci@8.../st@w500104f000b6d98d,0:cbn
```

3. 테스트 편집기에서 */etc/devlink.tab* 파일을 엽니다.

   예제에서는 *vi* 편집기를 사용합니다.

```
[sharefs-mds]root@solaris:~# vi /etc/devlink.tab
# Copyright (c) 1993, 2011, Oracle and/or its affiliates. All rights reserved.
# This is the table used by devlinks
# Each entry should have 2 fields; but may have 3.  Fields are separated
# by single tab ('/t') characters.
...
```

4. *device-mappings.txt* 파일을 안내서로 사용해서 Solaris 테이프 장치 트리 *rmt/node-number*에서 시작 노드를 라이브러리의 첫번째 드라이브에 재매핑합니다. */etc/devlink.tab* 파일에 *type=ddi_byte:tape; addr=device_address,0; rmt/node-number/M0* 형식으로 라인을 추가합니다. 여기서 *device_address*는 장치의 물리적 주소이고 *node-number*는 Solaris 장치 트리에서 장치의 위치입니다. Solaris에서 자동으로 구성하는 장치와 충돌하지 않도록 충분히 높은 숫자의 노드 번호를 선택합니다(Solaris는 노드 *0*부터 시작함).

   예제에서 라이브러리의 첫번째 장치 *1*에 대한 장치 주소는 *w500104f0008120fe*입니다. 장치가 현재 *rmt/1*에서 호스트에 연결되어 있습니다.

```
[sharefs-mds] vi /root/device-mappings.txt
LIBRARY SOLARIS          SOLARIS
DEVICE  LOGICAL          PHYSICAL
NUMBER  DEVICE           DEVICE
------- -------------    ----------------------------------------------------
   2    /dev/rmt/0cbn -> ../../devices/pci@8.../st@w500104f00093c438,0:cbn
   1    /dev/rmt/1cbn -> ../../devices/pci@8.../st@w500104f0008120fe,0:cbn
   3    /dev/rmt/2cbn -> ../../devices/pci@8.../st@w500104f000c086e1,0:cbn
   4    /dev/rmt/3cbn -> ../../devices/pci@8.../st@w500104f000b6d98d,0:cbn
```

   따라서 */etc/devlink.tab*에 행을 만들어서 라이브러리의 번호 *1* 드라이브 *w500104f0008120fe*에 *rmt/60*을 재매핑합니다.

```
[sharefs-mds]root@solaris:~# vi /etc/devlink.tab
# Copyright (c) 1993, 2011, Oracle and/or its affiliates. All rights reserved.
...
type=ddi_byte:tape;addr=w500104f0008120fe,0;    rmt/60/M0
:w
```

5. Oracle HSM 아카이빙을 위해 지정된 각 테이프 장치마다 */etc/devlink.tab* 파일에 행을 계속 추가합니다. 메타데이터 서버의 장치 트리의 드라이브 순서가 라이브러리의 설치 순서와 일치하도록 합니다. 파일을 저장합니다.

   예제에서는 세 개의 남은 장치의 순서와 주소로 라이브러리 드라이브 *2*가 *w500104f00093c438*에, 라이브러리 드라이브 *3*이 *w500104f000c086e1*에, 라이브러리 드라이브 *4*가 *w500104f000c086e1*에 있습니다.

```
[sharefs-mds]root@solaris:~# vi /root/device-mappings.txt
...
   2    /dev/rmt/0cbn -> ../../devices/pci@8.../st@w500104f00093c438,0:cbn
   1    /dev/rmt/1cbn -> ../../devices/pci@8.../st@w500104f0008120fe,0:cbn
   3    /dev/rmt/2cbn -> ../../devices/pci@8.../st@w500104f000c086e1,0:cbn
   4    /dev/rmt/3cbn -> ../../devices/pci@8.../st@w500104f000b6d98d,0:cbn
```

   그리고 다음 세 개의 Solaris 장치 노드에 장치 주소를 매핑하여 라이브러리와 동일한 순서가 유지되도록 합니다.

```
[sharefs-mds]root@solaris:~# vi /etc/devlink.tab
...
type=ddi_byte:tape;addr=w500104f0008120fe,0;    rmt/60/M0
type=ddi_byte:tape;addr=w500104f00093c438,0;    rmt/61/M0
type=ddi_byte:tape;addr=w500104f000c086e1,0;    rmt/62/M0
type=ddi_byte:tape;addr=w500104f000b6d98d,0;    rmt/63/M0
:wq
[sharefs-mds]root@solaris:~#
```

6. */dev/rmt*의 테이프 장치에 대한 기존 링크를 모두 삭제합니다.

```
[sharefs-mds]root@solaris:~# rm /dev/rmt/*
```

7. */etc/devlink.tab* 파일의 항목으로부터 새로운 지속 테이프 장치 링크를 만듭니다. *devfsadm -c tape* 명령을 사용합니다.

   devfsadm 명령을 실행할 때마다 */etc/devlink.tab* 파일에 지정된 구성을 사용하여 파일에 지정된 장치에 대해 새 테이프 장치 링크를 만듭니다. *-c tape* 옵션은 테이프 종류의 장치에만 새 링크를 만들도록 명령을 제한합니다.

```
[sharefs-mds]root@solaris:~# devfsadm -c tape
```

8. 공유 파일 시스템 구성에 있는 각 잠재적 메타데이터 서버 및 datamover에서 */etc/devlink.tab* 파일에 동일한 라인을 추가하고, */dev/rmt*에서 링크를 삭제하고, *devfsadm -c tape*를 실행합니다.

예제에서는 *ssh*를 실행하여 잠재적 메타데이터 서버 호스트 *sharefs-mds_alt* 및 클라이언트 호스트 *sharefs-client1*에 로그인합니다. 그런 다음 동일한 4개의 논리적 장치 *rmt/60/M0*, *rmt/61/M0*, *rmt/62/M0* 및 *rmt/63/M0*을 각 항목에 구성합니다.

```
[sharefs-mds]root@solaris:~# ssh root@sharefs-mds_alt
Password:
[sharefs-mds_alt]root@solaris:~# vi /etc/devlink.tab
...
type=ddi_byte:tape;addr=w500104f0008120fe,0;    rmt/60/M0
type=ddi_byte:tape;addr=w500104f00093c438,0;    rmt/61/M0
type=ddi_byte:tape;addr=w500104f000c086e1,0;    rmt/62/M0
type=ddi_byte:tape;addr=w500104f000b6d98d,0;    rmt/63/M0
:wq
[sharefs-mds_alt]root@solaris:~# rm /dev/rmt/*
[sharefs-mds_alt]root@solaris:~# devfsadm -c tape
[sharefs-mds_alt]root@solaris:~# exit
[sharefs-mds]root@solaris:~# ssh root@sharefs-client1
Password:
[sharefs-client1]root@solaris:~# vi /etc/devlink.tab
...
type=ddi_byte:tape;addr=w500104f0008120fe,0;    rmt/60/M0
type=ddi_byte:tape;addr=w500104f00093c438,0;    rmt/61/M0
type=ddi_byte:tape;addr=w500104f000c086e1,0;    rmt/62/M0
type=ddi_byte:tape;addr=w500104f000b6d98d,0;    rmt/63/M0
:wq
[sharefs-client1]root@solaris:~# rm /dev/rmt/*
[sharefs-client1]root@solaris:~# devfsadm -c tape
[sharefs-client1]root@solaris:~# exit
[sharefs-mds]root@solaris:~#
```

9. "분산 테이프 I/O를 위한 Datamover 클라이언트 구성 " 또는 "추가 파일 시스템 클라이언트 구성"으로 돌아갑니다.

활성 메타데이터 서버에서 잠재적 메타데이터 서버로 전환

이 절의 절차는 파일 시스템의 메타데이터 서비스를 현재 호스트(활성 메타데이터 서버)에서 대기 호스트(잠재적 메타데이터 서버)로 이동하는 것입니다. 어떤 절차를 사용할지는 교체하려는 서버 호스트의 상태에 따라 다릅니다.

- 고장난 활성 메타데이터 서버를 교체하기 위해 잠재적 메타데이터 서버 활성화
- 건강한 활성 메타데이터 서버를 교체하기 위해 잠재적 메타데이터 서버 활성화

고장난 활성 메타데이터 서버를 교체하기 위해 잠재적 메타데이터 서버 활성화

이 절차는 작동이 중지된 활성 메타데이터 서버 호스트의 메타데이터 서비스를 밖으로 이동하는 것입니다. 파일 시스템이 여전히 마운트된 경우에도 잠재적 메타데이터 서버를 활성화합니다. 다음과 같이 하십시오.

주의:

고장난 메타데이터 서버를 중지하거나 사용 안함으로 설정하거나 연결을 끊을 때까지 절대로 잠재적 메타데이터 서버를 활성화하지 마십시오!

파일 시스템이 마운트되었고 활성 메타데이터 서버가 다운되었을 때 잠재적 서버를 활성화하려면 *samsharefs* 명령을 *-R* 옵션(파일 시스템 인터페이스가 아닌 원시 장치에서 작동)과 함께 실행해야 합니다. 따라서 고장난 서버가 여전히 장치에 연결되어 있는 동안 잠재적 메타데이터 서버를 활성화하면 고장난 서버가 파일 시스템을 손상시킬 수 있습니다.

---

1. 활성 메타데이터 서버가 고장난 경우 작업을 진행하기 전에 메타데이터 장치에 액세스할 수 없는지 확인합니다. 영향을 받는 호스트의 전원을 끄거나, 호스트를 중지하거나, 메타데이터 장치에서 고장난 호스트의 연결을 끊습니다.
2. 모든 클라이언트 읽기, 쓰기, 첨부 임대가 만료될 수 있도록 적어도 최대 임대 시간이 소진될 때까지 기다립니다.
3. 잠재적 메타데이터 서버에 *root*로 로그인합니다.

   예제에서는 잠재적 메타데이터 서버 *sharefs-mds_alt*에 로그인합니다.

   ```
   [sharefs-mds_alt]root@solaris:~#
   ```

4. 잠재적 메타데이터 서버를 활성화합니다. 잠재적 메타데이터 서버에서 *samsharefs -R -s server file-system* 명령을 실행합니다. 여기서 *server*는 잠재적 메타데이터 서버의 호스트 이름이고 *file-system*은 Oracle HSM 공유 파일 시스템의 이름입니다.

   예제에서 잠재적 메타데이터 서버는 *sharefs-mds_alt*, 파일 시스템 이름은 *sharefs*입니다.

   ```
   [sharefs-mds_alt]root@solaris:~# samsharefs -R -s sharefs-mds_alt sharefs
   ```

5. 파일 시스템의 무결성을 검사하고 가능한 문제점을 교정해야 하는 경우 "공유 파일 시스템 마운트 해제" 절차를 사용하여 지금 파일 시스템을 마운트 해제합니다.
6. 파일 시스템을 마운트 해제했으면 파일 시스템 검사를 수행합니다. *samfsck -F file-system* 명령을 사용합니다. 여기서 *-F*는 오류 교정을 지정하고 *file-system*은 파일 시스템의 이름입니다.

   예제에서는 파일 시스템 이름이 *sharefs*인지 검사하고 오류를 교정합니다.

   ```
   [sharefs-mds_alt]root@solaris:~# samfsck -F sharefs
   ```

7. 여기서 중지합니다.

건강한 활성 메타데이터 서버를 교체하기 위해 잠재적 메타데이터 서버 활성화

필요한 경우 건강한 활성 메타데이터 서버 호스트의 메타데이터 서비스를 새로 활성화된 잠재적 메타데이터 서버로 이동할 수 있습니다. 예를 들어, 원래 서버 호스트 또는 일부 구성 요소를 업그레이드하거나 교체하는 동안, 파일 시스템의 가용성을 유지하기 위해 대체 호스트로 메타데이터 서비스를 전송할 수 있습니다. 다음과 같이 하십시오.

1. 활성/잠재적 메타데이터 서버에 *root*로 로그인합니다.

   예제에서는 활성 메타데이터 서버 *sharefs-mds*에 로그인합니다. 그리고 두번째 단말기 창에서 보안 셸(*ssh*)을 사용하여 잠재적 메타데이터 서버 *sharefs-mds_alt*에 로그인합니다.

   ```
   [sharefs-mds]root@solaris:~#

   [sharefs-mds]root@solaris:~# ssh root@sharefs-mds_alt
   Password:
   [sharefs-mds-alt]root@solaris:~#
   ```

2. 활성 메타데이터 서버가 Oracle HSM 아카이빙 파일 시스템을 마운트하는 경우 활성 아카이빙 및 스테이징 작업을 마치고 더 진행하기 전에 새로운 작동을 중지합니다. “아카이빙 및 스테이징 프로세스 유휴 설정”을 참조하십시오.
3. 활성 메타데이터 서버가 Oracle HSM 아카이빙 파일 시스템을 마운트하는 경우 이동식 매체 드라이브를 유휴 설정하고 라이브러리 제어 데몬을 중지합니다. “아카이빙 및 스테이징 프로세스 중지”를 참조하십시오.
4. *crontab* 항목을 사용하여 리사이클러 프로세스를 실행하는 경우 항목을 제거하고 리사이클러가 현재 실행 중이 아닌지 확인합니다.
5. 잠재적 메타데이터 서버를 활성화합니다. 잠재적 메타데이터 서버에서 *samsharefs -s server file-system* 명령을 실행합니다. 여기서 *server*는 잠재적 메타데이터 서버의 호스트 이름이고 *file-system*은 Oracle HSM 공유 파일 시스템의 이름입니다.

   예제에서 잠재적 메타데이터 서버는 *sharefs-mds_alt*, 파일 시스템 이름은 *sharefs*입니다.

   ```
   [sharefs-mds_alt]root@solaris:~# samsharefs -s sharefs-mds_alt sharefs
   ```

6. 구성 파일을 로드하고 잠재적 메타데이터 서버에서 Oracle HSM 프로세스를 시작합니다. *samd config* 명령을 사용합니다.

   아카이빙 공유 파일 시스템의 경우 *samd config* 명령이 아카이빙 프로세스와 라이브러리 제어 데몬을 다시 시작합니다. 그러나 테이프에서 주 디스크 캐시로 파일이 스테이

지되기를 기다리는 공유 파일 시스템 클라이언트의 경우 스테이지 요청을 다시 실행해야 합니다.

7. 여전히 *crontab* 항목을 사용하여 리사이클러 프로세스를 실행해야 하는 경우 항목을 복원합니다.
8. 여기서 중지합니다.

비공유 파일 시스템을 공유 파일 시스템으로 변환

비공유 파일 시스템을 공유 파일 시스템으로 변환하려면 다음 작업을 수행합니다.

- 활성/잠재적 메타데이터 서버에 호스트 파일 만들기
- 비공유 파일 시스템 공유 및 클라이언트 구성

활성/잠재적 메타데이터 서버에 호스트 파일 만들기

각 메타데이터 서버에서 공유 파일 시스템의 서버 및 클라이언트에 대한 네트워크 주소 정보를 나열하는 hosts 파일을 만들어야 합니다. hosts 파일은 */etc/opt/SUNWsamfs/* 디렉토리에 *mcf* 파일과 나란히 저장됩니다. 공유 파일 시스템의 초기 생성 중 *sammkfs -S* 명령은 이 파일에 저장된 설정을 사용하여 공유를 구성합니다. 따라서 아래 절차를 사용하여 지금 만듭니다.

1. 클라이언트로서 파일 시스템을 공유할 호스트에 대한 네트워크 호스트 이름과 IP 주소를 수집합니다.

   아래 예제에서는 *hsmfs1* 파일 시스템을 클라이언트 *hsmfs1-mds_alt*(잠재적 메타데이터 서버), *hsmfs1-client1*, *hsmfs1-client2*와 공유합니다.
2. 메타데이터 서버에 *root*로 로그인합니다.

   예제에서는 *hsmfs1-mds* 호스트에 로그인합니다.

   ```
   [hsmfs1-mds]root@solaris:~#
   ```

3. 텍스트 편집기를 사용하여 메타데이터 서버에 */etc/opt/SUNWsamfs/hosts.family-set-name* 파일을 만듭니다. *family-set-name*은 공유할 파일 시스템의 패밀리 세트 이름으로 바뀝니다.

   예제에서는 *vi* 텍스트 편집기를 사용하여 *hosts.hsmfs1* 파일을 만듭니다. 몇 가지 선택적 머리글을 추가하고, 각 라인은 주석을 나타내는 해시 기호(#)로 시작합니다.

   ```
   [hsmfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.hsmfs1
   # /etc/opt/SUNWsamfs/hosts.hsmfs1
   #                                          Server   On/  Additional
   #Host Name           Network Interface      Ordinal  Off  Parameters
   #------------------   ---------------------  -------  ---  ----------
   ```

4. 첫번째 열에 메타데이터 서버의 호스트 이름을 입력하고 두번째 열에는 해당 IP 주소 또는 도메인 이름을 입력합니다. 열을 공백 문자로 구분합니다.

   예제에서는 메타데이터 서버의 호스트 이름 및 IP 주소를 각각 *hsmfs1-mds* 및 *10.79.213.117*로 입력합니다.

```
[hsmfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.hsmfs1
# /etc/opt/SUNWsamfs/hosts.hsmfs1
#                                          Server   On/  Additional
#Host Name            Network Interface    Ordinal  Off  Parameters
#------------------   ---------------------  -------  ---  ----------
hsmfs1-mds           10.79.213.117
```

5. 네트워크 주소와 공백 문자로 구분된 세번째 열을 추가합니다. 이 열에서 서버의 순서 번호를 입력합니다(활성 메타데이터 서버는 *1*, 첫번째 잠재적 메타데이터 서버는 *2* 등등).

   이 예제에서는 메타데이터 서버가 하나뿐이므로 *1*을 입력합니다.

```
[hsmfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.hsmfs1
# /etc/opt/SUNWsamfs/hosts.hsmfs1
#                                          Server   On/  Additional
#Host Name            Network Interface    Ordinal  Off  Parameters
#------------------   ---------------------  -------  ---  ----------
hsmfs1-mds           10.79.213.117          1
```

6. 공백 문자로 서버 순서 번호와 구분된 네번째 열을 추가합니다. 이 열에서 *0*(제로)을 입력합니다.

   네번째 열의 *0*, *-*(하이픈) 또는 공백 값은 호스트가 *on*으로 구성되었고 공유 파일 시스템에 액세스할 수 있음을 나타냅니다. *1*(숫자 일)은 호스트가 *off*로 구성되었지만 파일 시스템에 액세스할 수 없음을 나타냅니다. 공유 파일 시스템을 관리할 때 해당 값 사용에 대한 자세한 내용은 *samhsmfs1* 매뉴얼 페이지를 참조하십시오.

```
[hsmfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.hsmfs1
# /etc/opt/SUNWsamfs/hosts.hsmfs1
#                                          Server   On/  Additional
#Host Name            Network Interface    Ordinal  Off  Parameters
#------------------   ---------------------  -------  ---  ----------
hsmfs1-mds           10.79.213.117          1        0
```

7. 공백 문자로 설정/해제 상태 열과 구분된 다섯번째 열을 추가합니다. 이 열에서 현재 활성 메타데이터 서버를 나타내는 *server* 키워드를 입력합니다.

```
[hsmfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.hsmfs1
# /etc/opt/SUNWsamfs/hosts.hsmfs1
```

```
#                                          Server   On/  Additional
#Host Name            Network Interface     Ordinal  Off  Parameters
#------------------   --------------------  -------  ---  ----------
hsmfs1-mds           10.79.213.117         1        0    server
```

8. 하나 이상의 호스트를 잠재적 메타데이터 서버로 포함하려면 각각 항목을 만듭니다. 매번 서버 순서를 증분합니다. 그러나 *server* 키워드는 포함하지 마십시오(파일 시스템당 하나의 활성 메타데이터 서버만 가능).

   예제에서는 *hsmfs1-mds_alt* 호스트가 서버 순서 *2*를 가진 잠재적 메타데이터 서버입니다. 메타데이터 서버로 활성화 때까지는 클라이언트로 유지됩니다.

```
[hsmfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.hsmfs1
# /etc/opt/SUNWsamfs/hosts.hsmfs1
#                                          Server   On/  Additional
#Host Name            Network Interface     Ordinal  Off  Parameters
#------------------   --------------------  -------  ---  ----------
hsmfs1-mds           10.79.213.117         1        0    server
hsmfs1-mds_alt       10.79.213.217         2        0
```

9. 각 클라이언트 호스트마다 서버 순서 값 *0*과 함께 행을 추가합니다.

   서버 순서 *0*은 호스트를 클라이언트로 식별합니다. 예제에서는 두 개의 클라이언트 *hsmfs1-client1* 및 *hsmfs1-client2*를 추가합니다.

```
[hsmfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.hsmfs1
# /etc/opt/SUNWsamfs/hosts.hsmfs1
#                                          Server   On/  Additional
#Host Name            Network Interface     Ordinal  Off  Parameters
#------------------   --------------------  -------  ---  ----------
hsmfs1-mds           10.79.213.17          1        0    server
hsmfs1-mds_alt       10.79.213.7           2        0
hsmfs1-client1       10.79.213.33          0        0
hsmfs1-client2       10.79.213.47          0        0
```

10. */etc/opt/SUNWsamfs/hosts.family-set-name* 파일을 저장하고 편집기를 종료합니다.

    예제에서는 */etc/opt/SUNWsamfs/hosts.hsmfs1*에 대한 변경사항을 저장하고 *vi* 편집기를 종료합니다.

```
[hsmfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.hsmfs1
# /etc/opt/SUNWsamfs/hosts.hsmfs1
#                                          Server   On/  Additional
#Host Name            Network Interface     Ordinal  Off  Parameters
```

```
#------------------   ---------------------  -------  ---  ----------
hsmfs1-mds           10.79.213.117          1        0    server
hsmfs1-mds           10.79.213.117          1        0    server
hsmfs1-mds_alt       10.79.213.217          2        0
hsmfs1-client1       10.79.213.133          0        0
hsmfs1-client2       10.79.213.147          0        0
:wq
[hsmfs1-mds]root@solaris:~#
```

11. 공유 파일 시스템 구성에 포함될 잠재적 메타데이터 서버에 새 파일 */etc/opt/SUNWsamfs/hosts.family-set-name*의 복사본을 배치합니다.

    예제에서는 *hsmfs1-mds_alt* 호스트에 복사본을 배치합니다.

```
[hsmfs1-mds]root@solaris:~# sftp root@hsmfs1-mds_alt
Password:
sftp> cd /etc/opt/SUNWsamfs/
sftp> put /etc/opt/SUNWsamfs/hosts.hsmfs1
sftp> bye
[hsmfs1-mds]root@solaris:~#
```

12. 이제 비공유 파일 시스템 공유 및 클라이언트 구성을 수행합니다.

비공유 파일 시스템 공유 및 클라이언트 구성

1. 메타데이터 서버에 *root*로 로그인합니다.

   예제에서는 *hsmfs1-mds* 호스트에 로그인합니다.

```
[hsmfs1-mds]root@solaris:~#
```

2. 현재 시스템 파일과 구성 파일의 백업 복사본이 없다면 지금 백업을 만듭니다. "Oracle HSM 구성 백업"을 참조하십시오.
3. 현재 파일 시스템 복구 지점 파일과 최근 아카이브 로그 복사본이 없다면 지금 만듭니다. "파일 시스템 백업"을 참조하십시오.

   초기 구성 중 파일 시스템에 대한 자동 백업 프로세스를 설정한 경우 추가 백업이 필요하지 않을 수도 있습니다.
4. 아카이빙 파일 시스템을 변환하는 경우 활성 아카이빙 및 스테이징 작업을 마치고 더 진행하기 전에 새로운 작동을 중지합니다. "아카이빙 및 스테이징 프로세스 유휴 설정" 및 "아카이빙 및 스테이징 프로세스 중지"를 참조하십시오.
5. 파일 시스템을 마운트 해제합니다. *umount family-set-name* 명령을 사용합니다. 여기서 *family-set-name*은 공유할 파일 시스템의 패밀리 세트 이름입니다.

Oracle HSM 파일 시스템 마운트 및 마운트 해제에 대한 자세한 내용은 *mount_samfs* 매뉴얼 페이지를 참조하십시오. 예제에서는 *hsmfs1* 파일 시스템을 마운트 해제합니다.

```
[hsmfs1-mds]root@solaris:~# umount hsmfs1
[hsmfs1-mds]root@solaris:~#
```

6. 파일 시스템을 Oracle HSM 공유 파일 시스템으로 변환합니다. *samfsck -S -F file-system-name* 명령을 사용합니다. 여기서 *file-system-name*은 파일 시스템의 패밀리 세트 이름입니다.

   예제에서는 *hsmfs1*이라는 파일 시스템을 변환합니다.

```
[hsmfs1-mds]root@solaris:~# samfsck -S -F hsmfs1
```

7. 텍스트 편집기에서 */etc/opt/SUNWsamfs/mcf* 파일을 열고 파일 시스템 행을 찾습니다.

   예제에서는 *vi* 편집기를 사용합니다:

```
[hsmfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment          Equipment  Equipment  Family   Device   Additional
# Identifier         Ordinal    Type       Set      State    Parameters
#------------------  ---------  ---------  -------  ------   -----------------
hsmfs1              200        ma         hsmfs1   on
 /dev/dsk/c0t0d0s0    201        mm         hsmfs1   on
 /dev/dsk/c0t3d0s0    202        md         hsmfs1   on
 /dev/dsk/c0t3d0s1    203        md         hsmfs1   on
```

8. *mcf* 파일에서 파일 시스템 항목의 마지막 열인 Additional Parameters 필드에 *shared* 매개변수를 추가합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

```
[hsmfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment          Equipment  Equipment  Family   Device   Additional
# Identifier         Ordinal    Type       Set      State    Parameters
#------------------  ---------  ---------  -------  ------   -----------------
hsmfs1              200        ma         hsmfs1   on       shared
 /dev/dsk/c0t0d0s0    201        mm         hsmfs1   on
 /dev/dsk/c0t3d0s0    202        md         hsmfs1   on
 /dev/dsk/c0t3d0s1    203        md         hsmfs1   on
:wq
[hsmfs1-mds]root@solaris:~#
```

9. 텍스트 편집기에서 */etc/vfstab* 파일을 열고 파일 시스템 행을 찾습니다.

이 예에서는 *vi* 편집기를 사용합니다.

```
[hsmfs1-mds]root@solaris:~# vi /etc/vfstab
#File
#Device    Device   Mount     System  fsck  Mount    Mount
#to Mount  to fsck  Point     Type    Pass  at Boot  Options
#--------  -------  --------  ------  ----  -------  ------------------------
/devices   -        /devices  devfs   -     no       -
/proc      -        /proc     proc    -     no       -
...
hsmfs1     -        /hsm/hsmfs1  samfs   -     yes
```

10. */etc/vfstab* 파일에서 파일 시스템 항목의 마지막 열인 Mount Options 필드에 *shared* 마운트 옵션을 추가합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

```
[hsmfs1-mds]root@solaris:~# vi /etc/vfstab
#File
#Device    Device   Mount     System  fsck  Mount    Mount
#to Mount  to fsck  Point     Type    Pass  at Boot  Options
#--------  -------  --------  ------  ----  -------  ------------------------
/devices   -        /devices  devfs   -     no       -
/proc      -        /proc     proc    -     no       -
...
hsmfs1     -        /hsm/hsmfs1  samfs   -     yes      shared
:wq
[hsmfs1-mds]root@solaris:~#
```

11. 공유 파일 시스템 및 호스트 구성을 초기화합니다. *samsharefs -u -R family-set-name* 명령을 사용합니다. 여기서 *family-set-name*은 파일 시스템의 패밀리 세트 이름입니다.

```
[hsmfs1-mds]root@solaris:~# samsharefs -u -R hsmfs1
```

12. Oracle HSM 소프트웨어에 *mcf* 파일을 다시 읽고 그에 따라 재구성하도록 지시합니다.

```
[hsmfs1-mds]root@solaris:~# samd config
```

13. 메타데이터 서버에 공유 파일 시스템을 마운트합니다.

```
[hsmfs1-mds]root@solaris:~# mount /hsm/hsmfs1
```

14. 호스트가 다중 네트워크 인터페이스로 구성된 경우 "로컬 hosts 파일을 사용하여 네트워크 통신 경로 지정 "을 참조하십시오.

15. "추가 파일 시스템 클라이언트 구성"에 설명된 절차를 사용하여 새로 공유된 파일 시스템에 필요한 클라이언트를 추가합니다.

로컬 hosts 파일을 사용하여 네트워크 통신 경로 지정

개별 호스트에는 로컬 hosts 파일이 필요 없습니다. 메타데이터 서버에서 파일 시스템의 전역 파일은 모든 파일 시스템 호스트에 대한 활성 메타데이터 서버와 활성/잠재적 메타데이터 서버의 네트워크 인터페이스를 식별합니다("활성/잠재적 메타데이터 서버에 호스트 파일 만들기" 참조). 그러나 다중 네트워크 인터페이스가 설치된 파일 시스템 호스트 간에 선택적으로 네트워크 트래픽 경로를 지정해야 하는 경우 로컬 hosts 파일이 유용할 수 있습니다.

각 파일 시스템 호스트는 메타데이터 서버에서 먼저 */etc/opt/SUNWsamfs/hosts.family-set-name* 파일을 확인하여 다른 호스트에 대한 네트워크 인터페이스를 식별합니다. 여기서 *family-set-name*은 */etc/opt/SUNWsamfs/mcf* 파일에 지정된 파일 시스템 패밀리의 이름입니다. 그런 다음 고유의 특정 */etc/opt/SUNWsamfs/hosts.family-set-name.local* 파일이 있는지 확인합니다. 로컬 hosts 파일이 없는 경우 전역 hosts 파일에 지정된 인터페이스 주소를 전역 파일에 지정된 순서대로 사용합니다. 그러나 로컬 hosts 파일이 있는 경우 로컬 파일을 전역 파일과 비교하여 양쪽 파일에 나열된 인터페이스만 로컬 파일에 지정된 순서대로 사용합니다. 따라서 각 파일에 다른 주소를 사용하여 다양한 호스트에서 사용되는 인터페이스를 제어할 수 있습니다.

로컬 hosts 파일을 구성하려면 아래 설명된 절차를 사용하십시오.

1. 메타데이터 서버 호스트와 각 잠재적 메타데이터 서버 호스트에서 "활성/잠재적 메타데이터 서버에 호스트 파일 만들기"에 설명된 대로 전역 hosts 파일 */etc/opt/SUNWsamfs/hosts.family-set-name*의 복사본을 만듭니다.

   이 절의 예제에서는 공유 파일 시스템 *sharefs2*에 활성 메타데이터 서버 *sharefs2-mds* 및 잠재적 메타데이터 서버 *sharefs2-mds_alt*가 각각 두 네트워크 인터페이스로 포함됩니다. 두 개의 클라이언트 *sharefs2-client1* 및 *sharefs2-client2*도 있습니다.

   활성/잠재적 메타데이터 서버에서 개인 네트워크 주소를 통해 서로 통신하고, DNS(도메인 이름 서비스)가 공용 LAN(근거리 통신망) 주소로 분석할 수 있는 호스트 이름을 통해 클라이언트와 통신하려고 합니다. 따라서 파일 시스템의 전역 호스트 파일인 */etc/opt/SUNWsamfs/hosts.sharefs2nics*는 *Network Interface* 필드에 활성/잠재적 서버 항목의 개인 네트워크 주소와 각 클라이언트의 인터페이스 주소 호스트 이름을 지정합니다. 파일은 다음과 같이 같습니다.

```
# /etc/opt/SUNWsamfs/hosts.sharefs2
#                                          Server   On/  Additional
#Host Name            Network Interface     Ordinal  Off  Parameters
#------------------   ---------------------  -------  ---  ----------
sharefs2-mds          172.16.0.129           1        0    server
sharefs2-mds_alt      172.16.0.130           2        0
sharefs2-client1      sharefs2-client1       0        0
```

```
sharefs2-client2      sharefs2-client2        0        0
```

2. 경로 및 파일 이름 */etc/opt/SUNWsamfs/hosts.family-set-name.local*을 사용해서 각 활성 및 잠재적 메타데이터 서버에 로컬 hosts 파일을 만듭니다. 여기서 *family-set-name*은 */etc/opt/SUNWsamfs/mcf* 파일에서 공유 파일 시스템에 대해 지정된 이름입니다. 활성/잠재적 서버에서 사용할 네트워크 인터페이스만 포함하십시오.

   예제에서는 활성 및 잠재적 메타데이터 서버가 개인 네트워크를 통해 서로 통신하도록 해야 하므로, 각 서버의 로컬 hosts 파일 *hosts.sharefs2.local*에는 활성 및 잠재적 서버에 대한 개인 주소만 나열됩니다.

```
[sharefs2-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sharefs2.local
# /etc/opt/SUNWsamfs/hosts.sharefs2
#                                          Server   On/  Additional
#Host Name            Network Interface     Ordinal  Off  Parameters
#------------------   ---------------------  -------  ---  ----------
sharefs2-mds          172.16.0.129            1        0    server
sharefs2-mds_alt      172.16.0.130            2        0
:wq
[sharefs2-mds]root@solaris:~# ssh root@sharefs2-mds_alt
Password:


[sharefs2-mds_alt]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sharefs2.local
# /etc/opt/SUNWsamfs/hosts.sharefs2
#                                          Server   On/  Additional
#Host Name            Network Interface     Ordinal  Off  Parameters
#------------------   ---------------------  -------  ---  ----------
sharefs2-mds          172.16.0.129            1        0    server
sharefs2-mds_alt      172.16.0.130            2        0
:wq
[sharefs2-mds_alt]root@solaris:~# exit
[sharefs2-mds]root@solaris:~#
```

3. 경로 및 파일 이름 */etc/opt/SUNWsamfs/hosts.family-set-name.local*을 사용해서 각 클라이언트에 로컬 hosts 파일을 만듭니다. 여기서 *family-set-name*은 */etc/opt/SUNWsamfs/mcf* 파일에서 공유 파일 시스템에 대해 지정된 이름입니다. 클라이언트에서 사용할 네트워크 인터페이스만 포함하십시오.

   예제에서는 클라이언트가 공용 네트워크를 통해서만 서버와 통신합니다. 따라서 파일에는 활성/잠재적 메타데이터 서버의 호스트 이름만 포함됩니다.

```
[sharefs2-mds]root@solaris:~# ssh root@sharefs2-client1
Password:
[sharefs2-client1]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sharefs2.local
# /etc/opt/SUNWsamfs/hosts.sharefs2
```

```
#                                              Server   On/  Additional
#Host Name            Network Interface       Ordinal  Off  Parameters
#------------------   ---------------------   -------  ---  ----------
sharefs2-mds          sharefs2-mds             1        0    server
sharefs2-mds_alt      sharefs2-mds_alt         2        0
:wq
[sharefs2-client1]root@solaris:~# exit
[sharefs2-mds]root@solaris:~# ssh root@sharefs2-client2
Password:

[sharefs2-client2]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sharefs2.local
# /etc/opt/SUNWsamfs/hosts.sharefs2
#                                              Server   On/  Additional
#Host Name            Network Interface       Ordinal  Off  Parameters
#------------------   ---------------------   -------  ---  ----------
sharefs2-mds          sharefs2-mds             1        0    server
sharefs2-mds_alt      sharefs2-mds_alt         2        0
:wq
[sharefs2-client2]root@solaris:~# exit
[sharefs2-mds]root@solaris:~#
```

4. 서버 구성을 마치는 동안 이 절차를 시작한 경우 클라이언트를 추가합니다. “추가 파일 시스템 클라이언트 구성”으로 이동합니다.

공유 파일 시스템을 비공유 파일 시스템으로 변환

파일 시스템 공유를 취소해야 하는 경우 다음과 같이 하십시오.

공유 메타데이터 서버를 비공유 시스템으로 변환

1. 메타데이터 서버에 *root*로 로그인합니다.

   예제에서는 *hsmfs1-mds* 호스트에 로그인합니다.

   ```
   [hsmfs1-mds]root@solaris:~#
   ```

2. “파일 시스템 호스트 파일에서 호스트 제거” 절차를 사용하여 메타데이터 서버 구성에서 클라이언트를 제거합니다.
3. 현재 시스템 파일과 구성 파일의 백업 복사본이 없다면 지금 백업을 만듭니다. “Oracle HSM 구성 백업”을 참조하십시오.
4. 현재 파일 시스템 복구 지점 파일과 최근 아카이브 로그 복사본이 없다면 지금 만듭니다. “파일 시스템 백업”을 참조하십시오.

   초기 구성 중 파일 시스템에 대한 자동 백업 프로세스를 설정한 경우 추가 백업이 필요하지 않을 수도 있습니다.

5. 아카이빙 파일 시스템을 변환하는 경우 활성 아카이빙 및 스테이징 작업을 마치고 더 진행하기 전에 새로운 작동을 중지합니다. "아카이빙 및 스테이징 프로세스 유휴 설정" 및 "아카이빙 및 스테이징 프로세스 중지"를 참조하십시오.
6. 파일 시스템을 마운트 해제합니다. *umount family-set-name* 명령을 사용합니다. 여기서 *family-set-name*은 */etc/opt/SUNWsamfs/mcf* 파일에서 공유 파일 시스템에 대해 지정된 이름입니다.

   Oracle HSM 파일 시스템 마운트 및 마운트 해제에 대한 자세한 내용은 *mount_samfs* 매뉴얼 페이지를 참조하십시오. 예제에서는 *hsmfs1* 파일 시스템을 마운트 해제합니다.

   ```
   [hsmfs1-mds]root@solaris:~# umount hsmfs1
   ```

7. Oracle HSM 공유 파일 시스템을 비공유 파일 시스템으로 변환합니다. *samfsck -F -U file-system-name* 명령을 사용합니다. 여기서 *file-system-name*은 */etc/opt/SUNWsamfs/mcf* 파일에서 공유 파일 시스템에 대해 지정된 이름입니다.

   예제에서는 *hsmfs1*이라는 파일 시스템을 변환합니다.

   ```
   [hsmfs1-mds]root@solaris:~# samfsck -F -U hsmfs1
   ```

8. 텍스트 편집기에서 */etc/opt/SUNWsamfs/mcf* 파일을 열고 파일 시스템 행을 찾습니다.

   예제에서는 *vi* 편집기를 사용합니다:

   ```
   [hsmfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
   # Equipment          Equipment  Equipment  Family   Device   Additional
   # Identifier         Ordinal    Type       Set      State    Parameters
   #------------------  ---------  ---------  -------  ------   -----------------
   hsmfs1               200        ma         hsmfs1   on       shared
    /dev/dsk/c0t0d0s0    201        mm         hsmfs1   on
    /dev/dsk/c0t3d0s0    202        md         hsmfs1   on
    /dev/dsk/c0t3d0s1    203        md         hsmfs1   on
   ```

9. *mcf* 파일에서 파일 시스템 항목의 마지막 열인 Additional Parameters 필드에서 *shared* 매개변수를 삭제합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   ```
   [hsmfs1-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
   # Equipment          Equipment  Equipment  Family   Device   Additional
   # Identifier         Ordinal    Type       Set      State    Parameters
   #------------------  ---------  ---------  -------  ------   -----------------
   hsmfs1               200        ma         hsmfs1   on
    /dev/dsk/c0t0d0s0    201        mm         hsmfs1   on
    /dev/dsk/c0t3d0s0    202        md         hsmfs1   on
    /dev/dsk/c0t3d0s1    203        md         hsmfs1   on
   ```

```
:wq
[hsmfs1-mds]root@solaris:~#
```

10. 텍스트 편집기에서 */etc/vfstab* 파일을 열고 파일 시스템 행을 찾습니다.

    예제에서는 *vi* 편집기를 사용합니다.

```
[hsmfs1-mds]root@solaris:~# vi /etc/vfstab
#File
#Device    Device   Mount     System  fsck  Mount    Mount
#to Mount  to fsck  Point     Type    Pass  at Boot  Options
#--------  -------  --------  ------  ----  -------  -------------------------
/devices   -        /devices  devfs   -     no       -
/proc      -        /proc     proc    -     no       -
...
hsmfs1     -        /hsm/hsmfs1  samfs   -     yes      shared
```

11. */etc/vfstab* 파일에서 파일 시스템 항목의 마지막 열에 있는 마운트 옵션 필드에서 *shared* 마운트 옵션을 삭제합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

    이 예에서는 *vi* 편집기를 사용합니다.

```
[hsmfs1-mds]root@solaris:~# vi /etc/vfstab
#File
#Device    Device   Mount     System  fsck  Mount    Mount
#to Mount  to fsck  Point     Type    Pass  at Boot  Options
#--------  -------  --------  ------  ----  -------  -------------------------
/devices   -        /devices  devfs   -     no       -
/proc      -        /proc     proc    -     no       -
...
hsmfs1     -        /hsm/hsmfs1  samfs   -     yes
:wq
[hsmfs1-mds]root@solaris:~#
```

12. */etc/opt/SUNWsamfs/hosts.file-system-name* 파일을 삭제합니다.
13. Oracle HSM 소프트웨어에 *mcf* 파일을 다시 읽고 그에 따라 재구성하도록 지시합니다.

```
[hsmfs1-mds]root@solaris:~# samd config
```

14. 파일 시스템을 마운트합니다.

```
[hsmfs1]root@solaris:~# mount /hsm/hsmfs1
```

15. 여기서 중지합니다.

4

4장. 파일 및 디렉토리 관리

이 장에서는 다음 항목을 다룹니다.

- Oracle HSM 파일 속성 설정
- 확장 파일 속성 사용
- 큰 파일 수용
- LTFS(Linear Tape File System) 볼륨 작업
- SMB/CIFS 공유에서 디렉토리 및 파일 관리
- 액세스 제어 목록 관리.

Oracle HSM 파일 속성 설정

친숙한 인터페이스(표준 UNIX 파일 시스템)를 통해 사용자와 상호 작용하는 능력은 Oracle Hierarchical Storage Manager and StorageTek QFS Software의 주요 장점입니다. 대부분의 사용자는 차이점을 인지할 필요도 없습니다. 그러나 Oracle HSM 파일 시스템은 필요 시 고급 사용자에게 엄청난 기능을 제공할 수 있습니다. Oracle HSM 파일 속성 을 사용하여 사용자는 개별 파일 및 디렉토리와 작동하도록 파일 시스템의 동작을 최적화할 수 있습니다. 본인의 업무량과 데이터 특성을 잘 이해하는 사용자는 파일 단위로 성능을 크게 향상시킬 수 있습니다. 예를 들어, 사용자는 주어진 파일이나 디렉토리에서 데이터 특성을 기반으로 직접 또는 버퍼 I/O를 지정할 수 있습니다. 큰 파일을 순차적으로 기록할 수 있도록 파일 시스템 공간을 미리 할당할 수 있고, 특정 파일이나 디렉토리를 기록할 때 사용되는 스트라이프 너비를 지정할 수 있습니다.

setfa 명령은 신규/기존 파일 및 디렉토리에 이러한 파일 속성을 설정합니다. 명령은 기존에 없는 지정된 파일이나 디렉토리를 만듭니다. 디렉토리에 적용할 경우, 디렉토리 아래의 모든 파일과 하위 디렉토리에 지정된 속성을 설정합니다. 그 후에 만들어진 파일과 디렉토리는 이러한 속성을 상속합니다.

기본 작업이 아래에 개략적으로 설명됩니다. 추가 정보는 *setfa* 매뉴얼 페이지를 참조하십시오.

- 기본 파일 속성값 복원
- 파일 시스템 공간 미리 할당
- 파일에 라운드 로빈 또는 스트라이프 할당 지정
- 지정된 스트라이프 그룹 장치에 파일 스토리지 할당.

기본 파일 속성값 복원

1. 파일 시스템 호스트에 로그인합니다.

   ```
   user@solaris:~#
   ```

2. 파일에 기본 속성값을 재설정하려면 *setfa -d file* 명령을 사용합니다. 여기서 *file*은 파일의 경로 및 이름입니다.

   예제에서는 */samfs1/data/2014/03/series3.15* 파일에서 기본값을 재설정합니다.

   ```
   user@solaris:~# setfa -d /samfs1/data/2014/03/series3.15
   ```

3. 디렉토리와 모든 내용에 기본 속성값을 재귀적으로 재설정하려면 *setfa -r directory* 명령을 사용합니다. 여기서 *directory*는 디렉토리의 경로 및 이름입니다.

   예제에서는 하위 디렉토리 */samfs1/data/2014/02*에 기본값을 재설정합니다.

   ```
   user@solaris:~# setfa -r /samfs1/data/2014/02/
   ```

4. 여기서 중지합니다.

파일 시스템 공간 미리 할당

파일에 공간을 미리 할당하면 파일을 기록할 때 전체 파일을 순차적으로 작성할 충분한 공간이 확보됩니다. 큰 파일을 순차적 블록에 읽기/쓰기하면 흩어진 작은 데이터 블록을 찾고 버퍼링하는 데 필요한 오버헤드가 줄어들므로 효율성과 전체 성능이 향상됩니다. 따라서 미리 할당은 예측 가능한 개수의 큰 데이터 블록을 쓰기에 가장 좋은 방법입니다. 미리 할당되었지만 사용되지 않은 공간은 파일을 닫을 때 파일의 일부로 남아서 전체 파일을 삭제할 때까지 다른 용도로 사용할 수 없습니다.

1. 파일 시스템 호스트에 로그인합니다.

   ```
   user@solaris:~#
   ```

2. 이미 데이터가 있는 기존 파일에 쓰기 공간을 미리 할당해야 하는 경우 *setfa -L number-bytes file* 명령을 사용합니다. 여기서 *number-bytes*는 정수 또는 정수에 *k*(킬로바이트), *m*(메가바이트), *g*(기가바이트) 단위를 더한 것입니다. *file*은 파일의 이름입니다.

   setfa -L 명령은 표준 할당을 사용합니다. 여기에서는 스트라이핑이 지원됩니다. 사전 할당된 파일은 사전 할당된 크기보다 커질 수 있습니다. 예제에서는 기존 파일 *tests/series119b*에 121메가바이트를 미리 할당합니다.

   ```
   user@solaris:~# setfa -L 121m tests/series119b
   ```

3. 스토리지 블록이 지정되지 않은 새 파일을 기록하기 위해 공간을 사전 할당해야 할 경우 *setfa -l number-bytes file* 명령을 사용합니다. 설명:
   - *l*은 "L"의 소문자입니다.
   - *number-bytes*는 정수이거나 *k*(KB), *m*(MB) 또는 *g*(GB)가 함께 표시된 정수입니다.
   - *file*은 파일의 이름입니다.

   setfa -l 명령은 지정된 개수의 바이트를 미리 할당합니다. 결과 파일은 미리 할당된 크기로 고정되므로 미리 할당된 크기를 넘어가거나 그 아래로 줄어들 수 없습니다. 예제에서는 *data/2014/a3168445* 파일을 만들고 그 내용에 2기가바이트의 공간을 미리 할당합니다.

   ```
   user@solaris:~# setfa -l 2g data/2014/a3168445
   ```

4. 여기서 중지합니다.

파일에 라운드 로빈 또는 스트라이프 할당 지정

기본적으로 Oracle HSM 파일 시스템은 마운트 시점에 파일 시스템에 지정된 할당 방식을 사용합니다. 그러나 사용자가 지정된 디렉토리나 파일에 대해 선호하는 할당 방식(지정된 스트라이프 너비로 라운드 로빈 또는 스트라이핑)을 지정할 수 있습니다.

1. 파일 시스템 호스트에 로그인합니다.

   ```
   user@solaris:~#
   ```

2. 라운드 로빈 할당을 지정하려면 스트라이프 너비 *0*(제로)을 지정합니다. *setfa -s 0 directory-or-file* 명령을 사용합니다. 여기서 *directory-or-file*은 지정된 할당 방법을 사용하여 쓰여질 디렉토리나 파일의 이름입니다.

   스트라이프 너비 *0*(제로)은 스트라이프되지 않은 라운드 로빈 할당을 지정합니다. 파일 시스템은 그 다음 사용 가능한 장치에 파일을 쓰기 시작합니다. 파일이 완료되거나 공간이 소진될 때까지 동일한 장치에 DAU(디스크 할당 단위)를 파일에 연속 기록합니다. 장치에 공간이 소진된 경우 파일 시스템은 그 다음 사용 가능한 장치로 이동하여 디스크 할당 단위 쓰기를 계속합니다. 파일이 완료될 때까지 프로세스를 반복합니다. 예제에서는 *data/field-reports* 디렉토리에 쓰여진 모든 파일에 대해 라운드 로빈 할당을 지정합니다.

   ```
   user@solaris:~# setfa -s 0 data/field-reports
   ```

3. 스트라이프 할당을 지정하려면 스트라이프 너비를 지정합니다. *setfa -s stripe-width directory-or-file* 명령을 사용합니다. 여기서 *stripe-width*는 [*1–255*] 범위의 정수이고 *directory-or-file*은 지정된 할당 방법을 사용하여 쓰여질 디렉토리나 파일의 이름입니다.

[*1–255*] 범위의 스트라이프 너비는 스트라이프 할당을 지정합니다. 파일 시스템은 파일이 완료될 때까지 스트라이프 너비에 지정된 개수의 DAU(디스크 할당 단위)를 여러 장치에 병렬로 기록합니다. 예제에서는 디렉토리에 쓰여진 모든 파일에 대해 스트라이프 너비 *1*로 스트라이프 할당을 지정합니다. 따라서 *data/field-reports* 디렉토리 *data/2014/*에 쓰여진 모든 파일에 대한 파일 할당 및 파일 시스템은 파일이 완료될 때까지 각 사용 가능한 장치에 하나의 디스크 할당 단위를 기록합니다.

```
user@solaris:~# setfa -s 1 data/2014/
```

4. 여기서 중지합니다.

지정된 스트라이프 그룹 장치에 파일 스토리지 할당

사용자는 라운드 로빈 또는 스트라이프 할당을 시작해야 할 스트라이프 그룹 장치를 지정할 수 있습니다. Oracle HSM 스트라이프 그룹은 여러 물리적 볼륨에 걸쳐 데이터를 분할하는 논리적 볼륨입니다. 라운드 로빈 파일 할당이 적용될 때 지정된 스트라이프 그룹에 전체 파일이 기록됩니다. 스트라이프 할당이 적용될 때 지정된 스트라이프 그룹에 첫번째 할당이 이루어집니다.

1. 파일 시스템 호스트에 로그인합니다.

```
user@solaris:~#
```

2. 특정 스트라이프 그룹에 전체 파일을 기록하려면 라운드 로빈 할당을 사용합니다. *setfa -s 0 -gstripe-group-number* 명령을 사용합니다. 여기서 *stripe-group-number*는 지정된 스트라이프 그룹을 식별하는 [*0-127*] 범위의 정수입니다.

   예제에서는 *reports/site51* 파일을 쓸 때 스트라이프 그룹 *0*에서 시작하는 라운드 로빈 할당을 지정합니다.

```
user@solaris:~# setfa -s 0 -g0 reports/site51
```

3. 지정된 스트라이프 그룹부터 시작하여 여러 스트라이프 그룹에 걸쳐 파일을 분할하려면 스트라이프 할당을 사용합니다. *setfa -s stripe-width -gstripe-group-number* 명령을 사용합니다. 여기서 *stripe-width*는 디스크 할당 단위 수를 지정하는 [*1–255*] 범위의 정수이고 *stripe-group-number*는 지정된 스트라이프 그룹을 식별하는 [*0-127*] 범위의 정수입니다.

   예제에서는 *assessments/site52* 파일에 대한 스트라이프 할당을 지정합니다. 스트라이프 그룹 *21*부터 시작하여 그룹당 3개의 디스크 할당 단위를 지정합니다.

```
user@solaris:~# setfa -s 3 -g21 assessments/site52
```

4. 여기서 중지합니다.

확장 파일 속성 사용

다른 Solaris 및 Linux 파일 시스템과 마찬가지로 Oracle HSM 파일 시스템은 확장 파일 속성을 지원합니다. 확장 속성은 파일 시스템 자체가 아닌, 사용자나 응용 프로그램에 의해 파일과 연관된 임의의 메타데이터를 보유합니다. 확장 속성은 파일 다이제스트, 저작자와 소스 응용 프로그램의 이름, 텍스트 파일에서 사용된 문자 인코딩을 보유하는 데 사용되었습니다.

릴리스 6.1부터 Oracle HSM은 데이터 분할 영역의 블록을 사용하는 대신, 메타데이터 분할 영역 내의 확장 inode에 464자 이하를 포함하는 작은 확장 속성 파일을 저장합니다. 새로운 접근법은 확장 속성이 사용 중이고 파일 시스템 메타데이터가 플래시 스토리지와 같은 빠른 장치에 저장될 때 파일 시스템 성능을 크게 향상시킵니다.

확장 파일 속성은 새 파일 시스템을 만들거나 복구 지점(*samfsdump*) 파일에서 구 파일 시스템을 복원할 때마다 자동으로 사용으로 설정됩니다. 확장 속성 사용에 대한 자세한 내용은 Solaris *fsattr(5)* 및 Linux *xattr(7)* 매뉴얼 페이지를 참조하십시오.

큰 파일 수용

Oracle HSM 파일 시스템은 엄청나게 큰 파일을 작업하기에 매우 적합합니다. 이 절에서는 다음 항목을 다룹니다.

- 매우 큰 파일로 디스크 캐시 관리
- 파일 세그먼트
- 큰 데이터 세트를 위해 이동식 매체 파일 사용

매우 큰 파일로 디스크 캐시 관리

매우 큰 파일을 조작할 때 사용 가능한 디스크 캐시 크기에 주의하십시오. 디스크 캐시보다 큰 파일을 쓰려고 시도하면 비아카이빙 파일 시스템은 *ENOSPC* 오류를 반환하고, 아카이빙 파일 시스템은 절대로 사용 가능해질 수 없는 공간을 기다리다가 응용 프로그램이 차단됩니다.

Oracle HSM는 디스크 캐시 크기를 늘릴 수 있는 두 가지 가능한 대안을 제공합니다.

- 파일 세그먼트 - 사용자가 주어진 시간에 큰 파일의 일부만 디스크에 스테이지합니다.
- 큰 데이터 세트를 위해 이동식 매체 파일 사용 - 사용자가 데이터를 디스크에 스테이지하지 않습니다.

파일 세그먼트

Oracle HSM 세그먼테이션 속성을 파일에 설정할 때 파일 시스템은 지정된 크기의 세그먼트로 파일을 분해해서 액세스 요청을 관리합니다. 그러면 주어진 시간에 현재 필요한 세그먼트만 디스크에 상주합니다. 파일의 나머지 부분은 이동식 매체에 상주합니다.

큰 파일을 세그먼트로 나누면 수많은 장점이 있습니다.

- 사용자는 사용 가능한 디스크 캐시보다 큰 파일을 만들고 액세스할 수 있습니다.

  주어진 시간에 세그먼트만 캐시에 상주하므로 디스크 캐시에 맞는 세그먼트 크기만 선택하면 됩니다. 전체 파일은 매체가 수용할 수 있는 크기까지 증가할 수 있습니다.
- 사용자는 디스크 캐시에서 해제된 큰 파일에 더 빠르게 액세스할 수 있습니다. 큰 파일의 일부를 디스크에 스테이지하면 전체 파일이 스테이지될 때까지 기다리는 것보다 훨씬 빠릅니다.
- 각 파일의 변경된 부분만 다시 아카이브하므로 파일을 세그먼트할 때 아카이빙 속도와 효율성이 향상될 수 있습니다.
- 여러 드라이브에 마운트된 이동식 매체 볼륨에 걸쳐 파일을 분할할 수 있습니다. 그런 다음 아카이빙 및 스테이징 작업을 병렬로 진행하면 더욱 성능이 향상될 수 있습니다.

두 가지 제한 사항이 있습니다.

- 공유 파일 시스템의 파일은 세그먼트할 수 없습니다.
- 이진 실행 파일은 세그먼트할 수 없습니다. Solaris 메모리 매핑 함수 *mmap()*은 세그먼트된 파일의 바이트를 프로세스 주소 공간으로 매핑할 수 없습니다.

세그먼트된 파일을 만들려면 다음과 같이 하십시오.

파일 세그먼트

1. 파일 시스템 호스트에 로그인합니다.

