Oracle® Hierarchical Storage Manager and StorageTek QFS Software

설치 및 구성 설명서

릴리스 6.1

E56769-03

2016년 3월

ORACLE

Oracle® Hierarchical Storage Manager and StorageTek QFS Software
설치 및 구성 설명서

E56769-03

차례

머리말

이 문서는 Oracle Hierarchical Storage Manager(이전의 StorageTek Storage Archive Manager) 및 Oracle StorageTek QFS 소프트웨어를 사용하여 파일 시스템 및 아카이빙 솔루션을 설치 및 구성하는 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 시스템, LAN과 SAN 둘 다의 관리에 익숙하다고 간주합니다. 관련 작업, 명령 및 절차에 대한 자세한 내용은 Solaris 설명서와 매뉴얼 페이지 및 스토리지 하드웨어 설명서를 참조하십시오.

규약

이 문서에서는 다음 텍스트 규약이 사용되었습니다.

- 기울임꼴 유형은 책 제목 및 강조를 나타냅니다.
- 굵은체 유형은 그래픽 사용자 인터페이스 요소를 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 설치 및 구성 설명서는 여러 볼륨으로 구성된 *Oracle HSM* 고객 설명서 라이브러리의 일부로, *docs.oracle.com/en/storage*에서 제공됩니다.

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

1

1장. Oracle HSM 솔루션 배치

Oracle Hierarchical Storage Manager and StorageTek QFS Software(Oracle HSM)를 배치하는 프로세스는 기본적으로 아주 간단합니다. 소프트웨어 패키지를 설치하고 몇몇 구성 파일을 편집하고 몇몇 명령을 실행한 다음 새 파일 시스템을 마운트하고 사용합니다. 그래도, Oracle HSM에서 옵션과 조정 매개변수를 다양하게 제공합니다. 이러한 추가 기능을 통해 특수한 요구 사항을 거의 모두 해결할 수 있습니다. 하지만 불필요한 기능으로 인해 배치가 복잡해지고 결과로 얻는 솔루션이 전혀 만족스럽지 않게 될 수도 있습니다.

따라서, 이 문서는 자세한 솔루션 요구 사항을 엄밀히 따르는 Oracle HSM 배치를 안내하도록 설계되었습니다. 먼저 기본 QFS 및 Oracle HSM 파일 시스템에 대한 작업, 설치 및 구성을 설명합니다. 이러한 파일 시스템은 자체적으로 요구 사항을 모두 충족하거나 더 특수화된 솔루션을 위한 기초를 형성합니다. 기본 사항이 마련되면 특정 환경 및 특수화된 비즈니스 요구를 지원하는 추가 기능을 구성하는 절차로 가지를 칠 수 있습니다. 다음과 같은 핵심 작업을 수행합니다.

- 하드웨어 및 운영체제 소프트웨어를 요구 사항에 맞게 구성합니다.
- 필요한 기본 QFS 및/또는 Oracle HSM 파일 시스템을 가능한 한 기본값을 적용하여 구성합니다.
- 요구 사항에 따라 필요한 추가 Oracle HSM 기능을 구성합니다.
- 완료된 구성을 백업하고 테스트 및 운용에 사용하도록 전달합니다.

계획 및 배치 프로세스 전체에서 QFS 및 Oracle HSM은 성능 최적화, 데이터 보호, 아카이빙의 복잡성을 단순한 UNIX 파일 시스템 인터페이스 뒤에 숨기도록 설계되었습니다. 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. 따라서 파일 시스템 및 스토리지 리소스를 너무 복잡하게 구현하고 지나치게 세부적으로 관리하면 Oracle HSM 배치의 핵심 목표가 약화될 뿐 아니라 성능, 용량 사용률 및/또는 데이터 보호가 손상되는 경우가 많습니다.

이 소개의 나머지 부분에서는 QFS 파일 시스템 및 Oracle HSM 아카이빙 파일 시스템에 대해 간략하게 설명하는 개요를 제공합니다. 이 정보의 주요 내용을 잘 알면 이어지는 구성 단계의 목적을 더 쉽게 이해할 수 있습니다.