   ```
   user@solaris:~#
   ```

2. 세그먼트할 파일을 선택하거나 필요한 경우 파일을 만듭니다.
3. 단일 파일을 세그먼트하려면 *segment [-s stage_ahead] -l segment_size file-path-name* 명령을 사용합니다. 설명:
   - *stage_ahead*(선택사항)는 주어진 세그먼트에 액세스할 때 읽을 연속적 여분 세그먼트 수를 지정하는 정수입니다. 잘 고른 값은 시스템 페이지 캐시의 사용률을 높여서 I/O 성능이 향상될 수 있습니다. 기본값은 *0*(사용 안함)입니다.
   - *segment_size*는 각 세그먼트의 크기를 지정하는 정수 및 단위입니다. 지원되는 단위는 *k*(킬로바이트), *m*(메가바이트), *g*(기가바이트)입니다. 최소 크기는 1메가바이트(*1m* 또는 *1024k*)입니다.
   - *file-path-name*은 파일의 경로 및 이름입니다.

   자세한 내용은 *segment* 매뉴얼 페이지를 참조하십시오. 예제에서는 1.5메가바이트(*1536k*) 세그먼트 크기를 사용하여 *201401.dat* 파일을 세그먼트합니다.

   ```
   user@solaris:~# segment -l 1536k 201401.dat
   ```

4. 디렉토리와 모든 하위 디렉토리의 파일을 반복적으로 세그먼트하려면 *segment [-s stage_ahead] -l segment_size -r directory-path-name* 명령을 사용합니다. 여기서 *directory-path-name*은 시작 디렉토리의 경로 및 이름입니다.

예제에서는 1메가바이트(*1m*) 세그먼트 크기를 사용하여 */hsm/hsmfs1/data* 디렉토리와 하위 디렉토리의 모든 파일을 세그먼트합니다.

```
user@solaris:~# segment -l 1m -r /hsm/hsmfs1/data
```

5. 여기서 중지합니다.

여러 볼륨에 걸쳐 세그먼트된 파일 분할

여러 드라이브를 가리키는 아카이브 세트에 세그먼트된 파일을 지정하여 스트라이프 I/O 용도로 구성합니다. 다음과 같이 하십시오.

1. 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. 텍스트 편집기에서 */etc/opt/SUNWsamfs/archiver.cmd* 파일을 엽니다.

   예제에서는 *vi* 편집기를 사용하여 파일을 엽니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/archiver.cmd
# Configuration file for Oracle HSM archiving file systems ...
```

3. 여러 드라이브에 걸쳐 세그먼트된 파일을 분할하려면 세그먼트된 파일을 포함하는 각 아카이브 세트의 복사본마다 적어도 2개의 드라이브를 사용하도록 지정합니다. *archiver.cmd* 파일에서 *params* 섹션을 찾습니다. 각 복사본의 매개변수에 *-drives number* 매개변수가 포함되는지 확인합니다. 여기서 *number*는 *2* 또는 기타 숫자입니다. 필요한 변경을 수행하고 파일을 저장하고 편집기를 닫습니다.

   예제에서는 *archiver.cmd* 파일에서 모든 구성된 아카이브 세트의 복사본 3부에 대해 두 개의 드라이브를 지정합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/archiver.cmd
# Configuration file for Oracle HSM archiving file systems ...
...
#------------------------------------------------------------------------
# Copy Parameters
params
allsets -sort path -offline_copy stageahead -reserve set
allsets.1 -startage 10m -drives 2
allsets.2 -startage 24h -drives 2
allsets.3 -startage 48h -drives 2
endparams
...
:wq
```

```
root@solaris:~#
```

4. *archiver.cmd* 파일에 오류가 있는지 확인합니다. *archiver -lv* 명령을 사용합니다.

   archiver -lv 명령은 *archiver.cmd* 파일을 화면에 출력하고 발견된 오류가 없으면 구성 보고서를 생성합니다. 그렇지 않고 오류가 발견되면 작업을 중지합니다.

```
root@solaris:~# archiver -lv
Reading '/etc/opt/SUNWsamfs/archiver.cmd'.
...
 Total space available:  300T
root@solaris:~#
```

5. Oracle HSM 소프트웨어에 *archiver.cmd* 파일을 다시 읽고 그에 따라 재구성하도록 지시합니다. */opt/SUNWsamfs/sbin/samd config* 명령을 사용합니다.

```
root@solaris:~# samd config
```

6. 여기서 중지합니다.

큰 데이터 세트를 위해 이동식 매체 파일 사용

Oracle HSM 이동식 매체 파일은 전적으로 이동식 매체에 상주하므로 절대로 파일 시스템 디스크 캐시의 공간을 차지하지 않습니다. 파일 시스템은 이동식 매체 파일을 메모리로 직접 읽습니다. 따라서 스토리지 매체가 파일 크기를 전혀 제한하지 않습니다. 단일 매체 카트리지의 용량을 초과하는 이동식 파일은 여러 카트리지에 걸쳐 볼륨 오버플로우 파일이 될 수 있습니다. 파일 시스템은 순차적으로 데이터를 읽고 매체에 씁니다.

모든 면에서 이동식 매체 파일은 표준 UNIX 파일과 비슷합니다. 권한, 사용자 이름, 그룹 이름, 파일 크기가 있습니다. 사용자나 응용 프로그램이 이동식 매체 파일을 요청하면 시스템이 자동으로 해당하는 볼륨을 마운트하고, 사용자는 마치 데이터가 디스크에 있는 것처럼 메모리에서 데이터를 액세스할 수 있습니다. 그러나 이동식 매체 파일은 다른 Oracle HSM 파일과 두 가지 측면에서 다릅니다. 즉, Oracle Hierarchical Storage Manager 소프트웨어로 아카이브되지 않고 NFS를 통해 지원되지 않습니다.

Oracle Hierarchical Storage Manager 소프트웨어는 이동식 매체 파일을 관리하지 않습니다. 파일은 절대로 아카이브되거나 해제되지 않으며, 파일을 포함하는 매체는 재활용할 수 없습니다. 따라서 아카이빙 이외의 목적으로 이동식 매체를 사용해야 하는 경우 이동식 매체 파일이 유용합니다. 이 파일은 Oracle HSM 구성 및 메타데이터 덤프 파일을 백업하는 이동식 재해 복구 볼륨을 만드는 데 이상적입니다. 또한 외래 볼륨(다른 응용 프로그램이 만든 볼륨)을 읽기 전용으로 로드하고 파일을 메모리로 읽어들이면 볼륨 데이터를 이동식 매체 파일로 읽을 수 있습니다.

이동식 매체 파일은 해제할 수 없고 연관된 볼륨을 재활용할 수 없으므로 일반적으로 이동식 매체 파일은 아카이브 복사본과 섞이지 않도록 전용 볼륨에 격리해야 합니다.

이동식 매체 또는 볼륨 오버플로우 파일 만들기

1. 파일 시스템 호스트에 로그인합니다.

   ```
   user@solaris:~#
   ```

2. 이동식 매체 파일을 위해 Oracle HSM 파일 시스템, 경로, 파일 이름을 선택합니다.

   일단 이동식 매체 파일이 만들어지면 파일 시스템이 이동식 매체의 데이터를 사용하여 이 경로 및 파일 이름에 대한 요청을 처리합니다.

3. 이동식 매체 파일을 만듭니다. *request -m media-type -v volume-specifier data-file* 명령을 사용합니다. 여기서 *mediatype*은 부록 A. 장비 유형 용어집에 나열된 2자 매체 유형 코드 중 하나이고, *data-file*은 이동식 매체 파일에 대해 선택한 경로 및 이름이고, *volume-specifier*는 다음 중 하나입니다.

   - 볼륨 일련 번호 또는 슬래시로 구분된 볼륨 일련 번호 목록

     첫번째 예제에서 LTO(*li*) 볼륨 *VOL080*에 *file1*을 만듭니다.

     ```
     user@solaris:~# request -m li -v VOL080 /hsm/hsmfs1/data/file1
     ```

     두번째 예제에서 LTO(*li*) 볼륨 *VOL080*, *VOL082*, *VOL098*에 *file2*를 만듭니다.

     ```
     user@solaris:~# request -m li -v VOL081/VOL082/VOL098 /hsm/hsmfs1/data/file2
     ```

   - *-l volume-list-file*. 여기서 *volume-list-file*은 각 라인의 단일 볼륨 일련 번호를 나열하는 파일의 경로 및 이름입니다. 선택적으로 공백과 지정된 볼륨에서 시작 위치를 지정하는 10진수 또는 16진수(16진수는 앞에 *0x*가 붙음)를 지정합니다.

     예제에서는 *vi* 편집기를 사용해서 *vsnsfile3* 파일에 나열된 LTO(*li*) 볼륨에 *file3*을 만듭니다.

     ```
     user@solaris:~# vi vsnsfile3
     VOL180
     VOL181
     VOL182
     :wq
     user@solaris:~# request -m li -v -l vsnsfile3 /hsm/hsmfs1/data/file3
     ```

4. 여기서 중지합니다.

외래 테이프 볼륨을 이동식 매체 파일로 읽기

1. 파일 시스템 호스트에 로그인합니다.

   ```
   user@solaris:~#
   ```

2. 외래 테이프에 바코드가 부착되고 쓰기 금지 상태이고 읽기 전용으로 열려 있고 *0*에 위치하는지 확인합니다.
3. 이동식 매체 파일을 위해 Oracle HSM 파일 시스템, 경로, 파일 이름을 선택합니다.

   이동식 매체 파일이 생성된 다음에는 파일 시스템이 외부 테이프의 데이터를 사용해서 이 경로 및 파일 이름에 대한 요청을 처리합니다.
4. *-N*(외래 매체) 옵션을 사용하여 이동식 매체 파일을 만듭니다. *request -m media-type -N -v volume-serial-number data-file* 명령을 사용합니다. 설명:
   - *mediatype*은 부록 A. 장비 유형 용어집에 나열된 2자 매체 유형 코드 중 하나입니다.
   - *volume-serial-number*는 외부 테이프의 볼륨 일련 번호입니다.
   - *data-file*은 이동식 매체 파일의 경로 및 이름입니다.

   예제에서는 외래 LTO(*li*) 볼륨 *FOR991*에 이동식 매체 파일을 만듭니다.

   ```
   user@solaris:~# request -m li -N -v FOR991 /hsm/hsmfs1/foreignfile
   ```

5. 여기서 중지합니다.

LTFS(Linear Tape File System) 볼륨 작업

Linear Tape File System은 순차 액세스 테이프 매체의 데이터를 파일 시스템에 구성하는 자체 설명 테이프 형식이므로, 파일이 랜덤 액세스 디스크에 있을 때와 같이 파일에 액세스할 수 있습니다. Oracle HSM에서는 LTFS에 폭넓은 지원을 제공합니다. Oracle HSM 파일 시스템에서 LTFS 파일을 사용할 수 있으며 소프트웨어에 LTFS 매체를 만들기/액세스/관리하는 도구가 제공됩니다.

이 절에서는 다음 항목을 다룹니다.

- LTFS 매체를 라이브러리로 가져오기
- Oracle HSM 파일 시스템에 LTFS 디렉토리 및 파일 연결
- Oracle HSM 소프트웨어를 사용하여 LTFS 매체 액세스
- Oracle HSM 소프트웨어를 사용하여 LTFS 매체 관리

LTFS 매체를 라이브러리로 가져오기

Oracle HSM 소프트웨어는 자동으로 LTFS 매체를 인식합니다. 따라서 다른 매체와 마찬가지로 *samimport* 명령을 사용하여 LTFS 볼륨을 가져올 수 있습니다. 추가 정보는 "이동식 매체 가져오기 및 내보내기" 및 *samimport* 매뉴얼 페이지를 참조하십시오.

Oracle HSM 파일 시스템에 LTFS 디렉토리 및 파일 연결

Oracle HSM 소프트웨어는 LTFS(Linear Tape File System) 디렉토리 및 파일을 Oracle HSM 파일 시스템에 연결할 수 있으므로 마치 Oracle HSM 파일처럼 액세스하고 관리할 수 있습니다. 소프트웨어는 LTFS 메타 데이터를 LTFS 볼륨에서 Oracle HSM 파일 시스템의

빈 디렉토리로 복사합니다. 이 메타데이터를 사용하여 Oracle HSM는 아카이브된 Oracle HSM 파일과 마찬가지로 LTFS 매체 및 파일을 관리할 수 있습니다. 사용자가 LTFS 파일에 액세스하거나 LTFS 메타데이터가 제자리에 놓이자마자 LTFS 매체에서 Oracle HSM 디스크 캐시로 파일이 스테이지됩니다. Oracle HSM 파일 시스템의 아카이빙 및 공간 관리 정책이 다른 Oracle HSM 파일과 마찬가지로 적용됩니다.

이 절에서는 다음 작업을 설명합니다.

- LTFS 파일을 요구 시 액세스 가능하도록 만들기
- LTFS 파일을 디스크 캐시에서 즉시 액세스 가능하도록 만들기

LTFS 파일을 요구 시 액세스 가능하도록 만들기

LTFS 파일을 Oracle HSM 파일 시스템에 연결할 때 Oracle HSM 소프트웨어는 LTFS 볼륨의 파일 시스템 메타데이터를 Oracle HSM 파일 시스템의 지정된 디렉토리로 복사합니다. 그러면 사용자가 파일에 액세스할 때 디스크 캐시로 파일이 스테이지됩니다. LTFS 파일을 연결하려면 다음과 같이 하십시오.

1. 파일 시스템 호스트에 로그인합니다.

   ```
   user@solaris:~#
   ```

2. LTFS 파일을 호스트하는 Oracle HSM 파일 시스템에서 LTFS 메타데이터를 보유할 디렉토리를 만듭니다.

   예제에서는 파일 시스템 마운트 지점 */hsm/hsmfs1* 아래에 *ltfs1/* 디렉토리를 만듭니다.

   ```
   user@solaris:~# mkdir /hsm/hsmfs1/ltfs1
   user@solaris:~#
   ```

3. Oracle HSM 파일 시스템에 LTFS 파일을 연결합니다. *samltfs attach LTFS-media-type.LTFS-volume-serial-number SAMQFS-directory* 명령을 사용합니다. 설명:
   - *LTFS-media-type*은 LTFS 데이터를 보유한 매체 유형에 대한 2자 매체 유형 코드입니다(부록 A. 장비 유형 용어집 참조).
   - *LTFS-volume-serial-number*는 LTFS 볼륨의 6자 영숫자 볼륨 일련 번호입니다.
   - 지정된 매체 유형 및 볼륨 일련 번호는 카탈로그에 LTFS 볼륨으로 나열되는 볼륨을 식별합니다.

     Oracle HSM 카탈로그에서 LTFS 매체는 레이블 없이 *non-SAM* 및 *tfs*로 표시됩니다.
   - *SAMQFS-directory*는 LTFS 메타데이터를 보유할 디렉토리의 경로 및 이름입니다.

   예제에서는 LTO(*li*) 볼륨 *TFS233*을 연결합니다.

```
user@solaris:~# samltfs attach li.TFS233 /hsm/hsmfs1/ltfs1
user@solaris:~#
```

4. 여기서 중지합니다.

LTFS 파일을 디스크 캐시에서 즉시 액세스 가능하도록 만들기

LTFS 파일을 Oracle HSM 파일 시스템으로 입수할 때 Oracle HSM 소프트웨어는 LTFS 볼륨의 파일 시스템 메타데이터를 Oracle HSM 파일 시스템의 지정된 디렉토리로 복사하고 즉시 모든 파일을 디스크 캐시로 스테이지합니다. LTFS 파일을 입수하려면 다음과 같이 하십시오.

1. 파일 시스템 호스트에 로그인합니다.

   ```
   user@solaris:~#
   ```

2. LTFS 파일을 호스트하는 Oracle HSM 파일 시스템에서 LTFS 메타데이터를 보유할 디렉토리를 만듭니다.

   예제에서는 파일 시스템 마운트 지점 */hsm/hsmfs1* 아래에 *ltfs2/* 디렉토리를 만듭니다.

   ```
   user@solaris:~# mkdir /hsm/hsmfs1/ltfs2
   user@solaris:~#
   ```

3. LTFS 파일을 Oracle HSM 파일 시스템으로 입수합니다. *samltfs ingest LTFS-media-type.LTFS-volume-serial-number SAMQFS-directory* 명령을 사용합니다. 설명:
   - *LTFS-media-type*은 LTFS 데이터를 보유한 매체 유형에 대한 2자 매체 유형 코드입니다(부록 A. 장비 유형 용어집 참조).
   - *LTFS-volume-serial-number*는 LTFS 볼륨의 6자 영숫자 볼륨 일련 번호입니다.
   - 지정된 매체 유형 및 볼륨 일련 번호는 카탈로그에 LTFS 볼륨으로 나열되는 볼륨을 식별합니다.

     Oracle HSM 카탈로그에서 LTFS 매체는 레이블 없이 *non-SAM* 및 *tfs*로 표시됩니다.
   - *SAMQFS-directory*는 LTFS 메타데이터를 보유한 디렉토리의 경로 및 이름입니다.

   예제에서는 LTO(*li*) 볼륨 *TFS234*를 입수합니다.

   ```
   user@solaris:~# samltfs ingest li.TFS234 /hsm/hsmfs1/ltfs2
   user@solaris:~#
   ```

4. 여기서 중지합니다.

Oracle HSM 소프트웨어를 사용하여 LTFS 매체 액세스

Oracle HSM 소프트웨어는 LTFS 매체를 로드/언로드하고 Oracle HSM *defaults.conf* 파일에 지정된 LTFS 마운트 지점을 사용하여 호스트에 LTFS 파일 시스템을 마운트/마운트 해제할 수 있습니다.

- LTFS 볼륨을 테이프 드라이브에 로드 및 LTFS 파일 시스템 마운트
- LTFS 파일 시스템 마운트 해제 및 테이프 드라이브에서 볼륨 언로드
- LTFS 구성 및 상태 정보 표시.

LTFS 볼륨을 테이프 드라이브에 로드 및 LTFS 파일 시스템 마운트

1. 파일 시스템 호스트에 로그인합니다.

   ```
   user@solaris:~#
   ```

2. LTFS 볼륨을 테이프 드라이브에 로드하고 *defaults.conf* 파일에 지정된 마운트 지점에 파일 시스템을 마운트합니다. *samltfs load LTFS-media-type.LTFS-volume-serial-number* 명령을 사용합니다. 설명:
   - *LTFS-media-type*은 LTFS 데이터를 보유한 매체 유형에 대한 2자 매체 유형 코드입니다(부록 A. 장비 유형 용어집 참조).
   - *LTFS-volume-serial-number*는 LTFS 볼륨의 6자 영숫자 볼륨 일련 번호입니다.
   - 지정된 매체 유형 및 볼륨 일련 번호는 카탈로그에 LTFS 볼륨으로 나열되는 볼륨을 식별합니다.

     Oracle HSM 카탈로그에서 LTFS 매체는 레이블 없이 *non-SAM* 및 *tfs*로 표시됩니다.

   예제에서는 LTO(*li*) 볼륨 *TFS434*를 로드하고 *defaults.conf* 파일에 지정된 디렉토리 */mnt/ltfs*에 마운트합니다.

   ```
   user@solaris:~# samltfs load li.TFS234
   ```

3. 여기서 중지합니다.

LTFS 파일 시스템 마운트 해제 및 테이프 드라이브에서 볼륨 언로드

1. 파일 시스템 호스트에 로그인합니다.

   ```
   user@solaris:~#
   ```

2. LTFS 파일 시스템을 마운트 해제하고 테이프 드라이브에서 해당 볼륨을 언로드합니다. *samltfs unload LTFS-media-type.LTFS-volume-serial-number* 명령을 사용합니다. 설명:

- *LTFS-media-type*은 LTFS 데이터를 보유한 매체 유형에 대한 2자 매체 유형 코드입니다(부록 A. 장비 유형 용어집 참조).
- *LTFS-volume-serial-number*는 LTFS 볼륨의 6자 영숫자 볼륨 일련 번호입니다.
- 지정된 매체 유형 및 볼륨 일련 번호는 카탈로그에 LTFS 볼륨으로 나열되는 LTFS 볼륨을 식별합니다.

  Oracle HSM 카탈로그에서 LTFS 매체는 레이블 없이 *non-SAM* 및 *tfs*로 표시됩니다.

예제에서는 LTFS 파일 시스템을 마운트 해제하고 LTO(*li*) 볼륨 *TFS435*를 언로드합니다.

```
user@solaris:~# samltfs unload li.TFS435
```

3. 여기서 중지합니다.

Oracle HSM 소프트웨어를 사용하여 LTFS 매체 관리

Oracle HSM 소프트웨어는 LTFS 매체를 만들기/지우기/검증하기에 필요한 기본 도구를 제공합니다.

- LTFS 파일 시스템으로 볼륨 포맷
- LTFS 데이터 지우기 및 볼륨에서 LTFS 포맷/분할 제거
- LTFS 파일 시스템의 무결성 검사

LTFS 파일 시스템으로 볼륨 포맷

1. 파일 시스템 호스트에 로그인합니다.

```
user@solaris:~#
```

2. LTFS 파일 시스템의 이동식 매체 볼륨을 분할하고 포맷합니다. *samltfs mkltfs media-type.volume-serial-number* 명령을 사용합니다. 설명:
   - *media-type*은 LTFS 호환 가능 매체 유형에 대한 2자 매체 유형 코드입니다(부록 A. 장비 유형 용어집 참조).
   - *volume-serial-number*는 볼륨의 6자 영숫자 볼륨 일련 번호입니다.

   예제에서는 LTO(*li*) 볼륨 *VOL234*를 분할하고 LTFS 볼륨으로 포맷합니다.

```
user@solaris:~# samltfs mkltfs li.VOL234
```

3. 여기서 중지합니다.

LTFS 데이터 지우기 및 볼륨에서 LTFS 포맷/분할 제거

1. 파일 시스템 호스트에 로그인합니다.

   ```
   user@solaris:~#
   ```

2. LTFS 볼륨을 지우고 일반 용도로 복원합니다. *samltfs unltfs media-type.volume-serial-number* 명령을 사용합니다. 설명:
   - *media-type*은 LTFS 호환 가능 매체 유형에 대한 2자 매체 유형 코드입니다(부록 A. 장비 유형 용어집 참조).
   - *volume-serial-number*는 볼륨의 6자 영숫자 볼륨 일련 번호입니다.

   예제에서는 LTFS 파일 시스템 데이터 및 메타데이터를 지우고 LTO(*li*) 볼륨 *VOL234*에서 분할 영역을 제거합니다.

   ```
   user@solaris:~# samltfs unltfs li.VOL234
   ```

3. 여기서 중지합니다.

LTFS 파일 시스템의 무결성 검사

1. 파일 시스템 호스트에 로그인합니다.

   ```
   user@solaris:~#
   ```

2. LTFS 파일 시스템의 무결성을 검사합니다. *samltfs ltfsck LTFS-media-type.LTFS-volume-serial-number* 명령을 사용합니다. 설명:
   - *LTFS-media-type*은 LTFS 데이터를 보유한 매체 유형에 대한 2자 매체 유형 코드입니다(부록 A. 장비 유형 용어집 참조).
   - *LTFS-volume-serial-number*는 LTFS 볼륨의 6자 영숫자 볼륨 일련 번호입니다.
   - 지정된 매체 유형 및 볼륨 일련 번호는 카탈로그에 LTFS 볼륨으로 나열되는 LTFS 볼륨을 식별합니다.

     Oracle HSM 카탈로그에서 LTFS 매체는 레이블 없이 *non-SAM* 및 *tfs*로 표시됩니다.

   예제에서는 LTO(*li*) 볼륨 *VOL234*에서 LTFS 파일 시스템을 검사합니다.

   ```
   user@solaris:~# samltfs ltfsck li.VOL234
   ```

3. 여기서 중지합니다.

LTFS 구성 및 상태 정보 표시

LTFS의 구성 및 상태를 표시하려면 *samltfs status* 명령을 사용합니다.

```
user@solaris:~# samltfs status
```

SMB/CIFS 공유에서 디렉토리 및 파일 관리

이 절에서는 다음 항목을 다룹니다.

- SMB/CIFS 공유에서 시스템 속성 관리
- 액세스 제어 목록 관리.

SMB/CIFS 공유에서 시스템 속성 관리

시스템 속성은 Oracle HSM 파일을 Microsoft Windows 파일 시스템에서 해석할 수 있는 비-UNIX 메타데이터와 연관시켜서 SMB/CIFS 파일 공유를 지원합니다. 이 절은 Oracle HSM에서 지원되는 시스템 속성에 대한 간략한 개요로 시작합니다. 그리고 다음 작업을 위한 기본 지침을 제공합니다.

- 시스템 속성 표시
- 시스템 속성 수정.

Oracle HSM에서 지원되는 시스템 속성

시스템 속성은 부울(true 또는 false) 값으로, 속성 *name*과 *true* 값으로 표현하거나 이름의 부정인 *noname*과 *false* 값으로 표현합니다. Oracle HSM은 SMB/CIFS 파일 공유 지원 시 다음 시스템 속성을 제공합니다.

- *appendonly*는 사용자가 파일에 데이터만 첨부할 수 있음을 의미합니다. *noappendonly*는 이 제한이 아무 효력이 없음을 의미합니다.
- *archive*는 마지막 복사 또는 백업된 이후 파일이 변경되었음을 의미합니다. *noarchive*는 마지막 복사 또는 백업된 이후 파일이 변경되지 않았음을 의미합니다. Oracle HSM은 현재 이 속성을 사용하지 않습니다.
- *hidden*은 파일이 기본적으로 파일 목록에 표시되지 않음을 의미합니다. *nohidden*은 파일이 기본적으로 표시됨을 의미합니다.
- *immutable*은 디렉토리나 파일의 내용을 변경하거나 삭제할 수 없음을 의미합니다. *noimmutable*은 디렉토리나 파일을 변경하거나 삭제할 수 있음을 의미합니다.
- *nodump*는 파일을 백업할 수 없음을 의미합니다. *nonodump*는 파일을 백업할 수 있음을 의미합니다. Oracle Solaris는 이 속성을 사용하지 않습니다.
- *nounlink*는 파일이나 디렉토리의 내용을 삭제하거나 이름을 바꿀 수 없음을 의미합니다. *nonounlink*는 파일이나 디렉토리의 내용을 삭제하거나 이름을 바꿀 수 있음을 의미합니다.
- *offline*은 파일이 Oracle HSM 파일 시스템에서 해제되었음을 의미합니다. Microsoft Windows 시스템은 파일 미리보기를 시행하지 않습니다. *nooffline*은 파일이 온라인이고 Oracle HSM 파일 시스템에서 해제되지 않았음을 의미합니다.
- *readonly*는 파일을 삭제하거나 수정할 수 없음을 의미합니다. *noreadonly*는 파일을 삭제하거나 수정할 수 있음을 의미합니다. 디렉토리에 적용할 경우 속성이 무시됩니다.

- *sparse*는 희소 파일을 지원하지 않는 파일 시스템으로 파일을 복사하거나 액세스할 때 파일 시스템에서 복원하는 범위까지 제로가 줄어들면서, 저장된 파일에 비제로 데이터만 있음을 의미합니다. *nosparse*는 희소 파일이 아님을 의미합니다.
- *system*은 파일이 Microsoft Windows 운영체제에 매우 중요하고, 변경하거나 삭제하지 않아야 하며, 기본적으로 목록에 표시되지 않아야 함을 의미합니다. *nosystem*은 파일이 시스템 파일이 아님을 의미합니다.

시스템 속성 표시

Oracle HSM 파일의 시스템 속성을 보려면 Solaris 명령 *ls -/v file*을 사용합니다. 여기서 *file*은 파일의 경로 및 이름입니다.

예제에서는 */hsm/hsmfs1/documents/master-plan.odt* 파일의 시스템 속성을 나열합니다.

```
user@solaris:~# ls -/v /hsm/hsmfs1/documents/master-plan.odt
-rw-r--r--   1 root root  40560 Mar 4 15:52 /hsm/hsmfs1/documents/master-plan.odt
{archive,nohidden,noreadonly,nosystem,noappendonly,nonodump,noimmutable,nonounlink,
 nooffline,nosparse}
user@solaris:~#
```

시스템 속성 수정

파일의 시스템 속성값을 지정된 값으로 변경하려면 Solaris 명령 *chmod S +v{attributes)*를 사용합니다. 여기서 *attributes*는 콤마로 구분된 Oracle HSM에서 지원되는 시스템 속성 목록입니다.

구문 및 사용 가능한 옵션에 대한 종합적 설명은 *chmod* 매뉴얼 페이지를 참조하십시오. 예제에서는 아카이브 속성을 *noarchive*(false)에서 *archive*(true)로 변경합니다.

```
root@solaris:~# ls -/v /hsm/hsmfs1/documents/master-plan.odt
-r-xr-xr-x 1 root root 40561 Mar 4 15:52 /hsm/hsmfs1/documents/master-plan.odt
{noarchive,nohidden,readonly,nosystem,noappendonly,nonodump,noimmutable,
 nonounlink,offline,nosparse}
root@solaris:~# chmod S+v{archive} /hsm/hsmfs1/documents/master-plan.odt
root@solaris:~# ls -/v /hsm/hsmfs1/documents/master-plan.odt
-r-xr-xr-x 1 root root 40561 Mar 4 15:52 /hsm/hsmfs1/documents/master-plan.odt
{archive,nohidden,readonly,nosystem,noappendonly,nonodump,noimmutable,
 nonounlink,offline,nosparse}
```

액세스 제어 목록 관리

ACL(액세스 제어 목록)은 파일이나 디렉토리에 액세스 권한을 정의하는 테이블입니다. 테이블의 각 레코드나 ACE(액세스 제어 항목)는 특정 사용자, 그룹 또는 사용자/그룹 클래스의 액세스 권한을 정의합니다. 기본적으로 Oracle HSM 릴리스 6.1으로 만든 새 파일 시스템은 NFS(네트워크 파일 시스템) 버전 4 및 Solaris 11에 도입된 ACL(액세스 제어 목록) 구현을 사용합니다.

Solaris ACL 관리, 구문, 사용법에 대한 종합적 설명은 이 문서에서 다루지 않습니다. 자세한 내용은 *docs.oracle.com*의 *Oracle Solaris 11.1 Information Library*에서 제공되는

Oracle Solaris 11.1 관리*: ZFS* 파일 시스템의 "ACL 및 속성을 사용하여 Oracle Solaris ZFS 파일 보호" 장을 참조하십시오. 또한 Solaris *`ls`* 및 *`chmod`* 매뉴얼 페이지를 참조하십시오.

5

5장. 라이브러리, 매체 및 드라이브 관리

이 장에서는 다음 항목을 다룹니다.

- 자동화된 매체 라이브러리 관리
- 드라이브 관리
- 이동식 매체 관리.

자동화된 매체 라이브러리 관리

이 절에서는 라이브러리 유지 관리 및 관리와 연관된 기본 작업을 다룹니다.

- 라이브러리를 온라인/오프라인으로 전환
- 이동식 매체 가져오기 및 내보내기
- 라이브러리 카탈로그 유지 관리
- 라이브러리에 드라이브가 설치되는 순서 결정.

라이브러리를 온라인/오프라인으로 전환

- 라이브러리를 오프라인으로 전환
- 라이브러리를 온라인으로 전환.

라이브러리를 오프라인으로 전환

한 라이브러리에서만 Oracle HSM 작동을 중지하거나 라이브러리 전원을 꺼야 하는 경우 아래 설명된 대로 라이브러리를 오프라인으로 전환하는 작업부터 시작합니다.

1. 파일 시스템 호스트에 *root*로 로그인합니다.

   ```
   root@solaris:~#
   ```

2. 활성 아카이빙 및 스테이징 작업을 마치고 새 작업이 시작되지 않도록 합니다. "아카이빙 및 스테이징 프로세스 유휴 설정"을 참조하십시오.
3. 드라이브 및 라이브러리 작동을 중지합니다. "아카이빙 및 스테이징 프로세스 중지"를 참조하십시오.

4. 라이브러리를 오프라인으로 가져옵니다. *samcmd off library-equipment-number* 명령을 사용합니다. 여기서 *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 라이브러리에 지정된 장비 순서 번호입니다.

   라이브러리를 *off* 상태에 놓으면 I/O 작업이 중지되고 Oracle HSM 소프트웨어 통제로부터 라이브러리가 제거됩니다. 전원이 꺼지지 않은 드라이브는 모두 *on* 상태로 남습니다. 예제에서는 *800* 라이브러리를 오프라인으로 가져오고 *samcmd c*를 사용하여 결과를 확인합니다.

```
root@solaris:~# samcmd off 800
root@solaris:~# samcmd c
Device configuration samcmd     5.4 14:34:04 Mar  7 2014
samcmd on hsmfs1host
Device configuration:
ty   eq state   device_name                       fs   family_set
sn  800 off     /dev/scsi/changer/c1t2d0          800  lib800
li  801 on      /dev/rmt/0cbn                     800  lib800
li  802 on      /dev/rmt/1cbn                     800  lib800
li  803 on      /dev/rmt/2cbn                     800  lib800
li  804 on      /dev/rmt/3cbn                     800  lib800
hy  900 on      historian                         900
root@solaris:~#
```

5. 준비가 되었으면 라이브러리를 온라인으로 전환을 수행합니다.

라이브러리를 온라인으로 전환

1. 파일 시스템 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. 라이브러리를 온라인으로 가져옵니다. *samcmd on library-equipment-number* 명령을 사용합니다. 여기서 *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 라이브러리에 지정된 장비 순서 번호입니다.

   라이브러리가 온라인이 됩니다. Oracle HSM 소프트웨어는 장치 상태를 질의하고 필요에 따라 카탈로그를 업데이트합니다. 예제에서는 *800* 라이브러리를 온라인으로 가져오고 *samcmd c*를 사용하여 결과를 확인합니다.

```
root@solaris:~# samcmd on 800
root@solaris:~# samcmd c
Device configuration samcmd     5.4 15:04:14 Mar  7 2014
samcmd on hsmfs1host
Device configuration:
ty   eq state   device_name                       fs   family_set
sn  800 on      /dev/scsi/changer/c1t2d0          800  lib800
```

```
li  801 on      /dev/rmt/0cbn                   800  lib800
li  802 on      /dev/rmt/1cbn                   800  lib800
li  803 on      /dev/rmt/2cbn                   800  lib800
li  804 on      /dev/rmt/3cbn                   800  lib800
hy  900 on      historian                       900
root@solaris:~#
```

3. 여기서 중지합니다.

이동식 매체 가져오기 및 내보내기

대부분의 자동화된 라이브러리에는 실제로 라이브러리에 들어가지 않고도 매체 카트리지를 추가/제거할 수 있는 적재 베이가 있습니다. 공급업체에 따라 이를 메일박스, 메일슬롯, MAP(매체 액세스 포트), CAP(카트리지 액세스 포트)라고 부릅니다. 이 라이브러리 유형을 사용하면 Oracle HSM 명령을 통해 다음 작업을 수행할 수 있습니다.

- 이동식 매체 카트리지 가져오기
- 이동식 매체 카트리지 내보내기.

라이브러리에 메일박스가 없으면 라이브러리 공급업체 설명서와 로컬 사이트 정책에서 라이브러리 매체 추가 및 제거 지침을 참조하십시오. 라이브러리가 변경 후 다시 초기화되고 그 내용을 감사할 때 Oracle HSM 소프트웨어는 라이브러리와 내역기 카탈로그를 자동으로 업데이트합니다.

이동식 매체 카트리지 가져오기

Oracle HSM 소프트웨어를 시작할 때 라이브러리 메일박스에 매체 카트리지가 있으면 소프트웨어는 이를 자동으로 라이브러리에 로드합니다. 소프트웨어가 실행 중인 경우 다음 절차를 사용하여 언제든지 메일박스에서 매체를 가져올 수 있습니다.

1. 라이브러리 공급업체의 지침에 따라 메일박스에 매체 카트리지를 놓습니다.
2. 파일 시스템 호스트에 *root*로 로그인합니다.

   ```
   root@solaris:~#
   ```

3. 카트리지를 자동화된 라이브러리로 가져옵니다. *samimport library-equipment-number* 명령을 사용합니다. 여기서 *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 라이브러리에 대해 지정된 장비 순서 번호입니다.

   Oracle HSM 소프트웨어는 매체를 스토리지 슬롯에 지정하고 해당 위치를 카탈로그에 저장합니다. 예제에서는 *800* 라이브러리로 매체를 가져옵니다.

   ```
   root@solaris:~# samimport 800
   ```

4. 여기서 중지합니다.

이동식 매체 카트리지 내보내기

1. 파일 시스템 호스트에 *root*로 로그인합니다.

   ```
   root@solaris:~#
   ```

2. 필요한 경우, 카트리지를 내보내기 전에 카탈로그 레코드에 정보 메모를 추가합니다. *chmed -I "note" identifier* 명령을 사용합니다. 여기서 *note*는 최대 128자 문자열이고 *identifier*는 다음 중 하나입니다.

   - *mediatype.volume-serial-number*. 여기서 *mediatype*은 부록 A. 장비 유형 용어집에 나열된 2자의 매체 유형 코드 중 하나이고 *volume-serial-number*는 라이브러리 내에서 볼륨을 고유하게 식별하는 6자의 영숫자 문자열입니다.
   - *library-equipment-number:slot*, 여기서 *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 자동화된 테이프 라이브러리에 대해 지정된 장비 순서 번호이고 *slot*은 카트리지가 라이브러리 내에 있는 슬롯 주소입니다.

   볼륨을 내보낸 후에 내역기 카탈로그에 메모가 보관됩니다. 예제에서는 LTO(*li*) 카트리지 *VOL054*의 카탈로그 항목에 메모를 추가합니다.

   ```
   root@solaris:~# chmed -I "To vault 20150411" li.VOL054
   ```

3. 카트리지를 지정된 스토리지 슬롯에서 메일박스로 이동하려면 *samexport library-equipment-number:slot* 명령을 사용합니다. 여기서 *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 자동화된 테이프 라이브러리에 대해 지정된 장비 순서 번호이고 *slot*은 라이브러리 내에서 카트리지가 있는 슬롯 주소입니다.

   예제에서는 *800* 라이브러리의 *11* 슬롯에 위치한 자기 테이프 카트리지를 내보냅니다.

   ```
   root@solaris:~# samexport 800:11
   ```

4. 지정된 카트리지를 메일박스로 이동하려면 *samexport mediatype.volume-serial-number* 명령을 사용합니다. 여기서 *mediatype*은 부록 A. 장비 유형 용어집에 나열된 2자 매체 유형 코드 중 하나이고 *volume-serial-number*는 라이브러리 내에서 볼륨을 고유하게 식별하는 영숫자 문자열입니다.

   Oracle HSM 소프트웨어는 Oracle HSM **historian(내역기)**로 유지 관리되는 카탈로그에 카트리지를 추가합니다. 예제에서는 LTO(*li*) 테이프 카트리지 *VOL109*를 내보냅니다.

   ```
   root@solaris:~# samexport li.VOL109
   ```

5. 라이브러리 공급업체의 지침에 따라 메일박스에서 매체 카트리지를 꺼냅니다.
6. 여기서 중지합니다.

라이브러리 카탈로그 유지 관리

Oracle Hierarchical Storage Manager 라이브러리 카탈로그는 자동화된 라이브러리 및 해당 컨텐츠에 관한 소프트웨어 내부 표현입니다. 자동화된 라이브러리가 직접 연결된 경우 Oracle HSM 소프트웨어는 라이브러리 및 해당 컨텐츠에 대한 모든 제어 권한을 갖습니다. 라이브러리 카탈로그 항목은 따라서 물리적 라이브러리의 슬롯에 대한 일대일 표현입니다. 자동화된 라이브러리가 네트워크 연결 방식인 경우 Oracle HSM은 라이브러리 소프트웨어가 가상 라이브러리나 라이브러리 분할 영역 형태로 사용 가능하게 만든 라이브러리 일부에만 액세스합니다. 따라서 Oracle HSM 라이브러리 카탈로그 항목에는 해당 라이브러리의 일정 부분에 대한 컨텐츠만 반영됩니다.

이 절에서는 다음 작업을 설명합니다.

- 라이브러리 카탈로그 보기
- 라이브러리 슬롯의 내용 감사
- 전체 직접 연결식 자동화된 라이브러리 감사
- 카탈로그에서 매체 오류 지우기.

라이브러리 카탈로그 보기

1. 가장 흔히 사용되는 라이브러리 카탈로그 정보를 보려면 *samcmd v library-equipment-number* 명령을 사용합니다. 여기서 *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 자동화된 테이프 라이브러리에 지정하는 장비 순서 번호입니다.

   예제에서는 *800* 라이브러리의 카탈로그를 표시합니다.

```
root@solaris:~# samcmd v 800
Robot catalog samcmd     5.4    16:45:25 Mar 14 2014
samcmd on samqfshost                          count 32
Robot VSN catalog by slot       : eq 800
slot          access time count use  flags         ty vsn
   0     2014/03/14 11:23  875   0%  -il-o-b-----  li VOL001
   1     2014/03/13 17:54  866   0%  -il-o-b-----  li VOL002
   2     2014/03/14 11:26    3   0%  -il-o-b-----  li VOL003
   3     2014/03/14 10:33    3   0%  -il-o-b-----  li VOL004
   4     2014/03/14 11:34    5   0%  -il-o-b-----  li VOL005
   5     2014/03/14 11:32    2   0%  -ilEo-b----f  li VOL006 MEDIA ERROR
   6     2014/03/13 18:07    2   0%  -il-o-b-----  li VOL007
   7     2014/03/13 18:07    1   0%  -il-o-b-----  li VOL008
   8     2014/03/13 18:07    1   0%  -il-o-b-----  li VOL009
...
  18     2014/03/13 18:16    1   0%  -il-o-b-----  li VOL019
  19     none               50   0%  -il-oCb-----  li CLN020
```

2. *samcmd v* 디스플레이를 사용하여 볼륨의 상태를 확인하려면 *flags* 열의 항목을 조사하고 아래 목록을 참조합니다.
   - *A*는 슬롯에 감사가 필요함을 의미합니다.
   - *C*는 슬롯에 청소 카트리지가 있음을 의미합니다.
   - *E*는 볼륨이 불량하거나 청소 매체가 만료되었음을 의미합니다.
   - *L*은 LTFS(Linear Tape File System) 볼륨임을 의미합니다.
   - *N*은 해당 볼륨이 Oracle HSM 형식이 아닌 외부 매체임을 의미합니다.
   - *R*은 볼륨이 읽기 전용임을 의미합니다(소프트웨어 플래그).
   - *U*는 볼륨을 사용할 수 없음을 의미합니다.
   - *W*는 볼륨이 물리적으로 쓰기 금지됨을 의미합니다.
   - *X*는 내보내기 슬롯임을 의미합니다.
   - *b*는 볼륨에 바코드가 있음을 의미합니다.
   - *c*는 볼륨 재활용이 예약되어 있음을 의미합니다.
   - *f*는 볼륨이 가득 차거나 손상된 것을 아카이버가 발견했음을 의미합니다.
   - *d*는 볼륨에 중복된 VSN(볼륨 일련 번호)이 있음을 의미합니다.
   - *l*은 볼륨에 레이블이 붙어 있음을 의미합니다.
   - *o*는 슬롯이 점유되었음을 의미합니다.
   - *p*는 높은 우선 순위 볼륨임을 의미합니다.
   - *-*는 해당 플래그가 설정되지 않았음을 의미합니다.
3. *samcmd v* 디스플레이를 사용하여 볼륨에 사용된 매체 유형을 식별하려면 *ty* 열을 참조하고 부록 A. 장비 유형 용어집 또는 *mcf* 매뉴얼 페이지에 표시된 코드를 조회합니다.
4. 카탈로그의 모든 정보를 나열하려면 *dump_cat catalog-path-name* 명령을 사용합니다. 여기서 *catalog-path-name*은 */etc/opt/SUNWsamfs/mcf* 파일에 지정된 대로 카탈로그 파일의 경로 및 이름입니다.

   예제에서는 카탈로그 파일 *catalog/800_cat*를 덤프합니다.

```
root@solaris:~# dump_cat catalog/800_cat
# audit_time Wed Dec 31 17:00:00 1969
# version 530  count 32 mediatype
#Index VSN    Barcode  Type PTOC  Access Capacity ...  LVTime LVPos
#
 0     S00001 S00001L4 li   0x747    875   512000 ...       0   0x3
 1     S00002 S00002L4 li   0x5db    866   512000 ...       0   0x3
13     S00014 S00014L4 li       0      4   512000 ...       0   0
17     S00018 S00018L4 li       0      1   512000 ...       0   0
18     S00003 S00003L4 li       0      3   512000 ...       0   0
```

5. 여기서 중지합니다.

라이브러리 슬롯의 내용 감사

이동식 매체 볼륨에 보고된 남은 공간으로 라이브러리 카탈로그를 업데이트하려면 라이브러리 슬롯을 감사합니다. *auditslot* 명령을 사용합니다.

1. 파일 시스템 호스트에 *root*로 로그인합니다.

   ```
   root@solaris:~#
   ```

2. 지정된 테이프 볼륨을 감사하려면 EOD(데이터 끝)로 건너뛰어서 사용 가능한 공간을 업데이트하고 *auditslot -e library-equipment-number:slot* 명령을 사용합니다. 여기서 *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 라이브러리에 지정하는 장비 순서 번호이고 *slot*은 라이브러리 내에서 카트리지의 위치입니다.

   auditslot 명령은 볼륨이 포함된 카트리지를 로드하고, 레이블을 읽고, 슬롯의 라이브러리 카탈로그 항목을 업데이트합니다. 일단 시작하면 EOD로 건너뛰기를 중단할 수 없으며, 특정 조건에서는 완료하는 데 몇 시간 걸릴 수 있습니다. 예제에서는 테이프 라이브러리 *800*의 *11* 슬롯을 감사합니다.

   ```
   root@solaris:~# auditslot -e 800:11
   root@solaris:~#
   ```

3. 지정된 광 볼륨을 감사하려면 *auditslot library-equipment-number:slot[:side]* 명령을 사용합니다. 여기서 *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 라이브러리에 지정하는 장비 순서 번호이고 *slot*은 라이브러리 내에서 카트리지의 위치이고 *side*(선택사항)는 양면 광 디스크의 지정된 면입니다.

   예제에서는 광 라이브러리 *700*의 *21* 슬롯의 볼륨 *1*면을 감사합니다.

   ```
   root@solaris:~# auditslot 800:21:1
   root@solaris:~#
   ```

4. 여기서 중지합니다.

전체 직접 연결식 자동화된 라이브러리 감사

전체 감사는 각 카트리지를 드라이브에 로드하고, 레이블을 읽고, 라이브러리 카탈로그를 업데이트합니다. 다음 상황에서 라이브러리를 감사합니다.

- Oracle HSM 명령을 사용하지 않고 자동화된 라이브러리에서 카트리지를 이동한 후
- 라이브러리 카탈로그를 신뢰할 수 없을 때(예를 들어 정전 이후)
- 메일박스가 없는 자동화된 라이브러리에서 카트리지를 추가, 제거, 이동했을 때

전체 감사를 수행하려면 *samcmd audit library-equipment-number* 명령을 사용합니다. 여기서 *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 라이브러리에 지정하는 장비 순서 번호입니다.

전체 감사는 매체를 포함하는 슬롯 수에 따라 시간이 오래 걸릴 수 있습니다.

예제에서는 테이프 라이브러리 *800*을 감사합니다.

```
root@solaris:~# audit 800
root@solaris:~#
```

카탈로그에서 매체 오류 지우기

Oracle HSM는 이동식 매체 카트리지 사용에 문제가 있는 경우 해당하는 카탈로그 항목에 오류 플래그를 설정합니다. 매체가 마모되었거나, 손상되었거나, 청소 매체의 경우 만료되었을 수 있습니다. 이 경우 매체를 재사용하면 안됩니다. 그러나 드라이브 결함으로 인해 매체 액세스에 문제가 생길 수도 있습니다. 이 경우 어려움 없이 매체를 재사용할 수 있습니다. 후자의 경우 카트리지의 오류 플래그를 지워야 합니다.

오류 플래그를 지우기 전에 문제의 특성을 알고 있어야 합니다. 오류 플래그는 Oracle HSM 작동과 데이터 보안에 매우 중요합니다. 카트리지에 실제로 결함이 있는 경우 이 플래그를 지우지 않아도 됩니다.

일단 확신이 들면 오류를 지우고 카트리지 사용을 시도할 수 있습니다. 다음과 같이 하십시오.

1. 파일 시스템 호스트에 *root*로 로그인합니다.

   ```
   root@solaris:~#
   ```

2. 이동식 매체 볼륨의 상태를 확인합니다. *samcmd r* 명령을 사용합니다.

   예제에서 *samcmd r* 명령은 *801* 드라이브가 LTO(*li*) 볼륨 *VOL004*에 오류 플래그를 설정했음을 보여줍니다.

   ```
   root@solaris:~# samcmd r
   Removable media status: all          samcmd 5.4        17:40:11 Mar 13 2014
   ty   eq  status      act  use  state   vsn
   li  801 -E-------r     0   0%  notrdy  VOL004  MEDIA ERROR
           MEDIA ERROR
   li  802  ---------p    0   0%  notrdy
           empty
   li  803  ---------p    0   0%  notrdy
           empty
   li  804  ---------p    0   0%  notrdy
           empty
   root@solaris:~#
   ```

3. 오류 플래그를 설정한 드라이브가 의심되면 카트리지를 언로드하고 오류 플래그를 지웁니다. *samcmd unload drive-number* 명령을 사용합니다. 여기서 *drive-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 드라이브에 대해 지정된 장비 순서 번호입니다.

   예제에서는 *801* 드라이브를 언로드합니다.

   ```
   root@solaris:~# samcmd unload 801
   ```

4. 지정된 볼륨 일련 번호 및 매체 유형에 대한 매체 오류 플래그를 지우려면 *chmed -E media-type.volume-serial-number* 명령을 사용합니다. 여기서 *mediatype*은 부록 A. 장비 유형 용어집에 나열된 2자의 매체 유형 코드 중 하나이고 *volume-serial-number*는 라이브러리 내에서 볼륨을 고유하게 식별하는 영숫자 문자열입니다.

   예제에서는 LTO(*li*) 볼륨 *VOL004*에서 오류 플래그를 지웁니다.

   ```
   root@solaris:~# chmed -E li.VOL004
      3:0 li VOL004     Ail---b-----    2.3T    2.3T    0          0 800 4  0 //
   root@solaris:~#
   ```

5. 지정된 라이브러리 슬롯에 있는 카트리지에 대한 매체 오류 플래그를 지우려면 *chmed -E library-equipment-number:slot[:disk-side]* 명령을 사용합니다. 여기서 *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일이 자동화된 테이프 라이브러리에 지정하는 장비 순서 번호이고, *slot*은 라이브러리 내에 대상 볼륨이 있는 슬롯 주소이고, 선택적인 *disk-side* 값(*1* 또는 *2*)은 양면 자기 광 디스크의 어느 한 면을 지정합니다.

   예제에서는 *800* 라이브러리의 *31* 슬롯에서 카트리지에 오류 플래그를 지웁니다.

   ```
   root@solaris:~# chmed -E 800:31
   ```

6. 변경사항이 반영되도록 라이브러리 카탈로그를 업데이트합니다. *auditslot -e library-equipment-number:slot[:disk-side]* 명령을 사용합니다.

   예제에서는 *800* 라이브러리의 *31* 슬롯을 감사하여 카탈로그를 업데이트합니다.

   ```
   root@solaris:~# auditslot -e 800:31
   root@solaris:~#
   ```

7. 다른 드라이브에서 카트리지를 마운트하고, 오류가 재발하는지 확인합니다. *samcmd load media-type.volume-serial-number* 명령을 사용합니다. 여기서 *mediatype*은 부록 A. 장비 유형 용어집에 나열된 2자 매체 유형 코드 중 하나이고 *volume-serial-number*는 라이브러리 내에서 볼륨을 고유하게 식별하는 영숫자 문자열입니다.

   ```
   root@solaris:~# samcmd load li.VOL004
   root@solaris:~#
   ```

8. 이동식 매체 볼륨의 상태를 다시 확인합니다. *samcmd r* 명령을 사용합니다.

```
root@solaris:~# samcmd r
Removable media status: all          samcmd 5.4         17:42:10 Mar 13 2014
ty   eq  status      act  use  state  vsn
li  801  ---------p    0   0%  notrdy
        empty
li  802  --l------r    0   0%  ready   VOL004
        idle
li  803  ---------p    0   0%  notrdy
        empty
li  804  ---------p    0   0%  notrdy
        empty
root@solaris:~#
```

9. 새 드라이브에서 오류가 재발하지 않으면 카트리지는 아마 괜찮을 것입니다.
10. 오류가 재발하면 이동식 매체 볼륨을 폐기하는 것을 고려하십시오.
11. 여기서 중지합니다.

내역기 카탈로그 관리

Oracle Hierarchical Storage Manager 내역기는 카탈로그가 있지만 장비는 없는 의사 라이브러리입니다. 내역기는 더 이상 Oracle HSM 직접 통제를 받지 않는 볼륨을 보관합니다. 따라서 라이브러리에서 내보내서 오프사이트 스토리지로 보낸 볼륨과 독립형 드라이브로 손수 로드한 볼륨의 레코드를 유지 관리합니다. Oracle HSM는 라이브러리에서 볼륨을 내보낼 때 내역기 카탈로그를 자동으로 업데이트합니다. 그러나 레코드를 추가/제거하고 메모를 첨부하여 수동 레코드 보관용으로 내역기를 사용할 수도 있습니다. 일반적으로 물리적 매체 라이브러리와 마찬가지 방법으로 내역기와 상호 작용합니다.

이 절에서는 다음 작업을 설명합니다.

- 내역기 카탈로그 보기
- 내역기 카탈로그에 항목 추가
- 내역기 카탈로그에서 항목 제거.

내역기 카탈로그 보기

물리적 라이브러리와 똑같은 방법으로 내역기 카탈로그를 볼 수 있습니다. *samcmd v historian-equipment-number* 명령을 사용합니다. 여기서 *historian-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 내역기에 지정하는 장비 순서 번호입니다.

예제에서는 장비 순서 번호 *900*을 가진 내역기의 카탈로그를 표시합니다.

```
root@solaris:~# samcmd v 900
```

```
Robot catalog samcmd     5.4    16:45:25 Mar 14 2014
samcmd on samqfshost                          count 32
Robot VSN catalog by slot       : eq 900
slot          access time count use  flags         ty vsn
   0     2014/03/14 11:23  875   0%  -il-o-b-----  li EXT001
   1     2014/03/13 17:54  866   0%  -il-o-b-----  li EXT002
```

내역기 카탈로그에 항목 추가

내역기 카탈로그에 항목을 추가하려면 다음과 같이 하십시오.

1. 지정된 볼륨 일련 번호에 대한 내역기 카탈로그에 항목을 추가하려면 *samimport -v volume-serial-number -m mediatype historian-equipment-number* 명령을 사용합니다. 설명:
   - *volume-serial-number*는 카탈로그 내에서 볼륨을 고유하게 식별하는 영숫자 문자열입니다.
   - *mediatype*은 부록 A. 장비 유형 용어집에 나열된 2자 매체 유형 코드 중 하나입니다.
   - *historian-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일이 내역기에 지정하는 장비 순서 번호입니다.

   예제에서는 LTO(*li*) 볼륨 *EXT003*의 레코드를 *900* 내역기 카탈로그에 추가합니다.

   ```
   root@solaris:~# samimport -v EXT003 -m li 900
   ]
   ```
2. 지정된 바코드에 대한 항목을 내역기 카탈로그에 추가하려면 *samimport -b barcode -m mediatype historian-equipment-number* 명령을 사용합니다. 여기서 *barcode*는 해당하는 물리적 카트리지에 부착된 바코드입니다.

   예제에서는 바코드 *EXT003L4*가 붙은 LTO(*li*) 볼륨의 레코드를 *900* 내역기 카탈로그에 추가합니다.

   ```
   root@solaris:~# samimport -b EXT003L4 -m li 900
   ```
3. 여기서 중지합니다.

내역기 카탈로그에서 항목 제거

내역기 카탈로그에서 항목을 제거하려면 *samexport historian-equipment-number:slot* 명령을 사용합니다. 여기서 *historian-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 내역기에 지정하는 장비 순서 번호이고 *slot*은 레코드의 내역기 슬롯 주소입니다.

예제에서는 *900* 내역기 카탈로그의 *1* 슬롯에서 *EXT002* 볼륨의 레코드를 제거합니다.

```
root@solaris:~# samcmd v 900
Robot catalog samcmd     5.4    16:45:25 Mar 14 2014
samcmd on samqfshost                          count 32
Robot VSN catalog by slot       : eq 900
```

```
slot          access time count use  flags         ty vsn
   0     2014/03/14 11:23  875   0%  -il-o-b-----  li EXT001
   1     2014/03/13 17:54  866   0%  -il-o-b-----  li EXT002
   2     2014/03/13 17:57  866   0%  -il-o-b-----  li EXT003
root@solaris:~# samexport 900:1
```

내역기 정보 업데이트

내보낸 볼륨의 처리나 상태 변경사항을 적어서 내역기 카탈로그 항목의 정보 필드를 업데이트할 수 있습니다. *chmed -I "note" identifier* 명령을 사용합니다. 여기서 *note*는 최대 128자 문자열이고 *identifier*는 다음 중 하나입니다.

- *mediatype.volume-serial-number*. 여기서 *mediatype*은 부록 A. 장비 유형 용어집에 나열된 2자 매체 유형 코드 중 하나이고 *volume-serial-number*는 라이브러리 내에서 볼륨을 고유하게 식별하는 영숫자 문자열입니다. 또는 다음 명령을 사용합니다.
- *library-equipment-number:slot*. 여기서 *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 자동화된 테이프 라이브러리에 지정하는 장비 순서 번호이고 *slot*은 라이브러리 내에서 카트리지가 상주하는 슬롯 주소입니다.

예제에서는 LTO(*li*) 카트리지 *VOL06E*를 보관소에서 회수하여 성공적으로 검증한 후 보관소로 반납했다고 적었습니다.

```
root@solaris:~# chmed -I "validated and revaulted 20150310" li.VOL06A
```

라이브러리에 드라이브가 설치되는 순서 결정

자동화된 라이브러리에 여러 개의 드라이브가 있는 경우 *mcf* 파일의 드라이브 순서가 라이브러리 컨트롤러에서 드라이브가 표시되는 순서와 같아야 합니다. 이 순서는 장치가 호스트에 표시되고 호스트의 */var/adm/messages* 파일에 보고되는 순서와 다를 수 있습니다. 따라서 Oracle Hierarchical Storage Manager 메타데이터 서버 및 datamover 호스트를 구성하거나 라이브러리를 변경하거나 라이브러리 구성을 변경할 때마다 아래 나열된 작업을 수행하여 드라이브 순서를 확인해야 합니다.

- 라이브러리 및 Solaris 호스트에 대한 드라이브 정보 수집
- 직접 연결 라이브러리의 드라이브를 Solaris 장치 이름에 매핑 또는 ACSLS 연결 라이브러리의 드라이브를 Solaris 장치 이름에 매핑(사용 중인 장비에 따라 선택)

라이브러리 및 Solaris 호스트에 대한 드라이브 정보 수집

1. 라이브러리 설명서를 참조하십시오. 드라이브 및 대상이 어떻게 식별되는지 확인합니다. 로컬 운영자 패널이 있는 경우 이를 사용하여 어떻게 드라이브 순서를 결정할 수 있는지 확인합니다.
2. 라이브러리에 마운트된 로컬 운영자 패널이 있는 경우 이를 사용하여 드라이브가 컨트롤러에 연결되는 순서를 결정합니다. 각 드라이브의 SCSI 대상 식별자 또는 World Wide Name을 확인합니다.

3. Solaris 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

4. */dev/rmt/*에 Solaris 논리 장치 이름을 나열하고 출력을 텍스트 파일로 재지정합니다.

   예제에서는 */dev/rmt/*의 목록을 *root* 사용자의 홈 디렉토리에 있는 *device-mappings.txt* 파일로 재지정합니다.

```
root@solaris:~# ls -l /dev/rmt/ > /root/device-mappings.txt
```

5. 이제, 직접 연결 테이프 라이브러리 또는 ACSLS 연결 라이브러리 장비에 따른 절차를 사용하여 Solaris 장치 이름에 드라이브를 매핑합니다.

직접 연결 라이브러리의 드라이브를 Solaris 장치 이름에 매핑

*/dev/rmt/*에 나열된 각 Solaris 논리 드라이브 이름과 라이브러리에서 Oracle HSM 서버 호스트에 지정하는 각 드라이브에 대해 다음 절차를 수행합니다.

1. Oracle HSM Solaris 호스트에 아직 로그인하지 않은 경우 *root*로 로그인합니다.

```
root@solaris:~#
```

2. 드라이브가 사용되지 않도록 실행 중인 아카이빙 프로세스를 모두 중지합니다. "아카이빙 및 스테이징 프로세스 유휴 설정" 및 “아카이빙 및 스테이징 프로세스 중지”를 참조하십시오.
3. 텍스트 편집기에서 “라이브러리 및 Solaris 호스트에 대한 드라이브 정보 수집” 절차로 만든 장치 매핑 파일을 엽니다. 파일을 간단한 테이블로 구성하고 변경사항을 저장합니다.

   후속 단계에서 이 정보를 참조해야 합니다. 예제에서는 라이브러리 장치 정보를 위한 머리글 및 공간을 추가하는 동안, *vi* 편집기를 사용하여 */dev/rmt/* 목록에서 권한, 소유권 및 날짜 속성을 삭제했습니다.

```
root@solaris:~# vi /root/device-mappings.txt
LIBRARY SOLARIS          SOLARIS
DEVICE  LOGICAL          PHYSICAL
NUMBER  DEVICE           DEVICE
------- -------------    ----------------------------------------------------
        /dev/rmt/0cbn -> ../../devices/pci@8.../st@w500104f00093c438,0:cbn
        /dev/rmt/1cbn -> ../../devices/pci@8.../st@w500104f0008120fe,0:cbn
        /dev/rmt/2cbn -> ../../devices/pci@8.../st@w500104f000c086e1,0:cbn
        /dev/rmt/3cbn -> ../../devices/pci@8.../st@w500104f000b6d98d,0:cbn
:w
```

4. 라이브러리에서 모든 드라이브가 비어 있는지 확인합니다.
5. 아직 Solaris 논리 장치 이름에 매핑되지 않은 라이브러리의 첫번째 드라이브로 테이프를 로드합니다.

   아래 예제의 목적상 HP Ultrium LTO4 테이프 드라이브로 LTO4 테이프를 로드합니다.
6. 테이프 라이브러리에서 드라이브를 매핑하는 경우 테이프를 마운트하는 드라이브에 해당하는 Solaris */dev/rmt/* 항목을 식별합니다. 드라이브를 식별할 때까지 *mt -f /dev/rmt/number status* 명령을 실행합니다. 여기서 *number*는 */dev/rmt/*에서 드라이브를 식별합니다.

   예제에서는 */dev/rmt/0*에 드라이브가 비어 있지만, */dev/rmt/1*에 드라이브가 테이프를 보유합니다. 따라서 라이브러리에서 드라이브 1로 식별한 드라이브는 Solaris */dev/rmt/1*에 해당합니다.

```
root@solaris:~# mt -f /dev/rmt/0 status
/dev/rmt/0: no tape loaded or drive offline
root@solaris:~# mt -f /dev/rmt/1 status
HP Ultrium LTO 4 tape drive:
   sense key(0x0)= No Additional Sense   residual= 0   retries= 0
   file no= 0   block no= 3
```

7. 이전 절차에서 만든 드라이브 매핑 파일에서 테이프를 보유한 Solaris 장치 항목을 찾아서 제공된 공간에 라이브러리의 장치 식별자를 입력합니다. 그런 다음 파일을 저장합니다.

   예제에서는 */dev/rmt/1* 행의 *LIBRARY DEVICE NUMBER* 필드에 *1*을 입력합니다.

```
root@solaris:~# vi /root/device-mappings.txt
LIBRARY SOLARIS          SOLARIS
DEVICE  LOGICAL          PHYSICAL
NUMBER  DEVICE           DEVICE
------- -------------    ------------------------------------------------------
        /dev/rmt/0cbn -> ../../devices/pci@8.../st@w500104f00093c438,0:cbn
   1    /dev/rmt/1cbn -> ../../devices/pci@8.../st@w500104f0008120fe,0:cbn
        /dev/rmt/2cbn -> ../../devices/pci@8.../st@w500104f000c086e1,0:cbn
        /dev/rmt/3cbn -> ../../devices/pci@8.../st@w500104f000b6d98d,0:cbn
:w
```

8. 테이프를 언로드합니다.
9. 장치 매핑 파일에서 모든 장치를 Solaris 논리 장치 이름에 매핑하는 항목을 보유할 때까지 이 절차를 반복합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

```
root@solaris:~# vi /root/device-mappings.txt
LIBRARY SOLARIS          SOLARIS
DEVICE  LOGICAL          PHYSICAL
```

```
NUMBER  DEVICE           DEVICE
------- -------------    ------------------------------------------------
   2    /dev/rmt/0cbn -> ../../devices/pci@8.../st@w500104f00093c438,0:cbn
   1    /dev/rmt/1cbn -> ../../devices/pci@8.../st@w500104f0008120fe,0:cbn
   3    /dev/rmt/2cbn -> ../../devices/pci@8.../st@w500104f000c086e1,0:cbn
   4    /dev/rmt/3cbn -> ../../devices/pci@8.../st@w500104f000b6d98d,0:cbn
:wq
root@solaris:~#
```

10. 여기서 중지합니다. 나중에 사용할 수 있도록 매핑 파일을 보관합니다.

ACSLS 연결 라이브러리의 드라이브를 Solaris 장치 이름에 매핑

1. Oracle HSM Solaris 호스트에 아직 로그인하지 않은 경우 *root*로 로그인합니다.

   ```
   root@solaris:~#
   ```

2. 드라이브가 사용되지 않도록 실행 중인 아카이빙 프로세스를 모두 중지합니다. "아카이빙 및 스테이징 프로세스 유휴 설정" 및 "아카이빙 및 스테이징 프로세스 중지"를 참조하십시오.
3. 텍스트 편집기에서 “라이브러리 및 Solaris 호스트에 대한 드라이브 정보 수집” 절차로 만든 장치 매핑 파일을 엽니다. 파일을 간단한 테이블로 구성합니다.

   후속 단계에서 이 정보를 참조해야 합니다. 예제에서는 라이브러리 장치 정보를 위한 머리글 및 공간을 추가하는 동안, *vi* 편집기를 사용하여 */dev/rmt/* 목록에서 권한, 소유권 및 날짜 속성을 삭제합니다.

   ```
   root@solaris:~# vi /root/device-mappings.txt
   SOLARIS LOGICAL DEVICE  DEVICE SERIAL NUMBER  ACSLS DEVICE ADDRESS
   ----------------------  --------------------  -------------------------------
   /dev/rmt/0
   /dev/rmt/1
   /dev/rmt/2
   /dev/rmt/3
   ```

4. */dev/rmt/*에 나열된 각 논리 장치 이름에 대해 *luxadm display /dev/rmt/number* 명령을 사용하여 일련 번호를 표시합니다. 여기서 *number*는 */dev/rmt/*에서 드라이브를 식별합니다.

   예제에서는 */dev/rmt/0* 장치의 일련 번호인 *HU92K00200*을 가져옵니다.

   ```
   root@solaris:~# luxadm display /dev/rmt/0
   DEVICE PROPERTIES for tape: /dev/rmt/0
   Vendor: HP
   Product ID: Ultrium 4-SCSI
   ```

```
Revision: G25W
Serial Num: HU92K00200
...
Path status: Ready
root@solaris:~#
```

5. 그런 후 텍스트 편집기를 사용해서 *device-mappings.txt* 파일의 해당 행에 각 장치의 일련 번호를 입력합니다.

   예제에서는 *vi* 편집기를 사용하여 *device-mappings.txt* 파일에서 */dev/rmt/0* 장치의 일련 번호를 기록합니다.

```
root@solaris:~# vi /root/device-mappings.txt
SOLARIS LOGICAL DEVICE  DEVICE SERIAL NUMBER  ACSLS DEVICE ADDRESS
----------------------  -------------------  -------------------------------
/dev/rmt/0              HU92K00200
/dev/rmt/1
/dev/rmt/2
/dev/rmt/3
```

6. */dev/rmt/*에 매핑된 각 장치 일련 번호에 대해 해당하는 ACSLS 드라이브 주소를 얻습니다. ACSLS 명령 *display drive * -f serial_num*을 사용합니다.

   예제에서는 *HU92K00200*(*/dev/rmt/0*), *HU92K00208*(*/dev/rmt/1*), *HU92K00339*(*/dev/rmt/2*), *HU92K00289*(*/dev/rmt/3*) 장치의 ACSLS 주소를 가져옵니다.

```
 ACSSA> display drive * -f serial_num
 2014-03-29 10:49:12 Display Drive
 Acs Lsm Panel Drive Serial_num
 0   2   10    16    331002031352
 0   2   10    17    HU92K00200
 0   2   10    18    HU92K00208
 0   3   10    10    HU92K00339
 0   3   10    11    HU92K00189
 0   3   10    12    HU92K00289
root@solaris:~#
```

7. 텍스트 편집기를 사용해서 *device-mappings.txt* 파일의 해당 행에 각 일련 번호에 대한 ACSLS 주소를 입력합니다. 파일을 저장하고 편집기를 닫습니다.

   예제에서는 *vi* 편집기를 사용하여 *device-mappings.txt* 파일에서 정보를 기록합니다.

```
root@solaris:~# vi /root/device-mappings.txt
SOLARIS LOGICAL DEVICE DEVICE SERIAL NUMBER ACSLS DEVICE ADDRESS
```

```
---------------------- -------------------- ---------------------------------
/dev/rmt/0             HU92K00200           (acs=0, lsm=2, panel=10, drive=17)
/dev/rmt/1             HU92K00208           (acs=0, lsm=2, panel=10, drive=18)
/dev/rmt/2             HU92K00339           (acs=0, lsm=2, panel=10, drive=10)
/dev/rmt/3             HU92K00289           (acs=0, lsm=2, panel=10, drive=12)
:wq
root@solaris:~#
```

8. 여기서 중지합니다. 나중에 사용할 수 있도록 매핑 파일을 보관합니다.

드라이브 관리

Oracle HSM 인터페이스에서 다음과 같은 다양한 드라이브 관리 작업을 처리할 수 있습니다.

- 드라이브 로드 및 언로드
- 테이프 드라이브 청소
- 암호화 기능을 가진 드라이브 사용
- 드라이브 문제 처리.

드라이브 로드 및 언로드

이동식 매체가 자동화된 라이브러리에 저장된 경우 파일 시스템 아카이빙 및 스테이징 프로세스가 필요에 따라 자동으로 카트리지를 드라이브에 로드합니다. 그러나 이동식 매체 파일을 관리하거나 Oracle HSM 구성을 백업하거나 파일 시스템을 복구할 때 요구 시 카트리지를 로드할 수도 있습니다. 이 절에서는 다음 항목을 다룹니다.

- 자동화된 라이브러리에 설치된 드라이브 로드 및 언로드
- 수동으로 독립형 드라이브 로드 및 언로드
- 볼륨을 수동으로 로드해야 할 때 운영자에게 알림.

자동화된 라이브러리에 설치된 드라이브 로드 및 언로드

- 지정된 라이브러리 위치에서 드라이브 로드
- 지정된 매체 유형 및 볼륨 일련 번호를 가진 드라이브 로드
- 라이브러리에서 지정된 드라이브 언로드.

지정된 라이브러리 위치에서 드라이브 로드

samcmd load library-equipment-number:slot[:disk-side] 명령을 사용합니다. 여기서 *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 자동화된 테이프 라이브러리에 지정하는 장비 순서 번호이고 *slot*은 라이브러리 내에서 대상 볼륨이 상주하는 슬롯 주소이고, 선택적 *disk-side* 값인 *1* 또는 *2*는 양면 광자기 디스크의 한쪽 면을 지정합니다.

라이브러리에서 그 다음 사용 가능한 드라이브에 카트리지가 로드됩니다. 예제에서는 *800* 라이브러리의 *11* 슬롯에 위치한 자기 테이프 카트리지를 로드합니다.

```
root@solaris:~# samcmd load 800:11
```

지정된 매체 유형 및 볼륨 일련 번호를 가진 드라이브 로드

samcmd load mediatype.volume-serial-number 명령을 사용합니다. 여기서 *mediatype*은 부록 A. 장비 유형 용어집에 나열된 2자 매체 유형 코드 중 하나이고 *volume-serial-number*는 라이브러리 내에서 볼륨을 고유하게 식별하는 영숫자 문자열입니다.

라이브러리에서 그 다음 사용 가능한 드라이브에 카트리지가 로드됩니다. 예제에서는 LTO(*li*) 테이프 카트리지 *VOL109*를 로드합니다.

```
root@solaris:~# samcmd load li.VOL109
```

라이브러리에서 지정된 드라이브 언로드

samcmd unload drive-equipment-number 명령을 사용합니다. 여기서 *drive-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 드라이브에 지정하는 장비 순서 번호입니다.

드라이브가 *unavail* 상태인 경우에도 카트리지가 언로드됩니다. 예제에서는 *801* 드라이브를 언로드합니다.

```
root@solaris:~# samcmd unload 801]
```

수동으로 독립형 드라이브 로드 및 언로드

Oracle HSM 소프트웨어는 독립형 이동식 매체 드라이브를 고유의 카탈로그를 가진 작은 단일 슬롯 라이브러리처럼 취급합니다.

독립형 드라이브로 카트리지 로드

독립형 드라이브를 로드하려면 제조업체의 지침에 따라 드라이브에 카트리지를 놓습니다. Oracle HSM 시스템은 카트리지가 로드된 것을 인식하고, 레이블을 읽고, 드라이브의 카탈로그를 업데이트합니다.

독립형 드라이브에서 카트리지 언로드

독립형 드라이브를 언로드하려면 다음과 같이 하십시오.

1. 드라이브를 유휴 설정합니다. *samcmd idle drive-equipment-number* 명령을 사용합니다. 여기서 *drive-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 드라이브에 지정하는 장비 순서 번호입니다.

드라이브가 유휴 상태일 때 Oracle HSM 소프트웨어는 드라이브를 사용 중인 현재 아카이빙 프로세스를 마치지만, 새로운 작업을 시작하지는 않습니다.

```
root@solaris:~# samcmd idle 801
```

2. Oracle HSM이 완료되고 드라이브를 *off*로 설정할 때까지 기다립니다.

   samcmd r 명령을 사용하여 드라이브의 상태를 확인할 수 있습니다.
3. 공급업체의 지침에 따라 카트리지를 꺼냅니다.
4. 여기서 중지합니다.

볼륨을 수동으로 로드해야 할 때 운영자에게 알림

독립형 드라이브를 사용하거나 필요한 카트리지를 보관소나 라이브러리 밖의 기타 다른 위치에 저장할 경우 Oracle HSM 소프트웨어는 운영자가 비상주 카트리지를 로드해야 할 때 지정된 주소로 전자 메일을 보낼 수 있습니다. 이 기능을 사용으로 설정하려면 아래 절차를 따르십시오.

로드 알림 사용으로 설정

1. 파일 시스템 호스트에 *root*로 로그인합니다.

   ```
   root@solaris:~#
   ```

2. */opt/SUNWsamfs/examples/* 디렉토리의 *load_notify.sh* 파일을 */etc/opt/SUNWsamfs/scripts/* 디렉토리로 복사합니다.

   아래의 명령은 단일 라인으로 입력됩니다. 줄바꿈은 백슬래시 문자로 이스케이프됩니다.

   ```
   root@solaris:~# cp /opt/SUNWsamfs/examples/load_notify.sh /
   /etc/opt/SUNWsamfs/scripts/
   root@solaris:~#
   ```

3. 텍스트 편집기에서 */etc/opt/SUNWsamfs/defaults.conf* 파일을 엽니다. *exported_media* 지시어를 검색합니다. 행 주석 처리를 해제하거나 필요한 경우 추가하고, 값을 *exported_media=available*로 설정합니다.

   이 예에서는 *vi* 편집기를 사용합니다.

   ```
   root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
   # These are the defaults.
   ...
   exported_media=available
   ```

4. */etc/opt/SUNWsamfs/defaults.conf* 파일에서 *attended* 지시어를 검색합니다. 행 주석 처리를 해제하거나 필요한 경우 행을 추가합니다. 값을 *attended=yes*로 설정합니다. 파일을 저장하고 편집기를 닫습니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults. ...
# These are the defaults. ...
exported_media=available
attended=yes
:wq
root@solaris:~#
```

5. 텍스트 편집기에서 */etc/opt/SUNWsamfs/scripts/load_notify.sh* 파일을 엽니다. 알림 전자 메일의 기본 수신자인 *root*를 찾습니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/scripts/load_notify.sh
#       Notify operator to load volume.
...
# Change the email address on the following line to send email to
# the appropriate recipient.
/bin/ppriv -s I=basic -e /usr/bin/mailx -s "SAM-FS needs VSN $5" root <<EOF
...
```

6. */etc/opt/SUNWsamfs/scripts/load_notify.sh* 파일에서 알림 전자 메일의 수신자를 기본값인 *root*에서 비상주 볼륨의 담당 운영자의 전자 메일 주소로 변경합니다. 파일을 저장하고 편집기를 닫습니다.

   예제에서는 수신자를 *tapetech*로 변경합니다.

```
#       Notify operator to load volume.
...
/bin/ppriv -s I=basic -e /usr/bin/mailx -s "SAM-FS needs VSN $5" tapetech <<EOF
...
:wq
root@solaris:~#
```

7. Oracle HSM 소프트웨어를 다시 초기화합니다. *sam-fsd* 명령을 사용합니다.

   *sam-fsd*는 Oracle HSM 구성 파일을 읽는 초기화 명령입니다. 오류가 발견되면 실행을 중지합니다.

```
root@solaris:~# sam-fsd
```

8. Oracle HSM 소프트웨어에 *mcf* 파일을 다시 읽고 그에 따라 파일 시스템 및 하드웨어를 재구성하도록 지시합니다. *samd config* 명령을 사용합니다.

```
root@solaris:~# samd config
```

9. 여기서 중지합니다.

테이프 드라이브 청소

최신 Oracle StorageTek T10000D 및 LTO(Linear Tape Open) 테이프 드라이브는 자체 모니터하다가 필요할 때 청소를 요청합니다. Oracle Hierarchical Storage Manager 소프트웨어는 이 요청을 수용하고 필요할 때 자동으로 청소 카트리지를 로드합니다. 따라서 대부분의 경우 사용자는 라이브러리에 충분한 청소 카트리지가 있고 Oracle HSM이 이를 찾을 수 있는지 확인하기만 하면 됩니다.

드라이브 요청 청소를 사용할 수 없는 경우 청소를 수동으로 시작할 수 있습니다. 그러나 대부분의 제조업체는 드라이브의 요청이 없을 때 일상적 청소를 강력히 억제합니다. 청소 카트리지는 연마재입니다. 남용 시 드라이브와 매체가 손상될 수 있습니다. 따라서 주의를 기울이고 제조업체의 권장 사항을 따르십시오.

이 절의 나머지 부분에서는 다음 작업을 위한 지침을 제공합니다.

- 충분한 청소 카트리지 공급
- 자동 테이프 드라이브 청소 사용(권장)
- 수동으로 테이프 드라이브 청소.

충분한 청소 카트리지 공급

1. 파일 시스템 호스트에 *root*로 로그인합니다.

   ```
   root@solaris:~#
   ```

2. 자동 청소(권장)를 구성하려는 경우 라이브러리에 드라이브가 2개 이상 있으면 라이브러리의 테이프를 나열하는 각 파일 시스템 카탈로그마다 적어도 2개의 청소 카트리지를 공급해야 합니다.

   드라이브에 청소가 필요할 때 청소 카트리지를 사용할 수 없으면 Oracle HSM 소프트웨어는 청소를 완료할 때까지 드라이브 상태를 *down*으로 설정합니다.
3. 청소 카트리지를 라이브러리 메일슬롯(카트리지 액세스 포트라고도 함)에 놓습니다.
4. 청소 카트리지를 자동화된 라이브러리로 가져옵니다. *samimport library-equipment-number* 명령을 사용합니다. 여기서 *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 라이브러리에 지정하는 장비 순서 번호입니다.

   예제에서는 청소 카트리지를 라이브러리 *800*의 메일슬롯에 놓고 라이브러리로 가져옵니다.

   ```
   root@solaris:~# samimport 800
   ```

5. 청소 카트리지 레이블이 *CLEAN*으로 읽히거나 *CLN* 문자로 시작하면 여기서 중지합니다.

   Oracle HSM 소프트웨어는 청소 카트리지를 인식하고 메일박스에서 스토리지 슬롯으로 이동합니다. Oracle HSM는 라이브러리 카탈로그를 업데이트하고, 청소 매체 플래그를 설정하고, 매체 유형에 권장된 최대 청소 횟수로 액세스 카운트를 설정합니다. 카트리지가 드라이브 청소에 사용될 때마다 이 카운트가 감소합니다.
6. 카트리지에 레이블이 없으면 청소 매체로 플래그를 지정합니다. *chmed +C library-equipment-number:slot* 명령을 사용합니다. 여기서 *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 라이브러리에 지정하는 장비 순서 번호이고 *slot*은 라이브러리 내에서 청소 카트리지의 위치입니다.

   예제에서는 *800* 라이브러리의 *31* 슬롯에서 카트리지에 *C*(청소 매체) 플래그를 설정합니다.

   ```
   root@solaris:~# chmed +C 800:31
   ```

7. 액세스 수를 매체 유형에 권장되는 최대 청소 수로 설정합니다. *chmed -count cleanings library-equipment-number:slot* 명령을 사용합니다. 설명:
   - *cleanings*는 제조업체가 카트리지당 권장하는 최대 청소 수입니다.
   - *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일이 라이브러리에 지정하는 장비 순서 번호입니다.
   - *slot*은 라이브러리 내의 청소 카트리지 위치입니다.

   카트리지가 드라이브 청소에 사용될 때마다 청소 카운트가 감소합니다. 예제에서는 *800* 라이브러리에 사용된 LTO(유형 *li*) 청소 카트리지에 권장되는 최대값으로 카운트를 최대 *50*회 청소로 설정합니다.

   ```
   root@solaris:~# chmed -count 50 800:31
   ```

8. 그런 다음 자동 테이프 드라이브 청소 사용(권장)을 수행하거나 여기서 중지합니다.

자동 테이프 드라이브 청소 사용(권장)

1. 파일 시스템 호스트에 *root*로 로그인합니다.

   ```
   root@solaris:~#
   ```

2. 라이브러리에 사용하려는 자동 청소 기능이 포함되어 있으면 라이브러리 제조업체의 권장 사항에 따라 기능을 구성합니다. 여기서 중지합니다.

   이제, 드라이브가 청소를 요청하면 라이브러리에서 자동으로 필요한 청소 매체를 공급합니다.
3. 라이브러리에 사용하지 않으려는 자동 청소 기능이 포함되어 있으면 제조업체의 권장 사항에 따라 기능을 사용 안함으로 설정합니다.

4. 텍스트 에디터에서 */etc/opt/SUNWsamfs/defaults.conf* 파일을 열고 Oracle HSM 자동 청소를 사용으로 설정합니다. *tapeclean = all autoclean on logsense on* 행을 추가합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   이 예에서는 *vi* 편집기를 사용합니다.

   ```
   root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
   # These are the defaults. ...
   #sef = all on once
   ...
   tapeclean = all autoclean on logsense on
   :wq
   root@solaris:~#
   ```

5. Oracle HSM 소프트웨어를 다시 초기화합니다. *sam-fsd* 명령을 사용합니다.

   *sam-fsd*는 Oracle HSM 구성 파일을 읽는 초기화 명령입니다. 오류가 발견되면 실행을 중지합니다.

   ```
   root@solaris:~# sam-fsd
   ```

6. Oracle HSM 소프트웨어에 *mcf* 파일을 다시 읽고 그에 따라 파일 시스템 및 하드웨어를 재구성하도록 지시합니다. *samd config* 명령을 사용합니다.

   ```
   root@solaris:~# samd config
   ```

7. 여기서 중지합니다.

수동으로 테이프 드라이브 청소

1. 계속하기 전에 드라이브 제조업체의 수동 청소 지침을 확인합니다.

   주의를 기울이십시오. 너무 잦은 청소는 드라이브 손상의 흔한 원인입니다. 현재 대부분의 제조업체는 주기적 또는 예약 청소를 권장하지 않습니다. 따라서 언제 드라이브를 청소해야 하는지 정확히 이해해야 합니다.
2. 드라이브에 청소가 필요하다는 표식이 있는지 장치 로그를 모니터합니다. 각 *drive-equipment-number*마다 */var/opt/SUNWsamfs/devlog/* 디렉토리에 하나의 로그가 있습니다. 여기서 *drive-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 드라이브에 지정하는 장비 순서 번호입니다.
3. 시스템 로그 파일 */var/adm/messages*에서 장치 오류를 모니터링합니다.
4. 테이프 드라이브를 청소합니다. *cleandrive drive-equipment-number* 명령을 사용합니다.

   예제에서는 *802* 드라이브를 청소합니다.

   ```
   root@solaris:~# cleandrive 802
   ```

5. 여기서 중지합니다.

암호화 기능을 가진 드라이브 사용

암호화 기능이 있는 드라이브에 파일을 아카이브할 때는 아카이빙 작업 계획 시 다음 사항을 고려하십시오.

- 한 라이브러리에 암호화 가능 드라이브와 암호화 불가능 드라이브를 섞지 마십시오.
- 드라이브에 암호화가 사용으로 설정된 후에는 사용 안함으로 설정할 수 없습니다.
- 한 테이프 볼륨에 암호화된 파일과 암호화되지 않은 파일을 섞지 마십시오.
- 암호화 가능 드라이브는 암호화되지 않은 데이터가 있는 테이프 볼륨에 암호화된 파일을 연결할 수 없습니다.
- 암호화 가능 드라이브는 암호화되지 않은 데이터를 읽을 수 있습니다.

자세한 내용은 드라이브 및 암호화 키 관리 시스템의 설명서를 참조하십시오.

드라이브 문제 처리

일반적으로 공급업체의 권장 사항에 따라 드라이브 문제를 처리합니다. 그러나 드라이브 유지 관리, 문제 해결 또는 수리를 시작하려면 먼저 다음 작업 중 하나 또는 모두를 수행해야 합니다.

- 유지 관리 또는 보수를 위해 드라이브를 오프라인으로 전환
- 드라이브 문제 이후 라이브러리로 매체 반납

유지 관리 또는 보수를 위해 드라이브를 오프라인으로 전환

1. 파일 시스템 호스트에 *root*로 로그인합니다.

   ```
   root@solaris:~#
   ```

2. "아카이빙 및 스테이징 프로세스 유휴 설정"에 설명된 대로 아카이빙 및 스테이징 프로세스를 중지합니다.
3. 아카이빙 및 스테이징 프로세스를 중지하고 드라이브를 오프라인으로 전환합니다. "아카이빙 및 스테이징 프로세스 중지" 절차를 사용하십시오.
4. 공급업체가 지정한 유지 관리, 진단, 수리 절차를 수행합니다.

   예를 들어, 끼인 카트리지를 빼려고 시도하기 전에 공급업체의 권장 사항을 확인해야 합니다. 끼인 카트리지를 잘못 빼면 카트리지와 드라이브가 손상될 수 있습니다.
5. 드라이브가 다시 작동하면 라이브러리 및 드라이브를 온라인으로 전환하고 아카이빙 및 스테이징 프로세스를 다시 시작합니다. "아카이빙 및 스테이징 프로세스 다시 시작" 절차를 사용하십시오.

6. 여기서 중지합니다.

드라이브 문제 이후 라이브러리로 매체 반납

드라이브에 마운트된 매체에 문제가 발생한 경우 수리 과정의 일부로 매체를 수동으로 꺼내야 할 수 있습니다. 그러면 카탈로그가 불일치 상태로 남을 수 있습니다. 따라서 아래의 적절한 절차를 따릅니다.

자동 감사를 수행하지 않은 라이브러리로 매체 반환

복구 후 라이브러리 및 드라이브를 다시 온라인으로 전환할 때 자동 감사가 수행되지 않을 경우 매체를 라이브러리로 반환하려면 다음과 같이 하십시오.

1. 손으로 카트리지를 스토리지 슬롯으로 반환합니다.

   이 경우에는 카탈로그가 업데이트되지 않고 라이브러리 컨텐츠 중에 카트리지가 계속 나열됩니다. 따라서 카트리지를 이전에 점유했던 해당 슬롯에 다시 넣어서 불일치를 해결합니다.
2. 슬롯이 다시 점유된 것으로 표시되도록 Oracle HSM 카탈로그를 업데이트합니다. *chmed library-equipment-number:slot* 명령을 사용합니다. 여기서 *slot*은 라이브러리 내 슬롯의 주소입니다.

   예제에서는 *800* 라이브러리에서 *42* 슬롯의 상태를 업데이트합니다.

   ```
   root@solaris:~# chmed +o 800:42
   root@solaris:~#
   ```

3. 여기서 중지합니다.

자동 감사 후 라이브러리에 매체 반환

복구 후 라이브러리 및 드라이브를 다시 온라인으로 전환할 때 라이브러리에서 자동 감사를 수행하는 경우 다음과 같이 하십시오.

1. 카트리지를 라이브러리 메일슬롯에 놓습니다.
2. 카트리지를 라이브러리로 가져옵니다. *samimport library-equipment-number* 명령을 사용합니다.

   이 경우 감사로 카탈로그를 조정했으므로 더 이상 라이브러리에 카트리지를 나열하지 않습니다. 따라서 카트리지를 가져오면 라이브러리와 Oracle HSM 카탈로그 양쪽에 추가됩니다. 예제에서는 *800* 라이브러리의 메일슬롯에 카트리지를 놓고 라이브러리로 가져왔습니다.

   ```
   root@solaris:~# samimport 800
   ```

3. 여기서 중지합니다.

이동식 매체 관리

이 절에서는 다음 항목을 다룹니다.

- 이동식 매체에 레이블 지정
- 데이터 무결성 유지 관리

이동식 매체에 레이블 지정

주의:

카트리지에 레이블을 지정하거나 재지정하면 카트리지의 데이터에 영구적으로 액세스할 수 없게 됩니다. 카트리지에 저장된 데이터가 필요 없다고 확신하는 경우에만 카트리지 레이블을 재지정하십시오.

레이블 지정 프로세스에서는 기록 매체에 식별 정보를 기록하고 사용할 수 있도록 초기화합니다. 자세한 내용은 ANSI X3.27-1987, *File Structure and Labeling of Magnetic Tapes for Information Interchange*를 참조하십시오.

매체에 레이블을 지정할 때는 아래에서 적합한 절차를 선택합니다.

- 바코드에서 레이블 생성
- 새 테이프에 레이블 지정 또는 기존 테이프에 레이블 재지정
- 새 광 디스크 레이블 지정 또는 기존 광 디스크 레이블 다시 지정.

바코드에서 레이블 생성

쓰기가 사용으로 설정되었고, 카트리지에 바코드에서 파생된 VSN(볼륨 일련 번호)으로 레이블이 지정되지 않은 카트리지를 자동으로 레이블 지정하려면 다음을 수행합니다.

1. 모든 바코드를 읽을 수 있는지 확인합니다.
2. 파일 시스템 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

3. 텍스트 편집기에서 */etc/opt/SUNWsamfs/defaults.conf* 파일을 엽니다.

   예제에서는 *vi* 편집기를 사용하여 파일을 봅니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults.
...
```

4. 해당 바코드의 처음 6자로부터 VSN(볼륨 일련 번호)을 생성하려면 먼저 Oracle HSM이 기본값인 *barcodes*로 설정되었는지 확인합니다. *defaults.conf* 파일에서 *labels* 지시어에 대한 행(있는 경우)을 찾습니다. *labels* 지시어가 *barcodes*로 설정되거나, 주석 처리되거나, 파일에 없는 경우 Oracle HSM은 기본값인 *barcodes*로 설정됩니다.

예제에서는 *defaults.conf* 파일에 *#labels* = *barcodes* 행이 포함되어 있습니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults.
...
#labels = barcodes
root@solaris:~#
```

5. 해당 바코드의 처음 6자로부터 VSN(볼륨 일련 번호)을 생성하려는 경우 Oracle HSM이 기본값으로 설정된 경우 변경 작업을 수행하지 않고 *defaults.conf* 파일을 닫습니다. 여기서 중지합니다.

   labels 지시어가 *barcodes*로 설정된 경우 소프트웨어는 해당 바코드의 첫 6자에서 필수 VSN(볼륨 일련 번호)을 자동으로 생성합니다. 예제에서는 Oracle HSM이 기본 설정을 사용합니다. 따라서 파일을 저장하지 않고 *vi* 편집기를 닫습니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults.
...
#labels = barcodes
:q
root@solaris:~#
```

6. 그렇지 않으면 해당 바코드의 처음 6자에서 VSN(볼륨 일련 번호)을 생성해야 하는 경우 *labels* = *barcodes*를 입력하거나, 기본값이 아닌 지시어를 주석 처리하거나, 기본값이 아닌 지시어를 삭제합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   예제에서는 지시어가 기본값이 아닌 *barcodes_low*로 설정되었습니다. 따라서 기본값이 아닌 행을 주석 처리합니다. *labels* = *barcodes* 행을 삽입합니다. 수정된 파일을 저장하고 편집기를 닫습니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults.
...
#labels = barcodes_low
labels = barcodes
:q
root@solaris:~#
```

7. 카트리지 바코드의 마지막 6자로부터 VSN(볼륨 일련 번호)을 생성하려면 *labels* 매개변수의 값을 *barcodes_low*로 설정합니다. 파일을 저장하고 편집기를 닫습니다.

   예제에서는 *labels = barcodes_low* 행을 삽입하고, 파일을 저장하고, 편집기를 닫습니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults.
...
labels = barcodes_low
:wq
root@solaris:~#
```

8. *defaults.conf* 파일을 편집했으면 *sam-fsd* 명령을 실행합니다.

   *sam-fsd*는 Oracle HSM 구성 파일을 읽는 초기화 명령입니다. 오류가 발견되면 실행을 중지합니다.

```
root@solaris:~# sam-fsd
```

9. *defaults.conf* 파일을 편집했으면 Oracle HSM 소프트웨어에 *mcf* 파일을 다시 읽고 그에 따라 재구성하도록 지시합니다. *samd config* 명령을 사용합니다.

```
[metadata-server]root@solaris:~# samd config
```

10. 여기서 중지합니다.

새 테이프에 레이블 지정 또는 기존 테이프에 레이블 재지정

주의:

카트리지에 레이블을 지정하거나 재지정하면 카트리지의 데이터에 영구적으로 액세스할 수 없게 됩니다. 카트리지에 저장된 데이터가 필요 없다고 확신하는 경우에만 카트리지 레이블을 재지정하십시오.

1. 파일 시스템 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. 드라이브에 이미 로드된 새 테이프에 레이블을 지정하려면 *tplabel -new volume-serial-number drive-equipment-number* 명령을 사용합니다. 설명:
   - *volume-serial-number*는 필수 볼륨 일련 번호입니다.
   - *drive-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일이 드라이브에 지정하는 장비 순서 번호입니다.

   예제에서는 *803* 드라이브의 새 테이프 카트리지에 볼륨 일련 번호 *VOL600*을 지정합니다.

```
root@solaris:~# tplabel -new -vsn VOL600 803
root@solaris:~#
```

3. 자동화된 매체 라이브러리에 있는 새 테이프에 레이블을 지정하려면 *tplabel -new volume-serial-number library-equipment-number:slot* 명령을 사용합니다. 설명:
   - *volume-serial-number*는 필수 볼륨 일련 번호입니다.
   - *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일이 드라이브에 지정하는 장비 순서 번호입니다.
   - *slot*은 라이브러리 내 카트리지 위치입니다.

   예제에서는 *800* 라이브러리의 *19* 슬롯에서 새 테이프 카트리지에 볼륨 일련 번호 *VOL601*을 지정합니다.

   ```
   root@solaris:~# tplabel -new -vsn VOL601 800:19
   root@solaris:~#
   ```

4. 드라이브에 로드된 테이프의 레이블을 재지정하려면 *tplabel -old old-volume-serial-number -new new-volume-serial-number drive-equipment-number* 명령을 사용합니다. 설명:
   - *volume-serial-number*는 필수 볼륨 일련 번호입니다.
   - *drive-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일이 드라이브에 지정하는 장비 순서 번호입니다.

   예제에서는 이전 볼륨 일련 번호 *AZ0001*을 새 일련 번호 *VOL120*으로 교체하여 *804* 드라이브에서 테이프 카트리지를 다시 초기화합니다.

   ```
   root@solaris:~# tplabel -old AZ0001 -vsn VOL120 804
   root@solaris:~#
   ```

5. 테이프 드라이브에 있는 테이프의 레이블을 재지정하려면 *tplabel -old old-volume-serial-number -new new-volume-serial-number library-equipment-number:slot* 명령을 사용합니다. 설명:
   - *volume-serial-number*는 필수 볼륨 일련 번호입니다.
   - *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일이 드라이브에 지정하는 장비 순서 번호입니다.
   - *slot*은 라이브러리 내 카트리지 위치입니다.

   필요한 경우 기존 볼륨 일련 번호를 재사용할 수 있습니다. 예제에서는 볼륨 일련 번호를 기존 볼륨 일련 번호 *VOL121*로 교체하여 *800* 라이브러리의 *23* 슬롯에 있는 테이프 카트리지를 다시 초기화합니다:

   ```
   root@solaris:~# tplabel -old VOL601 -vsn VOL601 800:23
   root@solaris:~#
   ```

6. 여기서 중지합니다.

새 광 디스크 레이블 지정 또는 기존 광 디스크 레이블 다시 지정

1. 파일 시스템 호스트에 *root*로 로그인합니다.

   ```
   root@solaris:~#
   ```

2. 드라이브에 로드된 새 광 카트리지에 레이블을 지정하려면 *odlabel -new volume-serial-number drive-equipment-number[:side]* 명령을 사용합니다. 설명:
   - *volume-serial-number*는 필수 볼륨 일련 번호입니다.
   - *drive-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일이 드라이브에 지정하는 장비 순서 번호입니다.
   - *side*(선택사항)는 양면 디스크에서 지정된 한 면입니다.

   예제에서는 *701* 드라이브의 새로운 단면 광 카트리지에 볼륨 일련 번호 *OD1700*을 지정합니다.

   ```
   root@solaris:~# odlabel -new -vsn OD1700 701
   root@solaris:~#
   ```

3. 자동화된 매체 라이브러리에 있는 새 광 카트리지에 레이블을 지정하려면 *odlabel -new volume-serial-number library-equipment-number:slot[:side]* 명령을 사용합니다. 설명:
   - *volume-serial-number*는 필수 볼륨 일련 번호입니다.
   - *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일이 드라이브에 지정하는 장비 순서 번호입니다.
   - *slot*은 라이브러리 내에 있는 카트리지의 위치이고 *side*(선택사항)는 양면 디스크의 지정된 한 면입니다.

   예제에서는 *700* 라이브러리의 *42* 슬롯에서 새로운 양면 광 카트리지의 *2*면에 볼륨 일련 번호 *OD1701*을 지정합니다.

   ```
   root@solaris:~# odlabel -new -vsn OD1701 700:42:2
   root@solaris:~#
   ```

4. 드라이브에 로드한 광자기 카트리지의 레이블을 재지정하려면 *odlabel -old old-volume-serial-number -new new-volume-serial-number drive-equipment-number[:side]* 명령을 사용합니다. 설명:
   - *volume-serial-number*는 필수 볼륨 일련 번호입니다.
   - *drive-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일이 드라이브에 지정하는 장비 순서 번호입니다.
   - *side*(선택사항)는 양면 디스크에서 지정된 한 면입니다.

예제에서는 이전 볼륨 일련 번호 *OD1120*을 새 볼륨 일련 번호 *OD1120*으로 교체하여 *702* 드라이브에서 광자기 카트리지를 다시 초기화합니다.

```
root@solaris:~# odlabel -old OD0001 -vsn OD1120 702
root@solaris:~#
```

5. 자동화된 매체 라이브러리에 있는 기존 광 카트리지에 레이블을 다시 지정하려면 *odlabel -old volume-serial-number library-equipment-number:slot[:side]* 명령을 사용합니다. 설명:
   - *volume-serial-number*는 필수 볼륨 일련 번호입니다.
   - *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일이 라이브러리에 지정하는 장비 순서 번호입니다.
   - *side*(선택사항)는 양면 디스크에서 지정된 한 면입니다.

   필요한 경우 기존 볼륨 일련 번호를 재사용할 수 있습니다. 예제에서는 볼륨 일련 번호를 기본 볼륨 일련 번호 *OD1121*로 교체하여 *700* 라이브러리의 *23* 슬롯에 있는 광자기 카트리지를 다시 초기화합니다.

```
root@solaris:~# odlabel -old OD1121 -vsn OD1121 800:23
root@solaris:~#
```

6. 여기서 중지합니다.

데이터 무결성 유지 관리

Oracle Hierarchical Storage Manager 소프트웨어는 이동식 테이프 매체에 저장된 데이터 파일의 무결성을 유지 관리하기 위한 요구 시/자동화된 도구를 제공합니다. 이 절에서는 다음 항목을 다룹니다.

- DIV(데이터 무결성 검증) 설정 및 상태 표시
- 주어진 테이프 볼륨의 무결성 검사
- 자동화된 무결성 검증 모니터링.

DIV(데이터 무결성 검증) 설정 및 상태 표시

이 절에서는 다음 작업을 다룹니다.

- DIV 설정 표시
- 아카이브 파일의 쓰기 후 확인 상태 모니터링
- 장치의 쓰기 후 확인 상태 모니터링.

DIV 설정 표시

DIV(데이터 무결성 검증) 설정을 표시하려면 *samcmd L* 명령을 사용하고 출력을 Solaris *grep* 명령 및 정규 표현식 *div*로 파이프합니다.

예제에서는 DIV가 *OFF*입니다.

```
root@solaris:~# samcmd L | grep div
div             OFF
root@solaris:~#
```

아카이브 파일의 쓰기 후 확인 상태 모니터링

아카이빙 동안 아카이브 파일의 확인 상태를 모니터하려면 *samu* 인터페이스를 사용합니다. *samu -d a* 명령을 사용합니다.

```
root@solaris:~# samu -d a
Archiver status                          samu 5.4         22:22:31 Mar 4 2014
sam-archiverd: Archiving files
sam-arfind:  hsmfs1 mounted at /hsm/hsmfs1
Files waiting to start     12,576  schedule     26,695  archiving     13,120
...
sam-arcopy: qfs.arset1.2.344 ti.TKC960
Verifying archive file at position 1175
```

장치의 쓰기 후 확인 상태 모니터링

아카이빙 동안 장치의 확인 상태를 모니터하려면 *samu* 인터페이스를 사용합니다. *samu -d s* 명령을 사용합니다.

```
root@solaris:~# samu -d s
Device status                          samu 5.4           22:27:53 Mar 4 2014
ty     eq state  device_name                      fs  status
sn    800 on     /dev/scsi/changer/c1t2d0        800 n--------r
ti    801 on     /dev/rmt/0cbn                   800 ---------p
...
hy    805 on     historian                       805 ----------
ti     91 on     /dev/rmt/4cbn                    90 -l----oVr
Verify averaging 240.9 MB/s
```

주어진 테이프 볼륨의 무결성 검사

특정 테이프 볼륨의 데이터 무결성을 확인해야 하는 경우 Oracle HSM *tpverify* 명령을 사용합니다. *tpverify* 명령은 Oracle T10000C/D, LTO 및 기타 흔히 사용되는 매체를 지원합니다. T10000C/D 매체는 Oracle 데이터 무결성 검증을 사용하여 확인됩니다. 기타 형식은 널리 지원되는 SCSI *verify(6)* 명령을 사용하여 검사합니다.

다음 절에서는 *tpverify*를 사용할 수 있는 방법을 간략히 설명합니다. 구문 및 옵션에 대한 자세한 내용은 *tpverify* 매뉴얼 페이지를 참조하십시오.

- 라이브러리 위치로 지정된 테이프에서 데이터 확인
- 매체 유형 및 볼륨 일련 번호로 지정된 테이프에서 데이터 확인
- 지정된 드라이브를 사용하여 테이프에서 데이터 확인

- 테이프 시작 부분부터 데이터 확인 다시 시작
- T10000C/D 테이프의 모든 블록에 대한 ECC 확인
- T10000C/D 테이프의 모든 블록에 대한 ECC 및 DIV 체크섬 확인
- T10000C/D 테이프의 MIR(매체 정보 영역) 재구축
- 지정된 테이프에 대한 데이터 확인 취소
- 테이프에 대한 DIV 상태 및 확인 진행률 표시

라이브러리 위치로 지정된 테이프에서 데이터 확인

tpverify library-equipment-number:slot 명령을 사용합니다. 여기서 *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 자동화된 테이프 라이브러리에 지정하는 장비 순서 번호이고 *slot*은 라이브러리 내에서 대상 볼륨이 상주하는 슬롯 주소입니다.

tpverify 명령은 라이브러리 매체 카탈로그를 검사하여 확인된 마지막 테이프 위치를 찾습니다. 그런 다음 첫번째 사용 가능한 드라이브로 테이프를 로드하고 마지막 중지된 지점부터 검증을 시작합니다. T10000C/D 매체는 기본 방식인 *tpverify* Standard 방식을 사용하고 기타 매체는 SCSI *verify(6)*를 사용합니다. Standard 방식은 속도에 최적화된 방식이며, Oracle HSM 매체의 모서리, 시작, 끝과 처음 1,000개 블록을 확인합니다.

예제에서는 Standard 방식을 사용하여 *800* 라이브러리의 *9* 슬롯에 저장된 T10000D 테이프를 검증합니다.

```
root@solaris:~# tpverify 800:9
```

매체 유형 및 볼륨 일련 번호로 지정된 테이프에서 데이터 확인

tpverify mediatype.volume-serial-number 명령을 사용합니다. 여기서 *mediatype*은 부록 A. 장비 유형 용어집에 나열된 2자의 매체 유형 코드 중 하나이고 *volume-serial-number*는 라이브러리 내에서 볼륨을 고유하게 식별하는 영숫자 문자열입니다.

tpverify 명령은 라이브러리 매체 카탈로그를 검사하여 확인된 마지막 테이프 위치를 찾습니다. 그런 다음 첫번째 사용 가능한 드라이브로 테이프를 로드하고 마지막 중지된 지점부터 검증을 시작합니다. T10000C/D 매체는 기본 방식인 *tpverify* Standard 방식을 사용하고 기타 매체는 SCSI *verify(6)*를 사용합니다.

예제에서는 SCSI **verify(6)** 명령을 사용하여 LTO(**li**) 볼륨 **VOL006**을 검증합니다.

```
root@solaris:~# tpverify li.VOL006
```

지정된 드라이브를 사용하여 테이프에서 데이터 확인

tpverify library-equipment-number:slot device-equipment-number 명령을 사용합니다. 설명:

- *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 자동화된 테이프 라이브러리에 지정하는 장비 순서 번호입니다.
- *slot*은 라이브러리 내에서 대상 볼륨이 상주하는 슬롯 주소입니다.
- *device-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 드라이브에 지정하는 장비 순서 번호입니다.

예제에서는 *803* 드라이브를 사용하여 *800* 라이브러리의 *17* 슬롯에 저장된 T10000D 테이프를 검증합니다.

```
root@solaris:~# tpverify 800:17 803
```

테이프 시작 부분부터 데이터 확인 다시 시작

tpverify -a library-equipment-number:slot 또는 *tpverify -a mediatype.volume-serial-number* 명령을 사용합니다. 설명:

- *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 자동화된 테이프 라이브러리에 지정하는 장비 순서 번호입니다.
- *slot*은 라이브러리 내에서 대상 볼륨이 상주하는 슬롯 주소입니다.
- *mediatype*은 부록 A. 장비 유형 용어집에 나열된 2자 매체 유형 코드 중 하나입니다.
- *volume-serial-number*는 라이브러리 내에서 볼륨을 고유하게 식별하는 영숫자 문자열입니다.

-a 옵션은 기본 동작을 대체하여 매체 시작 부분부터 확인하기 시작합니다. 매체 카탈로그에 기록된 위치는 무시됩니다.

예제에서는 테이프 시작 부분부터 LTO(*li*) 볼륨 *VOL016*을 검증합니다.

```
root@solaris:~# tpverify -a li.VOL016
```

T10000C/D 테이프의 모든 블록에 대한 ECC 확인

tpverify -C library-equipment-number:slot 또는 *tpverify -C mediatype.volume-serial-number* 명령을 사용합니다. 설명:

- *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 자동화된 테이프 라이브러리에 지정하는 장비 순서 번호입니다.
- *slot*은 라이브러리 내에서 대상 볼륨이 상주하는 슬롯 주소입니다.
- *mediatype*은 부록 A. 장비 유형 용어집에 나열된 2자 매체 유형 코드 중 하나입니다.
- *volume-serial-number*는 라이브러리 내에서 볼륨을 고유하게 식별하는 영숫자 문자열입니다.

tpverify 명령은 라이브러리 매체 카탈로그를 검사하여 확인된 마지막 테이프 위치를 찾습니다. 그런 다음 *-C* 옵션으로 지정된 Complete 방식을 사용하여 마지막 중지된 지점부터 검

증을 시작합니다. Complete 방식은 Standard 방식보다 정밀하지만 아주 느려질 수 있습니다. 매체의 모든 블록에서 ECC(Error Correction Codes)를 검사합니다.

예제에서는 Complete 방식을 사용하여 T10000D(*ti*) 볼륨 *VOL516*을 검증합니다.

```
root@solaris:~# tpverify -C ti.VOL516
```

T10000C/D 테이프의 모든 블록에 대한 ECC 및 DIV 체크섬 확인

tpverify -P library-equipment-number:slot 또는 *tpverify -P mediatype.volume-serial-number* 명령을 사용합니다. 설명:

- *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 자동화된 테이프 라이브러리에 지정하는 장비 순서 번호입니다.
- *slot*은 라이브러리 내에서 대상 볼륨이 상주하는 슬롯 주소입니다.
- *mediatype*은 부록 A. 장비 유형 용어집에 나열된 2자 매체 유형 코드 중 하나입니다.
- *volume-serial-number*는 라이브러리 내에서 볼륨을 고유하게 식별하는 영숫자 문자열입니다.

tpverify 명령은 라이브러리 매체 카탈로그를 검사하여 확인된 마지막 테이프 위치를 찾습니다. 그런 다음 *-P* 옵션으로 지정된 Complete Plus 방식을 사용하여 마지막 중지된 지점부터 검증을 시작합니다. Complete Plus 방식은 매우 철저한 방식이지만 다른 방식보다 속도가 느립니다. 매체의 모든 블록에서 ECC(Error Correction Codes) 및 데이터 무결성 검증 체크섬을 검사합니다.

예제에서는 Complete Plus 방식을 사용하여 T10000D(*ti*) 볼륨 *VOL521*을 검증합니다.

```
root@solaris:~# tpverify -P ti.VOL521
```

T10000C/D 테이프의 MIR(매체 정보 영역) 재구축

tpverify -M library-equipment-number:slot 또는 *tpverify -M mediatype.volume-serial-number* 명령을 사용합니다. 설명:

- *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 자동화된 테이프 라이브러리에 지정하는 장비 순서 번호입니다.
- *slot*은 라이브러리 내에서 대상 볼륨이 상주하는 슬롯 주소입니다.
- *mediatype*은 부록 A. 장비 유형 용어집에 나열된 2자 매체 유형 코드 중 하나입니다.
- *volume-serial-number*는 라이브러리 내에서 볼륨을 고유하게 식별하는 영숫자 문자열입니다.

tpverify 명령은 테이프가 매체 카탈로그에서 불량으로 표시된 경우에도 Oracle StorageTek 테이프 카트리지에서 누락되었거나 손상된 MIR(매체 정보 영역)을 재구축합니다. 재구축은 MIR 손상이 발견되었을 때 자동으로 지정됩니다.

예제에서는 MIR Rebuild 방식을 사용하여 T10000D(*ti*) 볼륨 *VOL523*을 검증합니다.