QFS 파일 시스템

QFS 파일 시스템을 사용하면 완전히 최적화된 사용자 정의 스토리지 솔루션을 표준 UNIX 인터페이스와 결합할 수 있습니다. 내부적으로, 이 파일 시스템은 까다롭고, 경우에 따라 매우 특수화된 성능 요구 사항을 충족하도록 물리적 스토리지 장치를 관리합니다. 하지만 외부 사용자, 응용 프로그램 및 운영체제에게는 자신을 일반 UNIX 파일 시스템으로 표시합니다. 따라서 복잡하고 다양한 스토리지 하드웨어를 사용하여 특수화된 성능 및 데이터 보호 요구 사항을 충족하면서도 기존 응용 프로그램 및 비즈니스 프로세스와의 간단한 통합을 보장할 수 있습니다.

QFS 파일 시스템은 필수 QFS 볼륨 관리자를 사용하여 고유한 물리적 스토리지를 관리합니다. QFS 소프트웨어는 물리적 표준 스토리지 장치를 표준 인터페이스와 완전히 호환되는 최적화된 논리 장치로 구성합니다. 특수 기능 및 사용자 정의는 캡슐화되므로 운영체제와 응용 프로그램에게는 숨겨진 상태로 유지됩니다. 응용 프로그램의 경우 QFS 소프트웨어는 표준 Solaris 장치 드라이버를 통해 I/O 요청을 단일 디스크처럼 처리하는 논리적 패밀리 세트 장치를 표시합니다. 이와 같이 표준 준수와 조정 가능성이 조합된 점이 QFS가 다른 UNIX 파일 시스템과 구분되는 점입니다.

이 절의 나머지 부분에서는 먼저 QFS 기본값 및 I/O 성능 조정에 대해 간략히 설명한 다음 만드는 파일 시스템의 I/O 동작을 제어하는 데 사용할 수 있는 핵심 도구를 설명합니다.

- 유연한 디스크 할당 단위 및 논리 장치 유형을 사용하면 읽기 및 쓰기 크기를 파일 크기에 일치시킬 수 있습니다.
- 스트라이프 및 라운드 로빈 파일 할당 방법을 사용하면 파일 I/O가 장치와 상호 작용하는 방식을 제어할 수 있습니다.
- 완전히 구성 가능한 스토리지 할당 및 통합 볼륨 관리를 사용하면 파일 시스템이 기본 물리적 스토리지와 상호 작용하는 방식을 제어할 수 있습니다.
- 범용 및 고성능 파일 시스템을 선택하면 별도의 장치에서 데이터 및 메타데이터 I/O를 수행하는 옵션이 제공됩니다.

QFS 기본값 및 I/O 성능 조정 목표

디스크 I/O(입출력)에는 CPU를 많이 사용하는 운영체제 요청과 시간이 많이 소요되는 기계적 프로세스가 더불어 발생합니다. 그래서 I/O 성능 조정에서는 I/O 관련 시스템 오버헤드를 최소화하고 기계적 작업을 주어진 양의 데이터를 전달하는 데 필요한 최소 수준으로 유지하는 데 주력합니다. 즉, 데이터 전송당 별도의 I/O 수(따라서 CPU에서 수행하는 작업 수)를 줄이고 각 개별 I/O 중 검색(읽기/쓰기 헤드의 재배치)을 최소화합니다. 따라서 I/O 조정의 기본 목표는 다음과 같습니다.

- 평균 파일 크기로 고르게 나눈 블록으로 데이터를 읽고 씁니다.
- 큰 데이터 블록을 읽고 씁니다.
- 기본 매체의 512바이트 섹터 경계에 맞는 단위로 블록을 쓰므로 디스크 컨트롤러에서 기존 데이터를 읽거나 수정하지 않아도 새 데이터를 쓸 수 있습니다.
- 작은 I/O는 캐시에 대기시키고 더 크고 결합된 I/O는 디스크에 씁니다.

Oracle HSM 기본 설정을 사용하면 대부분의 범용 파일 시스템에 일반적인 응용 프로그램 및 사용 패턴 범위에서 전반적으로 최상의 성능이 제공됩니다. 하지만 필요한 경우 응용 프로그램에서 생성하는 I/O 유형에 더 잘 맞게 기본 동작을 조정할 수 있습니다. 최소 연속 읽기 또는 쓰기의 크기를 지정할 수 있습니다. 파일이 장치에 저장되는 방식을 최적화할 수 있습니다. 일반적인 사용이나 고성능에 최적화된 파일 시스템 간에 선택할 수 있습니다.