```
root@solaris:~# tpverify -M ti.VOL523
```

지정된 테이프에 대한 데이터 확인 취소

tpverify -c library-equipment-number:slot 또는 *tpverify -c mediatype.volume-serial-number* 명령을 사용합니다. 설명:

- *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 자동화된 테이프 라이브러리에 지정하는 장비 순서 번호입니다.
- *slot*은 라이브러리 내에서 대상 볼륨이 상주하는 슬롯 주소입니다.
- *mediatype*은 부록 A. 장비 유형 용어집에 나열된 2자 매체 유형 코드 중 하나입니다.
- *volume-serial-number*는 라이브러리 내에서 볼륨을 고유하게 식별하는 영숫자 문자열입니다.

tpverify -c 명령은 현재 확인 작업을 취소하고 매체 카탈로그에 테이프의 마지막 확인 위치를 기록합니다. 따라서 아카이브/스테이지용 드라이브나 볼륨을 확보하기 위해 확인 작업을 중지했다가 나중에 동일 지점에서 확인을 재개할 수 있습니다.

예제에서는 T10000D(*ti*) 볼륨 *VOL533*의 확인을 취소합니다.

```
root@solaris:~# tpverify -c ti.VOL523
```

테이프에 대한 DIV 상태 및 확인 진행률 표시

itemize -2 library-equipment-number 명령을 사용합니다. 여기서 *library-equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에서 자동화된 테이프 라이브러리에 지정하는 장비 순서 번호입니다.

itemize -2 명령은 지정된 라이브러리에 매체를 보관하고 각 볼륨에 대한 DIV 상태 및 확인 진행률을 나열합니다.

예제에서는 장비 순서 번호 *800*을 가진 라이브러리에서 볼륨의 확인 상태를 표시합니다. *lvtime*(마지막 확인 시간) 필드는 *tpverify*가 전체 테이프 확인을 마지막으로 완료한 시간을 표시합니다. *status* 필드 값 *div*는 DIV 가능 테이프임을 나타내고 *none* 값은 그렇지 않음을 나타냅니다. *lvpos*(마지막 확인 위치) 필드는 *tpverify*가 마지막으로 취소되었고 다시 실행할 때 시작할 위치를 보여줍니다.

```
root@solaris:~# itemize -2 800
Robot VSN catalog: eq: 800       count: 60
slot    access_time  count  use  ty vsn
        lvtime      status          lvpos
   0    Apr  2 16:34     6   0%  ti VOL519
        Apr  2 09:23   div           0
   1    Apr  2 16:17    28  29%  ti VOL510
        Apr  2 16:17   div           0x9bb9
   2    none             0   0%  ti VOL511
```

```
        none          none          0
  ...
root@solaris:~#
```

자동화된 무결성 검증 모니터링

주기적 매체 확인은 *tpverify* 명령이 자동화된 형태입니다. 이 섹션에서는 상황에 따라 필요할 수 있는 유지 관리 작업에 대한 지침을 제공합니다. 이러한 작업에는 다음이 포함됩니다.

- **verifyd.cmd** 구성 파일 보기 및 검증
- **verifyd.cmd** 구성 파일 다시 로드
- 주기적 매체 확인 테이프 결함 데이터베이스에 나열된 모든 결함 표시
- 특정 볼륨에 대해 나열된 결함 표시
- 주기적 매체 확인 테이프 결함 데이터베이스에 나열된 결함 지우기.

주기적 매체 확인 구성에 대한 지침은 고객 설명서 라이브러리(*http://docs.oracle.com/en/storage/#sw*)의 *Oracle Hierarchical Storage Manager and StorageTek QFS* 설치 및 구성 설명서를 참조하십시오.

verifyd.cmd 구성 파일 보기 및 검증

언제든지 *verifyd.cmd* 파일을 보거나 편집 후 파일을 검증하려면 *tpverify -x* 명령을 사용합니다.

tpverify -x 명령은 */etc/opt/SUNWsamfs/verifyd.cmd* 파일을 검사하고 오류를 불러오거나 파일의 내용을 표시합니다.

```
root@solaris:~# tpverify -x
Reading '/etc/opt/SUNWsamfs/verifyd.cmd'.
PMV: off
     Run-time:
     Start Time: 2200
End Time: 0500
PMV Scan: all
PMV Method: Standard
STA Scan: off
Action: none
PMV VSNs: all
PMV Policy:
     Last Verified Age: 6m
root@solaris:~#
```

verifyd.cmd 구성 파일 다시 로드

확인 프로세스를 중지하지 않고 *verifyd.cmd* 파일을 다시 로드하려면 *tpverify -r* 명령을 사용합니다.

```
root@solaris:~# tpverify -r
```

```
root@solaris:~#
```

주기적 매체 확인 테이프 결함 데이터베이스에 나열된 모든 결함 표시

주기적 매체 확인으로 식별되고 테이프 결함 데이터베이스에 저장된 모든 결함을 나열하려면 *tpverify -l* 명령을 사용합니다.

예제에서는 데이터베이스에 결함이 없습니다.

```
root@solaris:~# tpverify -l
No defects found.
root@solaris:~#
```

특정 볼륨에 대해 나열된 결함 표시

특정 볼륨에서 식별된 결함을 모두 나열하려면 *tpverify -l mediatype.volume-serial-number* 명령을 사용합니다. 설명:

- *mediatype*(선택사항)은 부록 A. 장비 유형 용어집에 나열된 2자 매체 유형 코드 중 하나입니다.
- *volume-serial-number*는 라이브러리 내에서 볼륨을 고유하게 식별하는 영숫자 문자열입니다.

예제에서는 LTO (*ti*) 볼륨 *VOL514*에 대해 데이터베이스에 나열된 결함이 없습니다.

```
root@solaris:~# tpverify -l ti.VOL514
No defects found.
root@solaris:~#
```

주기적 매체 확인 테이프 결함 데이터베이스에 나열된 결함 지우기

주기적 매체 확인으로 식별된 모든 결함을 테이프 결함 데이터베이스에서 삭제하려면 *tpverify -d* 명령을 사용합니다.

특정 볼륨에 대해 나열된 결함을 모두 삭제하려면 *tpverify -d mediatype.volume-serial-number* 명령을 사용합니다. 설명:

- *mediatype*(선택사항)은 부록 A. 장비 유형 용어집에 나열된 2자 매체 유형 코드 중 하나입니다.
- *volume-serial-number*는 라이브러리 내에서 볼륨을 고유하게 식별하는 영숫자 문자열입니다.

```
root@solaris:~# tpverify -d
root@solaris:~# tpverify -d ti.VOL514
root@solaris:~#
```

6

6장. 디지털 보존용 아카이브 관리

지금까지 이 문서는 사용자와 응용 프로그램이 정기적으로 파일을 만들고 수정 및 삭제하는 일반적인 UNIX 파일 시스템으로서 Oracle Hierarchical Storage Manager and StorageTek QFS Software 솔루션 관리에 대해 설명했습니다. 고도로 통합된 백업 서비스로 주로 작용하는 아카이브와 함께 디스크 캐시에 초점을 맞췄습니다. 이 장에서는 장기간 데이터 보존을 위한 저장소 및 관리 솔루션으로서 아카이브에 다시 초점을 맞춥니다. 이전에 다루었던 관리 원칙과 기법은 관련 항목으로 남습니다. 하지만 지금 디스크 캐시는 아카이브로 파일을 입수하는 수단으로 주로 작용하며, 입수 이후 삭제나 수정은 허용되지 않습니다.

정확한 요구 사항은 다양합니다. 법적 의무 기간 동안 업무나 의료 기록을 보관하는 저장소는 정기적으로 레코드를 폐기해야 할 수 있습니다. 그러나 과학적 데이터, 역사적 계보 기록 또는 디지털 음악, 영화, 텔레비전 프로그램을 저장하는 아카이브는 사실상 영구히 컨텐츠를 저장해야 할 수 있습니다. 이러한 이유로 Oracle HSM은 여러 방법으로 디지털 보존을 지원합니다.

- 메시지 다이제스트(체크섬)는 파일 파손, 데이터 손상, 무단 수정을 감지할 수 있으므로 하드웨어 문제를 해결하고 부적합한 파일을 아카이브에 저장된 적합한 복사본으로 교체할 수 있습니다.
- 파일 고정성 속성은 메시지 다이제스트와 작동하여 수퍼 유저만 고정된 파일을 변경할 수 있도록 보장합니다. Oracle HSM이 고정된 파일을 스테이지/아카이브할 때마다 고정성 속성으로 저장된 체크섬을 재검증하여 파일이 변경되지 않았음을 입증합니다.
- Oracle HSM WORM(Write Once Read Many) 파일 시스템은 파일을 읽기 전용으로 만들고 지정된 기간 동안 강제 보존할 수 있습니다. 이 파일 시스템은 수퍼 유저가 파일이나 파일 속성(위에 설명된 고정성 속성)을 변경할 수 없도록 구성할 수 있습니다.

이 장은 장기간용 스토리지 솔루션의 기초를 이루는 기본적인 Oracle HSM 데이터 보호 조치(보존용 파일 시스템 구성)를 간략히 검토하는 것부터 시작합니다. 그 다음 구체적으로 데이터 보존을 처리하는 작업을 설명합니다.

보존용 파일 시스템 구성

모든 보존 솔루션은 적합한 고중복성 파일 시스템으로 시작합니다. 따라서 아직 수행하지 않은 경우 동봉된 Oracle Hierarchical Storage Manager and StorageTek QFS 설치 및 구성 설명서의 구현 장을 검토하십시오. 중복 서버, 네트워크 연결, 스토리지 장치를 제공하여

아카이브에 대한 액세스를 보호합니다. 파일마다 최소 2개 이상의 추가 복사본을 구성하고 각각 독립된 매체에 저장하여 파일 데이터를 보호합니다. 대부분의 경우 복사본 1개는 디스크나 솔리드 상태 스토리지 장치에, 복사본 2개는 테이프 매체에 아카이브하는 것이 이상적입니다. 가능한 경우 Oracle HSM 데이터 무결성 검증 기능을 구현하여 테이프 블록을 올바로 읽고 쓸 수 있도록 보장합니다. 정기적으로 덤프 파일을 생성하고 정기적으로 아카이브 로그를 백업하여 파일 시스템 메타데이터를 보호합니다.

메시지 다이제스트(체크섬) 사용

메시지 다이제스트(체크섬)로 보존자는 컨텐츠에 점진적 악화, 하드웨어/운영자 오류 또는 고의적 무단 변경을 나타낼 수 있는 변경사항이 있는지 아카이브된 파일을 테스트할 수 있습니다. 메시지 다이제스트는 간단히 단방향 암호화 해시 함수로 생성된 파일 컨텐츠를 수학적으로 요약한 것입니다. 암호화 해시 함수는 입력 데이터의 변경에 극도로 민감합니다. 입력을 조금만 변경해도 출력이 크게 변경됩니다. 따라서 메시지 다이제스트는 파일 손상 및 무단 변경을 감지하기에 이상적입니다. 파일의 다이제스트를 재계산하고 저장된 다이제스트 값과 결과 값을 비교하면 파일이 변경되었는지 여부가 나타납니다.

Oracle Hierarchical Storage Manager 파일 시스템은 다음 암호화 해시 함수 중 하나를 사용하여 메시지 다이제스트를 입수, 만들기, 저장 및 검증할 수 있습니다.

- SHA1 - 보안 해시 알고리즘 계열 암호화 함수의 160비트 멤버

  보안 해시 알고리즘은 NIST(National Institute of Standards and Technology)가 2012년 발행한 *FIPS(Federal Information Processing Standard) Publication 180-4*에 정의되어 있습니다. Oracle HSM은 기본적으로 SHA1을 사용합니다.
- SHA256 - 보안 해시 알고리즘 계열의 256비트 멤버
- SHA384 - 보안 해시 알고리즘 계열의 384비트 멤버
- SHA512 - 보안 해시 알고리즘 계열의 512비트 멤버
- MD5 - IETF(Internet Engineering Task Force)가 RFC(Request for Comment) 1321에 정의한 128비트 메시지 다이제스트 함수
- 지금은 주로 이전의 Storage Archive Manager 구현과 역호환성을 위해 사용되는 독점적 128비트 Oracle HSM 함수입니다.

저장소로 파일을 입수할 때 사용자가 기존 다이제스트 값을 제공하거나, 파일 시스템이 즉시 또는 파일을 처음 아카이브할 때 해당 값을 계산할 수 있습니다. Oracle HSM 파일 시스템은 특수 파일 속성을 사용하여 파일 시스템 메타데이터와 함께 다이제스트 값을 저장합니다. 일단 속성이 설정되면 파일 시스템은 다이제스트를 재계산하고, 해당 파일을 다시 아카이브할 때마다 또는 선택적으로 아카이브 매체에서 디스크 캐시로 파일을 스테이지할 때마다 저장된 값에 대해 결과를 검증합니다.

그러나 Oracle HSM 매체 마이그레이션 기능은 체크섬을 재계산하지 않고 새 매체로 파일을 복사합니다. 매체 마이그레이션에 대한 자세한 내용은 8장. 새 스토리지 매체로 마이그레이션을 참조하십시오. 파일을 제대로 복사하지 않으면 파일을 다시 스테이지하고 검증할 때까지 손상이 감지되지 않을 작은 위험이 있습니다. DIV(데이터 무결성 검증)를 사용하면 이 위험이 최소화됩니다(세부정보는 Oracle Hierarchical Storage Manager and StorageTek QFS 설치 및 구성 설명서 참조).

메시지 다이제스트 사용을 시작하기 전에 먼저 파일 시스템 호스트 성능이 적절한지 확인을 수행해야 합니다. 그리고 다음 절을 참조하여 다이제스트 제공, 생성, 검증 지침을 확인할 수 있습니다.

- 메시지 다이제스트를 제공하고 파일에 대한 검증을 사용으로 설정
- 메시지 다이제스트를 생성하고 파일에 대한 검증을 사용으로 설정
- 메시지 다이제스트를 생성하고 디렉토리의 각 파일에 대한 검증을 사용으로 설정
- 스테이지 동안 파일의 메시지 다이제스트 검증
- 파일을 아카이브하기 전에 메시지 다이제스트 및 검증 속성 변경

파일 시스템 호스트 성능이 적절한지 확인

메시지 다이제스트를 빈번히 사용할 계획이면 적절한 성능을 위해 파일 시스템 호스트에 충분한 컴퓨팅 리소스가 있는지 확인합니다. 대부분의 최신 플랫폼은 중앙 프로세서 주기를 소비하지 않고 특화된 계산을 효율적으로 수행할 수 있는 전용 암호화 하드웨어를 포함합니다. 가능한 경우 이 기능을 활용해야 합니다.

잠재적 파일 시스템 호스트의 기능을 확인하려면 다음과 같이 하십시오.

1. 파일 시스템 호스트에 *root*로 로그인합니다:

   ```
   root@solaris:~#
   ```

2. 호스트 운영체제가 Solaris 11.1 이상인지 확인합니다. *uname -v* 명령을 사용합니다.

   이전 버전의 운영체제는 해시 함수의 하드웨어 가속을 지원하지 않습니다. 예제에서는 호스트 운영체제가 Solaris 11.2입니다.

   ```
   root@solaris:~# uname -v
   11.2
   root@solaris:~#
   ```

3. 명령 세트 구조를 표시합니다. 명령 프롬프트에서 *isainfo -v* 명령을 입력합니다.

   ```
   root@solaris:~# isainfo -v
   ```

4. Solaris 11 호스트가 Oracle Sun SPARC T3 이상의 시스템인 경우 *isainfo -v* 명령의 출력은 *sha512*, *sha256*, *sha1*, *md5* 암호화 알고리즘을 지원하는 명령 세트를 나열해야 합니다.

   예제에서는 Sun SPARC T4-2 호스트가 SHA1, SHA2, MD5 알고리즘 계열에 대한 하드웨어 가속을 제공합니다.

   ```
   [Sun_SPARC_T4-2]root@solaris:~# isainfo -v
   64-bit sparcv9 applications
   ```

```
        crc32c cbcond pause mont mpmul sha512 sha256 sha1 md5 camellia kasumi
        des aes ima hpc vis3 fmaf asi_blk_init vis2 vis popc
root@solaris:~#
```

5. Solaris 호스트가 x86/64 시스템인 경우 *isainfo -v* 명령의 출력에 *ssse3*(Supplemental Streaming SIMD Extensions 3) 명령 세트가 포함될 경우 SHA-1 하드웨어 가속을 지원합니다.

   예제에서는 Sun X3-2 호스트가 SHA-1 다이제스트의 하드웨어 가속을 지원합니다.

```
[Sun_X3-2]root@solaris:~# isainfo -v
64-bit amd64 applications
        avx xsave pclmulqdq aes sse4.2 sse4.1 ssse3 popcnt tscp ahf cx16 sse3
        sse2 sse fxsr mmx cmov amd_sysc cx8 tsc fpu
root@solaris:~#
```

메시지 다이제스트를 제공하고 파일에 대한 검증을 사용으로 설정

이미 메시지 다이제스트와 연관된 파일을 아카이브하는 경우 다음과 같이 하십시오.

1. 파일 시스템 호스트에 *root*로 로그인합니다:

```
root@solaris:~#
```

2. 명령 프롬프트에서 *ssum -a algorithm -h digest -G [-u]filename* 명령을 입력합니다. 설명:
   - *-a algorithm*은 제공된 메시지 다이제스트에 대해 파일을 검증할 때 파일 시스템이 사용할 암호화 해시 함수를 식별합니다.
   - *-h digest*는 파일 시스템이 파일 검증에 사용할 메시지 다이제스트를 식별합니다.
   - *-G*는 즉시 검증을 지정합니다. 파일 시스템은 *hash* 파일 속성을 제공된 메시지 다이제스트 값으로 설정하고, 독립적으로 파일의 메시지 다이제스트를 계산하고, 저장된 값과 결과를 비교합니다. 제공된 다이제스트와 계산된 다이제스트가 일치하면 파일 시스템은 파일의 *validated* 속성을 설정합니다. 그 다음 *generate* 속성을 설정하여 파일을 다시 아카이브할 때마다 유효성을 다시 검사하도록 합니다.
   - *-u*는 *use* 파일 속성(선택사항)을 설정합니다. 파일을 스테이지할 때마다 파일 시스템은 다이제스트를 재계산하고 *hash* 속성에 저장된 값에 대해 결과를 검증합니다.
   - *filename*은 파일의 경로 및 이름입니다.

   예제에서는 SHA256 다이제스트를 제공하고 파일 시스템이 즉시 재계산하면 제공된 값에 대해 *data10* 파일의 다이제스트 값을 검증합니다. *sls -D -h data10* 명령으로 파일 속성을 확인하면 *generate* 및 *validated* 파일 속성이 설정되었고, *algorithm* 속성이 *SHA-256*으로 설정되었고, 다이제스트 값이 계산 후 *hash* 속성에 저장되었음을 알 수 있습니다.

```
root@solaris:~# ssum -h f03ce01b3828...f7459503007e -a sha256 -g data10
```

```
root@solaris:~# sls -D -h data10
data10:
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14975  admin id:      0  inode:    90217.1
  project: user.root(1)
  access:      Jul 16 16:14  modification: Jul 16 16:14
  changed:     Jul 16 16:15  attributes:   Jul 16 16:14
  creation:    Jul 16 16:14  residence:    Jul 16 16:14
  checksum: generate validated  algorithm: SHA-256
  hash: f03ce01b3828...f7459503007e
root@solaris:~#
```

3. 필요한 경우 평소에 하듯이 파일을 편집합니다.

   예제에서는 마지막 아카이브 이후 *data10m*이라는 파일을 수정했습니다. *sls -D -h* 명령이 보여주듯이 어느 쪽도 가장 최근 변경사항을 반영하지 않으므로 *S*(사용되지 않음) 플래그가 두 복사본에 설정되었습니다. Solaris *digest* 명령을 사용하여 수정된 파일에 대한 SHA-256 다이제스트 값을 확인하면 파일의 *hash* 속성이 오래된 다이제스트 값도 저장함을 알 수 있습니다.

```
root@solaris:~# sls -D -h data10m
data10m:
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14983  admin id:      0  inode:    90307.1
  project: user.root(1)
  copy 1: S----- Jul 17 16:47        dd.1    dk diskarchive f221
  copy 2: S----- Jul 20 11:31       a8d.1    li VOL002
  access:      Jul 20 11:32  modification: Jul 20 11:31
  changed:     Jul 17 16:37  attributes:   Jul 17 16:36
  creation:    Jul 17 16:36  residence:    Jul 17 16:36
  checksum: generate  algorithm: SHA-256
  hash: f03ce01b3828...f7459503007e
root@solaris:~# digest -a sha256 data10m
56c55bb421cc...71ac2ac0b7b0
root@solaris:~#
```

4. 필요한 경우 다시 아카이브하기 전에 수정된 파일의 다이제스트 속성을 변경할 수 있습니다.

   예제에서는 다이제스트 알고리즘을 SHA256에서 SHA1로 변경하고 즉시 효력을 발휘합니다.

```
root@solaris:~# ssum -a sha1 -G data10m
root@solaris:~# sls -D -h data10m
data10m:
```

```
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14983  admin id:      0  inode:    90307.1
  project: user.root(1)
  release -a;
  copy 1: S----- Jul 20 13:00        e0.1    dk diskarchive f224
  copy 2: S----- Jul 20 13:05       a93.1    li VOL002
  access:      Jul 20 16:39  modification: Jul 20 16:39
  changed:     Jul 17 16:37  attributes:   Jul 17 16:36
  creation:    Jul 17 16:36  residence:    Jul 20 16:29
  checksum: generate validated  algorithm: SHA-1
  hash: 92003525f0f8...53e29d0718c8
root@solaris:~#
```

5. 그렇지 않으면, 파일 시스템이 수정된 파일을 아카이브하기를 기다렸다가 자동으로 다이제스트 관련 속성을 업데이트합니다.

   수정된 파일을 아카이브할 때 파일 시스템은 다이제스트 값을 재계산하고, 새 값을 *hash* 속성에 저장하고, 이전 파일 버전의 아카이브된 복사본에 *S*(사용되지 않음) 플래그를 설정합니다. 예제에서는 다이제스트 속성을 변경하지 않고 *data10m* 파일을 편집했습니다. 아카이버는 예정대로 디스크에 *copy 1*을 새로 만들었고 *hash* 속성을 업데이트했습니다. 수정되지 않은 파일의 복사본은 아카이버가 *copy 2*를 만들 때까지 *S*(사용되지 않음) 플래그가 지정된 채 테이프에 남아 있습니다.

```
root@solaris:~# sls -D -h data10m
data10m:
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14983  admin id:      0  inode:    90307.1
  project: user.root(1)
  copy 1: ------ Jul 17 16:47        dd.1    dk diskarchive f221
  copy 2: S----- Jul 20 11:31       a8d.1    li VOL002
  access:      Jul 20 11:32  modification: Jul 20 11:31
  changed:     Jul 17 16:37  attributes:   Jul 17 16:36
  creation:    Jul 17 16:36  residence:    Jul 17 16:36
  checksum: generate  algorithm: SHA-256
  hash: 56c55bb421cc...71ac2ac0b7b0
```

메시지 다이제스트를 생성하고 파일에 대한 검증을 사용으로 설정

파일의 다이제스트를 생성하고 파일 검증을 사용으로 설정하려면 다음과 같이 하십시오.

1. 파일 시스템 호스트에 *root*로 로그인합니다:

```
root@solaris:~#
```

2. 명령 프롬프트에서 *ssum -a algorithm -g|G [-u] filename* 명령을 입력합니다. 설명:

- *-a algorithm*은 파일의 메시지 다이제스트를 생성할 때 파일 시스템이 사용할 암호화 해시 함수를 지정합니다.
- *-g*는 파일에 대한 *generate* 파일 속성을 설정합니다. 처음 파일을 아카이브할 때 파일 시스템은 메시지 다이제스트를 계산합니다. 파일을 다시 아카이브할 때마다 파일 시스템은 다이제스트를 재계산하고 저장된 값에 대해 결과를 검증합니다.
- *-G*는 파일에 대한 *generate* 및 *validate* 파일 속성을 설정합니다. 파일 시스템은 메시지 다이제스트를 즉시 계산하고 *hash* 속성에 결과를 저장합니다. 파일을 아카이브할 때마다 파일 시스템은 다이제스트를 재계산하고 저장된 값에 대해 결과를 검증합니다.
- *-u*는 *use* 파일 속성(선택사항)을 설정합니다. 파일을 스테이지할 때마다 파일 시스템은 다이제스트를 재계산하고 *hash* 속성에 저장된 값에 대해 결과를 검증합니다.
- *filename*은 파일의 경로 및 이름입니다.

예제에서는 파일 시스템이 SHA256 알고리즘을 사용하여 아카이브 전에 *data11* 파일의 다이제스트를 계산하도록 합니다. *sls -D -h data10* 명령으로 파일 속성을 확인하면 파일마다 *generate* 파일 속성이 설정되었고 *algorithm* 속성이 *SHA-256*으로 설정되었음을 알 수 있습니다. 파일이 아직 아카이브되지 않았으므로 다이제스트 값이 아직 계산 후 *hash* 속성에 저장되지 않았습니다.

```
root@solaris:~# ssum -a sha256 -g data11
root@solaris:~# sls -D -h data11
data11:
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14975  admin id:      0  inode:    90218.1
  project: user.root(1)
  access:      Jul 16 16:14  modification: Jul 16 16:14
  changed:     Jul 16 16:22  attributes:   Jul 16 16:14
  creation:    Jul 16 16:14  residence:    Jul 16 16:14
  checksum: generate  algorithm: SHA-256
  hash:
root@solaris:~#
```

3. 필요한 경우 평소에 하듯이 파일을 편집합니다.

예제에서는 마지막 아카이브 이후 *data11m*이라는 파일을 수정했습니다. *sls -D -h* 명령이 보여주듯이 어느 쪽도 가장 최근 변경사항을 반영하지 않으므로 *S*(사용되지 않음) 플래그가 두 복사본에 설정되었습니다. Solaris *digest* 명령을 사용하여 수정된 파일에 대한 SHA-256 다이제스트 값을 확인하면 파일의 *hash* 속성이 오래된 다이제스트 값도 저장함을 알 수 있습니다.

```
root@solaris:~# sls -D -h data11m
data11m:
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14983  admin id:      0  inode:    90307.1
```

```
  project: user.root(1)
  copy 1: S----- Jul 17 16:47        dd.1    dk diskarchive f221
  copy 2: S----- Jul 20 11:31       a8d.1    li VOL002
  access:      Jul 20 11:32  modification: Jul 20 11:31
  changed:     Jul 17 16:37  attributes:   Jul 17 16:36
  creation:    Jul 17 16:36  residence:    Jul 17 16:36
  checksum: generate  algorithm: SHA-256
  hash: f03ce01b3828...f7459503007e
root@solaris:~# digest -a sha256 data11m
56c55bb421cc...71ac2ac0b7b0
root@solaris:~#
```

4. 필요한 경우 다시 아카이브하기 전에 수정된 파일의 다이제스트 속성을 변경할 수 있습니다.

   예제에서는 다이제스트 알고리즘을 SHA256에서 SHA1로 변경하고 즉시 효력을 발휘합니다.

```
root@solaris:~# ssum -a sha1 -G data11m
root@solaris:~# sls -D -h data11m
data11m:
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14983  admin id:      0  inode:    90307.1
  project: user.root(1)
  release -a;
  copy 1: S----- Jul 20 13:00        e0.1    dk diskarchive f224
  copy 2: S----- Jul 20 13:05       a93.1    li VOL002
  access:      Jul 20 16:39  modification: Jul 20 16:39
  changed:     Jul 17 16:37  attributes:   Jul 17 16:36
  creation:    Jul 17 16:36  residence:    Jul 20 16:29
  checksum: generate validated  algorithm: SHA-1
  hash: 92003525f0f8...53e29d0718c8
root@solaris:~#
```

5. 그렇지 않으면, 파일 시스템이 수정된 파일을 아카이브하기를 기다렸다가 자동으로 다이제스트 관련 속성을 업데이트합니다.

   수정된 파일을 아카이브할 때 파일 시스템은 다이제스트 값을 재계산하고, 새 값을 *hash* 속성에 저장하고, 이전 파일 버전의 아카이브된 복사본에 *S*(사용되지 않음) 플래그를 설정합니다.

   예제에서는 다이제스트 속성을 변경하지 않고 *data11m* 파일을 편집했습니다. 아카이버는 예정대로 디스크에 *copy 1*을 새로 만들었고 *hash* 속성을 업데이트했습니다. 수정되지 않은 파일의 복사본은 아카이버가 *copy 2*를 만들 때까지 *S*(사용되지 않음) 플래그가 지정된 채 테이프에 남아 있습니다.

```
root@solaris:~# sls -D -h data11m
mdata11:
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14983  admin id:      0  inode:    90307.1
  project: user.root(1)
  copy 1: ------ Jul 17 16:47        dd.1    dk diskarchive f221
  copy 2: S----- Jul 20 11:31       a8d.1    li VOL002
  access:      Jul 20 11:32  modification: Jul 20 11:31
  changed:     Jul 17 16:37  attributes:   Jul 17 16:36
  creation:    Jul 17 16:36  residence:    Jul 17 16:36
  checksum: generate  algorithm: SHA-256
  hash: 56c55bb421cc...71ac2ac0b7b0
```

메시지 다이제스트를 생성하고 디렉토리의 각 파일에 대한 검증을 사용으로 설정

반복적으로 다이제스트를 생성하고 디렉토리의 모든 파일에 대한 검증 속성을 설정하려면 다음과 같이 하십시오.

1. 파일 시스템 호스트에 *root*로 로그인합니다:

   ```
   root@solaris:~#
   ```

2. 명령 프롬프트에서 *ssum -a algorithm -g|G [-u] -r directoryname* 명령을 입력합니다. 설명:
   - *-a algorithm*은 메시지 다이제스트를 생성할 때 파일 시스템이 사용할 암호화 해시 함수를 지정합니다.
   - *-g*는 각 파일마다 *generate* 파일 속성을 설정합니다. 처음 파일을 아카이브할 때 파일 시스템은 파일의 메시지 다이제스트를 계산합니다. 파일을 다시 아카이브할 때마다 파일 시스템은 다이제스트를 재계산하고 저장된 값에 대해 결과를 검증합니다.
   - *-G*는 각 파일마다 *generate* 및 *validate* 파일 속성을 설정합니다. 파일 시스템은 즉시 메시지 다이제스트를 계산하고 *hash* 속성에 결과를 저장합니다. 파일을 아카이브할 때마다 파일 시스템은 다이제스트를 재계산하고 저장된 값에 대해 결과를 검증합니다.
   - *-u*는 *use* 파일 속성(선택사항)을 설정합니다. 파일을 스테이지할 때마다 파일 시스템은 다이제스트를 재계산하고 저장된 값에 대해 결과를 검증합니다.
   - *-r*은 지정된 디렉토리의 모든 파일에 대해 명령을 반복적으로 적용합니다.
   - *directoryname*은 디렉토리의 경로 및 이름입니다.

   첫번째 예제에서는 파일 시스템이 SHA256 알고리즘을 사용하여 아카이브 전에 *datasetA* 디렉토리의 파일에 대한 다이제스트를 계산하도록 합니다. *sls -D -h datasetA* 명령으로 파일 속성을 확인하면 파일마다 *generate* 파일 속성이 설정되었고

algorithm 속성이 *SHA-256*으로 설정되었음을 알 수 있습니다. 파일이 아직 아카이브되지 않았으므로 다이제스트 값이 아직 계산 후 *hash* 속성에 저장되지 않았습니다.

```
root@solaris:~# ssum -a sha256 -g -r datasetA
root@solaris:~# sls -D -h datasetA
datasetA/pdata0:
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14983  admin id:      0  inode:    90232.1
  project: user.root(1)
  access:      Jul 16 16:47  modification: Jul 16 16:47
  changed:     Jul 16 16:47  attributes:   Jul 16 16:47
  creation:    Jul 16 16:47  residence:    Jul 16 16:47
  checksum: generate  algorithm: SHA-256
  hash:
...
datasetA/pdata20:
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14983  admin id:      0  inode:    90234.1
  project: user.root(1)
  access:      Jul 16 16:47  modification: Jul 16 16:47
  changed:     Jul 16 16:47  attributes:   Jul 16 16:47
  creation:    Jul 16 16:47  residence:    Jul 16 16:47
  checksum: generate  algorithm: SHA-256
  hash:
...
root@solaris:~#
```

두번째 예제에서는 파일 시스템이 SHA256 알고리즘을 사용하여 아카이브 전에 *datasetB* 디렉토리의 파일에 대한 다이제스트를 즉시 계산하도록 합니다. *sls -D -h datasetB* 명령으로 파일 속성을 확인하면 파일마다 *generate* 및 *validated* 파일 속성이 설정되었고, *algorithm* 속성이 *SHA-256*으로 설정되었고, 다이제스트 값이 계산 후 *hash* 속성에 저장되었음을 알 수 있습니다.

```
root@solaris:~# ssum -a sha256 -G -r datasetB
root@solaris:~# sls -D -h datasetB
datasetB/qdata0:
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14983  admin id:      0  inode:    90232.1
  project: user.root(1)
  access:      Jul 16 16:47  modification: Jul 16 16:47
  changed:     Jul 16 16:47  attributes:   Jul 16 16:47
  creation:    Jul 16 16:47  residence:    Jul 16 16:47
  checksum: generate validated  algorithm: SHA-256
  hash: 4d2800eb82b3...520341edde95
```

```
...
datasetB/qdata12:
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14983  admin id:      0  inode:    90234.1
  project: user.root(1)
  access:      Jul 16 16:47  modification: Jul 16 16:47
  changed:     Jul 16 16:47  attributes:   Jul 16 16:47
  creation:    Jul 16 16:47  residence:    Jul 16 16:47
  checksum: generate validated  algorithm: SHA-256
  hash: 5b057f1b7b48...88c590d47dec
...
root@solaris:~#
```

스테이지 동안 파일의 메시지 다이제스트 검증

필요한 경우 디스크 캐시에 스테이지하기 전에 파일을 검증할 수 있습니다. 다음과 같이 하십시오.

1. 파일 시스템 호스트에 *root*로 로그인합니다:

   ```
   root@solaris:~#
   ```

2. 명령 프롬프트에서 *ssum -u [-a algorithm [-h digest] -g|G] filename* 명령을 입력합니다. 설명:
   - *-u*는 *use* 파일 속성을 설정하여 스테이지 전에 검증을 지정합니다. *use* 속성이 파일에 설정된 경우, 파일 시스템은 메시지 다이제스트를 생성하고 파일의 *hash* 속성에 저장된 값에 대해 결과를 성공적으로 검증할 때까지 아카이브 매체에서 디스크 캐시로 파일을 복사하지 않습니다.
   - *-a algorithm*, *-h digest*, *-g|G*는 이전에 설정되지 않은 경우 파일에 필수 *algorithm*, *hash*, *generate* 속성을 설정하는 선택적 매개변수입니다.
   - *filename*은 파일의 경로 및 이름입니다.

   예제에서는 *data102* 파일에 대한 검증을 이미 사용으로 설정했습니다. *sls -D -h data102* 명령이 보여주듯이 *generate* 및 *validated* 파일 속성이 설정되었고, *algorithm* 속성이 *SHA-256*으로 설정되었고, 다이제스트 값이 계산 후 *hash* 속성에 저장되었습니다.

   ```
   root@solaris:~# ssum -a sha256 -F data102
   root@solaris:~# sls -D -h data102
   data102:
     mode: -rw-r--r--    links:   1  owner: root        group: root
     length:     14979  admin id:      0  inode:    90264.1
     project: user.root(1)
     access:      Jul 16 17:34  modification: Jul 16 17:34
     changed:     Jul 16 17:34  attributes:   Jul 16 17:34
   ```

```
  creation:    Jul 16 17:34  residence:    Jul 16 17:34
  checksum: generate validated  algorithm: SHA-256
  hash: baae932ce1cf...93166a2e36b5
root@solaris:~#
```

따라서 *use* 속성을 설정하여 파일 시스템이 스테이지 전에 파일을 검증함을 보장할 수 있습니다. *sls -D -h data102* 명령이 보여주듯이 *use* 속성이 지금 설정되어 있습니다.

```
root@solaris:~# ssum -u data102
root@solaris:~# sls -D -h data102
data102:
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14979  admin id:      0  inode:    90264.1
  project: user.root(1)
  access:      Jul 16 17:34  modification: Jul 16 17:34
  changed:     Jul 16 17:34  attributes:   Jul 16 17:34
  creation:    Jul 16 17:34  residence:    Jul 16 17:34
  checksum: generate use validated  algorithm: SHA-256
  hash: baae932ce1cf...93166a2e36b5
root@solaris:~#
```

파일을 아카이브하기 전에 메시지 다이제스트 및 검증 속성 변경

변경 불가능이 아닌 파일이 아직 아카이브되지 않은 경우 아래 절차를 사용하여 메시지 다이제스트 및 검증 속성을 변경할 수 있습니다.

1. 파일 시스템 호스트에 *root*로 로그인합니다:

   ```
   root@solaris:~#
   ```

2. 필요한 경우 다이제스트 알고리즘을 변경합니다. 명령 프롬프트에서 *ssum -a newalgorithm filename* 명령을 입력합니다. 설명:
   - *-a newalgorithm*은 이전에 지정된 다이제스트 알고리즘을 바꾸는 암호화 해시 함수를 지정합니다.
   - *filename*은 파일의 경로 및 이름입니다.

   예제에서는 보존 정책에 충돌 회피도가 높은 SHA256 함수가 필요합니다. 그러나 *sls -D -h* 명령이 보여주듯이 *data319* 파일의 다이제스트 속성을 설정할 때 실수로 SHA1 알고리즘을 지정했습니다. 파일이 아직 아카이브되지 않았으므로 알고리즘을 SHA256으로 성공적으로 변경할 수 있습니다.

   ```
   root@solaris:~# sls -D -h data319
   data319:
     mode: -rw-r--r--    links:   1  owner: root        group: root
     length:     14983  admin id:      0  inode:    90301.1
   ```

```
  project: user.root(1)
  access:      Jul 17 15:27  modification: Jul 17 15:27
  changed:     Jul 17 15:28  attributes:   Jul 17 15:27
  creation:    Jul 17 15:27  residence:    Jul 17 15:27
  checksum: generate  algorithm: SHA-1
  hash:
root@solaris:~# ssum -a sha256 data319
root@solaris:~# sls -D -h data319
data319:
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14983  admin id:      0  inode:    90301.1
  project: user.root(1)
  access:      Jul 17 15:27  modification: Jul 17 15:27
  changed:     Jul 17 15:28  attributes:   Jul 17 15:27
  creation:    Jul 17 15:27  residence:    Jul 17 15:27
  checksum: generate  algorithm: SHA-256
  hash:
root@solaris:~#
```

3. 필요한 경우 다이제스트 속성을 지우고 기본 파일 설정을 복원합니다. 명령 프롬프트에서 *ssum -d filename* 명령을 입력합니다. 설명:
   - *-d*는 파일 다이제스트 속성을 기본값으로 재설정합니다.
   - *filename*은 파일의 경로 및 이름입니다.

   예제에서는 *data44* 파일에 대한 메시지 다이제스트 및 검증을 구성하려는 의도가 아니었습니다. 그러나 *sls -D -h* 명령이 보여주듯이 실수로 이를 수행했습니다. 파일이 아직 아카이브되지 않았으므로 아카이브/스테이지 동안 다이제스트 검증을 제어하는 *generate* 및 *use* 속성을 성공적으로 지울 수 있습니다. *validated*, *algorithm*, *hash* 속성의 데이터는 남아 있지만 파일 시스템의 동작에 영향을 미치지 않습니다.

```
root@solaris:~# sls -D -h data44
data44:
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14983  admin id:      0  inode:    90292.1
  project: user.root(1)
  access:      Jul 17 14:58  modification: Jul 17 14:57
  changed:     Jul 17 14:58  attributes:   Jul 17 14:57
  creation:    Jul 17 14:57  residence:    Jul 17 14:57
  checksum: generate use validated  algorithm: SHA-256
  hash: 3b4b15f8f69c...bae62c7e7568
root@solaris:~# ssum -d data44
root@solaris:~# sls -D -h data44
data44:
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14983  admin id:      0  inode:    90292.1
```

```
  project: user.root(1)
  access:      Jul 17 14:58  modification: Jul 17 14:57
  changed:     Jul 17 14:58  attributes:   Jul 17 14:57
  creation:    Jul 17 14:57  residence:    Jul 17 14:57
  checksum: validated  algorithm: SHA-256
  hash:  3b4b15f8f69c...bae62c7e7568
root@solaris:~#
```

4. 필요한 경우 파일을 아카이브하기 전에 필요한 메시지 다이제스트 및 검증 속성을 재설정합니다. 명령 프롬프트에서 *ssum* 명령을 적절한 옵션 및 파일 이름과 함께 입력합니다.

   예제에서는 *qndat44* 파일에 메시지 다이제스트를 다시 사용으로 설정하고 아카이브 전에 다이제스트를 검증하기로 했습니다. 그러나 스테이지 전에 검증할 필요는 없습니다. 따라서 *generate* 속성을 복원하되, *use* 속성은 복원하지 않습니다.

```
root@solaris:~# ssum -g data44
root@solaris:~# sls -D -h data44
data44:
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14983  admin id:      0  inode:    90292.1
  project: user.root(1)
  access:      Jul 17 14:58  modification: Jul 17 14:57
  changed:     Jul 17 14:58  attributes:   Jul 17 14:57
  creation:    Jul 17 14:57  residence:    Jul 17 14:57
  checksum: generate validated  algorithm: SHA-256
  hash:  3b4b15f8f69c...bae62c7e7568
root@solaris:~#
```

파일을 변경 불가능으로 만들기

보존 요구 사항에서 종종 파일 고정성을 보장하는 메커니즘을 요구합니다. 아카이브는 변경을 방지해야 하고 해당 변경사항이 발생하지 않았음을 입증해야 합니다. 고정성을 제공하기 위해 Oracle HSM 아카이브 파일 시스템은 위에 설명된 메시지 다이제스트 및 다이제스트 관련 파일 속성을, 파일을 변경 불가능으로 만드는 추가 속성과 함께 결합합니다. 일단 파일을 변경 불가능으로 만들면 수퍼 유저 권한을 가진 사람만 상태를 변경할 수 있습니다. 불변성을 엄격한 WORM(Write Once Read Many) 파일 시스템과 결합하면 수퍼 유저조차도 변경할 수 없습니다(세부정보는 “WORM 파일 시스템 이해” 참조).

다음 방법 중 하나로 파일을 변경 불가능으로 만들 수 있습니다.

- 메시지 다이제스트를 제공하고 파일을 변경 불가능으로 만들기
- 메시지 다이제스트를 생성하고 파일을 변경 불가능으로 만들기

메시지 다이제스트를 제공하고 파일을 변경 불가능으로 만들기

아카이브로 입수한 후 파일이 변경되지 않았음을 보장해야 하는 경우 다음과 같이 하십시오.

1. 파일 시스템 호스트에 *root*로 로그인합니다:

   ```
   root@solaris:~#
   ```

2. 명령 프롬프트에서 *ssum -a algorithm [-h digest] -F filename* 명령을 입력합니다. 설명:
   - *-a algorithm*은 제공된 메시지 다이제스트에 대해 파일을 검증할 때 파일 시스템이 사용할 암호화 해시 함수를 식별합니다.
   - *-h digest*는 파일 시스템이 파일 검증에 사용할 메시지 다이제스트를 식별합니다.
   - *-F*는 즉시 검증과 불변성을 지정하고 *fixity*, *generate*, *validated*, *use* 파일 속성을 설정합니다. 파일 시스템은 즉시 메시지 다이제스트를 계산하고 검증합니다. 파일을 스테이지/아카이브할 때 파일 시스템은 메시지 다이제스트를 재계산하고 재검증합니다.
   - *filename*은 파일의 경로 및 이름입니다.

   예제에서는 SHA256 다이제스트를 제공하고 파일 시스템이 다이제스트를 재계산하면 *data20* 파일의 값을 검증하고 파일을 변경 불가능으로 만듭니다. *sls -D -h data10* 명령으로 파일 속성을 확인하면 파일마다 fixity, *generate*, *use*, *validated* 파일 속성이 설정되었고, *algorithm* 속성이 *SHA-256*으로 설정되었고, 다이제스트 값이 계산 후 *hash* 속성에 저장되었음을 알 수 있습니다.

   ```
   root@solaris:~# ssum -h bfaefde932cf...d450892eda63 -a sha256 -F data20
   root@solaris:~# sls -D -h data10
   data20:
     mode: -rw-r--r--    links:   1  owner: root        group: root
     length:     14979  admin id:      0  inode:    90264.1
     project: user.root(1)
     access:      Jul 16 17:34  modification: Jul 16 17:34
     changed:     Jul 16 17:34  attributes:   Jul 16 17:34
     creation:    Jul 16 17:34  residence:    Jul 16 17:34
     checksum: fixity generate use validated  algorithm: SHA-256
     hash: bfaefde932cf...d450892eda63
   root@solaris:~#
   ```

메시지 다이제스트를 생성하고 파일을 변경 불가능으로 만들기

이미 메시지 다이제스트와 연관된 파일을 아카이브할 때 아카이브로 입수한 후 파일이 변경되지 않았음을 보장해야 하는 경우 다음과 같이 하십시오.

1. 파일 시스템 호스트에 *root*로 로그인합니다:

   ```
   root@solaris:~#
   ```

2. 명령 프롬프트에서 *ssum -a algorithm [-h digest] -F filename* 명령을 입력합니다. 설명:
   - *-a algorithm*은 *-h digest* 매개변수에 지정된 다이제스트를 생성하는 데 사용된 암호화 해시 함수를 식별합니다.
   - *-F*는 *fixity*, *generate*, *validated*, *use* 파일 속성을 설정합니다. 파일 시스템은 즉시 메시지 다이제스트를 계산하고 검증합니다. 파일을 스테이지/아카이브할 때 파일 시스템은 메시지 다이제스트를 재계산하고 재검증합니다.
   - *filename*은 파일의 경로 및 이름입니다.

   예제에서는 파일 시스템이 SHA256 다이제스트를 계산하면 *data200* 파일의 값을 검증하고 파일을 변경 불가능으로 만듭니다. *sls -D -h data10* 명령으로 파일 속성을 확인하면 파일마다 *fixity*, *generate*, *validated*, *use* 파일 속성이 설정되었고, *algorithm* 속성이 *SHA-256*으로 설정되었고, 다이제스트 값이 계산 후 *hash* 속성에 저장되었음을 알 수 있습니다.

```
root@solaris:~# ssum -a sha256 -F data200
root@solaris:~# sls -D -h data200
data200:
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14979  admin id:      0  inode:    90264.1
  project: user.root(1)
  access:      Jul 16 17:34  modification: Jul 16 17:34
  changed:     Jul 16 17:34  attributes:   Jul 16 17:34
  creation:    Jul 16 17:34  residence:    Jul 16 17:34
  checksum: fixity generate use validated  algorithm: SHA-256
  hash: efde93cc12cf...d496602e36dd
root@solaris:~#
```

파일 다이제스트 및 고정성 속성 확인

하나 이상의 파일에 대한 메시지 다이제스트 및 고정성 속성을 보려면 Oracle HSM 디렉토리 나열 명령인 *sls*를 사용합니다. 다음과 같이 하십시오.

메시지 다이제스트 및 검증 속성 나열

1. 파일 시스템 호스트에 *root*로 로그인합니다:

```
root@solaris:~#
```

2. 명령 프롬프트에서 *sls -D -h filename* 명령을 입력합니다. 설명:
   - *-D*는 파일 속성의 세부 표시를 지정합니다.
   - *-h*는 화면에 해시(다이제스트) 값을 포함합니다.
   - *filename*은 하나 이상의 파일을 경로 및 이름으로 식별합니다.

예제에서는 화면의 *checksum* 및 *hash* 필드에 나열된 *data02* 파일의 다이제스트 속성을 보여줍니다.

```
root@solaris:~# sls -D -h data02
data02:
  mode: -rw-r--r--    links:   1  owner: root        group: root
  length:     14975  admin id:      0  inode:    90217.1
  project: user.root(1)
  access:      Jul 16 16:14  modification: Jul 16 16:14
  changed:     Jul 16 16:15  attributes:   Jul 16 16:14
  creation:    Jul 16 16:14  residence:    Jul 16 16:14
  checksum: generate use validated  algorithm: SHA-256
  hash: f03ce01b3828...f7459503007e
root@solaris:~#
```

- *hash* 속성은 *f03ce01b3828...f7459503007e* 파일의 메시지 다이제스트를 저장합니다.
- *algorithm* 속성은 *SHA-256* 암호화 해시 함수가 저장된 메시지 다이제스트를 생성했음을 보여줍니다.
- *generate* 속성은 파일 시스템이 독립적으로 메시지 다이제스트를 재계산하고 파일을 아카이브할 때마다 저장된 값에 대해 결과를 검증함을 보여줍니다.
- *use* 속성은 파일 시스템이 독립적으로 메시지 다이제스트를 재계산하고 파일을 스테이지할 때마다 저장된 값에 대해 결과를 검증함을 보여줍니다.
- *validated* 속성은 독립적으로 계산된 메시지 다이제스트가 마지막 검사 시 *hash* 속성에 저장된 값과 일치함을 보여줍니다.
- *fixity* 속성은 파일을 변경 불가능으로 만든 경우 나타납니다.

WORM 파일 시스템 사용

법적 또는 아카이브 고려 사항이 필요한 경우 WORM(Write Once Read Many)을 지원하도록 구성된 Oracle HSM 파일 시스템에 해당 디렉토리 및 파일을 만들 수 있습니다. 이 절에서는 WORM 파일 시스템 이해에 초점을 맞추고 WORM 파일 및 디렉토리를 사용할 때 수행할 특정 작업에 대해 설명합니다.

- 디렉토리에 WORM 사용으로 설정
- 파일에 WORM 보호 활성화
- WORM 파일 찾기 및 나열.

파일 시스템의 WORM 지원 사용에 대한 자세한 내용은 Oracle Hierarchical Storage Manager and StorageTek QFS 설치 및 구성 설명서를 참조하십시오.

WORM 파일 시스템 이해

WORM(Write Once Read Many) 파일 시스템은 사용자가 지정된 보존 기간 동안 파일을 읽기 전용으로 만들어 데이터를 보호할 수 있습니다. WORM 가능 Oracle HSM 파일 시스템은 기본/사용자 정의 가능 파일 보존 기간, 데이터 및 경로 불변성, 하위 디렉토리의 WORM 설정 상속을 지원합니다.

파일 시스템의 구성 방법에 따라 두 가지 Oracle HSM WORM 모드 중 하나를 사용할 수 있습니다.

- 표준 준수 모드(기본값)
- 에뮬레이션 모드

표준 WORM 모드로 마운트된 파일 시스템에서 사용자는 디렉토리에 WORM을 사용으로 설정하고 *chmod 4000 path_name* 명령(여기서 *path_name*은 파일/디렉토리의 경로 및 이름)을 실행하여 파일의 읽기 전용 보존 기간을 시작합니다. 이는 UNIX *setuid*(실행 시 사용자 *ID* 설정) 권한을 설정합니다. *execute* 권한이 있는 파일에 *setuid* 권한을 설정하면 보안 위험이 있으므로 표준 WORM 모드에서는 비실행 파일만 읽기 전용으로 만들 수 있습니다.

WORM 에뮬레이션 모드로 마운트된 파일 시스템에서 사용자는 디렉토리에 WORM을 사용으로 설정하고 *chmod 555 path_name* 명령(여기서 *path_name*은 쓰기 가능 파일/디렉토리의 경로 및 이름)을 실행하여 파일의 읽기 전용 보존 기간을 시작합니다. 에뮬레이션 모드에는 *setuid* 권한이 필요하지 않으므로 실행 파일을 읽기 전용으로 만들고 보존 기간을 지정할 수 있습니다.

표준 및 에뮬레이션 모드는 모두 엄격한 WORM 구현과 덜 제한적인 라이트 구현을 지원합니다. 엄격한 구현과 라이트 구현 모두 파일이나 디렉토리에 보존 기간이 트리거된 이후 데이터나 경로 변경을 허용하지 않습니다. 둘 다 기본 보존 기간을 43,200분(30일)으로 설정합니다. 그러나 라이트 구현은 *root* 사용자의 제한을 완화합니다.

엄격한 구현에서는 누구도 지정된 보존 기간을 단축하거나 보존 기간이 끝나기 전에 파일이나 디렉토리를 삭제할 수 없습니다. 또한 누구도 *sammkfs*를 사용하여 현재 보관된 파일과 디렉토리를 보유한 볼륨을 삭제할 수 없습니다. 따라서 엄격한 구현은 가장 까다로운 법적 규정 준수 및 보존 요구 사항을 충족하기에 매우 적합합니다.

라이트 구현에서는 *root* 사용자가 보존 기간을 단축하고, 파일 및 디렉토리를 삭제하고, *sammkfs* 명령을 사용하여 볼륨을 삭제할 수 있습니다. 이는 가벼운 데이터 손실에 대해 높은 수준의 보호를 제공하며 파일 시스템과 스토리지 리소스를 관리할 때 유연성이 증대됩니다. 그러나 수퍼 유저에게 이 정도의 통제를 허용하는 파일 시스템은 일부 법적 규정 준수 요구 사항을 충족할 수 없습니다.

WORM 파일에 대한 하드 및 소프트 링크를 만들 수 있습니다. WORM 가능 디렉토리에 상주하는 파일에는 하드 링크만 만들 수 있습니다. 하드 링크를 만든 후에 WORM 특성은 원래 파일과 동일합니다. 소프트 링크도 설정할 수 있지만, 소프트 링크는 WORM 기능을 사용할 수 없습니다. WORM 파일에 대한 소프트 링크는 Oracle HSM 파일 시스템의 아무 디렉토리에 만들 수 있습니다.

WORM 파일 시스템 만들기 및 구성에 대한 자세한 내용은 고객 설명서 라이브러리의 *Oracle Hierarchical Storage Manager and StorageTek QFS* 설치 및 구성 설명서를 참조하십시오.

디렉토리에 WORM 사용으로 설정

디렉토리에 WORM을 사용으로 설정할 때 WORM 파일 지원을 추가하되, 디렉토리의 특성은 변경하지 않습니다. 사용자는 계속해서 WORM 가능 디렉토리 내에 파일을 만들고 편집할 수 있으며, WORM 파일을 포함하지 않는 WORM 가능 디렉토리는 삭제할 수 있습니다. 디렉토리에 WORM을 사용으로 설정하려면 다음과 같이 하십시오.

1. 파일 시스템 서버에 로그인합니다.

   ```
   user@solaris:~#
   ```

2. 디렉토리에 이미 WORM이 사용으로 설정되었는지 확인합니다. *sls -Dd directory* 명령을 사용합니다. 여기서 *directory*는 디렉토리의 경로 및 이름입니다. 명령 출력에서 *worm-capable* 속성을 찾습니다.

   대개 디렉토리에는 WORM이 사용으로 설정됩니다. 사용자가 디렉토리에 WORM을 사용으로 설정할 때 모든 현재와 미래 하위 디렉토리가 WORM 기능을 상속하기 때문입니다. 명령에 대한 자세한 내용은 *sls* 매뉴얼 페이지를 참조하십시오. 첫번째 예제에서 대상 디렉토리 */hsm/hsmfs1/records*에 이미 WORM이 사용으로 설정되어 있음을 알 수 있습니다.

   ```
   user@solaris:~# sls -Dd /hsm/hsmfs1/records/2013/
   /hsm/hsmfs1/records/2013:
     mode: drwxr-xr-x    links:   2  owner: root        group: root
     length:      4096  admin id:      0  inode:     1048.1
     project: user.root(1)
     access:      Mar  3 12:15  modification: Mar  3 12:15
     changed:     Mar  3 12:15  attributes:   Mar  3 12:15
     creation:    Mar  3 12:15  residence:    Mar  3 12:15
     worm-capable        retention-period: 0y, 30d, 0h, 0m
   ```

   그러나 두번째 예제에서는 대상 디렉토리 */hsm/hsmfs1/documents*에 WORM이 사용으로 설정되지 않았습니다.

   ```
   user@solaris:~# sls -Dd /hsm/hsmfs1/documents
   /hsm/hsmfs1/documents
     mode: drwxr-xr-x    links:   2  owner: root        group: root
     length:      4096  admin id:      0  inode:     1049.1
     project: user.root(1)
     access:      Mar  3 12:28  modification: Mar  3 12:28
     changed:     Mar  3 12:28  attributes:   Mar  3 12:28
   ```

```
  creation:    Mar  3 12:28  residence:    Mar  3 12:28
```

3. 디렉토리에 WORM이 사용으로 설정되지 않았을 때 파일 시스템이 *worm_capable* 또는 *worm_lite* 마운트 옵션으로 마운트된 경우 Solaris 명령 *chmod 4000 directory-name*(여기서 *directory-name*은 WORM 파일을 보유할 디렉토리의 경로 및 이름)으로 WORM 지원을 사용으로 설정합니다.

   chmod 4000 명령은 디렉토리에 *setuid*(실행 시 사용자 *ID* 설정) 속성을 설정하고 표준 WORM 지원을 사용으로 설정합니다. 예제에서는 */hsm/hsmfs1/documents* 디렉토리에 WORM을 사용으로 설정하고 *sls -Dd*로 결과를 확인합니다. 작업을 성공하고 디렉토리에 WORM이 사용으로 설정됩니다.

```
user@solaris:~# chmod 4000 /hsm/hsmfs1/documents
user@solaris:~# sls -Dd /hsm/hsmfs1/documents
/hsm/hsmfs1/documents
  mode: drwxr-xr-x    links:   2  owner: root        group: root
  length:      4096  admin id:      0  inode:     1049.1
  project: user.root(1)
  access:      Mar  3 12:28  modification: Mar  3 12:28
  changed:     Mar  3 12:28  attributes:   Mar  3 12:28
  creation:    Mar  3 12:28  residence:    Mar  3 12:28
  worm-capable        retention-period: 0y, 30d, 0h, 0m
```

4. 디렉토리에 WORM이 사용으로 설정되지 않았을 때 파일 시스템이 *worm_emul* 또는 *emul_lite* 마운트 옵션으로 마운트된 경우 Solaris 명령 *chmod 555 directory-name*(여기서 *directory-name*은 WORM 파일을 보유할 디렉토리의 경로 및 이름)으로 WORM 지원을 사용으로 설정합니다.

   chmod 555 명령은 디렉토리에 쓰기 권한을 제거하고 WORM 에뮬레이션 지원을 사용으로 설정합니다. 예제에서는 */hsm/hsmfs1/documents* 디렉토리에 WORM을 사용으로 설정하고 *sls -Dd* 명령을 사용하여 결과를 확인합니다. 작업을 성공하고 디렉토리에 WORM이 사용으로 설정됩니다.

```
user@solaris:~# chmod 555 /hsm/hsmfs1/documents
user@solaris:~# sls -Dd /hsm/hsmfs1/documents
/hsm/hsmfs1/documents
  mode: drwxr-xr-x    links:   2  owner: root        group: root
  length:      4096  admin id:      0  inode:     1049.1
  project: user.root(1)
  access:      Mar  3 12:28  modification: Mar  3 12:28
  changed:     Mar  3 12:28  attributes:   Mar  3 12:28
  creation:    Mar  3 12:28  residence:    Mar  3 12:28
  worm-capable        retention-period: 0y, 30d, 0h, 0m
```

파일에 WORM 보호 활성화

WORM 가능 디렉토리의 파일에 WORM 보호를 활성화할 경우, 보존 기간이 만료될 때까지 파일 시스템은 더 이상 파일 데이터나 데이터 경로에 대한 수정을 허용하지 않습니다. 따라서 주의해서 사용해야 합니다. WORM 보호를 활성화하려면 다음과 같이 하십시오.

1. 파일 시스템 서버에 로그인합니다.

   ```
   user@solaris:~#
   ```

2. 기본값 이외의 일정 기간 동안 파일 시스템에 파일을 보관해야 하는 경우 파일에 액세스 시간을 변경하여 필요한 보존 시간을 지정합니다. Solaris 명령 *touch -a -texpiration-date path-name*을 사용합니다. 설명:
   - *expiration-date*는 4자리 연도, 2자리 월, 2자리 일, 2자리 시, 2자리 분, 선택적으로 2자리 초로 구성된 숫자 문자열입니다.
   - *path-name*은 파일의 경로 및 이름입니다.

   Oracle Solaris UNIX 유틸리티 *touch*는 2038년 1월 18일 오후 10시 14분을 넘어 보존 기간을 연장할 수 없습니다. 이 유틸리티는 부호 있는 32비트 숫자를 사용하여 1970년 1월 1일부터 시작하는 시간을 초 단위로 표현합니다. 따라서 이 마감일을 넘어 파일을 보관해야 하는 경우 기본 보존 기간을 사용합니다.

   예제에서는 4년 후 2019년 10월 4일 오전 11시 59분에 파일의 보존 기간이 만료되도록 설정합니다.

   ```
   user@solaris:~# touch -a -t201910141159  /hsm/hsmfs1/plans/master.odt
   ```

3. 파일 시스템이 *worm_capable* 또는 *worm_lite* 마운트 옵션으로 마운트된 경우 Solaris 명령 *chmod 4000 path-name*(여기서 *path-name*은 파일의 경로 및 이름)으로 WORM 보호를 활성화합니다.

   chmod 4000 명령은 지정된 파일에 *setuid*(실행 시 사용자 *ID* 설정) 속성을 설정합니다. 이 속성을 실행 파일에 설정하면 안전하지 않습니다. 따라서 파일 시스템이 *worm_capable* 또는 *worm_lite* 마운트 옵션으로 마운트된 경우 UNIX *execute* 권한이 있는 파일에 WORM 보호를 설정할 수 없습니다.

   예제에서는 *master.odt* 파일에 WORM 보호를 활성화합니다. *sls -D*로 결과를 확인합니다. *retention* 속성이 지금 *active*로 설정되고 *retention-period*가 4년으로 설정되어 있습니다.

   ```
   user@solaris:~# chmod 4000 /hsm/hsmfs1/plans/master.odt
   user@solaris:~# sls -Dd /hsm/hsmfs1/plans/master.odt
   /hsm/hsmfs1/plans/master.odt:
     mode: -r-xr-xr-x    links:   1  owner: root        group: root
     length:       104  admin id:      0  inode:     1051.1
     project: user.root(1)
   ```

```
  access:      Mar  4 2018  modification: Mar  3 13:14
  changed:     Mar  3 13:16  retention-end: Apr  2 14:16 2014
  creation:    Mar  3 13:16  residence:    Mar  3 13:16
  retention:   active        retention-period: 4y, 0d, 0h, 0m
```

4. 파일 시스템이 *worm_emul* 또는 *emul_lite* 마운트 옵션으로 마운트된 경우 Solaris 명령 *chmod 555 path-name*(여기서 *path-name*은 파일의 경로 및 이름)으로 WORM 보호를 활성화합니다.

   chmod 555 명령은 디렉토리에 쓰기 권한을 제거합니다. 따라서 필요한 경우 실행 파일을 WORM으로 보호할 수 있습니다. 예제에서는 *master-plan.odt* 파일에 WORM 보존 기간을 활성화합니다. *sls -D*로 결과를 확인합니다. *retention* 속성이 지금 *active*로 설정되고 *retention-period*가 4년으로 설정되어 있습니다.

```
user@solaris:~# chmod 555 /hsm/hsmfs1/plans/master.odt
user@solaris:~# sls -Dd /hsm/hsmfs1/plans/master.odt
/hsm/hsmfs1/plans/master.odt:
  mode: -r-xr-xr-x    links:   1  owner: root        group: root
  length:       104  admin id:      0  inode:     1051.1
  project: user.root(1)
  access:      Mar  4 2018  modification: Mar  3 13:14
  changed:     Mar  3 13:16  retention-end: Apr  2 14:16 2014
  creation:    Mar  3 13:16  residence:    Mar  3 13:16
  retention:   active        retention-period: 4y, 0d, 0h, 0m
```

WORM 파일 찾기 및 나열

지정된 검색 조건을 충족하는 WORM 파일을 찾고 나열하려면 *sfind* 명령을 사용합니다. 다음과 같이 하십시오.

1. 파일 시스템 서버에 로그인합니다.

```
user@solaris:~#
```

2. WORM으로 보호되고 활발히 보관 중인 파일을 나열하려면 *sfind starting-directory -ractive* 명령을 사용합니다. 여기서 *starting-directory*는 나열 프로세스를 시작하려는 디렉토리의 경로 및 이름입니다.

```
user@solaris:~# sfind /hsm/hsmfs1/ -ractive
/hsm/hsmfs1/documents/2013/master-plan.odt
/hsm/hsmfs1/documents/2013/schedule.ods
/samma1/records/2013/progress/report01.odt
/samma1/records/2013/progress/report02.odt
/samma1/records/2013/progress/report03.odt ...
user@solaris:~#
```

3. 보존 기간이 만료된 WORM 보호 파일을 나열하려면 *sfind starting-directory -rover* 명령을 사용합니다. 여기서 *starting-directory*는 나열 프로세스를 시작하려는 디렉토리의 경로 및 이름입니다.

```
user@solaris:~# sfind /hsm/hsmfs1/ -rover
/samma1/documents/2007/master-plan.odt
/samma1/documents/2007/schedule.ods
user@solaris:~#
```

4. 지정된 날짜 및 시간 후에 보존 기간이 만료될 WORM 보호 파일을 나열하려면 *sfind starting-directory -rafter expiration-date* 명령을 사용합니다. 설명:
   - *starting-directory*는 나열 프로세스를 시작하려는 디렉토리의 경로 및 이름입니다.
   - *expiration-date*는 4자리 연도, 2자리 월, 2자리 일, 2자리 시, 2자리 분, 선택적으로 2자리 초로 구성된 숫자 문자열입니다.

   예제에서는 2015년 1월 1일 자정을 지나 1분 후에 보존 기간이 만료되는 파일을 나열합니다.

```
user@solaris:~# sfind /hsm/hsmfs1/ -rafter 201501010001
/hsm/hsmfs1/documents/2013/master-plan.odt
user@solaris:~#
```

5. 적어도 지정된 시간 동안 파일 시스템에 남아야 하는 WORM 보호 파일을 나열하려면 *sfind starting-directory -rremain time-remaining* 명령을 사용합니다. 설명:
   - *starting-directory*는 검색이 시작되는 디렉토리 트리의 위치입니다.
   - *time-remaining*은 시간 단위와 쌍을 이루는 음이 아닌 정수 문자열로 *y*(연도), *d*(일), *h*(시간), *m*(분)을 나타냅니다.

   예제에서는 */hsm/hsmfs1/* 디렉토리에서 적어도 3년 이상 보관할 모든 파일을 찾습니다.

```
user@solaris:~# sfind /hsm/hsmfs1/ -rremain 3y
/hsm/hsmfs1/documents/2013/master-plan.odt
user@solaris:~#
```

6. 지정된 시간 이상 동안 파일 시스템에 남아야 하는 WORM 보호 파일을 나열하려면 *sfind starting-directory -rlonger time* 명령을 사용합니다. 설명:
   - *starting-directory*는 검색이 시작되는 디렉토리 트리의 위치입니다.
   - *time-remaining*은 시간 단위와 쌍을 이루는 음이 아닌 정수 문자열로 *y*(연도), *d*(일), *h*(시간), *m*(분)을 나타냅니다.

   예제에서는 */hsm/hsmfs1/* 디렉토리에서 3년 90일 이상 보관할 모든 파일을 찾습니다.

```
user@solaris:~# sfind /hsm/hsmfs1/ -rremain 3y90d
/hsm/hsmfs1/documents/2013/master-plan.odt
user@solaris:~#
```

7. 영구적으로 파일 시스템에 남아야 하는 WORM 보호 파일을 나열하려면 *sfind starting-directory -rpermanent* 명령을 사용합니다.

   예제에서는 */hsm/hsmfs1/* 디렉토리 아래의 어떤 파일도 영구적으로 보관되지 않습니다.

```
user@solaris:~# sfind /hsm/hsmfs1/ -rpermanent
user@solaris:~#
```

7

7장. 구성 및 파일 시스템 백업

Oracle Hierarchical Storage Manager and StorageTek QFS Software를 설치하고 구성할 때 복구 지점 파일 및 아카이버 로그의 복사본을 저장할 안전한 위치를 만들었습니다. 또한 복구 지점을 만들고 로그를 백업하고 시스템 구성을 보호하기 위한 자동화된 프로세스를 구성했습니다. 이러한 단계는 파일 시스템에 대한 핵심 보호를 제공합니다. 그러나 가끔씩 예정 외 보호 조치를 취해야 할 수도 있습니다.

- 주요 물리적 기반구조 변경 또는 데이터 센터 설비와 같이 잠재적으로 중단이 예상되는 이벤트에 앞서 Oracle HSM 구성 및 파일 시스템을 백업합니다.
- 소프트웨어, 운영체제, 호스트 플랫폼을 업그레이드하거나 재구성한 후에 Oracle HSM 구성 및 파일 시스템을 백업하여 현재 구성이 보호되도록 합니다.
- Oracle HSM 지원 서비스를 받기 전에 필요한 구성 및 상태 정보를 수집합니다.

이 장에서는 필요시 구성 및 파일 시스템 복구 파일을 수집하고 만들고 저장하기 위한 절차를 설명합니다. 여기에는 다음 세 가지 주요 섹션이 포함됩니다.

- 파일 시스템 백업
- Oracle HSM 구성 백업
- **`samexplorer`**로 구성 및 진단 정보 수집.

이 장에서는 모든 작업에 명령줄 인터페이스를 사용합니다. Oracle HSM Manager 그래픽 사용자 인터페이스를 사용하려면 세부 절차는 온라인 도움말을 참조하십시오.

파일 시스템 백업

이 절은 Oracle HSM 파일 시스템 보호에 대한 간략한 개요로 "복구 지점 및 아카이브 로그 이해"부터 시작합니다. 그리고 다음 작업을 수행하기 위한 절차를 제공합니다.

- 요구 시 복구 지점 만들기
- 아카이버 로그 백업
- **`samexplorer`**로 구성 및 진단 정보 수집
- 수동으로 Oracle HSM 구성 백업.

복구 지점 및 아카이브 로그 이해

파일 시스템을 보호하려면 다음 두 가지를 수행해야 합니다.

- 데이터를 보유한 파일을 보호해야 합니다.
- 데이터를 사용, 구성, 찾기, 액세스, 관리할 수 있도록 파일 시스템 자체를 보호해야 합니다.

Oracle HSM 아카이빙 파일 시스템에서는 파일 데이터가 아카이버에 의해 자동으로 보호되므로 수정된 파일은 테이프와 같은 아카이브 스토리지 매체로 자동 복사됩니다. 그러나 파일만 백업한 후 디스크 장치나 RAID 그룹에 복구할 수 없는 장애가 발생한 경우 데이터가 있어도 쉽게 사용할 방법이 없습니다. 대체 파일 시스템을 만들고 각 파일을 식별하고 새 파일 시스템 내에서 적절한 위치를 결정하고 파일을 입수하고 사용자, 응용 프로그램 및 기타 파일들 간에 유실된 관계를 다시 만들어야 합니다. 이러한 종류의 복구는 아무리 잘해도 힘겹고 지루한 프로세스입니다.

따라서 빠르고 효율적인 복구를 위해 파일 시스템 메타데이터를 적극 보호하고 파일 및 아카이브 복사본을 사용할 수 있어야 합니다. 디렉토리 경로, inode, 액세스 제어, 심볼릭 링크, 포인터를 이동식 매체에 아카이브된 복사본으로 백업해야 합니다.

복구 지점을 예약하고 아카이브 로그를 저장하여 Oracle HSM 파일 시스템 메타데이터를 보호합니다. 복구 지점은 Oracle HSM 파일 시스템에 대한 메타데이터의 적시 백업 복사본을 저장하는 압축 파일입니다. 사용자 파일을 실수로 삭제하는 것부터 전체 파일 시스템의 재해적 손실에 이르는 데이터 손실이 발생할 경우, 파일 또는 파일 시스템이 그대로 남아 있는 마지막 복구 지점을 찾아서 즉시 파일 또는 파일 시스템의 마지막 알려진 정상 상태로 복구할 수 있습니다. 그런 다음 해당 시점에 기록된 메타데이터를 복원하고 메타데이터에 지정된 파일을 아카이브 매체에서 디스크 캐시로 스테이지하거나 사용자 및 응용 프로그램이 액세스할 때 필요에 따라 파일 시스템에서 파일을 스테이지하도록 할 수 있으며 후자의 방법이 선호됩니다.

적시 백업 복사본과 마찬가지로, 복구 지점은 장애가 발생할 당시 파일 시스템의 상태를 완전히 기록하지 못합니다. 불가피하게, 한 복구 지점을 완료한 후 다음 지점을 만들기 전에 적어도 몇몇 파일이 만들어지고 변경됩니다. 이 문제를 최소화하려면 파일 시스템이 사용 중이 아닐 때 자주 복구 지점 만들기를 예약해야 합니다. 그러나 사실상 파일 시스템은 항상 사용되므로 예약을 절충해야 합니다.

이러한 이유로 아카이버 로그 파일의 적시 복사본을 저장해야 합니다. 각 데이터 파일이 아카이브될 때 로그 파일은 아카이브 매체의 볼륨 일련 번호, 아카이브 세트 및 복사본 번호, 매체에서 아카이브(*tar*) 파일의 위치 및 *tar* 파일 내에서 데이터 파일의 경로 및 이름을 기록합니다. 이 정보와 함께 Solaris 또는 Oracle HSM *tar* 유틸리티를 사용하여 복구 지점에서 누락된 파일을 복구할 수 있습니다. 그러나 이 정보는 휘발성 정보입니다. 대부분의 시스템 로그와 마찬가지로, 아카이버 로그는 급속히 늘어나므로 자주 덮어써야 합니다. 복구 지점을 보완하는 정기적 복사본을 만들지 않으면 필요할 때 로그 정보를 얻지 못합니다.

이 절의 나머지 부분에서는 요청 시 복구 지점 및 로그 복사본을 만들기 위한 지침을 제공합니다. 다음 세부 절이 포함됩니다.

- 요구 시 복구 지점 만들기
- 아카이버 로그 백업.

요구 시 복구 지점 만들기

정상 일정을 벗어난 시점에 아카이빙 파일 시스템에서 메타데이터를 캡처해야 할 경우도 있습니다. 예를 들어, 시스템 중단 또는 시설 유지 관리 등이 예상될 때마다 파일 시스템 보호를 위해 이전 및 이후 복구 지점을 만들 수 있습니다.

필요에 따라 예약되지 않은 요구 시 복구 지점 만들기를 시작하려면 다음을 수행합니다.

1. Oracle HSM 서버 호스트에 *root*로 로그인합니다.

   ```
   root@solaris:~#
   ```

2. 복구 지점을 저장할 독립적 위치를 선택합니다.

   예제에서는 처음에 파일 시스템을 구성할 때 복구 지점용으로 만든 디렉토리 아래에 하위 디렉토리 *unscheduled/*를 만듭니다. */zfs1* 파일 시스템은 원격에 위치하며 Oracle HSM 파일 시스템과 공통 구성 요소가 없습니다.

   ```
   root@solaris:~# mkdir /zfs1/samqfs_recovery/unscheduled
   root@solaris:~#
   ```

3. 파일 시스템의 루트 디렉토리로 변경합니다.

   예제에서는 마운트 지점 디렉토리 */samqfs*로 변경합니다.

   ```
   root@solaris:~# cd /samqfs
   root@solaris:~#
   ```

4. 데이터가 이동식 매체로 복사되는 아카이빙 파일 시스템을 백업하는 경우 메타데이터만 백업합니다. *samfsdump -f recovery-point* 명령을 사용합니다. 여기서 *recovery-point*는 완성된 복구 지점 파일의 경로 및 파일 이름입니다.

   자세한 내용은 *samfsdump* 매뉴얼 페이지를 참조하십시오. 예제에서는 예정된 유지 관리 관련 정전에 앞서 *samqfs* 파일 시스템에 대한 예정 외 복구 지점을 만듭니다. */zfs1/samqfs_recovery/unscheduled/* 디렉토리에서 복구 지점 파일 *20150315pre-outage*를 만듭니다. 아래의 두번째 명령은 단일 라인으로 입력됩니다. 줄바꿈은 백슬래시 문자로 이스케이프됩니다.

   ```
   root@solaris:~# cd /samqfs
   root@solaris:~# samfsdump -f /
   /zfs1/samqfs_recovery/unscheduled/20150315pre-outage
   root@solaris:~#
   ```

5. 데이터가 이동식 매체로 복사되지 않는 독립형 파일 시스템을 백업하는 경우 메타데이터와 데이터를 모두 백업합니다. *samfsdump -U -f recovery-point* 명령을 사용합니다. 여기서 *recovery-point*는 완성된 복구 지점 파일의 경로 및 파일 이름입니다.

데이터와 메타데이터가 포함된 복구 지점 파일은 엄청나게 클 수 있습니다. 자세한 내용은 *samfsdump* 매뉴얼 페이지를 참조하십시오. 예제에서는 *samqfs* 파일 시스템에 대한 예정 외 복구 지점을 만듭니다. 원격 디렉토리 */zfs1/samqfs_recovery/unscheduled/*에서 복구 지점 파일 *20150315pre-outage*를 만듭니다. 아래의 두번째 명령은 단일 라인으로 입력됩니다. 줄바꿈은 백슬래시 문자로 이스케이프됩니다.

```
root@solaris:~# cd /samqfs
root@solaris:~# samfsdump -f -U /
/zfs1/samqfs_recovery/unscheduled/20150315pre-outage
root@solaris:~#
```

6. 아카이빙 파일 시스템을 백업하는 경우 아카이버 로그 백업을 수행합니다.
7. 그렇지 않으면 상황에 따라 *samexplorer* 실행 및 수동으로 Oracle HSM 구성 백업을 수행할 수도 있습니다.

아카이버 로그 백업

복구 지점 파일에는 파일 시스템을 복원할 때 필요한 거의 모든 정보가 들어 있지만, 복구 지점을 만든 후에 만들거나 수정된 파일의 메타데이터는 보유하지 않습니다. 아카이버 로그에는 아카이브된 모든 파일과 카트리지에서의 위치가 나열되므로 아카이버 로그를 사용하여 복구 지점을 만든 후에 아카이브된 파일을 복구할 수 있습니다. 따라서 가능하면 예약되지 않은 복구 지점을 만들 때마다 예약되지 않은 아카이버 로그 파일의 복사본도 만드십시오. 다음과 같이 하십시오.

1. Oracle HSM 서버 호스트에 *root*로 로그인합니다.

   ```
   root@solaris:~#
   ```

2. 백업된 아카이버 로그를 저장할 독립적 위치를 선택합니다.

   예제에서는 위에서 만든 예정 외 복구 지점과 동일한 디렉토리에 로그를 저장하기로 결정했습니다. */zfs1* 파일 시스템은 원격에 위치하며 Oracle HSM 파일 시스템과 공통 구성 요소가 없습니다.

   ```
   root@solaris:~# ls /zfs1/samqfs_recovery/unscheduled
   20150315pre-outage
   root@solaris:~#
   ```

3. 현재 아카이버 로그를 선택한 위치로 복사하고 고유한 이름을 부여합니다. *cp /var/adm/samqfs.archive.log path/"date +%y%m%d";* 명령을 사용합니다. 여기서 *path*는 선택한 위치의 경로입니다.

   아래의 명령은 단일 라인으로 입력됩니다. 줄바꿈은 백슬래시 문자로 이스케이프됩니다.

   ```
   root@solaris:~# cp /var/adm/samqfs.archive.log /
   ```

```
/zfs1/samqfs_recovery/unscheduled/20150315pre-outage/"date +%y%m%d".archive.log
root@solaris:~#
```

4. 상황에 따라 *samexplorer* 실행 및 수동으로 Oracle HSM 구성 백업을 수행할 수도 있습니다.

Oracle HSM 구성 백업

Oracle HSM 구성을 변경할 때마다 모든 수정된 구성 파일과 관련 정보를 백업하여 투자를 보호합니다. 다음 작업을 수행합니다.

- 수동으로 Oracle HSM 구성 백업
- **samexplorer** 실행.

수동으로 Oracle HSM 구성 백업

전체 중복성을 위해 소프트웨어, 운영체제, 호스트를 대폭 변경할 때마다 구성 파일의 로컬 복사본을 만듭니다. 다음과 같이 하십시오.

1. 파일 시스템 호스트에 *root*로 로그인합니다.

   ```
   root@solaris:~#
   ```

2. 백업 구성 정보를 보유한 하위 디렉토리에서 Oracle HSM 구성의 수동 백업 복사본을 위한 하위 디렉토리를 만듭니다. *mkdir mount-point/path* 명령을 사용합니다. 여기서 *mount-point*는 선택한 독립적 파일 시스템에 대한 마운트 지점 디렉토리이고 *path*는 선택한 디렉토리의 경로 및 이름입니다.

   예제에서는 아카이빙 파일 시스템 */samqfs*에 대한 복구 지점을 구성하는 중입니다. 따라서 */zfs1/sam_config/samconfig* 디렉토리를 만들었습니다.

   ```
   root@solaris:~# mkdir /zfs1/sam_config/samconfig
   ```

3. Oracle HSM 구성의 수동 백업 복사본을 보유한 하위 디렉토리에서 현재 Oracle HSM 구성을 위한 하위 디렉토리를 만듭니다. *mkdir mount-point/path/subdirectory* 명령을 사용합니다. 여기서 *mount-point*는 선택한 독립적 파일 시스템에 대한 마운트 지점이고 *path/subdirectory*는 선택한 하위 디렉토리의 경로 및 이름입니다.

   예제에서는 초기 구성 중 이 목적으로 만든 디렉토리인 */zfs1/sam_config/samconfig* 아래에 하위 디렉토리를 만듭니다. 날짜를 사용하여 하위 디렉토리에 이름을 지정합니다.

   ```
   root@solaris:~# mkdir /zfs1/sam_config/samconfig/20150315
   ```

4. 구성 파일을 다른 파일 시스템으로 복사합니다.

```
/etc/opt/SUNWsamfs/
     mcf
     archiver.cmd
     defaults.conf
     diskvols.conf
     hosts.family-set-name
     hosts.family-set-name.local
     preview.cmd
     recycler.cmd
     releaser.cmd
     rft.cmd
     samfs.cmd
     stager.cmd
     inquiry.conf
     samremote                  # SAM-Remote server configuration file
     family-set-name           # SAM-Remote client configuration file
     network-attached-library   # Parameters file
     scripts/*                  # Back up all locally modified files
/var/opt/SUNWsamfs/
```

5. 내역기를 통해 유지 관리되는 데이터를 비롯하여 라이브러리 카탈로그 데이터를 모두 백업합니다. 각 카탈로그에 대해 */opt/SUNWsamfs/sbin/dump_cat -V catalog-file* 명령을 사용합니다. 여기서 *catalog-file*은 카탈로그 파일의 경로 및 이름입니다. 출력을 새 위치의 *dump-file*로 재지정합니다.

   예제에서는 *library1*의 카탈로그 데이터를 독립 NFS 마운트 파일 시스템인 *zfs1*에 있는 디렉토리의 *library1cat.dump* 파일에 덤프합니다. 아래의 명령은 단일 라인으로 입력됩니다. 줄바꿈은 백슬래시 문자로 이스케이프됩니다.

   ```
   root@solaris:~# dump_cat -V /var/opt/SUNWsamfs/catalog/library1cat > / /zfs1/sam
   _config/20150315/catalogs/library1cat.dump
   ```

6. Oracle HSM 설치 및 구성 중 수정했던 시스템 구성 파일을 복사합니다. 여기에는 다음이 포함될 수 있습니다.

   ```
   /etc/
        syslog.conf
        system
        vfstab
   /kernel/drv/
        sgen.conf
        samst.conf
        samrd.conf
        sd.conf
   ```

```
    ssd.conf
    st.conf
/usr/kernel/drv/dst.conf
```

7. Oracle HSM 구성의 일부로 만들었던 모든 사용자 정의 셸 스크립트 및 *crontab* 항목을 선택한 하위 디렉토리에 복사합니다.

   예를 들어, 복구 지점 만들기를 관리하기 위해 *crontab* 항목을 만든 경우 지금 복사본을 저장합니다.
8. Oracle HSM 구성의 일부로 만들었던 모든 사용자 정의 셸 스크립트 및 *crontab* 항목을 선택한 하위 디렉토리에 복사합니다.

   예를 들어, 복구 지점 만들기를 관리하기 위해 *crontab* 항목을 만든 경우 지금 복사본을 저장합니다.
9. Oracle HSM, Solaris 및 Solaris Cluster(적용 가능한 경우)를 포함한 현재 설치된 소프트웨어의 개정 레벨을 기록하고 *readme* 파일 정보의 복사본을 선택한 하위 디렉토리에 저장합니다.
10. 선택한 하위 디렉토리에서 다운로드한 Oracle HSM Oracle HSM, Solaris 및 Solaris Cluster 패키지의 복사본을 저장하여 필요할 때 소프트웨어를 빨리 복원할 수 있도록 합니다.
11. 그런 다음 *samexplorer* 실행을 수행합니다.

samexplorer로 구성 및 진단 정보 수집

*samexplorer*는 Oracle HSM 소프트웨어 및 파일 시스템에 대한 종합적 구성 및 상태 정보를 캡처하고 기록하는 진단 도구입니다. Oracle HSM 구성을 변경할 때마다 *samexplorer*를 실행하고 결과 보고서를 구성 파일의 백업 복사본으로 저장해야 합니다. 또한 문제를 해결하거나 Oracle HSM 지원 서비스 직원의 요청이 있을 때 *samexplorer*를 실행해야 합니다. 다음과 같이 하십시오.

samexplorer 실행

1. 파일 시스템 호스트에 *root*로 로그인합니다.
2. 백업 구성 정보를 보유한 디렉토리에서 *samexplorer* 보고서를 위한 하위 디렉토리를 만듭니다. *mkdir mount-point/path* 명령을 사용합니다. 여기서 *mount-point*는 선택한 독립적 파일 시스템에 대한 마운트 지점 디렉토리이고 *path*는 선택한 디렉토리의 경로 및 이름입니다.

   예제에서는 초기 구성 중 이 목적으로 만든 디렉토리인 */zfs1/sam_config/* 아래에 새 하위 디렉토리를 만듭니다. 새 하위 디렉토리의 이름을 *explorer/*로 지정합니다.

   ```
   root@solaris:~# mkdir /zfs1/sam_config/explorer
   ```

3. 선택한 디렉토리에 *samexplorer* 보고서를 만듭니다. *samexplorer path/hostname.YYYYMMDD.hhmmz.tar.gz* 명령을 사용합니다. 여기서 *path*는 선

택한 디렉토리의 경로이고 *hostname*은 Oracle HSM 파일 시스템 호스트의 이름이고 *YYYYMMDD.hhmmz*는 날짜 및 시간 기록입니다.

기본 파일 이름은 */tmp/SAMreport.hostname.YYYYMMDD.hhmmz.tar.gz*입니다. 예제에서는 2015년 3월 15일 오후 4시 59분 산지 표준시로 *samhost1* 호스트에 대한 보고서를 만듭니다. 아래의 명령은 단일 라인으로 입력됩니다. 줄바꿈은 백슬래시 문자로 이스케이프됩니다.

```
root@solaris:~# samexplorer /
/zfs1/sam_config/explorer/samhost1.20150315.1659MST.tar.gz

     Report name:     /zfs1/sam_config/explorer/samhost1.20150315.1659MST.tar.gz
     Lines per file:  1000
     Output format:   tar.gz (default) Use -u for unarchived/uncompressed.

     Please wait.....................................
     Please wait.....................................
     Please wait.....................................

     The following files should now be ftp'ed to your support provider
     as ftp type binary.

     /zfs1/sam_config/explorer/samhost1.20150315.1659MST.tar.gz
```

4. 파일 시스템을 대폭 재구성할 때마다 이 절차를 반복합니다.
5. 여기서 중지합니다. Oracle HSM 구성이 백업되었습니다.

8

8장. 새 스토리지 매체로 마이그레이션

테이프는 장기간 데이터 저장 및 보존을 위한 신뢰성 높은 안정적 매체입니다. 그러나 수명은 유한합니다. 시간이 갈수록 일상적인 기계적 프로세스(마운트, 장력, 읽기/쓰기)로부터 마모 손상이 누적됩니다. 새로운 고성능 드라이브가 제공되면서 이전 드라이브는 지원이 힘들고 호환 매체는 더 비싸고 찾기가 어렵습니다. 그래서 어떤 점에서 새 매체로 아카이브를 이관해야 합니다.

Oracle HSM 계층적 파일 시스템에서 구 매체를 새 매체로 교체하는 것은 복잡한 과정입니다. 테이프 매체는 파일 시스템의 중요한 부분입니다. 파일 시스템 메타데이터는 각 파일 데이터의 여러 복사본 위치(일부는 디스크에, 일부는 테이프에 있음)를 기록합니다. 따라서 테이프를 복사하면서 마이그레이션된 파일 복사본의 새 위치를 반영하도록 파일 시스템 inode가 업데이트되어야 합니다. 동시에, 아카이브 및 스테이지와 같은 정상적 파일 시스템 작동을 방해하지 않고 복사본을 만들도록 매체 및 드라이브를 관리해야 합니다.

Oracle Hierarchical Storage Manager는 두 가지 방법으로 매체 마이그레이션의 복잡성을 관리합니다. 각자 요구 사항에 따라 장점이 있습니다.

Oracle HSM 6.1에 도입된 매체 마이그레이션 기능은 한 라이브러리 드라이브에 마운트된 매체의 전체 볼륨을 다른 곳에 마운트된 매체로 복사합니다. 이 과정에서 파일 시스템 메타데이터를 업데이트하지만 수동으로 로드된 드라이브는 지원되지 않습니다. 이 방식은 시스템 오버헤드와 관리자 작업량을 최소화합니다. 드라이브가 아카이브/스테이지에 필요하지 않을 때 백그라운드에서 볼륨이 복사됩니다. 사용되는 드라이브 수와 하루 중 마이그레이션이 발생할 시간을 지정할 수 있습니다. 또는 드라이브가 유휴 상태일 때마다 Oracle HSM이 볼륨을 마이그레이션할 수 있습니다. 드라이브나 볼륨이 아카이브/스테이지 작업에 필요한 경우 높은 우선순위 작업에 매체 마이그레이션 프로세스를 양보합니다. 올바르게 구성된 StorageTek T1000D 이상의 대상 드라이브가 사용 가능한 경우 T10000 확장 복사 기능과 Oracle HSM *xcopy* 옵션을 사용하여 마이그레이션할 수 있습니다. 일단 요청하면 서버 리소스를 사용하지 않고 드라이브가 자체적으로 복사를 처리합니다. 그렇지 않으면, 여전히 Oracle HSM server-copy 옵션을 사용하여 서버 로드를 최소화할 수 있습니다. 그러면 파일 시스템 서버가 구성 가능한 I/O 버퍼를 통해 드라이브 간에 볼륨을 복사합니다.

정상적인 아카이브 프로세스에서 아카이브 파일을 한번에 하나씩, 데이터를 마이그레이션할 수도 있습니다. 구 매체에서 디스크 캐시로 파일을 스테이지하고 새 매체로 다시 아카이브하도록 시스템을 구성합니다. 이 파일별 접근법은 파일 그룹화/배포 방법을 효율적으로 제어할 수 있습니다. 그러나 더 많은 관리가 필요합니다. 관리자가 디스크 캐시 및 드라이브 리소스를 할당하므로 정상적 파일 시스템 작동에 방해를 최소화해야 하는 경우 신중히 계획해야 합니다.

이 문서의 나머지 부분에서 프로세스를 안내합니다.

- 마이그레이션 준비
- 마이그레이션 방식 선택
- 전체 볼륨 마이그레이션 또는 파일 스테이지 후 대체 매체로 다시 아카이브 방법으로 데이터 마이그레이션

마이그레이션 준비

더 진행하기 전에 다음 단계를 수행합니다.

- 파일 시스템이 백업되어 있는지 확인
- 필요한 매체 제공.

파일 시스템이 백업되어 있는지 확인

매체 마이그레이션을 시작하기 전에, 일반적으로 Oracle HSM 아카이브 데이터를 보호하는 복구 메커니즘이 전환 작업 동안과 그 후에 유효한지 확인합니다. 치명적인 하드웨어 장애와 사용자 오류는 마이그레이션 작업 동안 언제든 발생할 수 있습니다. 따라서 항상 현존하는 *samfsdump* 복구 지점 파일에서 파일 및/또는 전체 파일 시스템을 복구할 수 있는지 확인해야 합니다.

마이그레이션 후 얼마 동안 새로운 대상 볼륨이 아닌 소스 테이프 볼륨을 참조하는 복구 지점 파일에 따라 복구가 달라집니다. 주요 하드웨어 장애로 파일 시스템이 사용 안함으로 설정되고 구 테이프를 사용할 수 없는 경우 복구할 수 없습니다.

따라서 최소한, 새 매체에서 현재 파일 시스템을 복원하기에 충분한 복구 지점을 새로 만들 때까지 구 테이프를 보존할 계획을 세웁니다. 특정 시점으로 파일을 복원해야 하는 경우 구 매체를 더 오래(아니면 무기한으로) 보관할 수 있어야 합니다. 이상적으로, 구 볼륨에 쉽게 액세스할 수 있도록 라이브러리에서 유지 관리해야 합니다.

필요한 매체 제공

대상 라이브러리에 마이그레이션된 파일을 보유하기에 충분한 매체가 있는지 확인합니다. "새 테이프에 레이블 지정 또는 기존 테이프에 레이블 재지정"에 설명된 대로, 모든 볼륨에 올바로 레이블이 지정되었는지 확인합니다. 볼륨에 레이블이 없으면 마이그레이션을 실패합니다.

마이그레이션 방식 선택

선택할 마이그레이션 방식은 아카이브 상태와 사용자 및 응용 프로그램 요구 사항에 따라 달라집니다. 아래 절차를 사용하여 결정을 내리십시오.

사용자 요구에 가장 적합한 마이그레이션 방식 선택

1. 마이그레이션 동안 아카이브를 계속 작동할지 여부를 결정합니다.

아카이브를 중지하고 모든 리소스를 마이그레이션에 독점적으로 투여하면 작업을 간소화하고 빠르게 완성할 수 있습니다. 그러나 아카이브가 활발히 사용 중이면 실용적이지 않습니다.

2. 전체 볼륨이 아닌 아카이브 파일 그룹을 선택적으로 마이그레이션하거나 아카이브 파일 그룹 간에 지정된 관계를 유지해야 하는 경우 스테이지 후 다시 아카이브 방식을 사용합니다. "파일 스테이지 후 대체 매체로 다시 아카이브"로 이동합니다.
3. 간단히 구 볼륨을 새 매체로 복사하거나 마이그레이션이 파일 시스템 작동에 미치는 영향을 최소화해야 하는 경우 볼륨 마이그레이션 방식을 사용합니다.
4. 대상 드라이브로 사용할 수 있는 광 섬유 채널 Oracle StorageTek T10000D 이상의 드라이브가 없는 경우, 또는 소스와 대상 테이프가 공통 블록 크기를 공유하지 않는 경우 server-copy 방식을 사용합니다.

   이 모드에서 Oracle HSM 소프트웨어는 소스 드라이브에서 파일 시스템 서버에 구성 가능한 I/O 버퍼로 유효한 아카이브 파일만 복사합니다. 소스와 대상 블록 크기가 다르면 대상 블록 크기가 더 큰 경우에 한해 소프트웨어가 자동으로 조정합니다. 그러면 소프트웨어는 버퍼에서 대상 드라이브로 테이프 블록을 전송합니다.
5. 광 섬유 채널 StorageTek T10000D 이상의 대상 드라이브가 있고, 드라이브가 모두 현재 펌웨어를 실행 중이고, 소스와 대상 테이프가 동일한 블록 크기를 공유하며, 소스와 대상 드라이브가 동일한 SAN(스토리지 영역 네트워크) 스위치를 동해 연결된 경우 Oracle HSM *xcopy* 옵션을 사용합니다.

   *xcopy*를 지정하면 파일 시스템 서버가 SCSI 복사 요청을 드라이브로 보내고, T10000D 드라이브는 첫번째 유효한 아카이브 파일부터 시작해서 블록 단위로 소스를 대상 테이프로 복사합니다. 어떤 이유로 *xcopy* 작업을 실패하면 마이그레이션 소프트웨어가 자동으로 server-copy 방식으로 전환합니다. *xcopy* 방식은 성능을 극대화하고 서버 오버헤드를 최소화합니다.

   드라이브 및 펌웨어 요구 사항에 대한 자세한 내용은 다운로드 ZIP 파일 또는 파일 시스템 서버 */opt/SUNWsamfs/doc/README.txt*에서 릴리스 노트와 *README.txt*를 참조하십시오.
6. 소스 볼륨에 만료된 파일이 상대적으로 적은 경우 *xcopy* 옵션을 *eod*(데이터 끝) 모드로 사용합니다.

   이 모드에서 T10000 드라이브는 첫번째 유효 파일과 테이프의 EOD(데이터 끝) 표시 사이에 발견된 모든 아카이브 파일을 복사합니다. 이 파일 중 일부가 사용되지 않는 경우 유효 파일과 함께 대상 볼륨으로 복사됩니다.
7. 소스 볼륨에 만료된 파일이 많은 경우 *xcopy* 옵션을 *repack* 모드로 사용합니다.

   이 모드에서 T10000 드라이브는 만료되지 않은 아카이브 파일만 대상 볼륨으로 복사합니다.
8. 파일 스테이지 후 대체 매체로 다시 아카이브로 이동합니다.

전체 볼륨 마이그레이션

server-copy 또는 direct-copy 방식을 선택하고 *migrationd.cmd* 파일을 만들어 마이그레이션을 구성합니다. 다음 작업을 수행합니다.

- **migrationd.cmd** 파일 만들기
- 활성 마이그레이션 작업 확인
- 볼륨 마이그레이션.

migrationd.cmd 파일 만들기

1. Oracle HSM 메타데이터 서버에 *root*로 로그인합니다.

   ```
   root@solaris:~#
   ```

2. 텍스트 편집기에서 */etc/opt/SUNWsamfs/migrationd.cmd* 파일을 엽니다.

   예제에서는 *vi* 편집기에서 새 파일을 열고 처음 주석을 추가합니다.

   ```
   root@solaris:~# vi /etc/opt/SUNWsamfs/migrationd.cmd
   # /etc/opt/SUNWsamfs/migrationd.cmd
   # A configuration file for migrating data from old tape volumes to replacements
   ```

3. 소량의 볼륨만 마이그레이션해야 하는 경우 각 소스 볼륨, 대상 볼륨 및 마이그레이션 방향을 지정합니다. 각 소스 볼륨에 대해 *migrate = from source to destination* 형식의 행을 입력합니다. 설명:
   - *media_type*은 소스를 보유할 매체 종류를 식별하는 2자 코드입니다(세부정보는 부록 A. 장비 유형 용어집 참조).
   - *VSN*은 라이브러리에서 테이프 볼륨을 식별하는 고유한 볼륨 일련 번호입니다.

   예제에서는 구 LTO(*li*) 테이프 *VOL305*에서 새 Oracle StorageTek T10000(*ti*) 테이프 카트리지 *VOL820*으로 데이터를 마이그레이션합니다.

   ```
   root@solaris:~# vi /etc/opt/SUNWsamfs/migrationd.cmd
   # /etc/opt/SUNWsamfs/migrationd.cmd
   # A configuration file for migrating data from old tape volumes to replacements
   # Migrate a single volume.
   migrate = from li VOL305 to ti VOL820
   ```

4. 대량의 볼륨을 마이그레이션해야 하는 경우 소스 및 대상 볼륨에 대한 매체 풀을 정의합니다. *vsnpool = poolname library equipment_number media_type VSNlist* 형식의 행을 입력하여 각 풀을 정의합니다. 설명:
   - *name*은 풀을 고유하게 식별합니다.
   - *equipment_number*는 *mcf* 파일에서 소스 볼륨을 보유할 라이브러리에 지정하는 순서 번호입니다.

- *media_type*은 소스를 보유할 매체 종류를 식별하는 2자 코드입니다(세부정보는 부록 A. 장비 유형 용어집 참조).
- *VSNlist*는 공백으로 구분된 리터럴 VSN 목록이거나, VSN 그룹 및 범위를 식별하는 정규 표현식입니다.

예제에서는 구 LTO4(*li*) 테이프 볼륨에서 새 LTO6(*ti*) 테이프 카트리지로 데이터를 마이그레이션합니다. 마이그레이션할 라이브러리 *20*에서 LTO4 볼륨을 나타내는 소스 풀 *pool1* 행을 추가합니다. 이들은 *VOL000* ~ *VOL299* 범위의 VSN을 가진 볼륨과 2개의 단일 볼륨 *VOL300* 및 *VOL304*를 포함합니다. 그 다음 라이브러리 *30*에서 LTO6 볼륨 범위를 나타내는 대상 풀 *pool2* 행을 추가합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/migrationd.cmd
# /etc/opt/SUNWsamfs/migrationd.cmd
# A configuration file for migrating data from old tape volumes to replacements
# pool1 contains the source volumes
vsnpool = pool1 library 20 li ^VOL[0-2][0-9][0-9] VOL300 VOL304
# pool2 contains the destination volumes
vsnpool = pool2 library 30 li ^VOL50[0-9]
```

5. 소스 및 대상 매체 풀을 정의했으면 마이그레이션 방향을 지정합니다. *migrate* = *from sourcepool to destinationpool* 형식의 행을 입력합니다. 설명:
   - *sourcepool*은 마이그레이션할 데이터를 포함하는 매체 풀입니다.
   - *destinationpool*은 마이그레이션된 데이터를 수신할 매체 풀입니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/migrationd.cmd
...
vsnpool = pool1 library 20 li ^VOL[0-2][0-9][0-9] VOL300 VOL304
# pool2 contains the destination volumes
vsnpool = pool2 library 30 ti ^VOL50[0-9]
# Migrate data from tapes in pool1 to tapes in pool2.
migrate = from pool1 to pool2
```

6. server-copy 마이그레이션 방식을 배타적으로 사용하려면 *xcopy* 기능을 사용 안함으로 설정합니다. *xcopy* = *off* 형식의 행을 입력합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/migrationd.cmd
...
# Disable xcopy and the StorageTek T10000 Extended Copy feature.
xcopy = off
```

7. StorageTek T10000 확장 복사 기능을 배타적으로 사용하고 이 기능을 지원하는 드라이브를 사용할 수 없을 때 데이터를 마이그레이션하지 않으려면 *xcopy* 마이그레이션만 사용으로 설정합니다. *xcopy* = *only* 형식의 행을 입력합니다.

예제에서는 *xcopy*만 사용으로 설정합니다. 소스나 대상 드라이브 중 어느 것도 확장 복사 기능을 지원하지 않으면 마이그레이션 소프트웨어가 자동으로 마이그레이션을 취소합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/migrationd.cmd
...
# Enable xcopy, StorageTek T10000D Extended Copy feature.
# If the source or destination is not xcopy capable, cancel migration.
xcopy = only
```

8. 가능한 경우 StorageTek T10000 확장 복사 기능을 활용하려면 *xcopy* 마이그레이션 방식을 사용으로 설정합니다. *xcopy = on* 형식의 행을 입력합니다.

   예제에서는 마이그레이션 기간 동안 호환 드라이브를 항상 사용할 수 없더라도 *xcopy*를 사용으로 설정합니다. 소스나 대상 드라이브 중 어느 것도 확장 복사 기능을 지원하지 않으면 마이그레이션 소프트웨어가 자동으로 server-copy 모드로 전환합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/migrationd.cmd
...
# Enable xcopy, StorageTek T10000D Extended Copy feature.
# If the source or destination is not xcopy capable, automatically switch
# to the server buffer copy.
xcopy = on
```

9. *xcopy* 방식을 사용하여 소량의 만료된 파일을 포함한 테이프 볼륨을 마이그레이션하려면 *xcopy*가 데이터 끝(*eod*) 모드로 실행하도록 설정합니다. *xcopy_eod = on* 형식의 행을 입력합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/migrationd.cmd
...
xcopy = on
xcopy_eod = on
```

10. *xcopy* 방식을 사용하여 대량의 만료된 파일을 포함한 테이프 볼륨을 마이그레이션하려면 *xcopy*가 repack 모드로 실행하도록 설정합니다. *xcopy_eod = off* 형식의 행을 입력합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/migrationd.cmd
...
xcopy = on
xcopy_eod = off
```

11. *xcopy*가 높은 우선순위의 아카이브/스테이지 작업에 의해 중단되기 전에 복사해야 하는 최소량의 데이터를 지정합니다. *xcopy_minsize* = *amountunits* 형식의 행을 입력합니다. 설명:
    - *amount*는 정수입니다.
    - *units*는 킬로바이트 *k*, 메가바이트 *M*, 기가바이트 *G*, 테라바이트 *T*, 페타바이트 *P*, 엑사바이트 *E*입니다.

    이 값은 효율적인 T10000 드라이브 활용과 다른 작업을 위한 드라이브 가용성 사이의 절충안을 정의합니다. 값이 클수록 더 효율적으로 드라이브에 데이터를 쓸 수 있습니다. 값이 작을수록 아카이브/스테이지를 위한 드라이브 가용성이 증가합니다. 예제에서는 최소 복사 크기를 30GB로 설정합니다.

    ```
    root@solaris:~# vi /etc/opt/SUNWsamfs/migrationd.cmd
    ...
    xcopy_eod = on
    # xcopy can be interrupted after 30GB copied.
    xcopy_minsize = 30G
    ```

12. 매일 마이그레이션 작업 실행이 허용되는 기간을 정의합니다. *runtime* = *window* 형식의 행을 입력합니다. 여기서 *window*는 다음 값 중 하나입니다.
    - *always* - 드라이브와 매체가 아카이브/스테이지에 필요하지 않을 때마다 마이그레이션 데몬이 데이터를 마이그레이션합니다. 드라이브나 매체가 아카이브/스테이지에 필요할 때 마이그레이션 데몬이 이를 사용 중인 경우 마이그레이션 데몬이 양보합니다.
    - *start_time end_time* - 여기서 *start_time* 및 *end_time*은 각각 허용 기간이 시작하고 끝나는 시간으로, 24시간제의 시 분(*HHMM*)으로 표현합니다.

    samcmd, *migstart*, *migidle* 또는 *migstop* 명령을 실행하여 언제든 이 지시어를 대체할 수 있습니다.

    마이그레이션 서비스는 볼륨과 드라이브를 스테이저/아카이버가 요구할 때 이를 포기합니다. 따라서 피크 시간 동안 아카이브/스테이지에 문제가 발생하지 않는 한, 기본값인 *always*를 수락해야 합니다.

    ```
    root@solaris:~# vi /etc/opt/SUNWsamfs/migrationd.cmd
    ...
    xcopy_minsize = 30G
    # Run all of the time. Migration daemon will yield VSNs and drives when
    # resources are wanted by the SAM-QFS archiver and stager.
    run_time = always
    ```

13. 로그 디렉토리를 지정하여 로깅을 사용으로 설정합니다. *logdir* = *path* 형식의 행을 입력합니다. 여기서 *path*는 디렉토리 경로 및 디렉토리 이름입니다.

디렉토리가 정의되면 마이그레이션 데몬은 각 소스 볼륨에서 마이그레이션할 각 아카이브 파일의 대상을 기록합니다. 각 소스 볼륨에는 *media_type.VSN*이라는 고유의 로그 파일이 있습니다. 설명:

- *media_type*은 소스 매체의 종류를 식별하는 2자 코드입니다(세부정보는 부록 A. 장비 유형 용어집 참조).
- *VSN*은 소스 볼륨을 식별하는 고유한 볼륨 일련 번호입니다.

예를 들어, VSN *VOL300*을 가진 소스 볼륨의 로그 파일은 *li.VOL300*이 됩니다.

아카이버 로그와 마찬가지로 이 마이그레이션 로그는 재해 복구 동안 매우 유용할 수 있습니다. 자세한 내용은 "복구 지점 및 아카이브 로그 이해" 및 *Oracle Hierarchical Storage Manager and StorageTek QFS Software* 파일 시스템 복구 설명서를 참조하십시오. 따라서 가능한 경우 항상 로그 디렉토리를 지정하십시오. Oracle HSM 소프트웨어나 하드웨어 장애의 영향을 받지 않을 위치(예: */var/adm/*)를 선택합니다. 예제에서는 */var/adm/hsm_migration_logs* 디렉토리를 지정합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/migrationd.cmd
...
run_time = always
# Log directory for the migration logs.
logdir = /var/adm/hsm_migration_logs
```

14. 마이그레이션 inode 데이터베이스의 홈 디렉토리를 지정합니다. *dbdir = path* 형식의 행을 입력합니다. 여기서 *path*는 절대 디렉토리 경로 이름입니다.

    각 소스 볼륨에 대해 inode 데이터베이스가 만들어지고 마이그레이션 기간 동안 유지됩니다. 소스 볼륨에서 발견된 각 아카이브 복사본에 대해 224바이트의 데이터베이스 레코드 하나가 만들어집니다. 따라서 소스 매체에 맞는 가장 큰 복사본 수를 수용하기에 충분한 디스크 공간이 있는 위치를 선택해야 합니다. 예를 들어, 각 Oracle StorageTek T10000D 볼륨은 최대 8,200,104,892개의 아카이브 복사본을 보유할 수 있습니다. 따라서 주어진 시간에 마이그레이션할 각 T10000D 볼륨에 대해 약 1.67TB의 데이터베이스 공간이 필요합니다. 자세한 내용은 *migration.cmd* (1m) 매뉴얼 페이지를 참조하십시오.

    기본 데이터베이스 위치는 */var/opt/SUNWsamfs/sammig/db*입니다. 예제에서는 기본 디렉토리를 지정합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/migrationd.cmd
...
logdir = /var/adm/hsm_migration_logs
# database home directory
dbdir = /var/opt/SUNWsamfs/sammig/db
```

15. 대상 장치에 대한 마이그레이션 버퍼 크기를 설정합니다. *buffsize* = *media_type blocks* 형식의 행을 입력합니다. 설명:
    - *media_type*은 소스를 보유할 매체 종류를 식별하는 2자 코드입니다(세부정보는 부록 A. 장비 유형 용어집 참조).
    - *size*는 *[2-8192]* 범위의 정수입니다. 여기서 정수 값은 버퍼가 보유할 수 있는 테이프 블록 수를 지정합니다. 기본값은 *64*입니다.

    예제에서는 Oracle StorageTek T10000 테이프 블록의 기본 개수를 보유하기에 충분한 공간을 할당합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/migrationd.cmd
...
# database home directory
dbdir = /var/opt/SUNWsamfs/sammig/db
# allocate buffer space for 64 T10000D tape blocks
bufsize = ti 64
```

16. 라이브러리당 마이그레이션에 사용할 수 있는 최대 드라이브 수를 지정합니다. *max_drives* = *library-list* 형식의 행을 입력합니다. 설명:
    - *library-list*는 공백으로 구분된 라이브러리 항목 목록으로, 각각 *library equipment-number device-count* 형식을 사용합니다.
    - *equipment-number*는 *mcf* 파일에서 라이브러리에 지정된 장비 순서 번호입니다.
    - *device-count*는 지정된 라이브러리에서 사용할 수 있는 드라이브 수입니다. 기본적으로 *device-count*는 라이브러리의 드라이브 수로 설정됩니다.

    마이그레이션 서비스는 볼륨과 드라이브를 스테이저/아카이버가 요구할 때 이를 포기합니다. 따라서 아카이브/스테이지에 문제가 발생하지 않는 한, 기본 설정을 수락하고 사용 가능한 드라이브를 마이그레이션에 사용하도록 허용해야 합니다. 예제에서는 실제로 드라이브 사용량을 제한해야 합니다. 따라서 라이브러리 *20*에서 8개, 라이브러리 *30*에서 6개, 라이브러리 *40*에서 2개 드라이브를 마이그레이션에 할당합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/migrationd.cmd
...
dbdir = /var/opt/SUNWsamfs/sammig/db
# allocate buffer space for 64 T10000D tape blocks
bufsize = ti 64
# For migration, use 8 drives in library 20, 6 in 30, and 2 in 40
max_drives = library 20 8 library 30 6 library 40 2
```

17. 동시에 실행할 수 있는 마이그레이션 관련 복사 작업의 최대 개수를 지정합니다. *max_copy* = *processes* 형식의 행을 입력합니다. 여기서 *processes*는 정수입니다.