디스크 할당 단위 및 논리 장치 유형

파일 시스템에서는 디스크 스토리지를 균일한 크기의 블록으로 할당합니다. 이 크기 즉, DAU(디스크 할당 단위)는 쓰는 데이터 양에 상관없이 각 I/O 작업에서 사용하는 연속 공간의 최소 양과 지정된 크기의 파일을 전송할 때 필요한 최소 I/O 작업 수를 결정합니다. 블록 크기가 평균 파일 크기에 비해 너무 크면 디스크 공간이 낭비됩니다. 블록 크기가 너무 작으면 각 파일 전송에 더 많은 I/O 작업이 필요하므로 성능이 떨어집니다. 따라서 파일 크기가 기본 블록 크기의 짝수 배인 경우에 I/O 성능 및 스토리지 효율성이 가장 높습니다.

이 때문에 QFS 소프트웨어에서는 구성 가능한 DAU 크기를 다양하게 지원합니다. QFS 파일 시스템을 만드는 경우 먼저 액세스하고 저장해야 하는 데이터 파일의 평균 크기를 결정합니다. 그런 다음 평균 파일 크기로 가장 고르게 나누어지는 DAU를 지정합니다.

먼저 데이터에 가장 적합한 QFS 장치 유형을 선택합니다. 다음과 같은 세 가지 유형이 있습니다.

- *md* 장치
- *mr* 장치
- *gXXX* 스트라이프 그룹 장치(여기서 *XXX*는 *[0-127]* 범위의 정수)

파일 시스템에 대부분 작은 파일이거나 작은 파일과 큰 파일이 함께 있는 경우 일반적으로 *md* 장치를 선택하는 것이 가장 좋습니다. *md* 장치 유형은 유연한 이중 할당 체계를 사용합니다. 이 장치에 파일을 쓰면 파일 시스템에서는 처음 8회 쓰기에는 *4*킬로바이트의 작은 DAU를 사용합니다. 그런 다음 나머지 데이터는 *16*, *32* 또는 *64*킬로바이트의 큰 사용자 선택 DAU를 사용하여 씁니다. 따라서 작은 파일은 적당히 작은 블록에 기록되고, 큰 파일은 평균 크기에 맞게 더 큰 블록에 기록됩니다.

파일 시스템에 대부분 크거나 크기가 균일한 파일이 포함된 경우 *mr* 장치가 더 나은 선택일 수 있습니다. *mr* 장치 유형에서는 *[8-65528]*킬로바이트 범위에서 8킬로바이트 증분씩 조정할 수 있는 DAU를 사용합니다. 파일이 평균 파일 크기와 거의 같은 크고 균일한 블록에 기록되므로, 읽기-수정-쓰기 오버헤드가 최소화되고 성능이 최대화됩니다.

스트라이프 그룹은 최대 128개 장치를 묶은 것으로, 단일 논리 장치로 처리됩니다. *mr* 장치 유형처럼 스트라이프 그룹은 *[8-65528]*킬로바이트 범위에서 8킬로바이트 증분씩 조정할 수 있는 DAU를 사용합니다. 파일 시스템은 스트라이프 그룹의 멤버에 데이터를 기록할 때 디스크당 DAU 하나씩 병렬로 기록합니다. 따라서 쓰기를 집계하면 아주 커질 수 있습니다. 이 때문에 스트라이프 그룹은 매우 큰 데이터 파일을 처리해야 하는 응용 프로그램에서 유용할 수 있습니다.

파일 할당 방법

기본적으로 비공유 QFS 파일 시스템은 스트라이프 할당을, 공유 파일 시스템은 라운드 로빈 할당을 사용합니다. 하지만 필요한 경우 할당을 변경할 수 있습니다. 각 접근 방식은 일부 상황에서 장점을 발휘합니다.

스트라이프 할당

스트라이프 할당을 지정한 경우 파일 시스템에서는 사용 가능한 모든 장치에서 병렬로 공간을 할당합니다. 파일 시스템에서는 데이터 파일을 세그먼트화하고 각 장치에 세그먼트를 하나씩 기록합니다. 각 세그먼트 크기는 스트라이프 너비(장치당 기록되는 DAU 수)에 패밀리 세트의 장치 수를 곱해서 결정됩니다. 장치 유형은 *md* 디스크 장치, *mr* 디스크 장치 또는 스트라이프 그룹일 수 있습니다.