    기본값이 최대값이며, *mcf* 파일에 나열된 모든 라이브러리에 구성된 드라이브 수를 2로 나눈 값과 같습니다. 예제에서는 최대 8개의 동시 복사 프로세스를 할당합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/migrationd.cmd
...
bufsize = ti 64
# For migration, use 8 drives in library 20, 6 in 30, and 2 in 40
max_drives = library 20 8 library 30 6 library 40 2
# Up to 8 sam-migcopy process can be run simultaneously.
max_copy = 8
```

18. 동시에 실행할 수 있는 마이그레이션 관련 테이프 스캔 작업의 최대 개수를 지정합니다. *max_scan* = *processes* 형식의 행을 입력합니다. 여기서 *processes*는 정수입니다.

    마이그레이션 소스 VSN에 아카이브 복사본을 식별하기 위해 *sam-migrationd* 데몬은 *mcf*에 구성된 모든 파일 시스템을 스캔하고 디스크 캐시에서 모든 inode를 읽어서 각 inode의 *vsn* 필드를 마이그레이션 소스 볼륨의 VSN(볼륨 일련 번호)과 비교합니다. 이 과정에서 파일 시스템의 메타데이터 작동이 증가하므로 파일 시스템 성능에 악영향을 미칠 수 있습니다.

    수용 가능한 파일 시스템 성능과 마이그레이션 속도가 가장 균형을 이루는 값을 선택하거나, 대부분의 경우 기본값인 *4*를 수락합니다. 가장 빨리 마이그레이션하기 위해 파일 시스템을 중지하려면 *max_scan*을 *0*으로 설정합니다. 그러면 모든 소스 볼륨이 한꺼번에 스캔됩니다. 예제에서는 정상적 파일 시스템 작동에 영향을 미치지 않고 최대 8개 동시 스캔 프로세스가 가능한 것을 경험으로 알 수 있습니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/migrationd.cmd
...
bufsize = ti 64
# For migration, use 8 drives in library 20, 6 in 30, and 2 in 40
max_drives = library 20 8 library 30 6 library 40 2
# Run up to 8 sam-migcopy processes simultaneously.
max_copy = 8
# Scan up to 8 VSNs simultaneously.
max_scan = 8
```

19. 파일을 저장하고 편집기를 닫습니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/migrationd.cmd
...
max_copy = 8
# Scan up to 8 VSNs simultaneously.
max_scan = 8
:wq
root@solaris:~#
```

활성 마이그레이션 작업 확인

이 절의 지침은 셸 명령 프롬프트에서 *samcmd* 명령을 사용하여 명령을 입력하는 방법을 설명합니다. 그러나 모든 명령은 *samu* 인터페이스에서 *:command* 형식(여기서 *command*는 명령의 이름)으로도 입력할 수 있습니다.

1. 현재 Oracle HSM 메타데이터 서버에 *root*로 로그인하지 않은 경우 지금 로그인합니다.

   ```
   root@solaris:~#
   ```

2. 이전 마이그레이션이 현재 활성 또는 미완료 상태가 아닌지 확인합니다. 먼저, 현재 마이그레이션 상태를 확인합니다. *samcmd x* 명령을 사용합니다.

   다른 마이그레이션 복사 작업이 진행 중이면 이 명령은 소스와 대상 볼륨을 매체 유형 및 VSN, 복사 모드, 완료율, 현재 복사 상태에 따라 나열합니다.

   ```
   root@solaris:~# samcmd x
   Migration status   samcmd  version HH:MM:SS month day year
   samcmd on hsm61sol
   Status: Stop: Waiting for :migstart
   source    dest    cmod perc status
   li VOL004 li VOL042 -   60% Copy idled
   ```

   그렇지 않고 다른 마이그레이션 복사 작업이 실행 중이 아니면 이 명령은 아무 작업도 나열하지 않습니다.

   ```
   root@solaris:~# samcmd x
   Migration status     samu      ver  time date
   Source Vsns - wait:  0 fsscan: 0 copy: 0 update ino: 0 log: 0 done:  0
   Status: Idle: Waiting for :migstart
   source  dest  cmod  perc  status
   ```

3. 그 다음, 현재 소스(*S*) 및/또는 대상(*D*) 볼륨의 상태를 확인합니다. *samcmd y* 명령을 사용합니다.

   첫번째 예제에서는 나열된 소스와 대상 볼륨에 대한 *end time* 작업이 *10/16 12:14*입니다. 소스 볼륨의 복사가 완료 상태입니다. 따라서 현재 실행 중인 작업이 없습니다.

   ```
   root@solaris:~# samcmd y
   Migration vsn list   samcmd  version HH:MM:SS month day year
   Status:  Run  Vsns:2 src:1 dest:1 maxcopy:2
   ord m ty vsn     start time  end time    status  Inodes done/tot    bytes
     0 S li VOLa01  10/16 12:12 10/16 12:14 complete    35023/35023   550.00M
     0 D li VOLa80  10/16 12:12 10/16 12:14 avail                     550.00M
   ```

두번째 예제에서는 소스와 대상 볼륨에 대한 *end time* 작업이 *none*입니다. 소스 볼륨이 대상 볼륨으로 아직 복사되는 중입니다. 따라서 마이그레이션 작업이 아직 실행 중입니다.

```
root@solaris:~# samcmd y
Migration vsn list   samcmd  version HH:MM:SS month day year
Status:  Run  Vsns:2 src:1 dest:1 maxcopy:2
ord m ty vsn     start time  end time  status  Inodes done/tot    bytes
  0 S li VOLa02  10/16 12:12 none      copy            0/35023   164.50M
  0 D li VOLa81  10/16 12:12 none      copy                      148.75M
```

4. 마지막으로, 라이브러리 카탈로그에서 볼륨 목록을 확인합니다. *samcmd v* 명령을 사용합니다. 출력에서 다음 플래그를 찾습니다.
   - *R*은 볼륨이 읽기 전용임을 의미합니다. 마이그레이션을 시작할 때 소스 볼륨은 읽기 전용으로 표시됩니다.
   - *S*(소스)는 이 볼륨에서 아직 데이터가 복사 중임을 의미합니다.
   - *D*(대상)는 이 볼륨으로 아직 데이터가 복사 중임을 의미합니다.
   - *m*은 소스 볼륨이 마이그레이션을 마쳤음을 의미합니다.
   - *e*는 소스 볼륨이 오류로 인해 마이그레이션을 실패했음을 의미합니다.

   예제에서는 *VOLa01* 볼륨을 *VOLa80*으로 성공적으로 마이그레이션했습니다. *VOLa02* 볼륨은 *VOLa81*로 아직 마이그레이션되는 중입니다. 마이그레이션에 실패했습니다.

```
root@solaris:~# samcmd v
Robot catalog   samcmd  version HH:MM:SS month day year
Robot VSN catalog by slot       : eq 800
slot          access time count use flags         ty vsn
count 64
   0     2015/06/29 17:00    1  95%  -il---b--Rm-  li VOLa01
   1     2015/07/02 17:43    2  89%  -il-o-b--RS-  li VOLa02
   2     2015/07/02 18:31    2  89%  -il-o-b--Re-  li VOLa03
   ...
   51    2015/10/16 15:18    2   82%  -il-o-b-----  li VOLa80
   52    2015/10/16 15:25    2   84%  -il-o-b---D-  li VOLa81
```

5. 작업이 실행 중이면 완료할 때까지 기다립니다.
6. 그렇지 않고 이미 진행 중인 마이그레이션이 없다고 확신하면 볼륨 마이그레이션을 수행합니다.

볼륨 마이그레이션

이 절의 지침은 셸 명령 프롬프트에서 *samcmd* 명령을 사용하여 명령을 입력하는 방법을 설명합니다. 그러나 모든 명령은 *samu* 인터페이스에서 *:command* 형식(여기서 *command*는 명령의 이름)으로도 입력할 수 있습니다.

1. 현재 Oracle HSM 메타데이터 서버에 *root*로 로그인하지 않은 경우 지금 로그인합니다.

   ```
   root@solaris:~#
   ```

2. 소스 파일 시스템이 마운트되었는지 확인합니다.
3. *migrationd.cmd* 파일을 활성화합니다. *samcmd migconfig* 명령을 사용합니다.

   구성을 성공하면 이 명령은 *Configuring migration* 메시지를 표시하고 세부정보는 로그 파일을 참조하라고 알려줍니다.

   ```
   root@solaris:~# samcmd migconfig
   samcmd: migconfig: Configuring migration (see /var/opt/SUNWsamfs/sammig/logfile)
   root@solaris:~#
   ```

   그렇지 않으면 오류와 함께 명령이 중지됩니다. 구성 명령을 실행하기 전에 마이그레이션 프로세스를 중지하지 못했거나, 테이프 볼륨이 마이그레이션되기를 기다리는 동안 마이그레이션을 중지했습니다.

   ```
   root@solaris:~# samcmd migconfig
   samcmd: migconfig: Can't configure migration, migration status is not stop, or
    migration job is pending
   root@solaris:~#
   ```

4. 마이그레이션 구성을 표시합니다. *samcmd y* 명령을 사용합니다.

   구성을 성공했으면 지정된 볼륨이 나열되고 소스 볼륨의 상태가 *sched_wait*(예약됨, 대기 중)이고, 대상 볼륨의 상태가 *avail*(사용 가능)입니다. 예제에서는 구성을 성공했습니다.

   ```
   root@solaris:~# samcmd y
   Migration vsn list   samcmd  version HH:MM:SS month day year
   samcmd on hsm61sol
   Status: Stop: Waiting for :migstart  Vsns:2 src:1 dest:1 maxcopy:2
   ord m ty vsn    start time  end time    status  Inodes done/tot      bytes
     0 S li VOL001 none        none        sched_wait        0/0            0
     0 D li VOL501 none        none        avail                            0
   ```

5. 구성을 성공했으면 마이그레이션을 시작합니다. *samcmd migstart* 명령을 사용합니다.

```
root@solaris:~# samcmd migstart
samcmd: migstart: State changed to start
root@solaris:~#
```

6. 마이그레이션의 상태를 확인합니다. *samcmd x* 및 *samcmd y* 명령을 사용합니다.

   예제에서는 마이그레이션을 방금 시작 중입니다. *Migration status* 화면이 보여주듯이 작업 상태가 지금 *Run*이고, *1*개 복사가 *s*(서버) 복사 모드(*cmod*)를 사용하여 진행 중이고, 복사가 *0%* 완료되었고, *0*개 inode가 업데이트되었고, 소스 볼륨이 아직 *Loading* 상태입니다.

```
root@solaris:~# samcmd x
Migration status    samcmd  version HH:MM:SS month day year
Source Vsns - wait:  0 fsscan: 0 copy: 1 update ino: 0 log: 0 done:  0
Status: Run
source     dest       cmod perc status
li VOL001 li VOL501 s     0%   Loading li.VOL001
```

   Migration vsn list 화면이 보여주듯이 *2*개 볼륨(*1*개 소스와 *1*개 대상)이 현재 처리되는 중입니다. 두 볼륨의 상태는 지금 *copy*이며 소스가 대상으로 복사되는 중임을 보여줍니다. 이 시점에서 *0* 바이트가 소스에서 대상으로 복사되었고 *35023*개 inode 중 아무것도 업데이트되지 않았습니다.

```
root@solaris:~# samcmd y
Migration vsn list   samcmd  version HH:MM:SS month day year
Status: Run  Vsns:2 src:1 dest:1 maxcopy:2
ord m ty vsn     start time  end time  status  Inodes done/tot    bytes
  0 S li VOL001  10/16 12:12 none      copy            0/35023        0
  0 D li VOL501  10/16 12:12 none      copy                           0
```

7. 다시 *samcmd x* 및 *samcmd y* 명령을 사용하여 마이그레이션 상태를 정기적으로 재확인합니다.

   예제에서는 *Migration status* 화면이 보여주듯이 복사가 지금 *23%* 완료되었고 560개(*0x00000230*) 테이프 블록을 소스에서 읽어왔습니다.

```
root@solaris:~# samcmd x
Migration status    samcmd  version HH:MM:SS month day year
Source Vsns - wait:  0 fsscan: 0 copy: 1 update  ino: 0 log: 0 done:  0
Status:  Run
source    dest       cmod perc status
li VOL001 li VOL501 s      24% 0x00000230 blocks read
```

Migration vsn list 화면이 보여주듯이 *164.50* 메가바이트를 소스 볼륨에서 읽어왔고 *148.75* 메가바이트를 대상 볼륨에 썼습니다.

```
root@solaris:~# samcmd y
Migration vsn list   samcmd  version HH:MM:SS month day year
Status:  Run  Vsns:2 src:1 dest:1 maxcopy:2
ord m ty vsn     start time  end time  status  Inodes done/tot    bytes
  0 S li VOL001  10/16 12:12 none      copy            0/35023   164.50M
  0 D li VOL501  10/16 12:12 none      copy                      148.75M
```

8. 마이그레이션이 완료되면 종료 상태를 확인합니다. *samcmd x* 및 *samcmd y* 명령을 사용하고 마이그레이션 로그 파일을 조사합니다.

   예제에서는 소스와 대상 볼륨이 더 이상 *Migration status* 화면에 나열되지 않고 지금 *1*개 복사가 완료되었음을 보여줍니다. 마이그레이션 상태는 아직 *Run*이고 *migidle* 또는 *migstop* 명령을 입력할 때까지 계속 유지됩니다.

```
root@solaris:~# samcmd x
Migration status    samcmd  version HH:MM:SS month day year
Source Vsns - wait:  0 fsscan: 0 copy: 0 update ino: 0 log: 0 done:  1
Status: Run
source   dest   cmod perc status
```

   Migration vsn list 화면이 보여주듯이 *550.00* 메가바이트를 소스 볼륨에서 읽어왔고 *550.50* 메가바이트를 대상 볼륨에 썼습니다. 모든 *35023*개 inode가 마이그레이션된 아카이브 복사본의 새 위치를 반영하도록 업데이트되었습니다.

```
root@solaris:~# samcmd y
Migration vsn list   samcmd  version HH:MM:SS month day year
Status:  Run  Vsns:2 src:1 dest:1 maxcopy:2
ord m ty vsn     start time  end   time  status  Inodes done/tot    bytes
  0 S li VOL001  10/16 12:12 10/16 12:14 complete    35023/35023   550.00M
  0 D li VOL012  10/16 12:12 10/16 12:14 avail                      550.00M
```

   마이그레이션 데몬의 로그 파일은 각 마이그레이션 단계를 나열하고 요약 결론을 보여줍니다. 예제에서는 Solaris *tail* 명령을 사용하여 가장 최근 항목을 봅니다.

```
root@solaris:~# tail /var/opt/SUNWsamfs/sammig/logfile
date time Info: Schedule: Create VsnList file.
date time Info: Schedule: VsnList file created, source: 1, destination: 1.
date time Info: Schedule: Migration status changed to Start.
date time Info: 'li.VOL001' Filesystem scan: Started
```

```
date time Info: 'li.VOL001' Filesystem scan: Completed, total copy bytes: 517.2M,
 inodes: 35023, multi vsn copy: 0, removable-media file: 0, obsolete copy: 0
date time Info: 'li.VOL001' Copy: Started, pid: 2459 destination 'li.VOL012'
date time Info: 'li.VOL001' Copy: Mode - server copy
date time Info: 'li.VOL001' Copy: Server copy started from position 0x4.
date time Info: 'li.VOL001' Copy: Tar header check started from position 0x4.
date time Info: 'li.VOL001' Copy: Tar header check succeeded, 5 inodes checked, 0
 tar header error found.
date time Info: 'li.VOL001' Copy: Completed, pid: 2459, exit status: 12, signal: 0
date time Info: 'li.VOL001' Update inode: Started, source position: 0
date time Info: 'li.VOL001' Update inode: Completed.
date time Info: 'li.VOL001' Log: Started, source position: 0
date time Info: 'li.VOL001' Log: Completed.
date time Summ: 'li.VOL001'
date time Summ: 'li.VOL001' =============== Summary ===============
date time Summ: 'li.VOL001' Status:    Complete
date time Summ: 'li.VOL001' Copy mode: Server copy
date time Summ: 'li.VOL001' Start at:  date time
date time Summ: 'li.VOL001' End at:    date time
date time Summ: 'li.VOL001' Bytes:     550.00M
date time Summ: 'li.VOL001' Archive copies:                   35023
date time Summ: 'li.VOL001' Read error copies:                    0
date time Summ: 'li.VOL001' Multi vsn copies:                     0
date time Summ: 'li.VOL001' Removable-Media file:                 0
date time Summ: 'li.VOL001' ---Dest---   ---Bytes---   ---Copies---
date time Summ: 'li.VOL001' li VOL501        550.00M          35023
root@solaris:~#
```

9. 마지막으로, 볼륨 마이그레이션 로그를 안전한 위치로 복사해야 합니다.

   이 로그는 각 소스 볼륨에서 마이그레이션된 각 아카이브 파일 복사본의 대상 볼륨과 시작 위치를 기록합니다. 이 정보는 파일이나 파일 시스템을 복구해야 하는 경우 매우 중요합니다. 따라서 Oracle은 복구 지점 및 아카이버 로그 파일과 함께 이 파일의 백업 복사본을 보관할 것을 강력히 권장합니다. 7장. 구성 및 파일 시스템 백업 및 Oracle Hierarchical Storage Manager and StorageTek QFS 설치 및 구성 설명서의 해당 장을 참조하십시오.

   마이그레이션 데몬은 *migrationd.cmd* 파일에 지정된 디렉토리에 마이그레이션 로그 파일을 만듭니다. 마이그레이션된 각 볼륨에 대해 *media_type*.*VSN*이라는 파일을 만듭니다. 설명:

   - *media_type*은 소스 매체의 종류를 식별하는 2자 코드입니다(세부정보는 부록 A. 장비 유형 용어집 참조).
   - *VSN*은 소스 볼륨을 식별하는 고유한 볼륨 일련 번호입니다.

예제에서는 지정된 로그 디렉토리 */var/adm/hsm_migration_logs/*에서 NFS 마운트 원격 파일 시스템의 파일 시스템 복구 리소스를 보관할 디렉토리로 볼륨 로그를 복사합니다.

```
root@solaris:~# ls /var/adm/hsm_migration_logs/
li.VOL001  li.VOL002  li.VOL003  li.VOL004  li.VOL005  li.VOL006 ... ti.801 ...
root@solaris:~# cp /var/adm/hsm_migration_logs/*.* /zfs/recover/hsmfs1/2015mig/
```

10. 모든 파일을 다시 아카이브했으면 요구 사항에 따라 테이프를 처리합니다("마이그레이션 이후 구 매체 처리 " 참조).
11. 여기서 중지합니다. 마이그레이션이 완료되었습니다.

파일 스테이지 후 대체 매체로 다시 아카이브

스테이지 후 다시 아카이브 방식을 사용하여 구 매체에서 새 매체로 아카이브 파일을 마이그레이션하려면 정상적 파일 시스템 작동을 방해하지 않고 마이그레이션할 파일을 식별하고, 디스크 캐시로 스테이지하고, 새 매체에 기록해야 합니다. 이 장에서는 프로세스의 다음 단계를 다룹니다.

- 사용 가능한 리소스 추정
- 새 매체를 사용하도록 아카이빙 프로세스 구성
- 대체 매체로 데이터 마이그레이션.

사용 가능한 리소스 추정

스테이지 후 다시 아카이브 프로세스의 세부정보는 주로 사용 가능한 디스크 스토리지 양과 사용 가능한 이동식 매체 드라이브 수에 따라 달라집니다. 데이터 마이그레이션 동안 Oracle HSM 스테이저는 이전 이동식 볼륨을 이전 매체 형식을 읽을 수 있는 드라이브로 로드하고 아카이브된 파일을 디스크 캐시로 복원합니다. 그런 다음 Oracle HSM 아카이버는 새 매체 형식을 쓸 수 있는 드라이브를 사용하여 파일을 새 이동식 볼륨으로 다시 아카이브합니다. 따라서 이상적으로, 주어진 테이프 볼륨의 모든 파일을 한꺼번에 디스크로 스테이지한 후 즉시 새 매체로 아카이브하게 됩니다.

이를 위해 마이그레이션 기간 동안 엄청난 리소스를 투입해야 합니다.

- 전체 테이프 용량에 해당하는 디스크 공간
- 구 테이프 형식을 읽는 드라이브의 배타적 사용
- 새 형식에 쓰는 드라이브의 배타적 사용

마이그레이션이 완료될 때까지 파일 시스템을 중지할 수 있으면 위 사항은 문제가 되지 않습니다. 그러나 진행 중인 파일 시스템 및 아카이브 작업을 지나치게 방해하지 않고 운용 설정의 데이터를 마이그레이션하려면 약간 생각할 필요가 있습니다. 디스크 공간이나 테이프 드라이브에 공급이 부족한 경우 마이그레이션에 상당량 할애할 수 있는 리소스를 식별하고 마이그레이션 프로세스를 조정해야 합니다. 따라서 다음과 같이 하십시오.

1. 정상적 파일 시스템 작동을 방해하지 않고 마이그레이션에 사용할 수 있는 디스크 캐시 양을 추정합니다.
2. 마이그레이션에 투입할 여유가 있는 테이프 드라이브 수를 추정합니다.

   제한된 수의 테이프 드라이브만 사용할 수 있는 경우 스테이징 및 아카이빙 프로세스에 스로틀링을 적용하여 마이그레이션 프로세스가 정상적 작동을 방해하지 않도록 합니다.
3. 위의 예상치를 바탕으로 스테이징 및 아카이빙 매개변수를 결정합니다. 주어진 시간에 사용 가능한 디스크 공간이 보유할 마이그레이션 파일의 최대 개수와, 파일이 캐시에서 새 매체로 이동할 수 있는 최대 속도를 결정합니다.
4. 리소스를 추정했으면 마이그레이션 사후 구 매체 처리 계획을 수행합니다.

새 매체를 사용하도록 아카이빙 프로세스 구성

새 매체를 *archiver.cmd* 파일에 추가하고 항상 새 매체를 사용하여 복사본을 만들도록 아카이브 복사 지시어를 수정합니다.

1. 텍스트 편집기에서 */etc/opt/SUNWsamfs/archiver.cmd* 파일을 엽니다.

   아카이빙 정책은 복사본 2부를 명시합니다. 둘 다 교체하려는 매체 유형에 쓰여집니다. 예제에서는 *vi* 편집기에서 파일을 엽니다. DLT 카트리지(유형 *lt*)를 교체하려고 합니다.

   ```
   root@solaris: vi /etc/opt/SUNWsamfs/archiver.cmd
   # ============================================
   # /etc/opt/SUNWsamfs/archiver.cmd
   # --------------------------------------------
   ...
   # --------------------------------------------
   # VSN Directives
   vsns
   allfiles.1 lt .*
   allfiles.2 lt .*
   endvsns
   ```

2. 복사본 *2*의 지시어에서 지정된 매체 유형을 새 매체의 식별자로 변경하고 파일을 저장하고 텍스트 편집기를 닫습니다.

   예제에서는 구 DLT 테이프에서 새 LTO 카트리지로 데이터를 마이그레이션하려고 합니다. 따라서 복사본 *2*에서 구 매체 유형 *lt*(DLT)를 *li*(LTO)로 변경합니다.

   ```
   root@solaris: vi /etc/opt/SUNWsamfs/archiver.cmd
   # ============================================
   # /etc/opt/SUNWsamfs/archiver.cmd
   # --------------------------------------------
   ...
   # --------------------------------------------
   ```

```
# VSN Directives
vsns
allfiles.1 lt .*
allfiles.2 li .*
endvsns
:wq
root@solaris:~#
```

3. *archiver.cmd* 파일에 구문 오류가 있는지 확인합니다. *archiver -lv* 명령을 실행하고 오류가 발견되지 않을 때까지 오류를 수정합니다.

   archiver -lv 명령은 한 행씩 파일을 출력합니다. 만일 오류가 발생하면 오류가 발생한 지점에서 실행이 중지됩니다.

```
root@solaris:~# archiver -lv
Reading '/etc/opt/SUNWsamfs/archiver.cmd'.
1: # ============================================
2: # /etc/opt/SUNWsamfs/archiver.cmd
3: # --------------------------------------------
4: # Global Directives
5: logfile = /var/opt/SUNWsamfs/archiver.log
6: # --------------------------------------------
7: # File System Directives:
8: fs = samqfsms
9: all .
10: 1 5m ...
root@solaris:~#
```

4. 수정된 *archiver.cmd* 파일에 오류가 없으면 *samd config* 명령을 사용하여 이를 현재 구성에 로드합니다.

```
root@solaris:~# samd config
Configuring SAM-FS
root@solaris:~#
```

5. 그런 다음 카트리지에서 카트리지로 데이터 마이그레이션을 수행합니다.

대체 매체로 데이터 마이그레이션

데이터 마이그레이션을 위한 스테이징 및 아카이빙 방식은 GNU *find* 명령의 Oracle HSM 확장인 *sfind*를 사용하는 것입니다. *sfind* 명령을 사용하여 지정된 테이프 볼륨에서 파일을 찾고 모든 발견된 파일에 대해 *stage* 및 *rearchive* 명령을 실행할 수 있습니다.

sfind, *stage*, *rearchive* 명령에 익숙하지 않으면 지금 각각의 매뉴얼 페이지를 검토해야 합니다. 그런 다음 마이그레이션할 데이터를 보유한 각 테이프 카트리지에 대해 다음과 같이 하십시오

한 카트리지에서 다른 카트리지로 데이터 마이그레이션

1. 파일 시스템 호스트에 *root*로 로그인합니다.

   ```
   root@solaris:~#
   ```

2. 마이그레이션하려는 파일이 저장된 파일 시스템의 마운트 지점 디렉토리로 이동합니다.

   예제에서는 */hsm/hsmfs1*에 마운트된 *hsmfs1* 파일 시스템에 저장되어 있는 파일의 아카이브된 복사본을 마이그레이션합니다.

   ```
   root@solaris:~# cd /hsm/hsmfs1
   root@solaris:~#
   ```

3. 테이프 볼륨을 선택합니다.

   매체 유형에서 매체 유형으로 데이터를 마이그레이션할 때 한번에 하나씩 볼륨을 작업합니다. 아래 예제에서는 볼륨 일련 번호 *VOL008*을 사용합니다.

4. 먼저, 선택한 볼륨에서 성공적으로 스테이지할 수 없는 손상된 파일이 있는지 검색합니다. Oracle HSM 명령 *sfind . -vsn volume-serial-number -damaged*를 사용합니다. 여기서 *volume-serial-number*는 라이브러리에서 볼륨을 고유하게 식별하는 영숫자 문자열입니다.

   예제에서는 현재 작업 디렉토리(*.*)에서 검색을 시작합니다. *-vsn* 매개변수는 현재 테이프 *VOL008*에서 발견된 파일로 검색을 제한합니다. *-damaged* 플래그는 성공적으로 스테이지할 수 없는 파일로 검색을 제한합니다.

   ```
   root@solaris:~# sfind . -vsn VOL008 -damaged
   ```

5. 손상된 파일에 대한 *sfind* 검색 작업으로 결과가 반환되면 파일을 수정합니다. *undamage -m media-type -vsn volume-serial-number file* 명령을 사용합니다. 설명:

   - *media-type*은 부록 A. 장비 유형 용어집에 나열된 2자리 매체 유형 코드 중 하나입니다.
   - *volume-serial-number*는 볼륨을 고유하게 식별하는 영숫자 문자열입니다.
   - *file*은 손상된 파일의 경로 및 이름입니다.

   일부 경우에는 일시적 I/O 오류로 인해 복사본이 손상된 것으로 표시될 수 있습니다. Oracle HSM *undamage* 명령은 이러한 조건을 해결합니다. 예제에서는 아카이브 파일 복사본 */hsm/hsmfs1/data0008/20131025DAT*가 손상된 것으로 보고되었습니다. 따라서 이 파일의 손상을 해결하고 손상된 파일을 다시 검색해봅니다.

   ```
   root@solaris:~# sfind . -vsn VOL008 -damaged
   /hsm/hsmfs1/data0008/20131025DAT
   root@solaris:~# undamage -m lt -vsn VOL008 /hsm/hsmfs1/data0008/20131025DAT
   ```

```
root@solaris:~# sfind . -vsn VOL008 -damaged
```

6. *sfind* 명령에서 이 파일이 다시 손상된 것으로 나열되면 복사본을 사용할 수 없는 것입니다. 아카이브에 다른 손상되지 않은 파일 복사본이 있는지 확인합니다. 사용 가능한 복사본을 나열하려면 *sls -D file* 명령을 사용합니다. 여기서 *file*은 파일의 경로 및 이름입니다. 발견된 복사본의 상태를 확인하려면 *sfind file -vsn volume-serial-number* 명령을 사용합니다.

   예제에서는 *undamage* 명령이 복사본을 수정할 수 없습니다. 따라서 *sls*를 사용하여 */hsm/hsmfs1/data0008/20131025DAT* 파일의 모든 복사본을 나열합니다.

```
root@solaris:~# undamage -m lt -vsn VOL008 /hsm/hsmfs1/data0008/20131025DAT
root@solaris:~# sfind . -vsn VOL008 -damaged
/hsm/hsmfs1/data0008/20131025DAT
root@solaris:~# sls -D /hsm/hsmfs1/data0008/20131025DAT
20131025DAT:
mode: -rw-r--r--  links:   1  owner: root      group: other
            length:    319279  admin id:      7  inode: 1407.5
            project: system(0)
            offline;  archdone;  stage -n;
            copy 1: ---- May 21 07:12     1e4b1.1    lt VOL008
            copy 2: ---- May 21 10:29     109c6.1    lt VOL022
...
```

   테이프 볼륨 *VOL022*는 파일의 두번째 복사본을 보유합니다. 따라서 *sfind*를 사용하여 두번째 복사본을 확인합니다.

```
root@solaris:~# sfind /hsm/hsmfs1/data0008/20131025DAT -vsn VOL022 -damaged
```

7. 복사본을 사용할 수 없으며 파일의 손상되지 않은 복사본이 하나 존재하는 경우 파일을 다시 아카이브합니다. 그런 다음 아카이브가 두 개의 올바른 복사본을 보유하면 손상된 복사본을 아카이브 해제합니다.

   예제에서는 *VOL008* 볼륨의 */hsm/hsmfs1/data0008/20131025DAT* 파일 복사본 1을 사용할 수 없지만 *sfind* 명령에서 복사본 2에 대한 손상을 찾지 못했습니다. 따라서 *VOL008* 볼륨에서 손상된 복사본을 아카이브 해제하기 전에 *-c* 옵션과 함께 *archive* 명령을 실행하여 유효한 복사본 1을 만듭니다.

```
root@solaris:~# sfind /hsm/hsmfs1/data0008/20131025DAT -vsn VOL022 -damaged
root@solaris:~# archive -c 1 /hsm/hsmfs1/data0008/20131025DAT
...
root@solaris:~# unarchive -m lt -vsn VOL008 /hsm/hsmfs1/data0008/20131025DAT
```

8. 사용 가능한 복사본이 존재하지 않으면 파일이 캐시에 있는지 확인합니다. *sfind . -vsn volume-serial-number -online* 명령을 사용합니다.

   예제에서는 *VOL008* 볼륨의 복사본 1과 *VOL022* 볼륨의 복사본 2 모두 손상되고 사용할 수 없습니다. 따라서 디스크 캐시에서 파일을 온라인으로 사용할 수 있는지 확인합니다.

```
root@solaris:~# undamage -m lt -vsn VOL008 /hsm/hsmfs1/data0008/20131025DAT
root@solaris:~# sfind . -vsn VOL008 -damaged
/hsm/hsmfs1/data0008/20131025DAT
root@solaris:~# undamage -m lt -vsn VOL022 /hsm/hsmfs1/data0008/20131025DAT
root@solaris:~# sfind /hsm/hsmfs1/data0008/20131025DAT -vsn VOL022 -damaged
/hsm/hsmfs1/data0008/20131025DAT
root@solaris:~# sfind /hsm/hsmfs1/data0008/20131025DAT -online
```

9. 사용 가능한 복사본이 존재하지 않지만 파일이 캐시에 있는 경우 파일을 아카이브합니다. 그런 다음 아카이브가 두 개의 올바른 복사본을 보유하면 손상된 복사본을 아카이브 해제합니다.

   예제에서는 *VOL008* 볼륨의 복사본 1과 *VOL022* 볼륨의 복사본 2가 모두 사용 불가능하므로 *VOL008* 볼륨에서 손상된 복사본을 아카이브 해제하기 전에 *archive* 명령을 실행하여 두 개의 유효한 복사본을 만듭니다.

```
root@solaris:~# undamage -m lt -vsn VOL008 /hsm/hsmfs1/data0008/20131025DAT
root@solaris:~# sfind . -vsn VOL008 -damaged
/hsm/hsmfs1/data0008/20131025DAT
root@solaris:~# undamage -m lt -vsn VOL022 /hsm/hsmfs1/data0008/20131025DAT
root@solaris:~# sfind /hsm/hsmfs1/data0008/20131025DAT -vsn VOL022 -damaged
/hsm/hsmfs1/data0008/20131025DAT
root@solaris:~# sfind /hsm/hsmfs1/data0008/20131025DAT -online
/hsm/hsmfs1/data0008/20131025DAT
root@solaris:~# archive /hsm/hsmfs1/data0008/20131025DAT
root@solaris:~# unarchive -m lt -vsn VOL008 /hsm/hsmfs1/data0008/20131025DAT
```

10. 사용 가능한 복사본이 존재하지 않고, 파일이 디스크 캐시에 없으면 데이터가 손실된 것입니다. 중요한 데이터인 경우 데이터 복구 회사 전문가에게 도움을 요청하십시오. 그렇지 않으면 손상된 복사본을 아카이브 해제합니다.

    예제에서는 *VOL008* 볼륨의 복사본 1과 *VOL022* 볼륨의 복사본 2 모두 사용할 수 없습니다. *sfind* 명령은 디스크 캐시에서 파일을 찾을 수 없습니다. 데이터가 중요하지 않습니다. 따라서 *VOL008* 볼륨에서 손상된 복사본을 아카이브 해제합니다.

```
root@solaris:~# undamage -m lt -vsn VOL008 /hsm/hsmfs1/data0008/20131025DAT
root@solaris:~# sfind . -vsn VOL008 -damaged
/hsm/hsmfs1/data0008/20131025DAT
root@solaris:~# undamage -m lt -vsn VOL022 /hsm/hsmfs1/data0008/20131025DAT
```

```
root@solaris:~# sfind /hsm/hsmfs1/data0008/20131025DAT -vsn VOL022 -damaged
/hsm/hsmfs1/data0008/20131025DAT
root@solaris:~# sfind /hsm/hsmfs1/data0008/20131025DAT -online
root@solaris:~# archive /hsm/hsmfs1/data0008/20131025DAT
root@solaris:~# unarchive -m lt -vsn VOL008 /hsm/hsmfs1/data0008/20131025DAT
```

11. 손상된 파일에 대한 *sfind* 검색이 결과를 반환하지 않으면 현재 테이프의 파일을 디스크 캐시에 스테이지합니다. *sfind . -vsn volume-serial-number -offline -exec stage {}/;* 명령을 사용합니다.

    -vsn 매개변수는 현재 테이프에서 발견된 파일로 검색을 제한합니다. 항상 한 번에 테이프 하나씩 마이그레이션합니다.

    -offline 매개변수는 이미 캐시에 없는 파일로 *sfind* 출력을 한층 더 제한하므로 데이터를 덮어쓰지 않습니다.

    -exec stage {}/; 인수는 *sfind*가 반환하는 각 경로 및 파일 이름을 사용하며 이를 Oracle HSM *stage* 명령에 대한 인수로 사용합니다. 그런 다음 *stage* 명령은 지정된 파일을 디스크 캐시로 복원합니다. 적합한 파일이 모두 스테이지될 때까지 프로세스가 반복됩니다.

    예제에서는 *sfind -vsn VOL008 -damaged* 명령이 출력을 반환하지 않습니다. 따라서 *sfind*를 사용하여 *VOL008*에서 검색되고 캐시에 없는 파일을 모두 스테이지합니다.

    ```
    root@solaris:~# sfind . -vsn VOL008 -damaged
    root@solaris:~# sfind . -vsn VOL008 -offline -exec stage {}/;
    ```

12. 테이프에서 파일이 스테이지되었으면 선택적으로 다시 아카이브합니다. *sfind . -vsn volume-serial-number -online -exec rearch -r -m media-type {}/;* 명령을 사용합니다. 여기서 *media-type*은 마이그레이션 중인 매체 유형입니다.

    -vsn 매개변수는 현재 테이프에서 발견된 파일로 검색을 제한합니다. 항상 한 번에 테이프 하나씩 마이그레이션합니다.

    -online 매개변수는 캐시에 있는 파일로 *sfind* 출력을 한층 더 제한하므로 데이터를 덮어쓰지 않습니다.

    -exec rearch -r -m media-type {}/; 인수는 *sfind*가 반환하는 각 경로 및 파일 이름을 사용하며 이를 Oracle HSM *rearch -r -m media-type* 명령에 대한 인수로 사용합니다. *-r* 인수는 하위 디렉토리를 통해 프로세스를 반복적으로 실행합니다. *-m* 인수는 소스 매체에 있는 파일만 다시 아카이브합니다.

    예제에서 *-vsn* 매개변수 값은 *VOL008*이고 *-m* 매개변수 값은 DLT 매체인 *lt*를 지정합니다.

    ```
    root@solaris:~# sfind . -vsn VOL008 -online -exec rearch -r -m lt {}/;
    ```

13. *sfind* 검색이 더 이상 파일을 발견하지 않을 때까지 앞의 단계를 반복합니다.
14. 모든 파일이 다시 아카이브되었을 때 계획한 대로 테이프를 처리합니다("마이그레이션 이후 구 매체 처리 " 참조).
15. 모든 구 매체에서 새 매체로 데이터가 마이그레이션될 때까지 이 절차를 반복합니다.

마이그레이션 이후 구 매체 처리

마이그레이션을 완료한 후 구 매체가 반드시 모든 가치를 잃는 것은 아닙니다. 따라서 이들의 처리 방법을 신중히 고려하십시오.

- 최소한, 새로운 대체 매체만 사용하여 파일 시스템의 모든 파일을 복구하기에 충분한 새 복구 지점 파일을 축적할 때까지 구 매체를 보관하십시오.
- 스토리지 공간이 허용한다면 구 매체를 무기한으로 보관하십시오. 호환 드라이브를 사용할 수 있는 한, 구 매체는 유용한 백업 및 복구 리소스가 될 수 있습니다.
- 라이브러리 공간이 부족한 경우 구 매체를 내보내고 오프사이트 스토리지에 보관하십시오.
- 구 매체를 재사용할 수 있고 보유 데이터가 더 이상 필요하지 않으면 구 볼륨에 레이블을 재지정하십시오. 예를 들어, 이전 Oracle StorageTek T10000C 드라이브의 매체에 레이블을 재지정하여 새로운 T10000D 드라이브와 함께 사용할 수 있습니다.
- 그렇지 않고 구 볼륨이나 매체의 데이터에 더 이상 남은 가치가 없으면 라이브러리에서 볼륨을 내보내고 적절히 처리하십시오.

부록 A

부록 A. 장비 유형 용어집

마스터 구성 파일(*mcf*)의 *Equipment Type* 필드 값은 Oracle Hierarchical Storage Manager and StorageTek QFS Software에서 장치 및 장치 구성을 식별합니다. 장비 유형은 2자리 문자 코드로 지정됩니다. 이 용어집에서는 샘플로 작업하거나 기존 *mcf*(자세한 내용은 *mcf(4)* 매뉴얼 페이지 참조)를 해석할 때 빠른 참조를 위한 코드를 나열합니다.

편의상 코드는 세 섹션으로 나눈 다음 알파벳순으로 나열되어 있습니다.

권장 장비 및 매체 유형

이 절에서는 일반적으로 필요한 일반 장비 코드(*rb*, *tp* 및 *od*) 및 네트워크 연결 라이브러리 인터페이스와 Oracle HSM 내역기를 식별하기 위한 코드 등 모든 장비 코드를 설명합니다.

일반 장비 코드인 *rb*, *tp* 및 *od*는 모든 SCSI 연결 라이브러리, 테이프 드라이브 및 옵티컬 디스크 장치에 대한 선호 장비 유형 코드입니다. 일반 장비 유형을 지정할 경우 Oracle HSM에서 SCSI 공급업체 코드를 기준으로 올바른 유형을 자동으로 설정할 수 있습니다.

gXXX
여기서 *XXX*는 *[0-127]* 범위의 정수이며, *ma* 디스크 캐시 패밀리 세트의 일부인 스트라이프된 디스크 장치 그룹입니다.

hy
Oracle HSM 내역기이며, 매체 카탈로그를 유지 관리하지만 연관된 하드웨어가 없는 선택적 가상 라이브러리입니다. 내보낸 매체를 추적하는 데 사용됩니다.

ma
하나 이상의 전용 *mm* 디스크 장치에서 파일 시스템 메타데이터를 유지 관리하는 고성능 QFS 파일 시스템입니다. 파일 데이터는 별도의 *md*, *mr* 또는 *gXXX* 데이터 장치에 상주합니다.

md
ma 파일 시스템에 대한 파일 데이터 또는 *ms* 파일 시스템에 대한 데이터 및 메타데이터를 저장하는 디스크 장치입니다. *md* 장치는 파일 데이터를 작은 4KB DAU(디스크 할당 단위) 및 큰 16KB, 32KB 또는 64KB DAU로 저장합니다. 기본 DAU는 64KB입니다.

mm
고성능 *ma* 파일 시스템에 대한 파일 시스템 메타데이터를 저장하는 디스크 장치입니다.

mr
ma 파일 시스템에 대한 파일 데이터를 저장하는 디스크 장치입니다. *mr* 장치는 파일 데이터를 완전히 조정 가능한 8-65528KB 범위에서 8KB의 배수인 큰 DAU(디스크 할당 단위)로 저장합니다. 기본 DAU는 64KB입니다.

ms
파일 데이터를 저장하는 동일한 장치에서 파일 시스템 메타데이터를 유지 관리하는 Oracle HSM 파일 시스템입니다.

od
모든 SCSI 연결 옵티컬 디스크입니다. Oracle HSM는 SCSI 공급업체 코드를 사용하여 적절한 장비 유형을 자동으로 설정합니다.

rb
모든 SCSI 연결 테이프 라이브러리입니다. Oracle HSM는 SCSI 공급업체 코드를 사용하여 적절한 장비 유형을 자동으로 설정합니다.

rd
SAM-Remote 의사 장치입니다. 마스터 구성 파일(*mcf*)에서 해당 *Equipment Identifier* 필드에 의사 장치의 경로를 포함해야 합니다(예: */dev/samrd/rd2*). 해당 *Family Set* 필드에 SAM-Remote 서버의 호스트 이름을 포함해야 합니다.

sc
SAM-Remote 클라이언트 시스템입니다. 마스터 구성 파일(*mcf*)에서 해당 *Equipment Identifier* 필드에 SAM-Remote 클라이언트에 대한 클라이언트 구성 파일의 경로를 포함해야 합니다. 해당 *Family Set* 필드에 서버의 패밀리 세트 이름을 포함해야 합니다. *Additional Parameters* 필드에 클라이언트의 라이브러리 카탈로그 파일에 대한 전체 경로를 포함해야 합니다.

sk
네트워크 연결 테이프 라이브러리에 대한 Oracle StorageTek ACSLS 인터페이스입니다. 마스터 구성 파일(*mcf*)에서 해당 *Equipment Identifier* 필드에 ACSLS 인터페이스에 대한 매개변수 파일의 경로를 포함해야 합니다. 자세한 내용은 *stk(7)* 매뉴얼 페이지를 참조하십시오.

ss
SAM-Remote 서버입니다. 마스터 구성 파일(*mcf*)에서 해당 *Equipment Identifier* 필드에 SAM-Remote 서버 구성 파일의 경로를 포함해야 합니다. 해당 *Family Set* 필드에 서버의 패밀리 세트 이름을 포함해야 하며, 이는 클라이언트에서 *mcf*의 *Family Set* 필드에 사용된 이름과 일치해야 합니다.

tp
모든 SCSI 연결 테이프 드라이브입니다. Oracle HSM는 SCSI 공급업체 코드를 사용하여 적절한 장비 유형을 자동으로 설정합니다. 하지만 *li* 및 *ti*와 같이 더 구체적인 장비 코드를 사용할 경우 일관성 있게 사용해야 합니다. 예를 들어, *mcf* 파일에서 *li*(LTO) 테이프 장비를 지정할 경우 *archiver.cmd* 파일에서 *tp* 장비와 동일한 장비를 참조할 수 없습니다.

기타 장비 및 매체 유형

이 절에 나열된 장비 유형도 지원됩니다.

대부분의 경우 일반 장비 유형 *rb*, *tp* 및 *od*를 사용하여 SCSI 연결 라이브러리, 테이프 드라이브 및 광 디스크 장치를 식별하는 것이 좋습니다. 일반 장비 유형은 Oracle HSM가 SCSI 공급업체 ID를 사용하여 하드웨어를 동적으로 식별하도록 합니다. 아래의 유형 코드는 한 매체 유형에서 다른 매체 유형으로 마이그레이션하는 경우 필수적이며 경우에 따라 관리 목적에 유용할 수 있습니다. 하지만 이러한 코드를 마스터 구성 파일(*mcf*)에서 사용하면 일정 시점에서 실제 하드웨어와 더 이상 일치하지 않는 정적 장비 구성이 하드 코딩됩니다.

ac
Sun 1800, 3500 또는 L11000 테이프 라이브러리입니다.

at
Sony AIT-4 또는 AIT-5 테이프 드라이브입니다.

cy
Cygnet 옵티컬 디스크 라이브러리입니다.

d3
StorageTek D3 테이프 드라이브입니다.

dm
Sony DMF 라이브러리입니다.

ds
DocuStore 또는 Plasmon 옵티컬 디스크 라이브러리입니다.

dt
DAT 4mm 테이프 드라이브입니다.

e8
Exabyte X80 라이브러리입니다.

fd
Fujitsu M8100 128트랙 테이프 드라이브입니다.

h4
HP SL48 또는 SL24 라이브러리입니다.

hc
Hewlett Packard L9-/L20-/L60 시리즈 라이브러리입니다.

i7
IBM 3570 테이프 드라이브입니다.

ic
IBM 3570 매체 교환기입니다.

il
IBM 3584 테이프 라이브러리입니다.

li
LTO-3 이상의 테이프 드라이브입니다.

lt
DLT(Digital Linear Tape), Super DLT 또는 DLT-S4 테이프 드라이브입니다.

me
Metrum 라이브러리입니다.

mf
IBM 다기능 옵티컬 드라이브입니다.

mo
5.25인치 지우기 가능 옵티컬 드라이브입니다.

o2
12인치 WORM 드라이브입니다.

ov
Overland Data Inc. Neo Series 테이프 라이브러리입니다.

pd
Plasmon D-Series DVD-RAM 라이브러리입니다.

q8
Qualstar 42xx, 62xx 또는 82xx 라이브러리입니다.

s3
StorageTek SL3000 라이브러리입니다.

s9
Oracle StorageTek 97xx 시리즈 라이브러리입니다.

se
StorageTek 9490 테이프 드라이브입니다.

sf
StorageTek T9940 테이프 드라이브입니다.

sg
StorageTek 9840C 이상 테이프 드라이브입니다.

sl
Spectra Logic 또는 Qualstar 테이프 라이브러리입니다.

st
StorageTek 3480 테이프 드라이브입니다.

ti
StorageTek T10000(Titanium) 테이프 드라이브입니다.

vt
Metrum VHS(RSP-2150) 테이프 드라이브입니다.

wo
5.25인치 옵티컬 WORM 드라이브입니다.

xt
Exabyte (850x) 8mm 테이프 드라이브입니다.

부록 B

부록 B. 매체 상태 플래그

매체 플래그는 다음 의미를 제공합니다.

- *A*는 슬롯에 감사가 필요함을 의미합니다.
- *C*는 슬롯에 청소 카트리지가 있음을 의미합니다.
- *D*는 볼륨이 매체 마이그레이션 대상임을 의미합니다.
- *E*는 볼륨이 불량하거나 청소 매체가 만료되었음을 의미합니다.
- *L*은 LTFS(Linear Tape File System) 볼륨임을 의미합니다.
- *N*은 볼륨이 Oracle HSM 형식이 아님을 의미합니다.
- *R*은 볼륨이 읽기 전용임을 의미합니다(소프트웨어 플래그).
- *S*는 볼륨이 매체 마이그레이션 소스임을 의미합니다.
- *U*는 볼륨을 사용할 수 없음을 의미합니다.
- *W*는 볼륨이 물리적으로 쓰기 금지됨을 의미합니다.
- *X*는 내보내기 슬롯임을 의미합니다.
- *b*는 볼륨에 바코드가 있음을 의미합니다.
- *c*는 볼륨 재활용이 예약되어 있음을 의미합니다.
- *f*는 볼륨이 가득 차거나 손상된 것을 아카이버가 발견했음을 의미합니다.
- *d*는 볼륨에 중복된 VSN(볼륨 일련 번호)이 있음을 의미합니다.
- *l*은 볼륨에 레이블이 붙어 있음을 의미합니다.
- *o*는 슬롯이 점유되었음을 의미합니다.
- *p*는 높은 우선 순위 볼륨임을 의미합니다.
- *-*는 디스플레이에 사용될 때 해당 플래그가 설정되지 않았음을 의미합니다.

예를 들어, *samcmd v*는 각 보관된 볼륨에 대해 매체 플래그를 포함한 카탈로그 정보를 나열합니다.

```
root@solaris:~# samcmd v 800
Robot catalog samcmd     6.1    16:45:25 Feb 14 2016
samcmd on samqfshost                        count 32
Robot VSN catalog by slot       : eq 800
slot          access time count use  flags         ty vsn
   0     2014/03/14 11:23  875   0%  -il-o-b-----  li VOL001
   1     2014/03/13 17:54  866   0%  -il-o-b-----  li VOL002
   2     2014/03/14 11:26    3   0%  -il-o-b-----  li VOL003
   3     2014/03/14 10:33    3   0%  -il-o-b-----  li VOL004
   4     2014/03/14 11:34    5   0%  -il-o-b-----  li VOL005
   5     2014/03/14 11:32    2   0%  -ilEo-b----f  li VOL006 MEDIA ERROR
   6     2014/03/13 18:07    2   0%  -il-o-b-----  li VOL007
   7     2014/03/13 18:07    1   0%  -il-o-b-----  li VOL008
```

```
10     2014/03/13 18:10    2   0%  -il-o-b-----  li VOL011
```

부록 C

부록 C. 공유 파일 시스템의 마운트 옵션

Oracle Hierarchical Storage Manager and StorageTek QFS Software 공유 파일 시스템은 여러 마운트 옵션을 사용해서 마운트할 수 있습니다. 이 장에서는 해당 역할의 컨텍스트 내에서 이러한 옵션 중 일부를 설명합니다.

공유 파일 시스템 마운트 옵션

mount 명령을 사용하거나 */etc/vfstab* 파일에 마운트 옵션을 입력하거나 *samfs.cmd* 파일에 마운트 옵션을 입력하여 대부분의 마운트 옵션을 지정할 수 있습니다. 예를 들어, 다음 */etc/vfstab* 파일에는 공유 파일 시스템에 대한 마운트 옵션이 포함되어 있습니다.

```
sharefs - /sfs samfs - no shared,mh_write
```

samu 운영자 유틸리티를 사용하여 일부 마운트 옵션을 동적으로 변경할 수 있습니다. 이러한 옵션에 대한 자세한 내용은 *Oracle Hierarchical Storage Manager and StorageTek QFS samu* 명령 참조를 참조하십시오.

이러한 마운트 옵션에 대한 자세한 내용은 *mount_samfs* 매뉴얼 페이지를 참조하십시오.

bg: 백그라운드에서 마운트

bg 마운트 옵션은 첫번째 마운트 작업이 실패할 경우 후속 마운트 시도가 백그라운드에서 실행되도록 지정합니다. 기본적으로 *bg*는 적용되지 않으며 마운트 시도가 포그라운드에서 계속 수행됩니다.

retry: 파일 시스템 마운트 재시도

retry 마운트 옵션은 시스템에서 파일 시스템 마운트를 시도해야 하는 횟수를 지정합니다. 기본값은 10000입니다.

shared: Oracle HSM 공유 파일 시스템 선언

shared 마운트 옵션은 파일 시스템을 Oracle HSM 공유 파일 시스템으로 선언합니다. 파일 시스템을 Oracle HSM 공유 파일 시스템으로 마운트하려면 이 옵션을 */etc/vfstab* 파일에 지정해야 합니다. 이 옵션이 *samfs.cmd* 파일 또는 *mount* 명령에 있을 경우 오류가 발생하지는 않지만 파일 시스템이 공유 파일 시스템으로 마운트되지 않습니다.

minallocsz 및 maxallocsz: 할당 크기 조정

mount 명령에 대한 *minallocsz* 및 *maxallocsz* 옵션은 공간 크기를 킬로바이트 단위로 지정합니다. 이러한 옵션은 최소 블록 할당 크기를 설정합니다. 첨부 임대가 허용될 경우 파

일이 커지면 메타데이터 서버는 블록을 할당합니다. *-o minallocsz=n*을 사용하여 이 할당의 초기 크기를 지정합니다. 메타데이터 서버는 *-o maxallocsz=n* 설정을 초과하지 않는 한도 내에서 응용 프로그램의 액세스 패턴에 따라 블록 할당의 크기를 늘릴 수 있습니다.

이러한 *mount* 옵션을 *mount* 명령줄, */etc/vfstab* 파일 또는 *samfs.cmd* 파일에서 지정할 수 있습니다.

rdlease, wrlease 및 aplease: Oracle HSM 공유 파일 시스템에서 임대 사용

기본적으로 호스트가 파일을 공유할 때는 Oracle HSM 메타데이터 서버가 서버 자체 및 해당 클라이언트에 대해 I/O *leases*를 실행하여 파일 시스템 일관성을 유지 관리합니다. 임대는 지정된 기간 동안 특정 파일에서 특정 작업을 수행하기 위한 권한을 공유 호스트에 부여합니다. 읽기 임대는 호스트가 파일 데이터를 읽을 수 있게 하고, 쓰기 임대는 호스트가 기존 파일 데이터를 겹쳐쓸 수 있게 합니다. 첨부 임대는 호스트가 파일 끝에 추가 데이터를 기록할 수 있게 합니다. 메타데이터 서버는 필요에 따라 임대를 갱신할 수 있습니다.

Oracle HSM 공유 파일 시스템에 대한 읽기 및 쓰기는 데이터에 대해 POSIX와 비슷한 동작을 제공해야 합니다. 하지만 메타데이터의 경우 액세스 시간 변경은 다른 호스트에 즉시 표시되지 않을 수 있습니다. 파일에 대한 변경사항은 쓰기 임대 종료 시에 디스크로 이동됩니다. 읽기 임대를 얻은 경우, 시스템은 새로 기록된 데이터를 볼 수 있도록 오래된 캐시 페이지를 무효화합니다.

다음 마운트 옵션은 임대 기간을 설정합니다.

- *-o rdlease=number-seconds*는 읽기 임대의 최대 시간(초)을 지정합니다.
- *-o wrlease=number-seconds*는 쓰기 임대의 최대 시간(초)을 지정합니다.
- *-o aplease=number-seconds*는 첨부 임대의 최대 시간을 초 단위로 지정합니다.

세 가지 경우 모두에서 *number-seconds*는 [*15-600*] 범위의 정수입니다. 각 임대의 기본 시간은 *30*초입니다. 임대가 적용된 경우 파일을 자를 수 없습니다. 이러한 임대를 설정하는 방법에 대한 자세한 내용은 *mount_samfs* 매뉴얼 페이지를 참조하십시오.

현재 메타데이터 서버가 다운되었기 때문에 메타데이터 서버를 변경할 경우 모든 임대가 만료된 후에만 대체 메타데이터 서버가 제어를 수행할 수 있으므로 전환 시간에 임대 시간을 추가해야 합니다.

임대 시간을 짧게 설정하면 만료 후 임대를 갱신해야 하기 때문에 클라이언트 호스트와 메타데이터 서버 간 트래픽이 증가합니다.

mh_write: 여러 호스트 읽기 및 쓰기를 사용으로 설정

mh_write 옵션은 동일한 파일에 대한 여러 호스트의 쓰기 권한을 제어합니다. *mh_write*를 메타데이터 서버 호스트에서 마운트 옵션으로 지정할 경우 Oracle HSM 공유 파일 시스템은 동일한 파일에 대한 여러 호스트의 동시 읽기 및 쓰기를 사용으로 설정합니다. *mh_write*를

메타데이터 서버 호스트에 지정하지 않을 경우 특정 시점에 하나의 호스트만 파일에 쓸 수 있습니다.

기본적으로 *mh_write*는 사용 안함으로 설정되고 *wrlease* 마운트 옵션 기간 동안 하나의 호스트만 파일에 대한 쓰기 권한을 가집니다. *mh_write* 옵션이 사용으로 설정된 상태에서 메타데이터 서버에 Oracle HSM 공유 파일 시스템이 마운트될 경우 동일한 파일에 대한 여러 호스트의 동시 읽기 및 쓰기가 발생할 수 있습니다.

*mh_write*가 메타데이터 서버에 사용으로 설정된 경우 Oracle HSM에서는 다음이 지원됩니다.

- 다중 판독기 호스트 및 페이징된 I/O
- 쓰기 장치가 있는 경우에 한하여 여러 읽기 장치 및/또는 쓰기 장치 호스트 및 직접 I/O
- 쓰기 장치가 있는 경우에 한하여 하나의 첨부 호스트(다른 호스트가 읽거나 씀) 및 직접 I/O

mh_write 옵션을 사용하여 파일 시스템은 마운트할 경우 잠금 동작이 변경되지 않습니다. 파일 잠금은 *mh_write*가 적용되는지 여부에 상관없이 동일하게 작동합니다. 그러나 다른 측면에서 동작의 일관성이 떨어질 수 있습니다. 동시 읽기 장치 및 쓰기 장치가 있는 경우 Oracle HSM 공유 파일 시스템은 파일에 대한 모든 호스트 액세스에 직접 I/O를 사용합니다. 따라서 페이지 정렬된 I/O를 다른 호스트에서 즉시 볼 수 있어야 합니다. 그러나 비페이지 정렬 I/O로 인해 사용되지 않은 데이터가 표시되거나 심지어 파일에 기록될 수 있는데 이는 이러한 동작을 방지하는 일반 임대 메커니즘이 사용 안함으로 설정되었기 때문입니다.

이러한 이유로 여러 호스트가 동일한 파일에 동시에 써야 하고 호스팅된 응용 프로그램이 페이지 정렬된 I/O를 수행하면서 충돌하는 쓰기를 조정할 경우에만 *mh_write* 옵션을 지정해야 합니다. 다른 경우에는 데이터 불일치가 발생할 수 있습니다. *mh_write*와 함께 *flock()*을 사용하여 호스트 간에 조정해도 일관성이 보장되지 않습니다. 자세한 내용은 *mount_samfs* 매뉴얼 페이지를 참조하십시오.

min_pool: 최소 동시 스레드 수 설정

min_pool 마운트 옵션은 Oracle HSM 공유 파일 시스템에 대한 최소 동시 스레드 수를 설정합니다. Oracle Solaris 시스템에서 기본 설정은 *min_pool=64*입니다. 이 설정은 Oracle Solaris에서 최소 64개의 활성 스레드가 스레드 풀에 있다는 것을 의미합니다. *min_pool* 설정은 공유 파일 시스템 작업에 따라 [*8-2048*] 범위의 값으로 조정할 수 있습니다.

min_pool 마운트 옵션은 *samfs.cmd* 파일에 설정해야 합니다. */etc/vfstab* 파일 또는 명령줄에 설정할 경우 무시됩니다.

meta_timeo: 캐시된 속성 유지

meta_timeo 마운트 옵션은 메타데이터 정보에 대한 검사 사이에 시스템이 대기하는 시간 간격을 결정합니다. 기본적으로 시스템은 3초마다 메타데이터 정보를 새로 고칩니다. 예를 들어, 몇 개 파일이 새로 생성된 공유 파일 시스템에서 3초가 경과하기 전에 *ls* 명령

을 입력하면 모든 파일에 대한 정보가 반환되지 않을 수 있습니다. 이 옵션의 구문은 *meta_timeo=seconds*입니다. 여기서 *seconds*는 [*0-60*] 범위의 정수입니다.

stripe: 스트라이프 할당 지정

기본적으로 공유 파일 시스템의 데이터 파일은 라운드 로빈 파일 할당 방법을 사용하여 할당됩니다. 디스크 간에 파일 데이터가 스트라이프되도록 지정하려면 메타데이터 호스트 및 모든 잠재적 메타데이터 호스트에서 *stripe* 마운트 옵션을 지정합니다. 기본적으로 비공유 파일 시스템은 스트라이프 방법을 사용하여 파일 데이터를 할당합니다.

라운드 로빈 할당에서는 파일이 각 슬라이스 또는 스트라이프 그룹에서 라운드 로빈 방식으로 만들어집니다. 한 파일의 최대 성능은 슬라이스 또는 스트라이프 그룹의 속도가 됩니다. 파일 할당 방법에 대한 자세한 내용은 Oracle Hierarchical Storage Manager and StorageTek QFS 설치 및 구성 설명서(*Oracle HSM* 고객 설명서 라이브러리, *docs.oracle.com/en/storage*)를 참조하십시오.

sync_meta: 메타데이터가 기록되는 빈도 지정

sync_meta 옵션을 *sync_meta=1* 또는 *sync_meta=0*으로 설정할 수 있습니다.

기본 설정은 *sync_meta=1*이며 이는 메타데이터가 변경될 때마다 Oracle HSM 공유 파일 시스템이 파일 메타데이터를 디스크에 기록한다는 것을 의미합니다. 이 설정으로 인해 데이터 성능이 저하되지만 데이터 일관성이 보장됩니다. 메타데이터 서버를 변경하려는 경우 이 설정을 적용해야 합니다.

*sync_meta=0*을 설정할 경우 Oracle HSM 공유 파일 시스템은 메타데이터를 디스크에 기록하기 전에 버퍼에 기록합니다. 이렇게 지연된 쓰기는 향상된 성능을 제공하지만 예약되지 않은 시스템 중단 후 데이터 일관성이 저하됩니다.

worm_capable 및 def_retention: WORM 기능을 사용으로 설정

worm_capable 마운트 옵션은 파일 시스템에서 WORM 파일을 지원할 수 있게 합니다. *def_retention* 마운트 옵션은 *def_retention=MyNdOhPm* 형식을 사용하여 기본 보존 시간을 설정합니다.

이 형식에서 *M*, *N*, *O* 및 *P*는 음수가 아닌 정수이고 *y*, *d*, *h* 및 *m*은 각각 년, 일, 시간 및 분을 나타냅니다. 이러한 단위를 임의로 조합하여 사용할 수 있습니다. 예를 들어, *1y5d4h3m*은 1년, 5일, 4시간 및 3분을 나타내고 *30d8h*는 30일 및 8시간을 나타내며 *300m*은 300분을 나타냅니다. 이 형식은 보존 기간이 분 단위로 지정되었던 이전 소프트웨어 버전의 공식과 호환됩니다.

자세한 내용은 Oracle Hierarchical Storage Manager and StorageTek QFS 설치 및 구성 설명서(*Oracle HSM* 고객 설명서 라이브러리, *docs.oracle.com/en/storage*)를 참조하십시오.

부록 D

부록 D. 아카이빙을 위한 구성 지시어

이 부록에는 Oracle Hierarchical Storage Manager 파일 시스템 및 관련 소프트웨어 작업을 구성하는 지시어가 나열되어 있습니다. 각 지시어는 쉼표로 구분된 하나 이상의 필드로 구성된 한 라인의 텍스트입니다. 관련 지시어는 Oracle HSM 명령(*.cmd*) 파일에 함께 저장됩니다.

이 부록의 나머지 부분에서는 세 가지 기본 지시어 유형의 개요를 제공합니다.

- 아카이빙 지시어
- 스테이징 지시어
- 미리보기 요청 지시어

자세한 내용은 Oracle HSM 매뉴얼 페이지를 참조하십시오.

Oracle HSM 명령 파일은 여기에 설명된 대로 명령줄을 사용하거나 Oracle HSM Manager 소프트웨어를 사용해서 구성할 수 있습니다. Oracle HSM Manager에 대한 자세한 내용은 온라인 도움말을 참조하십시오.

아카이빙 지시어

이 절에서는 *archiver.cmd* 파일을 구성하는 아카이빙 지시어의 사용법 정보가 제공됩니다. 아카이빙 지시어는 파일 복사, 사용된 매체 및 아카이빙 소프트웨어의 전반적인 동작을 제어하는 아카이브 세트를 정의합니다.

다음과 같은 네 가지 기본 유형의 아카이빙 지시어가 있습니다.

- 전역 아카이빙 지시어
- 파일 시스템 지시어
- 복사 매개변수
- VSN(볼륨 일련 번호) 연관 지시어

전역 및 파일 시스템 지시어는 둘 다 파일이 아카이브되는 방법을 제어합니다. 그러나 아카이버는 전역 지시어를 평가하기 전에 파일 시스템 특정 지시어를 평가합니다. 따라서 충돌이 있는 경우 파일 시스템 지시어가 전역 지시어를 대체합니다. 마찬가지로 파일 시스템 지시어 내에서 먼저 나열된 지시어가 이후의 충돌하는 지시어를 대체합니다.

전역 아카이빙 지시어

전역 지시어는 전체 아카이버 작업을 제어하고 구성된 모든 파일 시스템의 작업 최적화를 사용으로 설정합니다. 전역 지시어는 단독 키워드나 키워드 뒤에 등호(=)와 추가 데이터 필드로

구성됩니다. 전역 지시어는 *archiver.cmd* 파일을 시작하고 첫번째 파일 시스템 지시어에서 종료합니다.

archivemeta: 메타데이터가 아카이브되는지 여부 제어

archivemeta 지시어는 파일 시스템 메타데이터가 아카이브되는지 여부를 제어합니다. 파일을 자주 이동하고 파일 시스템의 디렉토리 구조를 자주 변경할 경우 파일 시스템 메타데이터를 아카이브합니다. 그러나 디렉토리 구조가 상당히 안정적인 경우 메타데이터 아카이빙을 사용 안함으로 설정하여 이동식 매체 드라이브가 수행하는 작업을 줄일 수 있습니다. 기본적으로 메타데이터는 아카이브되지 않습니다.

이 지시어의 형식은 다음과 같습니다.

```
archivemeta=state
```

*state*에 대해서는 *on* 또는 *off*를 지정합니다. 기본값은 *off*입니다.

메타데이터의 아카이빙 프로세스는 다음과 같이 버전 1 또는 버전 2 수퍼 블록을 사용하는지에 따라 달라집니다.

- 버전 1 파일 시스템의 경우 아카이버가 디렉토리, 이동식 매체 파일, 세그먼트 인덱스 inode 및 심볼릭 링크를 메타데이터로 아카이브합니다.
- 버전 2 파일 시스템의 경우에는 아카이버가 디렉토리 및 세그먼트 인덱스 inode를 메타데이터로 아카이브합니다. 이동식 매체 파일 및 심볼릭 링크는 데이터 블록이 아닌 inode에 저장됩니다. 이러한 항목은 아카이브되지 않습니다. 심볼릭 링크는 데이터로 아카이브됩니다.

archmax: 아카이브 파일 크기 제어

archmax 지시어는 아카이브(*.tar*) 파일의 최대 크기를 지정합니다. *target-size* 값에 도달한 다음에는 아카이브 파일에 다른 사용자 파일이 추가되지 않습니다. 큰 사용자 파일은 단일 아카이브 파일에 기록됩니다.

기본값을 변경하려면 다음 지시어를 사용합니다.

```
archmax=media target-size
```

여기서 *media*는 부록 A. 장비 유형 용어집 및 *mcf* 매뉴얼 페이지에 정의된 매체 유형 중 하나이고 *target-size*는 아카이브 파일의 최대 크기입니다. 이 값은 매체에 종속적입니다. 기본적으로 광 디스크에 기록되는 아카이브 파일은 5MB를 초과하지 않습니다. 테이프의 기본 최대 아카이브 파일 크기는 512MB입니다.

아카이브 파일의 크기를 크거나 작게 설정하는 것은 장점과 단점이 있습니다. 예를 들어, 테이프에 아카이빙 중이며 *archmax*를 큰 값으로 설정한 경우 테이프 드라이버가 중지 및 시작하는 빈도가 줄어듭니다. 그러나 큰 아카이브 파일을 기록하는 경우 테이프 끝에 너무 일찍 도달해서 많은 양의 테이프가 낭비됩니다. 최선의 방법은 *archmax* 지시어를 매체 용량의 5%보다 크지 않게 설정하는 것입니다.

또한 *archmax* 지시어를 개별 아카이브 세트에 대해 설정할 수 있습니다.

bufsize: 아카이버 버퍼 크기 설정

기본적으로 아카이브하는 파일은 메모리 버퍼를 사용하여 아카이브 매체에 복사됩니다. *bufsize* 지시어를 사용하여 기본값이 아닌 버퍼 크기를 지정하고 선택적으로 버퍼를 잠글 수 있습니다. 이러한 작업은 일부 경우에서 성능을 향상시킬 수 있습니다. 여러 다른 *number-blocks* 값을 사용해 볼 수 있습니다. 이 지시어의 형식은 다음과 같습니다.

```
bufsize=media number-blocks [lock]
```

설명:

- *media*는 부록 A. 장비 유형 용어집 및 *mcf* 매뉴얼 페이지에 정의된 매체 유형 중 하나입니다.
- *number-blocks*는 [*2-1024*] 범위의 숫자입니다. 기본값은 *4*입니다. 이 값에 매체 유형의 *dev_blksize* 값을 곱하여 얻은 결과 버퍼 크기가 사용됩니다. *dev_blksize* 값은 *defaults.conf* 파일에 지정됩니다. 자세한 내용은 *defaults.conf* 매뉴얼 페이지를 참조하십시오.
- *lock*은 아카이브 복사본을 만들 때 아카이버가 잠긴 버퍼를 사용할 수 있는지 여부를 나타냅니다.

  *lock*이 지정된 경우 아카이버는 *sam-arcopy* 작업 기간 동안에 메모리의 아카이브 버퍼에 파일 잠금을 설정합니다. 이 작업으로 인해 각 I/O 요청에 대한 버퍼 잠금 및 잠금 해제와 연관된 오버헤드가 방지되고 결과적으로 시스템 CPU 시간이 감소합니다.

  lock 인수는 많은 양의 메모리가 있는 대형 시스템에만 지정해야 합니다. 메모리가 충분하지 않으면 메모리 부족 상태가 될 수 있습니다. *lock* 인수는 아카이브하려는 파일에 대해 직접 I/O가 사용으로 설정된 경우에만 유효합니다. 기본적으로 *lock*이 지정되지 않으며 파일 시스템은 아카이빙을 위한 버퍼를 비롯한 모든 직접 I/O 버퍼에 잠금을 설정합니다.

아카이브 세트 복사 매개변수 *-bufsize* 및 *-lock*을 사용하여 각 아카이브 세트에 대한 버퍼 크기 및 잠금을 지정할 수 있습니다. 자세한 내용은 "아카이브 복사 지시어"를 참조하십시오.

drives: 아카이빙에 사용되는 드라이브 수 제어

기본적으로 아카이버는 자동화된 라이브러리의 모든 드라이브를 아카이빙에 사용합니다. 사용되는 드라이브 수를 제한하려면 *drives* 지시어를 사용합니다. 이 지시어의 형식은 다음과 같습니다.

```
drives=media-library count
```

여기서 *media-library*는 *mcf* 파일에 정의된 대로 자동화된 라이브러리의 패밀리 세트 이름이고 *count*는 아카이빙에 사용할 수 있는 드라이브 수입니다.

또한 이 목적을 위해 아카이브 세트 복사 매개변수 *-drivemax*, *-drivemin* 및 *-drives*를 사용할 수 있습니다. 자세한 내용은 "아카이브 복사 지시어"를 참조하십시오.

examine: 아카이브 스캔 제어

examine 지시어는 아카이빙 준비가 완료된 파일을 아카이버에서 식별하는 데 사용되는 *method*를 설정합니다.

```
examine=method
```

여기서 *method*는 다음 지시어 중 하나입니다.

- 기본값인 *noscan*은 연속 아카이빙을 지정합니다. 초기 스캔 이후 디렉토리는 컨텐츠가 변경되고 아카이빙이 필요한 경우에만 스캔됩니다. 디렉토리 및 inode 정보는 스캔되지 않습니다. 이 아카이빙 방법은 특히 1,000,000개 이상의 파일이 있는 파일 시스템의 경우 스캔 아카이빙보다 나은 성능을 제공합니다.
- *scan*은 스캔 아카이빙을 지정합니다. 파일 시스템 디렉토리의 초기 스캔 이후 inode는 항상 스캔됩니다.
- *scandirs*는 스캔 아카이빙을 지정합니다. 디렉토리는 항상 스캔됩니다. inode 정보는 스캔되지 않습니다.

  no_archive 속성이 설정된 디렉토리는 아카이버에서 스캔되지 않습니다. 따라서 변경되지 않는 파일이 들어 있는 디렉토리에 이 속성을 설정하여 스캔 시간을 줄일 수 있습니다.
- *scaninodes*는 스캔 아카이빙을 지정합니다. Inode는 항상 스캔됩니다. 디렉토리 정보는 스캔되지 않습니다.

interval: 아카이브 간격 지정

아카이버는 아카이브가 사용으로 설정된 모든 마운트된 파일 시스템의 상태를 주기적으로 검사합니다. 이러한 검사 시간은 각 파일 시스템에서 스캔 작업을 수행하는 간격인 아카이브 간격에 따라 제어됩니다. 아카이브 간격을 변경하려면 *interval* 지시어를 사용합니다.

interval 지시어는 연속 아카이빙이 설정되지 않았으며 *startage*, *startsize* 또는 *startcount* 매개변수가 지정되지 않은 경우에만 전체 스캔을 시작합니다. 연속 아카이빙이 설정된 경우(*examine=noscan*) *interval* 지시어는 기본 *startage* 값으로 작동합니다. 이 지시어의 형식은 다음과 같습니다.

```
interval=time
```

*time*에는 파일 시스템에서 수행되는 스캔 작업 간의 시간 간격을 지정합니다. 기본적으로 *time*은 초 단위로 해석되고 10분에 해당하는 값 *600*이 지정됩니다. 분 또는 시간과 같은 다른 시간 단위를 지정할 수 있습니다.

아카이버는 *samu* 유틸리티의 *arrun* 명령을 수신할 경우 모든 파일 시스템의 스캔을 즉시 시작합니다. 또한 *examine=scan* 지시어가 *archiver.cmd* 파일에 지정된 경우 *arrun* 또는 *arscan*이 실행된 후에 스캔이 수행됩니다.

hwm_archive 마운트 옵션이 파일 시스템에 설정된 경우 아카이브 간격이 자동으로 단축될 수 있습니다. 파일 시스템 사용률이 고수위를 넘을 때 아카이버가 스캔을 시작합니다. *high=percent* 마운트 옵션은 파일 시스템에 대한 고수위를 설정합니다.

아카이브 간격을 지정하는 방법에 대한 자세한 내용은 *archiver.cmd* 및 *mount_samfs* 매뉴얼 페이지를 참조하십시오.

logfile: 아카이버 로그 파일 지정

아카이버는 아카이브, 다시 아카이브 또는 아카이브 취소되는 각 파일에 대한 정보가 포함된 로그 파일을 생성할 수 있습니다. 이 로그 파일은 아카이브 작업의 연속 레코드입니다. 기본적으로 아카이버 로그 파일은 사용으로 설정되지 않습니다. 로그 파일을 지정하려면 *logfile* 지시어를 사용합니다. 이 지시어의 형식은 다음과 같습니다.

```
logfile=pathname
```

*pathname*에는 로그 파일의 절대 경로 및 이름을 지정합니다. 또한 *logfile* 지시어를 개별 파일 시스템에 대해 설정할 수 있습니다.

아카이버 로그 파일은 손상 또는 손실된 파일 시스템을 복구하는 데 필수적이며 모니터링 및 분석을 위해 중요할 수 있습니다. 따라서 아카이버 로그를 사용으로 설정하고 백업해야 합니다. 자세한 내용은 Oracle Hierarchical Storage Manager and StorageTek QFS 설치 및 구성 설명서를 참조하십시오.

notify: 이벤트 알림 스크립트 이름 바꾸기

notify 지시어는 아카이버의 이벤트 알림 스크립트 파일에 대한 이름을 설정합니다. 이 지시어의 형식은 다음과 같습니다.

```
notify=filename
```

*filename*에는 아카이버 이벤트 알림 스크립트를 포함하는 파일의 이름 또는 이 파일의 전체 경로를 지정합니다. 기본 파일 이름은 */etc/opt/SUNWsamfs/scripts/archiver.sh*입니다.

아카이버는 이 스크립트를 실행하여 사이트 특정 방식으로 여러 이벤트를 처리합니다. 이 스크립트는 첫번째 인수에 대해 *emerg*, *alert*, *crit*, *err*, *warning*, *notice*, *info* 및 *debug* 키워드 중 하나를 사용해서 호출됩니다.

추가 인수는 기본 스크립트에 설명되어 있습니다. 자세한 내용은 *archiver.sh* 매뉴얼 페이지를 참조하십시오.

ovflmin: 볼륨 오버플로우 제어

볼륨 오버플로우가 사용으로 설정된 경우 아카이버는 여러 볼륨에 걸쳐 있는 아카이브된 파일을 만들 수 있습니다. 파일 크기가 지정된 최소 크기를 초과할 경우 아카이버는 이 파일의

나머지 부분을 동일한 유형의 또 다른 볼륨에 기록합니다. 각 볼륨에 기록된 파일의 일부를 섹션이라고 부릅니다. *sls* 명령은 각 볼륨에 있는 파일의 각 섹션을 보여주는 아카이브 복사본을 나열합니다.

아카이버는 *ovflmin* 지시어를 통해 볼륨 오버플로우를 제어합니다. 기본적으로 볼륨 오버플로우는 사용 안함으로 설정됩니다. 볼륨 오버플로우를 사용으로 설정하려면 *archiver.cmd* 파일에서 *ovflmin* 지시어를 사용합니다. 이 지시어의 형식은 다음과 같습니다.

```
ovflmin = media minimum-file-size
```

여기서 *media*는 부록 A. 장비 유형 용어집 및 *mcf* 매뉴얼 페이지에 정의된 매체 유형 중 하나이고 *minimum-file-size*는 볼륨 오버플로우를 트리거하는 가장 작은 파일 크기입니다. 또한 *ovflmin* 지시어를 개별 아카이브 세트에 대해 설정할 수 있습니다.

볼륨 오버플로우는 해당 효과를 평가한 후 신중하게 사용합니다. 여러 볼륨에 걸쳐 있는 파일의 경우 재해 복구 및 재활용이 훨씬 더 어렵습니다. 볼륨 오버플로우 파일은 체크섬을 생성하지 않습니다. 체크섬 사용에 대한 자세한 내용은 *ssum* 매뉴얼 페이지를 참조하십시오.

scanlist_squash: 스캔 목록 통합 제어

scanlist_squash 매개변수는 스캔 목록 통합을 제어합니다. 기본 설정은 *off*입니다. 이 매개변수는 전역 또는 파일 시스템 특정 매개변수일 수 있습니다.

*on*인 경우 이 지시어는 디렉토리 트리의 하위 디렉토리에 대한 스캔 목록을 통합하므로 아카이버가 공통 상위 디렉토리에서 아래로 재귀적으로 스캔합니다. 파일 시스템 내에서 많은 파일과 하위 디렉토리가 변경된 경우 스캔 목록 통합 시 아카이빙 성능이 훨씬 저하될 수 있습니다.

setarchdone: archdone 플래그의 설정 제어

setarchdone 전역 지시어는 아카이브되지 않는 파일에 *archdone* 플래그를 설정할지 여부를 제어합니다. 이 지시어의 형식은 다음과 같습니다.

```
setarchdone=state
```

여기서 *state*는 *on* 또는 *off*입니다. *examine* 지시어가 *scandirs* 또는 *noscan*으로 설정된 경우 기본값은 *off*입니다.

archdone 플래그는 아카이빙 프로세스에 플래그가 표시된 파일을 무시하도록 지시합니다. 일반적으로 지정된 모든 파일 복사본이 생성되면 아카이빙 프로세스에서 *archdone* 플래그를 설정하여 나중에 수정될 때까지 해당 파일을 후속 아카이빙 작업에서 건너뛰게 합니다.

그러나 *setarchdone*이 *on*으로 설정된 경우에는 아카이빙 기준을 충족하지 않아 아카이브되지 않았으며 아카이브되지 않는 파일을 아카이빙 프로세스에서 식별하고 플래그를 표시합니다. 이 경우 이후의 아카이빙 오버헤드를 줄일 수 있는 반면 파일 평가로 인해 오버헤드가 즉시 증가하며 성능이 저하될 수 있습니다.

wait: 아카이버 시작 지연

wait 지시어는 아카이버가 *samcmd* 명령, *samu* 인터페이스 또는 Oracle HSM Manager로부터 시작 신호를 기다리게 만듭니다. 이 지시어의 형식은 다음과 같습니다.

```
wait
```

기본적으로 *sam-fsd* 초기화 명령이 실행되면 아카이버가 자동으로 시작됩니다.

또한 *wait* 지시어를 개별 파일 시스템에 대해 설정할 수 있습니다.

파일 시스템 지시어

파일 시스템 지시어는 특정 파일 시스템에 대한 아카이빙 동작을 정의합니다.

- **fs**: 파일 시스템 지정
- ***copy-number* [*archive-age*]**: 파일 시스템 메타데이터의 여러 복사본 지정
- 파일 시스템 지시어 **interval**, **logfile** 및 **scanlist**

fs: 파일 시스템 지정

각 *fs=file-system-name* 지시어에는 이름이 지정된 파일 시스템, *file-system-name*에만 적용되는 일련의 아카이빙 지시어가 사용됩니다. 이 지시어의 형식은 다음과 같습니다.

```
fs=file-system-name
```

여기서 *file-system-name*은 *mcf* 파일에 정의된 파일 시스템 이름입니다.

fs= 지시어 이후에 오는 전역 지시어 및 아카이브 세트 연관 지시어는 지정된 파일 시스템에만 적용됩니다.

copy-number [*archive-age*]: 파일 시스템 메타데이터의 여러 복사본 지정

파일 시스템 메타데이터에는 파일 시스템의 경로 이름이 포함됩니다. 둘 이상의 메타데이터 복사본이 필요한 경우 *fs=* 지시어 바로 뒤에 *archiver.cmd* 파일의 복사본 정의를 포함합니다.

```
copy-number [archive-age]
```

여기서 시간은 하나 이상의 정수 및 시간 단위 조합으로 표시됩니다. 단위에는 *s*(초), *m*(분), *h*(시간), *d*(일), *w*(주) 및 *y*(년)가 포함됩니다. 디렉토리가 자주 변경되는 경우 여러 메타데이터 복사본을 지정하면 파일 시스템이 메타데이터 테이프 볼륨을 너무 자주 마운트할 수 있습니다. 따라서 기본적으로 Oracle HSM은 메타데이터 복사본을 하나만 만듭니다.

예제에서는 *fs=samma1* 파일 시스템에 대한 메타데이터의 복사본 *1*이 4시간(*4h*) 후에 작성되고 복사본 *2*가 12시간(*12h*) 후에 작성됩니다.

```
# General Directives
archivemeta = off
examine = noscan
# Archive Set Assignments
fs = samma1
1 4h
2 12h
```

파일 시스템 지시어 interval, logfile 및 scanlist

여러 지시어를 모든 파일 시스템에 대한 전역 지시어 및 단일 파일 시스템에 대한 특정 지시어로 지정할 수 있습니다. 다음 절에서는 이러한 지시어에 대해 설명합니다.

- **interval**: 아카이브 간격 지정
- **logfile**: 아카이버 로그 파일 지정
- **scanlist_squash**: 스캔 목록 통합 제어
- **wait**: 아카이버 시작 지연

archive-set-name: 아카이브 세트 지정 지시어

아카이브 세트 지정 지시어는 함께 아카이브할 파일을 지정합니다. 아래 설명된 다양한 선택 조건을 사용해서 매우 세밀하게 파일을 지정할 수 있습니다. 하지만 반드시 필요한 경우를 제외하고는 이렇게 세밀한 지정을 피하십시오. 일반적으로 가능한 한 가장 적은 개수로 가장 포괄적인 아카이브 세트를 구성해야 합니다. 아카이브 세트에는 아카이브 매체 세트에 대한 배타적 사용이 포함됩니다. 따라서 과도하게 제한적인 지정 조건에 따라 정의된 각 아카이브 세트 숫자가 많을수록 매체 활용률 저하, 시스템 오버헤드 증가, 및 성능 감소의 원인이 됩니다. 극단적인 경우에는 라이브러리에 많은 용량이 있어도 사용 가능한 매체 부족으로 인해 작업이 실패할 수 있습니다.

각 아카이브 세트 지정 지시어의 형식은 다음과 같습니다.

```
archive-set-name path [-access interval [-nftv]] [-after date-time] [-minsize size] [-
maxsize size] [-user username] [-group groupname] [-name regex]
```

설명:

- *archive-set-name*은 아카이브 세트의 관리자 정의 이름입니다.

  이름은 대문자 및/또는 소문자[*A-Za-z*], 숫자[*0-9*] 및 밑줄(_)을 임의로 조합한 최대 29자를 포함할 수 있으며 첫 글자가 문자여야 합니다. 공백과 같은 다른 문자는 포함할 수 없으며 Oracle HSM 특수 아카이브 세트 *no_archive* 및 *all*의 이름을 고유한 아카이브 세트에 사용할 수 없습니다.
- *path*는 파일 시스템 내에서 아카이빙이 시작되는 하위 디렉토리의 마운트 지점을 기준으로 경로를 지정합니다. 시작 디렉토리 및 해당 하위 디렉토리의 모든 파일이 아카이브됩니

다. 파일 시스템의 모든 파일을 포함하려면 점(*.*) 문자를 사용합니다. 선행 슬래시(*/*)는 경로에서 허용되지 않습니다.

- *-access*는 *interval*로 지정된 시간 동안 액세스되지 않은 파일을 다시 아카이브합니다. 여기서 *interval*은 정수와 그 뒤에 오는 *s*(초), *m*(분), *h*(시), *d*(일), *w*(주), *y*(년) 단위 중 하나로 표시되는 정수입니다.

  이 매개변수를 사용하면 덜 사용되는 파일을 높은 비용의 매체에서 낮은 비용의 매체로 다시 아카이브하도록 예약할 수 있습니다. 소프트웨어는 파일의 액세스 및 수정 시간을 검증하여 파일 생성 시간 이후이고 파일 검사 시간 이전인지 확인합니다. *-nftv*(파일 시간 검증 없음) 매개변수는 이 검증을 사용 안함으로 설정합니다.
- *-after*는 *date-time* 이후에 작성 또는 수정된 파일만 아카이브합니다. 여기서 *date-time*은 *YYYY-MM-DD [hh:mm:ss] [Z]* 형태의 표현식이고 *YYYY*, *MM*, *DD*, *hh*, *mm* 및 *ss*는 각각 년, 월, 일, 시간, 분 및 초를 나타내는 정수입니다. 선택적 *Z* 매개변수는 시간대를 UTC(협정 세계시)로 설정합니다. 기본값은 *00:00:00* 및 로컬 시간입니다.
- *-minsize* 및 *-maxsize*는 지정된 *size*보다 크거나 작은 파일만 아카이브합니다. 여기서 *size*는 정수와 그 뒤에 오는 *b*(바이트), *k*(킬로바이트), *M*(메가바이트), *G*(기가바이트), *T*(테라바이트), *P*(페타바이트) 및 *E*(엑사바이트) 단위 중 하나로 구성됩니다.
- *-user username* 및 *-group groupname*은 지정된 사용자 및/또는 그룹에 속하는 파일만 아카이브합니다.
- *-name*은 경로 및 파일 이름이 정규 표현식 *regex*로 정의된 패턴과 일치하는 모든 파일을 아카이브합니다.

아카이브 복사 지시어

기본적으로 아카이버는 파일의 아카이브 기간이 4분 이상 되었을 때 아카이브 세트에서 파일에 대해 단일 아카이브 복사본을 기록합니다. 기본 동작을 변경하려면 아카이브 복사본 지시어를 사용합니다. 아카이브 복사본 지시어는 연관된 아카이브 세트 지정 지시어 바로 다음에 나타나야 합니다.

아카이브 복사본 지시어는 *copy-number* 값 *1*, *2*, *3* 또는 *4*로 시작합니다. 숫자 다음에는 해당 복사본의 아카이브 특성을 지정하는 하나 이상의 인수가 옵니다. 각 아카이브 복사본 지시어의 형식은 다음과 같습니다.

```
copy-number [archive-age] [-release [attribute] [-norelease][-stage[attribute] [unarchive-age]
```

설명:

- 선택적인 *archive-age* 매개변수는 새 파일 또는 수정된 파일이 아카이브에 적합해질 때까지 디스크 캐시에서 기다려야 하는 시간입니다. 정수와 시간 단위에 대한 하나 이상의 조합으로 *archive-age*를 지정합니다. 여기서 시간 단위에는 **s**(초), **m**(분), *h*(시), *d*(일), *w*(주) 및 *y*(년)가 포함됩니다. 기본값은 *4m*(4분)입니다.
- 선택적인 *-release* 매개변수는 아카이브 복사본이 생성되는 즉시 파일에 사용된 디스크 공간을 비우기 위해 Oracle HSM 릴리서 소프트웨어를 지웁니다. 선택적 릴리스 *attribute*는 *-a*, *-n* 또는 *-d*입니다. *-a*(연관 스테이징) 속성을 사용하면 파일 중 하나가

액세스될 때 아카이브 세트에서 릴리스된 모든 파일이 소프트웨어에서 스테이지되어야 합니다. *-n* 속성을 사용하면 소프트웨어가 아카이브 매체에서 직접 읽어야 하며, 파일을 스테이지해서는 안됩니다. *-d* 속성은 기본 스테이징 동작을 재설정합니다.

- 선택적인 *-norelease* 매개변수는 *-norelease*로 표시된 모든 복사본이 생성될 때까지 파일에 사용된 디스크 공간을 비우기 위해 Oracle HSM 릴리서 소프트웨어를 지우지 않습니다.
- 함께 사용되는 *-release -norelease*의 경우 Oracle HSM 소프트웨어가 *-release -norelease*로 플래그 지정된 모든 복사본이 생성된 바로 다음 파일에 사용된 디스크 공간을 비워야 합니다. Oracle HSM는 릴리서 프로세스가 실행될 때까지 기다리지 않습니다.
- 선택적인 *-stage* 매개변수. 선택적인 릴리스 *attribute*는 *-a*, *-c copy-number*, *-f*, *-I*, *-i input_file*, *-w*, *-n*, *-p*, *-V*, *-x*, *-r*, *-d*입니다. 설명:

  *-a*를 사용하면 파일 중 하나가 액세스될 때 아카이브 세트에서 모든 파일을 스테이지해야 합니다.

  *-c copy-number*를 사용하면 소프트웨어가 지정된 복사본 번호로부터 스테이지를 수행해야 합니다.

  *-n*을 사용하면 소프트웨어가 아카이브 매체로부터 직접 읽어야 하며, 파일을 스테이지해서는 안됩니다.

  *-w*를 사용하면 소프트웨어가 작업을 계속하기 전에 각 파일이 성공적으로 스테이지될 때까지 기다려야 합니다(*-d* 또는 *-n*에 적합하지 않음).

  *-d*는 기본 스테이징 동작을 재설정합니다.
- *unarchive-age* 매개변수는 재사용을 위해 매체에서 공간을 비울 수 있도록 아카이브 해제되기 전에 아카이브에서 대기할 수 있는 시간을 지정합니다. 시간은 정수와 시간 단위에 대한 하나 이상의 조합으로 표현되고, 여기서 단위에는 *s*(초), *m*(분), *h*(시), *d*(일), *w*(주) 및 *y*(년)이 있습니다.

아래 예제에는 아카이브 세트 *allsamma1*에 대해 두 개의 복사본 지시어가 포함됩니다. 첫번째 지시어는 아카이브 기간이 5분(*5m*)이 될 때까지 복사본 *1*을 릴리스하지 않습니다. 두번째 지시어는 아카이브 기간이 1시간(*1h*)이 될 때까지 복사본 *2*를 릴리스하지 않고 아카이브 해제 기간이 7년 6개월(*7y6m*)이 된 다음 복사본 *2*를 아카이브 해제합니다.

```
# Archive Set Assignments
fs = samma1
logfile = /var/adm/samma1.archive.log
allsamma1 .
    1 -norelease 5m
    2 -norelease 1h 7y6m
```

복사 매개변수

복사 매개변수는 아카이브 세트로 지정된 복사본이 생성되는 방법을 정의합니다. *archiver.cmd* 파일의 아카이브 세트 복사 매개변수 섹션은 *params* 지시어로 시작해서 *endparams* 지시어로 끝납니다.

```
params
allsets -sort path -offline_copy stageahead
allfiles.1 -startage 10m -startsize 10M -drives 10 -archmax 1G
allfiles.2 -startage 1h -startsize 1G -drives 2 -archmax 10G -reserve set
endparams
```

각 복사 매개변수의 형태는 다음과 같습니다.

archive-set-name[.copy-number][R] [-startage time] [-startcount count] [-startsize size] [-archmax maximum-size] [-bufsize=number-blocks] [-drivemax maximum-size] [-drivemin minimum-size] [-drives number] [-fillvsns] [-lock] [-offline_copy method] [-sort criterion] [-rsort criterion] [-recycle_dataquantity size] [-recycle_hwm percent] [-recycle_ignore] [-recycle_mailaddr mail-address] [-recycle_mingainpercentage] [-recycle_vsncountcount] [-recycle_minobs percentage] [-unarchagetime_ref] [-tapenonstop] [-reserve keyword] [-priority multiplier ranking]

설명:

- *archive-set-name*은 파일 시스템 지시어의 아카이브 세트 지정 지시어 또는 모든 정의된 아카이브 세트에 지정된 복사 매개변수를 적용하는 특수 지시어 *allsets*로 정의된 아카이브 세트의 이름입니다. 개별 아카이브 세트에 대한 매개변수를 지정하기 전에 먼저 *allsets*의 매개변수를 설정합니다. 그렇지 않은 경우 개별 아카이브 세트에 대한 매개변수가 *allsets* 지정으로 대체되므로 해당 목적에 맞지 않습니다.
- *.copy-number*는 응용 프로그램의 지정된 복사 매개변수가 *copy-number*로 지정된 아카이브 복사본에만 적용되도록 제한합니다. 여기서 *copy-number*는 [*1-4*] 범위의 정수이고, 선택적인 *R*은 응용 프로그램의 매개변수가 다시 아카이브된 복사본에만 적용되도록 제한합니다.
- *-startage time*은 첫번째 파일이 아카이브 요청에 추가되는 시점과 아카이빙이 실제로 시작되는 시점 간의 간격을 지정합니다. *time*을 하나 이상의 정수 및 시간 단위 조합으로 지정합니다. 여기서 단위에는 *s*(초), *m*(분), *h*(시간), *d*(일), *w*(주) 및 *y*(년)가 포함됩니다. 기본값은 *2h*(2시간)입니다.
- *-startcount count*는 아카이브 요청의 최소 파일 수를 지정합니다. 아카이빙 대기 중인 파일 수가 이 임계값에 도달할 경우 아카이빙이 시작됩니다. 기본적으로 *count*는 설정되지 않습니다.
- *-startsize size*는 아카이브 요청의 최소 크기를 바이트 단위로 지정합니다. 아카이빙 대기 중인 파일의 총 크기가 이 임계값에 도달할 경우 아카이빙이 시작됩니다. 기본적으로 *size*는 설정되지 않습니다.
- *-archmax*는 아카이브 파일 크기를 *maximum-size* 이하로 제한합니다. 여기서 *maximum-size*는 매체에 종속적입니다. 자기 테이프의 기본 최대 아카이브 파일 크기는 512MB입니다. 광 디스크에 기록되는 아카이브 파일은 5MB를 초과하지 않습니다.

  동일한 이름의 전역 아카이빙 지시어에 대한 설명은 “**archmax**: 아카이브 파일 크기 제어”를 참조하십시오.
- *-bufsize= media-type number-blocks*는 버퍼가 *number-blocks*dev_blksize*에 따라 아카이브 매체에 기록될 때 아카이브 파일을 보유하는 버퍼의 크기를 설정합니다. 여기서 *number-blocks*는 [*2-32*] 범위의 정수이고 *dev_blksize*는 *defaults.conf* 파일에서 매체 유형에 지정된 블록 크기입니다. 기본값은 *4*입니다.

- *-drivemax*는 하나의 드라이브를 사용하여 아카이브된 데이터 양을 *maximum-size*MB 이하로 제한합니다. 여기서 *maximum-size*는 정수입니다. 기본적으로 *maximum-size*는 지정되지 않습니다.

  -drives 매개변수를 사용해서 여러 드라이브가 지정된 경우, 어느 한 드라이브에 기록되는 데이터 양을 제한하면 드라이브 성능을 향상시키고, 작업 로드의 균형을 조정하고, 전반적인 드라이브 사용률을 높이는 데 도움이 됩니다.
- *-drivemin minimum-size*는 하나의 드라이브를 사용하여 아카이브된 데이터 양을 최소 *minimum-size*MB 이상으로 제한합니다. 여기서 *minimum-size*는 정수입니다. 기본값은 *-archmax*(지정된 경우)의 값 또는 *defaults.conf* 파일에서 매체 유형에 대해 나열된 값입니다.

  드라이브에 기록되는 데이터의 양을 더 작게 제한하면 드라이브 사용률 및 효율성이 향상될 수 있습니다. *minimum-size*는 전송 시간이 매체 로드, 배치 및 언로드 시간을 초과하도록 충분히 크게 설정합니다. *-drivemin*이 지정되었으면 데이터 전송이 충분히 큰 경우에만 여러 드라이브가 사용됩니다.
- *-drives number*는 아카이빙에 사용되는 드라이브 수를 *number* 이하로 제한합니다. 여기서 *number*는 정수입니다. 기본값은 **1**입니다.

  드라이브 최대 개수를 더 높게 설정하면 아카이브 세트에 큰 파일 또는 대량의 파일이 포함된 경우 성능이 향상될 수 있습니다. 사용 가능한 드라이브의 작동 속도가 서로 다를 경우, 여러 드라이브를 지정하면 이러한 차이가 균형적으로 조정되고 아카이빙 효율성이 향상됩니다.
- *-fillvsns*는 아카이빙 프로세스에서 더 작은 아카이브 파일을 사용하도록 강제로 지정하여 아카이브 매체 볼륨을 보다 완전하게 채웁니다.

  기본적으로 아카이버는 아카이브 복사본의 모든 파일을 저장하기에 공간이 충분한 볼륨을 선택합니다. 따라서 아카이브 파일이 클수록 여러 카트리지의 남은 용량에 들어가지 못할 수 있습니다. 그 결과 전반적으로 매체 활용률이 낮아집니다. *-fillvsns* 매개변수는 이 문제를 해결하지만 매체 마운트, 배치 작업 및 마운트 해제 비용이 추가되며, 아카이빙 및 스테이징 성능을 저하시킵니다.
- *-lock*은 직접 I/O를 사용해서 아카이브 복사본을 만들 때 잠긴 버퍼 사용을 강제로 적용합니다. 잠긴 버퍼는 버퍼 페이징을 방지하고 직접 I/O 성능을 향상시킵니다.

  -lock 매개변수는 사용 가능한 메모리가 제한적인 시스템에서 지정될 경우 메모리 부족 조건을 일으킬 수 있습니다. 기본적으로 잠긴 버퍼는 필수가 아니며, 파일 시스템이 아카이빙 버퍼를 계속 제어합니다.
- *-offline_copy method*는 파일이 이미 디스크 캐시에서 릴리스되었을 때 아카이브 복사본의 생성 방법을 지정합니다. 지정된 *method*는 *direct*, *stageahead*, *stageall* 또는 *none*일 수 있습니다.

  단일 아카이브 복사본이 생성되는 즉시 파일을 릴리스할 수 있으므로 남은 복사본은 오프라인 복사본으로부터 생성되어야 합니다. 지정된 *-offline_copy* 방법을 사용하면 사용할 수 있는 드라이브 수 및 디스크 캐시에서 사용 가능한 공간에 맞게 복사 프로세스를 조정할 수 있습니다.

*direct*는 2개의 드라이브를 사용해서 오프라인 볼륨에서 아카이브 볼륨으로 파일을 직접 복사합니다. 적절한 버퍼 공간을 보장하기 위해서는 이 방법을 사용할 때 *stage_n_window* 마운트 옵션으로 설정된 값을 늘립니다.

*stageahead*는 대상에 아카이브 파일을 기록하는 동안 다음 아카이브 파일을 스테이지합니다.

*stageall*은 아카이빙 전에 하나의 드라이브를 사용해서 디스크 캐시에 모든 파일을 스테이지합니다. 이 방법을 사용할 때는 해당 디스크 캐시가 파일을 저장하기에 충분히 큰지 확인해야 합니다.

none(기본값)은 아카이브 볼륨에 복사하기 전 필요에 따라 디스크 캐시에 파일을 스테이지합니다.

- *-sort*는 파일을 아카이브하기 전에 *criterion*에 따라 파일을 정렬합니다. 여기서 *criterion*은 *age*, *priority*, *size* 또는 *none*입니다.

  *age*는 가장 오래된 수정 시간부터 최신 수정 시간 순으로 정렬을 지정합니다.

  path(기본값)는 전체 파일 이름으로 정렬을 지정하며 동일한 디렉토리에 있는 파일을 아카이브 매체에 유지합니다.

  *priority*는 가장 높은 우선순위부터 가장 낮은 순으로 아카이빙 우선순위로 정렬을 지정합니다.

  *size*는 가장 작은 파일 크기부터 가장 큰 순으로 파일 크기별로 파일을 정렬합니다.

  *none*은 정렬을 지정하지 않고 파일 시스템에서 파일이 발견되는 순서로 파일을 아카이브합니다.
- *-rsort criterion*은 *-sort*와 같이 *criterion*으로 파일을 정렬하지만 정렬 순서가 반대입니다.
- *-recycle_dataquantity size*는 리사이클러가 재아카이빙을 예약할 데이터 양을 *size*바이트로 제한합니다. 여기서 *size*는 정수입니다.

  리사이클러는 적합한 아카이브 파일의 아카이브 볼륨을 비우기 위해 필요할 때 재아카이빙을 예약합니다. 재활용하도록 선택되는 실제 볼륨 수는 *-recycle_vsncount* 매개변수에 따라 달라질 수도 있습니다. 기본값은 *1073741824*(1GB)입니다.
- *-recycle_hwm percent*는 이동식 매체의 재활용을 시작하는 최대 매체 활용률(고수위 또는 *hwm*)을 설정합니다. 이 매개변수는 디스크 매체의 경우 무시됩니다(아래 *-recycle_minobs* 참조). 기본값은 *95*입니다.
- *-recycle_ignore*는 아카이브 세트의 매체가 실제로 재활용되지 않도록 방지하면서 재활용 프로세스는 정상적으로 실행되도록 허용합니다. 테스트 목적으로 사용됩니다.
- *-recycle_mailaddr mail-address*는 *mail-address*로 리사이클러 정보 메시지를 전송합니다. 메일은 기본적으로 설정되지 않습니다.

- *-recycle_mingain*은 지정된 *percentage* 이상 여유 공간을 늘릴 수 있도록 재활용에 사용할 볼륨 선택을 제한합니다. 기본값은 *50*입니다.
- *-recycle_vsncount*는 리사이클러가 재아카이빙하도록 예약하는 볼륨 수를 *count*로 제한합니다. 재활용하도록 선택되는 실제 볼륨 수는 *-recycle_dataquantity* 매개변수에 따라 달라질 수도 있습니다. 이 매개변수는 디스크 매체의 경우 무시됩니다. 기본값은 *1*입니다.
- *-recycle_minobs*는 디스크에 있는 아카이브 파일에서 적합한 파일을 재아카이빙하고 원본 *tar* 파일을 삭제하도록 트리거하는 오래된 파일의 *percentage*를 설정합니다. 이 매개변수는 이동식 매체의 경우 무시됩니다(위 *-recycle_hwm* 참조). 기본값은 *50*입니다.
- *-unarchage*는 아카이브 해제 시간을 계산하기 위한 참조 시간을 *time_ref*로 설정합니다. 여기서 *time_ref*는 파일 액세스 시간의 경우 *access*(기본값) 또는 수정 시간의 경우 *modify*입니다.
- *-tapenonstop*은 이동식 매체 파일을 닫지 않고 단일 테이프 표시 및 EOF(파일 끝) 레이블을 아카이브 파일의 끝에 기록합니다. 이렇게 하면 여러 아카이브 파일의 전송 속도가 빨라지지만 전체 아카이브 세트를 테이프에 쓰기 전까지 테이프 카트리지를 언로드할 수 없습니다. 기본적으로 Oracle HSM 소프트웨어는 아카이브 파일의 끝에서 파일 끝 레이블 뒤에 추가 테이프 표시 2개를 기록하여 테이프 파일을 닫습니다.
- *-reserve keyword*는 지정된 아카이브 세트의 배타적 사용을 위해 이동식 매체 볼륨을 예약합니다. 아카이브 세트의 파일을 보유하기 위해 볼륨이 처음 사용될 경우 소프트웨어는 하나 이상의 지정된 키워드 *fs*, *set* 및/또는 *dir*(디렉토리), *user* 또는 *group* 중 하나에 기초하여 고유한 예약 이름을 볼륨에 지정합니다.

  *fs*는 파일 시스템 이름을 예약 이름에 포함합니다(*arset.1 -reserve fs*).

  *set*는 예약 이름에 아카이브 세트 지정 지시어의 아카이브 세트 이름을 포함합니다(예: *all -reserve set*).

  *dir*은 아카이브 세트 지정 지시어에 지정된 디렉토리 경로의 처음 31자를 예약 이름에 포함합니다.

  *user*는 아카이브 파일과 연관된 사용자 이름을 포함합니다(*arset.1 -reserve user*).

  *group*은 아카이브 파일과 연관된 그룹 이름을 포함합니다(*arset.1 -reserve group*).

  경우에 따라서는 세트별로 볼륨을 예약하는 것이 유리할 수 있습니다. 그러나 기본적으로 이 방법은 소프트웨어에서 매체를 선택하도록 허용하는 것보다 비효율적입니다. 볼륨이 예약된 경우 시스템은 카트리지를 더 자주 마운트, 마운트 해제 및 배치해야 하므로 오버헤드가 증가하고 성능이 저하됩니다. 매우 제한적인 예약 체계의 경우 사용 가능한 매체의 활용도가 낮아지고 극단적으로는 사용 가능한 매체의 부족으로 인해 아카이브 실패가 발생할 수 있습니다.
- *-priority multiplier ranking*은 위에 나열된 *sort priority* 매개변수와 함께 사용될 경우 파일의 아카이빙 우선 순위를 변경합니다. *ranking*은 [*(-3.400000000E+38)-3.400000000E+38*](-3.402823466x10^{38} - 3.402823466x10^{38}) 범위의 실수이고 *multiplier*는 상대 *ranking*을 변경하려는 *age*, *archive_immediate*, *archive*

_overflow, *archive_loaded*, *copies*, *copy1*, *copy2*, *copy3*, *copy4*, *offline*, *queuewait*, *re-archive*, *reqrelease*, *size*, *stage_loaded* 및 *stage_overflow* 중에서 선택된 아카이브 특성입니다.

우선 순위에 대한 자세한 내용은 *archiver* 및 *archiver.cmd* 매뉴얼 페이지를 참조하십시오.

VSN(볼륨 일련 번호) 풀 지시어

archiver.cmd 파일의 VSN 풀 섹션은 VSN(볼륨 일련 번호) 연관 지시어에서 단위로 지정할 수 있는 아카이브 매체 볼륨의 명명된 모음을 정의합니다.

이 섹션은 *vsnpools* 지시어로 시작하고 *endvsnpools* 지시어 또는 *archiver.cmd* 파일의 끝으로 끝납니다. VSN 풀 정의에 대한 구문은 다음과 같습니다.

```
vsn-pool-name media-type volume-specification
```

설명:

- *vsn-pool-name*은 풀에 지정하는 이름입니다.
- *media-type*은 부록 A. 장비 유형 용어집 및 *mcf* 매뉴얼 페이지에 나열된 2자로 된 Oracle HSM 매체 유형 식별자 중 하나입니다.
- *volume-specification*은 볼륨 일련 번호와 일치하는 하나 이상의 정규 표현식이 공백으로 구분된 목록입니다. 정규 표현식 구문에 대한 자세한 내용은 Solaris *regcmp* 매뉴얼 페이지를 참조하십시오.

이 예제에서는 4개의 VSN 풀인 *users_pool*, *data_pool*, *proj_pool* 및 *scratch_pool*을 정의합니다. 스크래치 풀은 VSN 연관에서 특정 볼륨이 소진되었을 때 또는 다른 VSN 풀이 소진되었을 때 사용되는 일련의 볼륨입니다. 3개의 특정 풀 중 하나에서 볼륨이 부족해지면 아카이버가 스크래치 풀 VSN을 선택합니다.

```
vsnpools
users_pool li ^VOL2[0-9][0-9]
data_pool li ^VOL3.*
scratch_pool li ^VOL4[0-9][0-9]
proj_pool li ^VOL[56].*
endvsnpools
```

VSN(볼륨 일련 번호) 연관 지시어

archiver.cmd 파일의 VSN 연관 섹션은 아카이브 매체 볼륨을 아카이브 세트에 지정합니다. 이 섹션은 *vsns* 지시어로 시작하고 *endvsns* 지시어로 끝납니다.

볼륨 지정 지시어의 형식은 다음과 같습니다.

```
archive-set-name.copy-number [media-type volume-specification] [-pool vsn-pool-name]
```

설명:

- *archive-set-name*은 사용자가 지정된 볼륨과 연관 중인 아카이브 세트에 아카이브 세트 지정 지시어가 지정한 이름입니다.
- *copy-number*는 사용자가 지정된 볼륨과 연관 중인 복사본에 아카이브 복사 지시어가 지정한 숫자입니다. 이 숫자는 [*1-4*] 범위의 정수입니다.
- *media-type*은 부록 A. 장비 유형 용어집 및 *mcf* 매뉴얼 페이지에 나열된 2자로 된 Oracle HSM 매체 유형 식별자 중 하나입니다.
- *volume-specification*은 볼륨 일련 번호와 일치하는 하나 이상의 정규 표현식이 공백으로 구분된 목록입니다. 정규 표현식 구문에 대한 자세한 내용은 Solaris *regcmp* 매뉴얼 페이지를 참조하십시오.
- *-pool vsn-pool-name*은 하나의 단위로 지정될 수 있는 아카이브 매체 볼륨에 대해 이전에 지정된, 이름이 지정된 모음입니다. VSN(볼륨 일련 번호) 풀 지시어를 참조하십시오.

이 예제에서는 매체를 VSN 사양의 2개 행과 연관시킬 수 있는 여러 방법을 보여줍니다.

```
vsns
archiveset.1 lt VSN001 VSN002 VSN003 VSN004 VSN005
archiveset.2 lt VSN0[6-9] VSN10
archiveset.3 -pool data_pool
endvsns
```

스테이징 지시어

스테이징은 니어라인 또는 오프라인 스토리지의 파일 데이터를 다시 온라인 스토리지로 복사하는 과정을 의미합니다.

스테이저는 *samd* 데몬이 실행될 때 시작됩니다. 스테이저의 기본 동작은 다음과 같습니다.

- 스테이저가 라이브러리의 모든 드라이브를 사용하려고 시도합니다.
- 스테이지 버퍼 크기는 매체 유형에 따라 결정되고, 스테이지 버퍼가 잠기지 않습니다.
- 로그 파일은 기록되지 않습니다.
- 어느 시점에서든 활성화할 수 있는 스테이지 요청 수는 최대 1000개입니다.

/etc/opt/SUNWsamfs/stager.cmd 파일에 지시어를 삽입하여 사이트에 대한 스테이저 작업을 사용자 정의할 수 있습니다.

응용 프로그램에 오프라인 파일이 필요하면 파일이 *-n*(*never stage*) 옵션으로 아카이브되지 않은 한 해당 아카이브 복사본이 디스크 캐시에 스테이지됩니다. 파일을 응용 프로그램에서 즉시 사용하기 위해 스테이징 작업 바로 뒤에 읽기 작업이 추적되므로 전체 파일이 스테이지되기 전에 액세스를 시작할 수 있습니다.

스테이지 오류에는 매체 오류, 매체의 사용 불가, 자동화된 라이브러리의 사용 불가 등이 포함됩니다. 스테이지 오류가 반환될 경우 Oracle HSM 소프트웨어는 사용 가능한 다음 파일 복사본을 찾으려고 합니다(복사본이 있고 아카이브 복사본의 매체를 읽는 데 사용할 수 있는 장치가 있는 경우).

stager.cmd 파일

stager.cmd 파일에서 기본 동작을 대체하기 위한 지시어를 지정합니다. 스테이저를 구성하여 파일을 즉시 스테이지하거나 파일을 스테이지하지 않거나 부분적으로 스테이지하거나 다른 스테이징 작업을 지정할 수 있습니다. 예를 들어, 큰 파일의 작은 레코드에 액세스하는 응용 프로그램의 경우 never-stage 속성을 지정하는 것이 유리한데 이는 파일을 온라인으로 스테이지하지 않고 아카이브 매체에서 데이터에 직접 액세스하기 때문입니다.

이 절에서는 스테이저 지시어에 대해 설명합니다. 스테이저 지시어에 대한 추가 정보는 *stager.cmd* 매뉴얼 페이지를 참조하십시오. Oracle HSM Manager 소프트웨어를 사용하는 중이면 File System Summary 또는 File System Details 페이지에서 스테이징을 제어할 수 있습니다. 파일 시스템을 탐색하여 개별 파일의 상태를 보거나 필터를 사용하여 특정 파일을 보거나 스테이지할 특정 파일을 선택할 수 있습니다. 스테이지를 시작할 복사본을 선택하거나 시스템에서 복사본을 선택하게 할 수 있습니다.

예제에서는 가능한 모든 지시어가 설정된 후의 *stager.cmd* 파일을 보여줍니다.

```
drives=dog 1
bufsize=od 8 lock
logfile=/var/adm/stage.log
maxactive=500
```

drives: 스테이지를 위한 드라이브 수 지정

기본적으로 스테이저는 파일을 스테이지할 때 사용 가능한 모든 드라이브를 사용합니다. 스테이저가 모든 드라이브를 사용 중인 경우 아카이버의 작업에 방해가 될 수 있습니다. *drives* 지시어는 스테이저에 사용할 수 있는 드라이브 수를 지정합니다. 이 지시어의 형식은 다음과 같습니다.

```
drives=library count
```

설명:

- *library*는 *mcf* 파일에 표시된 대로 라이브러리의 패밀리 세트 이름입니다.
- *count*는 사용되는 최대 드라이브 수입니다. 기본적으로 이 숫자는 이 라이브러리에 대해 *mcf* 파일에 구성된 드라이브 수입니다.

이 예제에서는 *dog* 패밀리 세트의 라이브러리에서 드라이브 하나만 파일을 스테이지하는 데 사용되도록 지정합니다.

```
drives = dog 1
```

bufsize: 스테이지 버퍼 크기 설정

기본적으로 스테이지되는 파일은 아카이브 매체에서 디스크 캐시로 복원되기 전에 버퍼의 메모리로 읽혀집니다. *bufsize* 지시어를 사용하면 버퍼 크기를 지정하고, 선택적으로 버퍼

를 잠글 수 있습니다. 이러한 작업은 성능을 향상시킬 수 있습니다. 다양한 *number-blocks* 값을 사용해 볼 수 있습니다. 이 지시어의 형식은 다음과 같습니다.

```
bufsize= media-type number-blocks [lock]
```

설명:

- *media-type*은 부록 A. 장비 유형 용어집 및 *mcf* 매뉴얼 페이지에 나열된 2자로 된 Oracle HSM 매체 유형 식별자 중 하나입니다.
- *number-blocks*는 [*2-8192*] 범위의 정수입니다. 이 값은 *defaults.conf* 파일에 지정된 *media-type_blksize* 값으로 곱해집니다. *number-blocks*에 지정된 숫자가 높을수록 더 많은 메모리가 사용됩니다. 기본값은 *16*입니다.
- *lock*은 각 스테이징 작업의 기간 중 잠긴 버퍼 사용을 강제합니다. 이렇게 하면 각 I/O 요청에 대해 스테이징 버퍼 잠금을 설정 및 해제하는 것과 관련된 오버헤드를 방지하며 성능을 향상시켜 줍니다. *lock* 매개변수는 사용 가능한 메모리가 제한적인 시스템에서 지정될 경우 메모리 부족 조건을 일으킬 수 있습니다. 기본적으로 잠긴 버퍼는 필수가 아니며, 파일 시스템이 아카이빙 버퍼를 계속 제어합니다.

  lock 인수는 스테이지된 파일에 대해 직접 I/O가 사용으로 설정된 경우에만 효과가 있습니다. 직접 I/O를 사용으로 설정하는 방법에 대한 자세한 내용은 *setfa*, *sam_setfa* 및 *mount_samfs* 매뉴얼 페이지를 참조하십시오.

logfile: 스테이징 로그 파일 지정

Oracle HSM 소프트웨어가 파일 스테이징 이벤트 정보를 수집하고 이를 로그 파일에 기록하도록 요청할 수 있습니다. 기본적으로 로그 파일은 기록되지 않습니다. *logfile* 지시어는 스테이저가 로깅 정보를 기록할 수 있는 로그 파일을 지정합니다. 스테이저는 스테이지된 각 파일에 대해 로그 파일에 하나 이상의 라인을 기록합니다. 이 라인에는 파일의 이름, 스테이지 날짜 및 시간, VSN(볼륨 일련 번호)과 같은 정보가 포함됩니다. 이 지시어의 형식은 다음과 같습니다.

```
logfile=filename [event-list]
```

여기서 *filename*은 로그 파일의 전체 경로 이름이고 *event-list*는 기록할 이벤트 유형이 공백으로 구분된 목록입니다.

- *all*은 모든 스테이징 이벤트를 기록합니다.
- *start*는 파일의 스테이징이 시작될 때 기록합니다.
- *finish*(기본값)는 파일의 스테이징이 종료될 때 기록합니다.
- *cancel*(기본값)은 작업자가 스테이지를 취소할 때 기록합니다.
- *error*(기본값)는 스테이징 오류를 기록합니다.

다음 지시어는 */var/adm/* 디렉토리에 스테이지 로그를 만듭니다.

```
logfile=/var/adm/stage.log
```

스테이저 로그 항목이 형식은 다음과 같습니다.

```
status date time media-
type volume position.offset inode filesize filename copy user group requestor equipment-
number validation
```

설명:

- *status*는 *S*(시작), *C*(취소), *E*(오류), *F*(마침)입니다.
- *date*는 *yyyy/mm/dd* 형식의 날짜이고, 여기서 *yyyy*는 연도를 나타내는 4자리 숫자이고, *mm*은 월을 나타내는 2자리 숫자이고, *dd*는 월 중 일자를 나타내는 2자리 숫자입니다.
- *time*은 *hh:mm:ss* 형식의 시간이고, 여기서 *hh*, *mm* 및 *ss*는 각각 시, 분 및 초를 나타내는 2자리 숫자입니다.
- *media-type*은 부록 A. 장비 유형 용어집 및 *mcf* 매뉴얼 페이지에 나열된 2자로 된 Oracle HSM 매체 유형 식별자 중 하나입니다.
- *volume*은 스테이지 중인 파일이 저장된 매체의 VSN(볼륨 일련 번호)입니다.
- *position.offset*은 볼륨에서 아카이브 (*tar*) 파일의 시작 위치 및 아카이브 파일의 시작 위치에 상대되는 스테이지된 파일의 오프셋 위치를 나타내는 마침표로 구분된 16진수 숫자의 쌍입니다.
- *inode*는 마침표로 구분된 inode 번호 및 스테이지된 파일의 생성 번호입니다.
- *filesize*는 스테이지된 파일의 크기입니다.
- *filename*은 스테이지된 파일의 이름입니다.
- *copy*는 스테이지된 파일을 포함하는 복사본의 아카이브 복사 번호입니다.
- *user*는 파일을 소유하는 사용자입니다.
- *group*은 파일을 소유하는 그룹입니다.
- *requestor*는 파일을 요청한 그룹입니다.
- *equipment-number*는 파일이 스테이지된 드라이브의 *mcf* 파일에 정의된 장비 순서 번호입니다.
- *validation*은 스테이지된 파일이 검증되었거나(*V*) 또는 검증되지 않았는지(*-*) 여부를 나타냅니다.

이 예제에서는 일반적인 스테이저 로그의 일부를 보여줍니다.

```
S 2014/02/16 14:06:27 dk disk01 e.76d 2557.1759 1743132 /sam1/testdir0/filebu 1 root
 other root 0 -
F 2014/02/16 14:06:27 dk disk01 e.76d 2557.1759 1743132 /sam1/testdir0/filebu 1 root
 other root 0 -
S 2014/02/16 14:06:27 dk disk02 4.a68 1218.1387 519464 /sam1/testdir1/fileaq 1 root
 other root 0 -
S 2014/02/16 14:06:43 dk disk01 13.ba5 3179.41 750880 /sam1/testdir0/filecl 1 root
 other root 0 -
F 2014/02/16 14:06:43 dk disk01 13.ba5 3179.41 750880 /sam1/testdir0/filecl 1 root
 other root 0 -
```

maxactive: 스테이지 요청 수 지정

maxactive 지시어를 사용하면 어느 시점에서든 활성화할 수 있는 스테이지 요청 수를 지정할 수 있습니다. 이 지시어의 형식은 다음과 같습니다.

```
maxactive=number
```

여기서 *number*는 [*1-500000*] 범위의 정수입니다. 기본값은 *4000*입니다.

이 예제에서는 최대 500개의 스테이지 요청이 큐에 동시에 존재할 수 있도록 지정합니다.

```
maxactive=500
```

copysel: 스테이징 중 복사본 선택 순서 지정

스테이징 지시어 *copysel*은 파일 시스템당 스테이저 복사본 선택 시퀀스를 설정합니다.

```
copysel=selection-order
```

여기서 *selection-order*는 처음부터 마지막까지의 순서로 나열된 복사본 번호의 콜론으로 구분된 목록입니다. 기본 선택 순서는 *1:2:3:4*입니다.

자세한 내용은 *stager.cmd* 매뉴얼 페이지를 참조하십시오. 예제에서는 파일 시스템 *samfs1* 및 *samfs2*에 대한 기본값이 아닌 복사본 선택 순서를 설정하는 *stager.cmd* 파일을 보여줍니다.

```
logfile = /var/opt/SUNWsamfs/log/stager
drives = hp30 1
fs = samfs1
copysel = 4:3:2:1
fs = samfs2
copysel = 3:1:4:2
```

미리보기 요청 지시어

Oracle HSM 프로세스가 드라이브에 현재 로드되지 않은 이동식 매체 볼륨을 요청하면 요청이 미리보기 대기열에 추가됩니다. 대기열에 있는 요청은 기본적으로 FIFO(선입선출) 순서로 충족됩니다. 그러나 */etc/opt/SUNWsamfs/preview.cmd* 파일을 편집하여 기본 동작을 대체할 수 있습니다. Oracle HSM 라이브러리 제어 데몬(*sam-amld*)은 시작할 때 이러한 지시어를 읽고 중지될 때까지 사용합니다. 대기열 우선 순위를 동적으로 변경할 수 없습니다.

다음과 같은 두 가지 유형의 지시어가 있습니다.

- 전역 지시어는 파일의 위쪽에 배치되어 모든 파일 시스템에 적용됩니다.
- 파일 시스템 지시어는 *fs=directive* 형태이며 개별 파일 시스템과 관련됩니다.

다음 절에서는 *preview.cmd* 파일을 편집해서 미리보기 대기열을 제어하는 방법에 대해 설명합니다.

전역 지시어

다음은 순수한 전역 지시어입니다.

vsn_priority: 볼륨 우선순위 조정

vsn_priority 지시어는 높은 우선순위 볼륨으로 플래그가 지정된 볼륨(VSN)의 우선순위를 지정된 값만큼 늘립니다. 이 지시어의 형식은 다음과 같습니다.

```
vsn_priority=value
```

여기서 *value*는 실수입니다. 기본값은 *1000.0*입니다.

명령을 사용해서 볼륨에 높은 우선순위 플래그를 설정할 수 있습니다.

```
chmed +p media-type.volume-serial-number
```

여기서 *media-type*은 부록 A. 장비 유형 용어집 및 *mcf* 매뉴얼 페이지에 나열된 2문자 Oracle HSM 매체 유형 중 하나이며, 여기서 *volume-serial-number*는 라이브러리에서 높은 우선순위 볼륨을 고유하게 식별하는 영숫자 문자열입니다. 자세한 내용은 *chmed* 매뉴얼 페이지를 참조하십시오.

age_priority: 대기열에서의 대기 소비 시간에 대한 우선순위 조정

age_priority 지시어는 요청이 대기열에서 기다리는 시간에 따라 제공된 상대적 우선순위를 변경합니다. 예를 들어, 오래된 요청이 우선순위가 높은 새로운 요청에 밀려 무제한으로 늦춰지지 않도록 할 수 있습니다. 이 지시어는 대기열에서 보낸 시간의 상대적 가중치를 변경하는 배수를 지정합니다. 형식은 다음과 같습니다.

```
age_priority=weighting-factor
```

여기서 *weighting-factor*는 *1.0*보다 크거나, 작거나, 동일한 실수입니다. 그리고 이에 대한 설명은 다음과 같습니다.

- 값이 *1.0*보다 크면 집계 우선순위를 계산할 때 대기열에서 보낸 시간에 따라 가중치가 올라갑니다.
- 값이 *1.0*보다 작으면 총 우선순위를 계산할 때 대기열에서 보낸 시간에 따라 가중치가 내려갑니다.

- 값이 *1.0*이면 대기열에서 보낸 시간에 따른 상대적 가중치가 변경되지 않습니다.

기본값은 *1.0*입니다.

전역 및/또는 파일 시스템 특정 지시어

다음 지시어는 전역 또는 파일 시스템별 기준에 따라 적용될 수 있습니다.

- **hwm_priority**: 디스크 캐시가 거의 꽉 찼을 때 우선순위 조정
- **lwm_priority**: 디스크 캐시가 거의 비어 있을 때 우선순위 조정
- **lhwm_priority**: 디스크 캐시가 채워질 때 우선순위 조정
- **hlwm_priority**: 디스크 캐시가 비워질 때 우선순위 조정

hwm_priority: 디스크 캐시가 거의 꽉 찼을 때 우선순위 조정

hwm_priority 지시어는 파일 시스템 사용률이 고수위(*hwm*) 즉, 릴리서 프로세스가 시작되고 아카이브 매체에 복사본이 있는 파일로 점유된 디스크 공간의 재확보를 시작하는 지점을 초과할 때 아카이빙 요청과 스테이징 요청에 지정된 상대적 가중치를 조정합니다. 이 경우 아카이빙에 지정된 상대적 가중치를 늘리면 릴리스 프로세스가 스테이지된 아카이브 복사본 및 새 파일을 위해 더 많은 공간을 비울 수 있습니다. 이 지시어의 형식은 다음과 같습니다.

```
hwm_priority=weighting-factor
```

여기서 *weighting-factor*는 실수입니다. 기본값은 *0.0*입니다.

lwm_priority: 디스크 캐시가 거의 비어 있을 때 우선순위 조정

lwm_priority 지시어는 파일 시스템 사용률이 저수위(*lwm*) 즉, 릴리서 프로세스가 중지되는 지점 아래로 내려갈 때 아카이빙 요청과 스테이징 요청에 지정된 상대적 가중치를 조정합니다. 이 경우 아카이빙에 지정된 상대적 가중치를 줄이고 그 결과 스테이징 요청의 우선순위가 높아지면 디스크 캐시에 더 많은 파일이 배치되고, 매체 마운트 요구가 줄어들고, 파일 시스템 성능은 향상됩니다. 이 지시어의 형식은 다음과 같습니다.

```
lwm_priority=weighting-factor
```

여기서 *weighting-factor*는 실수입니다. 기본값은 *0.0*입니다.

lhwm_priority: 디스크 캐시가 채워질 때 우선순위 조정

hlwm_priority 지시어는 디스크 캐시가 채워지고 캐시 사용률이 저수위 및 고수위(*lwm* 및 *hwm*) 사이에 있을 때 아카이빙 요청과 스테이징 요청에 지정된 상대적 가중치를 조정합니다. 이 경우 아카이빙에 지정된 상대적 가중치를 늘리면 릴리스 프로세스가 스테이지된 아카이브 복사본 및 새 파일을 위해 더 많은 공간을 비울 수 있습니다. 이 지시어의 형식은 다음과 같습니다.

```
lhwm_priority=weighting-factor
```

여기서 *weighting-factor*는 실수입니다. 기본값은 *0.0*입니다.

hlwm_priority: 디스크 캐시가 비워질 때 우선순위 조정

hlwm_priority 지시어는 디스크 캐시가 비워지고 캐시 사용률이 저수위 및 고수위(*hwm* 및 *lwm*) 사이에 있을 때 아카이빙 요청과 스테이징 요청에 지정된 상대적 가중치를 조정합니다. 이 경우 아카이빙에 지정된 상대적 가중치를 줄이고 그 결과 스테이징 요청의 우선순위가 높아지면 디스크 캐시에 더 많은 파일이 배치되고, 매체 마운트 요구가 줄어들고, 파일 시스템 성능은 향상됩니다. 이 지시어의 형식은 다음과 같습니다.

```
hlwm_priority=weighting-factor
```

여기서 *weighting-factor*는 실수입니다. 기본값은 *0.0*입니다.

샘플 preview.cmd 파일

지정된 매체 마운트 요청에 대한 집계 우선순위는 다음 공식에 따라 모든 가중 인자로 설정된 값을 사용해서 결정됩니다.

```
priority = vsn_priority + wm_priority + (age_priority * time-waiting-in-queue)
```

여기서 *wm_priority*는 현재 적용된 수위 우선순위(*hwm_priority*, *lwm_priority*, *hlwm_priority* 또는 *lhwm_priority*)이고 *time-waiting-in-queue*는 볼륨 요청이 대기열에 들어간 시간(초)입니다. 우선순위 계산에 대한 자세한 설명은 *preview.cmd* 매뉴얼 페이지의 *PRIORITY CALCULATION* 섹션을 참조하십시오.

데이터 액세스가 매우 중요하거나 이동식 매체 드라이브 공급이 부족한 일부 특수한 경우에는 *preview.cmd* 파일에서 지시어를 사용하여 운영 요구 사항 및 사용 가능한 리소스에 보다 효과적인 파일 시스템 작업을 찾을 수 있습니다. 저장된 데이터의 무결성은 *preview.cmd* 파일의 설정에 영향을 받지 않으므로, 아카이빙 요청과 스테이징 요청 사이의 적절한 균형을 찾을 때까지 자유롭게 실험을 해볼 수 있습니다.

기본 우선순위 계산을 조정해야 하는 이유에는 다음이 포함될 수 있습니다.

- 사용자 및 응용 프로그램이 파일에 액세스할 때 이를 사용할 수 있도록 아카이브 요청 전에 스테이징 요청을 처리해야 합니다.
- 파일 시스템이 거의 채워졌을 때는 아카이브 요청이 가장 높은 우선순위를 갖도록 해야 합니다.

아래의 샘플 *preview.cmd* 파일에서는 위에 설명한 조건들을 보여줍니다.

```
# Use default weighting value for vsn_priority:
vsn_priority=1000.0
age_priority = 1.0
# Insure that staging requests are processed before archive requests:
```

```
lwm_priority = -200.0
lhwm_priority = -200.0
hlwm_priority = -200.0
# Insure that archive requests gain top priority when a file system is about to fill
 up:
hwm_priority = 500.0
```

lwm_priority, *lhwm_priority* 및 *hlwm_priority*에 음수 가중치 값을 사용하면 디스크 캐시에서 공간을 사용할 수 있을 때마다 아카이브 요청보다 스테이지 요청의 우선순위가 더 높도록 보장하여 항상 요청 시 데이터에 액세스할 수 있습니다. 일부 요청이 대기열에서 100초 이상 유지되고 파일 시스템이 저수위 아래에 있으면 다음과 같이 됩니다.

- 우선순위 볼륨에 대한 아카이빙 마운트 요청이 *1000+(-200)+(1x100)=900*의 집계 우선순위를 갖습니다.
- 우선순위 볼륨에 대한 스테이징 마운트 요청은 *1000+0+(1x100)=1100*의 집계 우선순위를 갖습니다.
- 비우선순위 볼륨에 대한 스테이징 마운트 요청은 *0+0+(1x100)=100*의 집계 우선순위를 갖습니다.

하지만 디스크 캐시가 거의 용량에 도달하면 아카이빙 요청이 우선순위를 갖습니다. 파일 시스템이 채워질 때 아카이브되는 파일 수가 너무 적으면 아카이브된 파일을 스테이징하거나 새 파일을 입수할 때 사용할 수 있는 공간이 부족해집니다. 일부 요청이 대기열에서 100초 이상 유지되고 파일 시스템이 고수위 위에 있으면 다음과 같이 됩니다.

- 우선순위 볼륨에 대한 아카이빙 마운트 요청이 *1000+500+(1x100)=1600*의 집계 우선순위를 갖습니다.
- 우선순위 볼륨에 대한 스테이징 마운트 요청은 *1000+0+(1x100)=1100*의 집계 우선순위를 갖습니다.
- 비우선순위 볼륨에 대한 스테이징 마운트 요청은 *0+0+(1x100)=100*의 집계 우선순위를 갖습니다.

부록 E. 제품 접근성 기능

저시력, 맹인, 색맹 또는 기타 시각 장애를 가진 사용자는 명령줄 인터페이스를 통해 Oracle Hierarchical Storage Manager and StorageTek QFS Software (Oracle HSM)에 액세스할 수 있습니다. 이 텍스트 기반 인터페이스는 화면 읽기 프로그램과 호환되며 모든 기능을 키보드로 제어합니다.

용어집

이 용어집에서는 Oracle HSM 소프트웨어 및 파일 시스템과 관련된 용어에 초점을 맞춥니다. 업계 표준 정의는 *http://www.snia.org/education/dictionary/*에서 Storage Networking Industry Association이 제공하는 사전을 참조하십시오.

커널 기본적인 운영체제 기능을 제공하는 프로그램입니다. UNIX 커널은 프로세스를 만들고 관리하며, 파일 시스템 액세스 기능, 일반적인 보안 및 통신 기능을 제공합니다.

addressable storage(주소 지정 가능한 스토리지) Oracle HSM 파일 시스템을 통해 사용자가 참조하는 온라인, 니어라인, 오프사이트 및 오프라인 스토리지를 포함하는 스토리지 공간입니다.

admin set ID(관리 세트 ID) 공통 특성을 공유하는 사용자 및/또는 그룹의 스토리지 관리자 정의 세트입니다. 관리 세트는 대개 여러 그룹의 사용자가 관여하고 여러 파일 및 디렉토리에 걸친 프로젝트에 대한 스토리지를 관리하기 위해 만들어집니다.

archival media(아카이브 매체) 아카이브 파일이 쓰여지는 매체입니다. 아카이브 매체에는 이동식 테이프 또는 마그네틱-옵티컬 카트리지 및 아카이브를 위해 구성된 디스크 파일 시스템이 모두 포함됩니다.

archival storage(아카이브 스토리지) 아카이브 매체에 만든 데이터 스토리지 공간입니다.

archive set(아카이브 세트) 아카이브 세트는 아카이브할 파일 그룹을 식별하며 파일은 크기, 소유권, 그룹, 디렉토리 위치와 관련한 공통 기준을 공유합니다. 아카이브 세트는 여러 파일 시스템 그룹에 걸쳐 정의할 수 있습니다.

archiver(아카이버) 이동식 카트리지에 파일 복사를 자동으로 제어하는 아카이브 프로그램입니다.

associative staging(연관 스테이징) 그룹의 한 멤버가 스테이징된 경우 관련 파일의 그룹을 스테이징합니다. 파일이 동일한 디렉토리에 있고 함께 자주 사용되는 경우 파일 소유자는 Oracle HSM 연관 스테이징 파일 속성을 설정하여 연관시킬 수 있습니다. 그런 다음 그룹의 한 파일이 응용 프로그램에서 액세스될 때 그룹의 파일이 오프라인인 경우 Oracle HSM는 전체 그룹을 아카이브 매체에서 디스크 캐시로 스테이징합니다. 그러면 모든 필요한 파일을 동시에 다시 사용할 수 있게 됩니다.

audit (full)(전체 감사) VSN을 확인하기 위해 카트리지를 로드하는 프로세스입니다. 마그네틱-옵티컬 카트리지의 경우, 용량 및 공간 정보가 확인되고 자동화 라이브러리의 카탈로그에 입력됩니다. **volume serial number(VSN, 볼륨 일련 번호)**를 참조하십시오.

automated library(자동화된 라이브러리) 작업자 개입 없이 이동식 매체 카트리지를 자동으로 로드 및 언로드하도록 설계된 로봇으로 제어되는 장치입니다. 자동화 라이브러리에는 하나 이상의

	드라이브와 카트리지를 스토리지 슬롯 및 드라이브에서 이동하는 전송 메커니즘이 포함됩니다.
backup(백업)	의도치 않은 손실을 막기 위한 용도의 파일 모음 스냅샷입니다. 백업에는 파일의 속성 및 연관된 데이터가 모두 포함됩니다.
block allocation map(블록 할당 맵)	디스크에서 각 사용 가능한 스토리지 블록을 나타내고 블록이 사용 중인지 또는 사용 가능한지 나타내는 비트맵입니다.
block size(블록 크기)	하드 디스크 또는 마그네틱 테이프 카트리지와 같은 블록 장치에서 가장 작은 주소 지정 가능한 데이터 단위의 크기입니다. 디스크 장치에서는 대개 512바이트인 섹터 크기와 같습니다.
cartridge(카트리지)	마그네틱 테이프 또는 옵티컬 매체와 같은 데이터 스토리지 매체에 대한 컨테이너입니다. **volume(볼륨)**, 테이프 또는 매체 조각이라고도 합니다. **volume serial number(VSN, 볼륨 일련 번호)**를 참조하십시오.
catalog(카탈로그)	자동화 라이브러리에서 이동식 매체 볼륨의 레코드입니다. 각 자동화 라이브러리에 대해 하나의 카탈로그가 있고, 사이트에는 모든 자동화 라이브러리에 대해 하나의 내역기가 있습니다. 볼륨은 **volume serial number(VSN, 볼륨 일련 번호)**를 사용하여 식별 및 추적됩니다.
client-server(클라이언트-서버)	한 사이트의 프로그램이 다른 사이트의 프로그램에 요청을 보내고 응답을 기다리는 분산된 시스템의 상호 작용 모델입니다. 요청 프로그램을 클라이언트라고 합니다. 응답을 충족시키는 프로그램을 서버라고 합니다.
connection(연결)	안정적인 스트림 전달 서비스를 제공하는 두 프로토콜 모듈 사이의 경로입니다. TCP 연결은 한 컴퓨터의 TCP 모듈에서 다른 컴퓨터의 TCP 모듈로 확장됩니다.
data device(데이터 장치)	파일 시스템에서 파일 데이터가 저장되는 장치 또는 장치 그룹입니다.
DAU	**disk allocation unit(DAU, 디스크 할당 단위)**를 참조하십시오.
device logging(장치 로깅)	Oracle HSM 파일 시스템을 지원하는 하드웨어 장치에 대한 특정 오류 정보를 제공하는 구성 가능한 기능입니다.
device scanner(장치 스캐너)	모든 수동으로 마운트된 이동식 장치의 유무를 정기적으로 모니터하고 사용자 또는 기타 프로세스가 요청할 수 있는 마운트된 카트리지의 유무를 감지하는 소프트웨어입니다.
direct access(직접 액세스)	니어라인 파일을 아카이브 매체에서 직접 액세스할 수 있고 디스크 캐시로 검색할 필요가 없도록 지정하는 파일 속성(스테이징 안함)입니다.
direct attached library(직접 연결 라이브러리)	SCSI 인터페이스를 사용하여 서버에 직접 연결된 자동화 라이브러리입니다. SCSI 연결 라이브러리는 Oracle HSM 소프트웨어에 의해 직접 제어됩니다.

direct I/O(직접 I/O)

큰 블록이 정렬된 순차적 I/O에 사용되는 속성입니다. *setfa* 명령의 *-D* 옵션은 직접 I/O 옵션입니다. 이 옵션은 파일 또는 디렉토리에 대해 직접 I/O 속성을 설정합니다. 디렉토리에 적용되면 직접 I/O 속성이 상속됩니다.

directory(디렉토리)

파일 시스템 내에서 다른 파일 및 디렉토리를 가리키는 파일 데이터 구조입니다.

disk allocation unit(DAU, 디스크 할당 단위)

Oracle HSM 파일 시스템에서 쓰여진 데이터의 크기에 상관 없이 각 I/O 작업이 소비하는 최소 연속 공간 크기입니다. 따라서 데이터 할당 단위는 제공된 크기의 파일을 전송할 때 필요한 최소 I/O 작업의 수를 결정합니다. 디스크 장치의 **block size(블록 크기)** 배수이어야 합니다.

디스크 할당 단위는 선택한 Oracle HSM 장치 유형 및 사용자 요구 사항에 따라 달라집니다. *md* 장치 유형은 이중 할당 단위를 사용합니다. 처음 8번 쓰기에 대한 DAU는 *4*KB이고 후속 쓰기에 대해서는 사용자 지정 *16*, *32* 또는 *64*KB이므로 작은 파일은 작은 블록에 기록되지만 더 큰 파일은 더 큰 블록에 기록됩니다. *mr* 및 **striped group(스트라이프 그룹)** 장치 유형은 *[8-65528]*KB 범위 내에서 *8*의 증분으로 조정할 수 있는 DAU를 사용합니다. 따라서 파일은 큰 균일 크기의 파일 크기에 근접하는 큰 균일 블록으로 쓰여집니다.

disk buffer(디스크 버퍼)

Sun SAM-Remote 구성에서 클라이언트에서 서버로 데이터 아카이브에 사용되는 서버 시스템의 버퍼입니다.

disk cache(디스크 캐시)

온라인 디스크 캐시와 아카이브 매체 사이에서 데이터 파일을 만들고 관리하는 데 사용되는 파일 시스템 소프트웨어의 디스크 상주 부분입니다. 개별 디스크 분할 영역 또는 전체 디스크를 디스크 캐시로 사용할 수 있습니다.

disk space threshold(디스크 공간 임계값)

관리자가 정의하는 디스크 캐시 사용률의 최대 또는 최소 레벨입니다. 릴리서는 이와 같이 미리 정의된 디스크 공간 임계값을 기준으로 디스크 캐시 사용률을 제어합니다.

disk striping(디스크 스트라이핑)

여러 디스크에 걸쳐 파일을 기록하는 프로세스로, 액세스 성능이 높아지고 전체적인 스토리지 용량이 증가합니다. **striping(스트라이핑)**을 참조하십시오.

drive(드라이브)

이동식 매체 볼륨 사이에 데이터를 전송하기 위한 메커니즘입니다.

Ethernet(이더넷)

패킷 스위칭 로컬 영역 네트워크 기술입니다.

extent array(익스텐트 어레이)

파일에 지정된 각 데이터 블록의 디스크 위치를 정의하는 파일 inode 내의 어레이입니다.

family device set(패밀리 장치 세트)

family set(패밀리 세트)를 참조하십시오.

family set(패밀리 세트)

디스크 모음이나 자동화된 라이브러리 내의 드라이브와 같은 독립된 물리적 장치의 논리적 그룹입니다. 또한 **storage family set(스토리지 패밀리 세트)**를 참조하십시오.

FDDI

FDDI(Fiber-distributed Data Interface)는 최고 200km(124마일)까지 범위를 확장할 수 있는 로컬 영역 네트워크의 데이터 전송 표준입니다. FDDI 프로토콜은 토큰 링 프로토콜을 기반으로 합니다.

Fibre Channel(광 섬유 채널)

장치 사이에 고속 직렬 통신을 지정하는 ANSI 표준입니다. 광 섬유 채널은 SCSI-3에서 버스 아키텍처 중 하나로 사용됩니다.

file system(파일 시스템)

파일 및 디렉토리의 계층적 모음입니다.

file-system-specific directives(파일 시스템 특정 지시어)

archiver.cmd 파일에서 전역 지시어 다음에 오는 아카이버 및 릴리서 지시어는 특정 파일 시스템에 따라 다르고 *fs =*로 시작됩니다. 파일 시스템 특정 지시어는 다음 **fs =** 지시어 라인이 오거나 파일의 끝에 도달할 때까지 적용됩니다. 여러 지시어가 파일 시스템에 영향을 미칠 경우 파일 시스템 특정 지시어는 전역 지시어보다 우선합니다.

ftp

FTP(File Transfer Protocol)는 두 호스트 사이에 파일을 전송하기 위한 인터넷 프로토콜입니다. 보다 안전한 대체 프로토콜은 **sftp**를 참조하십시오.

global directives(전역 지시어)

모든 파일 시스템에 적용되고 첫번째 **fs=** 라인 앞에 나타나는 아카이버 및 릴리서 지시어입니다.

grace period(유예 기간)

quota(할당량)에서 파일 시스템이 지정된 사용자, 그룹 및/또는 **admin set ID(관리 세트 ID)**에 속하는 총 파일 크기가 쿼터에 지정된 **soft limit(소프트 한계)**을 초과하도록 허용하는 시간입니다.

hard limit(하드 한계)

quota(할당량)에서 지정된 사용자, 그룹 및/또는 **admin set ID(관리 세트 ID)**가 소비할 수 있는 스토리지 리소스의 절대 최대 용량입니다. **soft limit(소프트 한계)**를 참조하십시오.

high-water mark(고수위)

1. 아카이브 파일 시스템에서 Oracle HSM 파일 시스템이 이전에 아카이브된 파일을 디스크에서 삭제하면서 릴리서 프로세스를 시작하는 디스크 캐시 사용률입니다. 제대로 구성된 고수위는 파일 시스템이 새 파일 및 새로 스테이징된 파일에 사용 가능한 공간을 항상 충분히 유지하도록 합니다. 자세한 내용은 *sam-releaser* 및 *mount_samfs* 매뉴얼 페이지를 참조하십시오. **low-water mark(저수위)**와 비교하십시오.
2. 아카이브 파일 시스템의 일부인 이동식 매체 라이브러리에서 리사이클러 프로세스를 시작하는 매체 캐시 사용률입니다. 리사이클링은 현재 데이터의 일부 가득 찬 볼륨을 비워 새 매체로 교체하거나 다시 레이블을 지정할 수 있도록 합니다.

historian(내역기)

Oracle HSM 내역기가 */etc/opt/SUNWsamfs/mcf* 파일에 정의된 자동화 매체 라이브러리에서 내보낸 볼륨의 카탈로그입니다. 기본적으로 Oracle HSM 파일 시스템 호스트의 */var/opt/SUNWsamfs/catalog/historian*에 위치합니다. 자세한 내용은 Oracle HSM *historian* 매뉴얼 페이지를 참조하십시오.

hosts file(hosts 파일)

hosts 파일에는 공유 파일 시스템의 모든 호스트 목록이 포함됩니다. 파일 시스템을 Oracle HSM 공유 파일 시스템으로 초기화하는 경우 파일 시

	스템이 만들어지기 전에 hosts 파일 */etc/opt/SUNWsamfs/hosts.fs-name*을 만들어야 합니다. *sammkfs* 명령은 파일 시스템을 만들 때 hosts 파일을 사용합니다. *samsharefs* 명령을 사용하여 나중에 hosts 파일의 내용을 바꾸거나 업데이트할 수 있습니다.
indirect block(간접 블록)	스토리지 블록의 목록을 포함하는 디스크 블록입니다. 파일 시스템에는 최대 세 레벨의 간접 블록이 있습니다. 첫번째 레벨 간접 블록은 데이터 스토리지에 사용되는 블록 목록을 포함합니다. 두번째 레벨 간접 블록은 첫번째 레벨 간접 블록 목록을 포함합니다. 세번째 레벨 간접 블록은 두번째 레벨 간접 블록 목록을 포함합니다.
inode	인덱스 노드입니다. 파일을 기술하기 위해 파일 시스템에서 사용되는 데이터 구조입니다. inode는 이름 이외의 파일과 관련된 모든 속성을 기술합니다. 속성에는 소유권, 액세스, 권한, 크기 및 디스크 시스템의 파일 위치가 포함됩니다.
inode file(inode 파일)	파일 시스템에 상주하는 모든 파일에 대한 inode 구조를 포함하는 파일 시스템의 특수 파일(*.inodes*)입니다. inode 길이는 512바이트입니다. inode 파일은 파일 시스템의 파일 데이터에서 분리된 메타데이터 파일입니다.
LAN	로컬 영역 네트워크입니다.
lease(임대)	지정된 시간 동안 파일에서 작업을 수행할 클라이언트 호스트 권한을 부여하는 기능입니다. 메타데이터 서버는 각 클라이언트 호스트에게 임대를 부여합니다. 파일 작업을 계속 수행할 수 있도록 필요에 따라 임대를 갱신할 수 있습니다.
library catalog(라이브러리 카탈로그)	**catalog(카탈로그)**를 참조하십시오.
library(라이브러리)	**automated library(자동화된 라이브러리)**를 참조하십시오.
local file system(로컬 파일 시스템)	Sun Cluster 시스템의 한 노드에 설치되고 다른 노드에서 별로 사용되지 않는 파일 시스템입니다. 또한 서버에 설치된 파일 시스템입니다.
low-water mark(저수위)	아카이브 파일 시스템에서 Oracle HSM 파일 시스템이 릴리서 프로세스를 중지하고 이전에 아카이브된 파일을 디스크에서 삭제를 중지하는 디스크 캐시 사용률입니다. 제대로 구성된 저수위는 파일 시스템이 최상의 성능을 위해 가능한 많은 파일을 캐시에 유지하고 새 파일 및 새로 스테이징된 파일에 사용 가능한 공간을 유지하도록 합니다. 자세한 내용은 *sam-releaser* 및 *mount_samfs* 매뉴얼 페이지를 참조하십시오. **high-water mark(고수위)**와 비교하십시오.
LUN	Logical Unit Number(논리 장치 번호)의 약어입니다.
mcf	Master Configuration File(마스터 구성 파일)의 약어입니다. 파일 시스템 환경에서 장치 사이의 관계(토폴로지)를 정의하는, 초기화 시 읽는 파일입니다.

media recycling(매체 재활용) 활성 파일이 적은 아카이브 매체를 재활용하거나 재사용하는 프로세스입니다.

media(매체) 테이프 또는 광 디스크 카트리지입니다.

metadata device(메타데이터 장치) 파일 시스템 메타데이터가 저장되는 장치(예: 솔리드 스테이트 디스크 또는 미러 장치)입니다. 파일 데이터 및 메타데이터를 별도의 장치에 두면 성능이 향상될 수 있습니다. *mcf* 파일에서 메타데이터 장치는 *ma* 파일 시스템 내에서 *mm* 장치로 선언됩니다.

metadata(메타데이터) 데이터에 대한 데이터입니다. 메타데이터는 디스크에서 파일의 정확한 데이터 위치를 찾는 데 사용되는 인덱스 정보입니다. 파일, 디렉토리, 액세스 제어 목록, 심볼릭 링크, 이동식 매체, 세그먼트화된 파일 및 세그먼트화된 파일의 인덱스에 대한 정보로 구성됩니다.

mirror writing(미러 쓰기) 단일 디스크 실패로부터 손실을 막기 위해 별도의 디스크 세트에 두 개의 파일 복사본을 유지하는 프로세스입니다.

mount point(마운트 지점) 파일 시스템이 마운트되는 디렉토리입니다.

multireader file system(다중 읽기 파일 시스템) 다중 호스트에 마운트될 수 있는 파일 시스템을 지정할 수 있는 단일 쓰기, 다중 읽기 기능입니다. 여러 호스트가 파일 시스템을 읽을 수 있지만, 하나의 호스트만 파일 시스템에 쓸 수 있습니다. 다중 리더는 *mount* 명령에서 *-o reader* 옵션으로 지정됩니다. 단일 쓰기 호스트는 *mount* 명령에서 *-o writer* 옵션으로 지정됩니다. 자세한 내용은 *mount_samfs* 매뉴얼 페이지를 참조하십시오.

name space(이름 공간) 파일, 속성 및 스토리지 위치를 식별하는 파일 모음의 메타데이터 부분입니다.

nearline storage(니어라인 스토리지) 액세스하려면 먼저 로봇 마운트가 필요한 이동식 매체 스토리지입니다. 니어라인 스토리지는 일반적으로 온라인 스토리지보다 저렴하지만 액세스 시간이 다소 길어집니다.

network attached automated library(네트워크 연결식 자동화된 라이브러리) StorageTek, ADIC/Grau, IBM, Sony 등의 공급업체에서 제공한 소프트웨어 패키지를 사용하여 제어되는 라이브러리입니다. QFS 파일 시스템은 자동화된 라이브러리용으로 특별히 설계된 Oracle HSM 매체 교환기 데몬을 사용하여 공급업체 소프트웨어와 상호 작용합니다.

NFS 네트워크 파일 시스템(Network File System)의 약어로 이기종 네트워크 환경에서 원격 파일 시스템에 대한 투명한 액세스를 제공하는 파일 시스템입니다.

NIS 네트워크 정보 서비스(Network Information Service)의 약어로 네트워크의 시스템 및 사용자에 대한 주요 정보를 포함하는 분산된 네트워크 데이터베이스입니다. NIS 데이터베이스는 마스터 서버 및 모든 슬레이브 서버에 저장됩니다.

offline storage(오프라인 스토리지) 로드를 위해 작업자 개입이 필요한 스토리지입니다.

offsite storage(오프사이트 스토리지) 서버와 원격으로 떨어져서 재해 복구에 사용되는 스토리지입니다.

online storage(온라인 스토리지) 디스크 캐시 스토리지와 같이 즉시 사용 가능한 스토리지입니다.

Oracle HSM

1. Oracle Hierarchical Storage Manager에 대한 일반 약어입니다.
2. 아카이브용으로 구성되고 Oracle HSM 소프트웨어에서 관리되는 **QFS** 파일 시스템을 설명하는 형용사입니다.

partition(분할 영역) 장치의 일부분 또는 광자기 카트리지의 한쪽 면입니다.

preallocation(미리 할당) 파일을 쓰기 위해 디스크 캐시에 연속 공간을 예약하는 프로세스입니다. 미리 할당은 제로 크기 파일에만 지정할 수 있습니다. 자세한 내용은 *setfa* 매뉴얼 페이지를 참조하십시오.

pseudo device(의사 장치) 연결된 하드웨어가 없는 소프트웨어 부속 시스템 또는 드라이버입니다.

QFS Oracle HSM QFS 소프트웨어 제품을 가리키며, 그 자체로 또는 Oracle Hierarchical Storage Manager에서 제어하는 아카이빙 파일 시스템으로 사용할 수 있는 고성능 고용량 UNIX 파일 시스템입니다.

qfsdump **samfsdump(qfsdump)**을 참조하십시오.

qfsrestore **samfsrestore(qfsrestore)**을 참조하십시오.

quota(할당량) 지정된 사용자, 그룹 또는 **admin set ID(관리 세트 ID)**가 소비할 수 있는 스토리지 리소스 용량입니다. **hard limit(하드 한계)** 및 **soft limit(소프트 한계)**를 참조하십시오.

RAID RAID(Redundant array of independent disks)입니다. 여러 독립된 디스크를 사용하여 파일을 안정적으로 저장하는 디스크 기술입니다. 단일 디스크 실패로부터 데이터 손실을 막을 수 있고, 내결함성 디스크 환경을 제공할 수 있으며, 개별 디스크보다 높은 처리량을 제공할 수 있습니다.

recovery point(복구 지점) Oracle HSM 파일 시스템에 대한 메타데이터의 특정 시점 백업 복사본을 저장하는 압축 파일입니다.

사용자 파일의 의도치 않은 삭제부터 전체 파일 시스템의 재해 손실에 이르기까지 모든 데이터 손실 발생 시 관리자는 파일 또는 파일 시스템이 영향을 받지 않은 마지막 복구 지점을 찾아 파일 또는 파일 시스템의 마지막으로 알려진 양호한 상태로 거의 즉시 복구할 수 있습니다. 그런 다음 관리자는 해당 시점에 기록된 메타데이터를 복원하고 메타데이터에 나타난 파일을 아카이

	브 매체에서 디스크 캐시로 스테이징하거나 사용자 및 응용 프로그램이 액세스할 때 필요에 따라 파일 시스템에서 파일을 스테이징할 수 있습니다.
recycler(리사이클러)	만료된 아카이브 복사본이 차지하고 있는 카트리지의 공간을 확보하는 Oracle HSM 유틸리티입니다.
regular expression(정규 표현식)	파일 이름 및 구성 파일과 같이 다른 문자열을 검색, 선택 및 편집하기 위해 마련된 표준화된 패턴 일치 언어의 문자열입니다. Oracle HSM 파일 시스템 작업에서 사용되는 정규 표현식 구문에 대한 자세한 내용은 Oracle HSM Solaris *regex* 및 *regcmp* 매뉴얼 페이지를 참조하십시오.
release priority(릴리스 우선 순위)	파일 시스템의 파일이 아카이브된 후 릴리스되는 우선 순위입니다. 릴리스 우선 순위는 등록 정보의 다양한 가중치를 곱한 다음 결과를 더하여 계산됩니다.
releaser(릴리서)	더 많은 디스크 캐시 공간을 사용할 수 있도록 아카이브된 파일을 식별하고 디스크 캐시 복사본을 릴리스하는 Oracle HSM 구성 요소입니다. 릴리서는 상한 및 하한 임계값에 따라 온라인 디스크 스토리지의 용량을 자동으로 조절합니다.
remote procedure call(원격 프로시저 호출)	**RPC**를 참조하십시오.
removable media file(이동식 매체 파일)	마그네틱 테이프 또는 옵티컬 디스크 카트리지와 같이 이동식 매체 카트리지에 상주하는 위치에서 직접 액세스할 수 있는 특수한 사용자 파일 유형입니다. 아카이브 및 스테이지 파일 데이터를 쓰는 데도 사용됩니다.
robot(로봇)	스토리지 슬롯과 드라이브 사이에서 카트리지를 이동하는 **automated library(자동화된 라이브러리)** 구성 요소입니다. **transport(전송)**라고도 합니다.
round-robin(라운드 로빈)	전체 파일이 순차적으로 논리 디스크에 쓰여지는 데이터 액세스 방식입니다. 단일 파일이 디스크에 쓰여질 때 전체 파일이 첫번째 논리 디스크에 쓰여집니다. 두번째 파일은 그 다음 논리 디스크에 쓰여지는 방식으로 이어집니다. 각 파일의 크기는 I/O의 크기를 결정합니다. **disk striping(디스크 스트라이핑)** 및 **striping(스트라이핑)**을 참조하십시오.
RPC	Remote Procedure Call(원격 프로시저 호출)의 약어입니다. NFS에서 사용자 정의 네트워크 데이터 서버를 구현하기 위해 사용되는 기본 데이터 교환 메커니즘입니다.
SAM	Oracle Hierarchical Storage Manager 제품의 이전 이름인 Storage Archive Manager에 대한 일반 약어입니다.
SAM-QFS	1. Oracle Hierarchical Storage Manager 제품의 이전 버전에 대한 일반 약어입니다.

	2. 아카이브용으로 구성되고 Oracle HSM 소프트웨어에서 관리되는 **QFS** 파일 시스템을 설명하는 형용사입니다.
SAM-Remote client(SAM-Remote 클라이언트)	수많은 의사 장치를 포함하는 클라이언트 데몬이 설치된 Oracle HSM 시스템으로, 자체 라이브러리 장치를 가질 수도 있습니다. 클라이언트는 하나 이상의 아카이브 복사본에 대해 아카이브 매체용 SAM-Remote 서버에 의존합니다.
SAM-Remote server(SAM-Remote 서버)	전체 용량 Oracle HSM 스토리지 관리 서버 및 SAM-Remote 클라이언트 사이에 공유할 라이브러리를 정의하는 SAM-Remote 서버 데몬입니다.
samfsdump(qfsdump)	제공된 파일 그룹에 대한 제어 구조 덤프를 만들고 모든 제어 구조 정보를 복사하는 프로그램입니다. 일반적으로 파일 데이터는 복사하지 않습니다. *-U* 옵션을 사용하면 이 명령은 데이터 파일도 복사합니다. Oracle Hierarchical Storage Manager 패키지가 설치되지 않은 경우 이 명령을 *qfsdump*라고 합니다.
samfsrestore(qfsrestore)	제어 구조 덤프에 대한 inode 및 디렉토리 정보를 복원하는 프로그램입니다. 또한 **samfsdump(qfsdump)**를 참조하십시오.
SAN	Storage Area Network(스토리지 영역 네트워크)의 약어입니다.
SCSI	Small Computer System Interface(소형 컴퓨터 시스템 인터페이스)의 약어로, 디스크 및 테이프 드라이브나 자동화된 라이브러리와 같은 주변 장치에 흔히 사용되는 전기 통신 사양입니다.
seeking(탐색)	랜덤 액세스 I/O 작업 중 한 디스크 위치에서 다른 디스크 위치로 디스크 장치의 읽기/쓰기 헤드를 이동하는 것입니다.
sftp	Secure File Transfer Protocol의 약어로 **ftp**를 기반으로 하는 **ssh**의 보안 구현입니다.
shared hosts file(공유 hosts 파일)	공유 파일 시스템을 만들 때 시스템은 hosts 파일에서 메타데이터 서버의 공유 hosts 파일로 정보를 복사합니다. **`samsharefs -u`** 명령을 실행할 때 이 정보를 업데이트합니다.
Small Computer System Interface(소형 컴퓨터 시스템 인터페이스)	**SCSI**를 참조하십시오.
soft limit(소프트 한계)	**quota(할당량)**에서 지정된 사용자, 그룹 및/또는 **admin set ID(관리 세트 ID)**가 무한대 기간 동안 채울 수 있는 스토리지 공간의 최대 용량입니다. 파일은 최대 하드 한계까지 소프트 한계에서 허용하는 것보다 많은 공간을 사용할 수 있지만 쿼터에 정의된 짧은 **grace period(유예 기간)** 동안만 가능합니다. **hard limit(하드 한계)**를 참조하십시오.

ssh

Secure Shell의 약어로 안전한 원격 명령줄 로그인 및 명령 실행을 허용하는 암호화된 네트워크 프로토콜입니다.

staging(스테이징)

니어라인 또는 오프라인 파일을 아카이브 스토리지에서 온라인 스토리지로 다시 복사하는 프로세스입니다.

Storage Archive Manager

Oracle Hierarchical Storage Manager 제품의 이전 이름입니다.

storage family set(스토리지 패밀리 세트)

단일 논리 장치로 통칭되는 디스크 세트입니다.

storage slots(스토리지 슬롯)

드라이브에서 사용 중인 아닐 때 카트리지를 저장하는 자동화된 라이브러리 안의 위치입니다.

stripe size(스트라이프 크기)

쓰기가 다음 스트라이프 장치로 이동하기 전에 할당할 디스크 할당 단위(DAU)의 수입니다. *`stripe=0`* 마운트 옵션이 사용되는 경우 파일 시스템은 스트라이프 액세스가 아닌 라운드 로빈 액세스를 사용합니다.

striped group(스트라이프 그룹)

`mcf` 파일에서 하나 이상의 *`gXXX`* 장치로 정의되는 파일 시스템 내의 장치 모음입니다. 스트라이프 그룹은 하나의 논리적 장치로 취급되고 언제나 디스크 할당 단위(DAU)와 동일한 크기로 스트라이프됩니다.

striping(스트라이핑)

파일이 인터레이스 방식으로 논리 디스크에 동시에 쓰여지는 데이터 액세스 방식입니다. Oracle HSM 파일 시스템은 스트라이프 그룹을 사용하는 "하드 스트라이핑"과 *`stripe=x`* 마운트 매개변수를 사용하는 "소프트 스트라이핑"의 두 유형의 스트라이핑을 제공합니다. 하드 스트라이핑은 파일 시스템이 설정될 때 활성화되며 *`mcf`* 파일 안에 스트라이프 그룹이 정의되어야 합니다. 소프트 스트라이핑은 *`stripe=x`* 마운트 매개변수를 통해 활성화되며 파일 시스템 또는 개별 파일에 대해 변경될 수 있습니다. *`stripe=0`*을 설정하면 사용 안함으로 설정됩니다. 파일 시스템이 동일한 수의 요소를 갖는 다중 스트라이프 그룹으로 구성되는 경우 하드 및 소프트 스트라이핑을 둘 다 사용할 수 있습니다. **round-robin(라운드 로빈)**을 참조하십시오.

SUNW.qfs

Oracle HSM 공유 파일 시스템을 지원하는 Solaris Cluster 리소스 유형입니다. *`SUNW.qfs`* 리소스 유형은 공유 파일 시스템의 메타데이터 서버(MDS)에 대한 페일오버 리소스를 정의합니다.

superblock(수퍼 블록)

파일 시스템의 기본적인 매개변수를 정의하는 파일 시스템의 데이터 구조입니다. superblock은 스토리지 패밀리 세트의 모든 분할 영역에 쓰여지고 세트에서 분할 영역의 멤버쉽을 식별합니다.

tar

Tape archive의 약어입니다. 아카이브 이미지에 사용되는 표준 파일 및 데이터 기록 형식입니다.

TCP/IP

Transmission Control Protocol/Internet Protocol. 호스트간 주소 지정 및 경로 지정, 패킷 전달(IP) 및 응용 프로그램 지점간의 데이터 전달(TCP)을 담당하는 인터넷 프로토콜입니다.

timer(타이머)

사용자가 소프트 한계에 도달하는 시간부터 사용자에게 부여된 하드 한계 사이에 경과된 시간을 추적하는 쿼터 소프트웨어입니다.

transport(전송)

robot(로봇)을 참조하십시오.

`vfstab` file(vfstab 파일)

`vfstab` 파일에는 파일 시스템에 대한 마운트 옵션이 포함됩니다. 명령줄에 지정된 마운트 옵션은 */etc/vfstab* 파일에 지정된 마운트 옵션보다 우선하지만, */etc/vfstab* 파일에 지정된 마운트 옵션은 *samfs.cmd* 파일에 지정된 마운트 옵션보다 우선합니다.

volume overflow(볼륨 오버플로우)

시스템에서 단일 파일을 여러 **volume(볼륨)**에 걸쳐 분산시킬 수 있는 기능입니다. 볼륨 오버플로우는 개별 카트리지의 용량을 초과하는 매우 큰 파일을 사용하는 사이트에 유용합니다.

volume serial number(VSN, 볼륨 일련 번호)

1. 테이프 또는 디스크 스토리지 볼륨에 지정된 일련 번호입니다. 볼륨 일련 번호는 최대 6자리의 영숫자 대문자로 구성될 수 있으며, 문자로 시작되어야 하고 테이프 라이브러리 또는 분할 영역과 같이 제공된 컨텍스트 내에서 고유하게 볼륨을 식별해야 합니다. 볼륨 일련 번호는 볼륨 레이블에 쓰여집니다.
2. 느슨하게 사용될 경우 특정 스토리지 **volume(볼륨)**, 특히 이동식 매체 **cartridge(카트리지)**입니다.

volume(볼륨)

1. 스토리지 매체에서 단일의 액세스 가능한 논리적 스토리지 영역이며, 대개 **volume serial number(VSN, 볼륨 일련 번호)** 및/또는 볼륨 레이블로 주소가 지정됩니다. 스토리지 디스크 및 마그네틱 테이프 카트리지는 하나 이상의 볼륨을 포함할 수 있습니다. 사용을 위해 볼륨은 파일 시스템의 지정된 에 마운트**mount point(마운트 지점)**됩니다.
2. 단일 논리적 볼륨을 포함하는 마그네틱 테이프 **cartridge(카트리지)**입니다.
3. 랜덤 액세스 디스크 장치에서 파일 시스템, 디렉토리 또는 파일은 테이프와 같은 순차 액세스, 이동식 매체 카트리지인 것처럼 구성되고 사용됩니다.

WORM

Write-Once-Read-Many의 약어입니다. 한 번만 쓸 수 있지만 여러 번 읽을 수 있는 매체에 대한 스토리지 분류입니다.

색인