스트라이프하면 파일 시스템에서 여러 파일 세그먼트를 순서대로가 아니라 동시에 읽으므로 일반적으로 성능이 향상됩니다. 여러 I/O 작업이 별도의 장치에서 병렬로 수행되므로 장치당 검색 오버헤드가 줍니다.

하지만, 여러 파일을 한번에 기록하는 경우에 스트라이프 할당을 사용하면 검색이 상당히 더 많아질 수 있습니다. 검색이 과도하면 성능이 심하게 저하될 수 있으므로, 여러 파일로의 동시 I/O가 예상되는 경우에는 라운드 로빈 할당을 고려하는 것이 좋습니다.

라운드 로빈 할당

라운드 로빈 할당을 지정한 경우 파일 시스템에서는 한 번에 한 파일씩, 그리고 한 번에 한 장치씩 직렬로 스토리지 공간을 할당합니다. 파일 시스템에서는 사용 가능한 공간이 있는 첫번째 장치에 파일을 기록합니다. 파일이 장치에 남은 공간보다 큰 경우에는 초과분을 가용 공간이 있는 다음 장치에 기록합니다. 각 후속 파일에 대해 파일 시스템은 사용 가능한 다음 장치로 이동하여 프로세스를 반복합니다. 사용 가능한 마지막 장치가 사용되면 첫번째 장치에서 다시 시작됩니다. 장치 유형은 *md* 디스크 장치, *mr* 디스크 장치 또는 스트라이프 그룹일 수 있습니다.

응용 프로그램에서 여러 파일로의 I/O를 동시에 수행하는 경우 라운드 로빈 할당을 사용하면 성능이 향상될 수 있습니다. 이 방법은 공유 QFS 파일 시스템의 기본 방법이기도 합니다(공유 파일 시스템에 대한 자세한 내용은 “Oracle HSM 소프트웨어를 사용하여 여러 호스트에서 파일 시스템 액세스” 및 *mount_samfs* 매뉴얼 페이지 참조).

스토리지 할당 및 통합 볼륨 관리

장치 하나당 장치의 일부만 처리하는 UNIX 파일 시스템과 달리 QFS 파일 시스템은 자체 볼륨 관리를 수행합니다. 각 파일 시스템은 내부적으로 물리적 스토리지를 제공하는 장치 간의 관계를 처리한 다음 운영체제에 스토리지를 단일 패밀리 세트로 표시합니다. UNIX 파일 시스템과 마찬가지로 I/O 요청은 표준 Solaris 장치-드라이버 인터페이스를 통해 이루어집니다.

파일 시스템 유형

QFS 파일 시스템의 유형은 다음 두 가지입니다. 각각 고유한 장점이 있습니다.

범용 ms 파일 시스템

QFS *ms* 파일 시스템은 구현하기가 가장 간단하고 가장 일반적인 용도에 적합합니다. 동일한 이중 할당 *md* 디스크 장치에 파일 시스템 메타데이터와 파일 데이터를 모두 저장합니다. 이 접근법은 하드웨어 구성을 간소화하고 대부분의 요구를 충족합니다.

고성능 ma 파일 시스템

QFS *ma* 파일 시스템은 조건이 까다로운 응용 프로그램의 데이터 전송 속도를 향상시킬 수 있습니다. 이러한 파일 시스템은 메타데이터와 데이터를 전용 장치에 별도로 저장합니다. 메타데이터는 *mm* 장치에 유지되고 데이터는 *md* 디스크 장치, *mr* 디스크 장치 또는 스트라이프 그룹 세트에 유지됩니다. 따라서 메타데이터 업데이트가 사용자 및 응용 프로그램 I/O와 경합하지 않으며, 장치 구성이 두 종류의 서로 다른 I/O 작업 로드를 수용할 필요가 없습니다. 예를 들어, 메타데이터는 중복성이 뛰어나고 읽기 속도가 빠른 RAID-10 미러링된 디스크에 배치하고 데이터는 공간 효율이 더 나은 RAID-5 디스크 어레이에 유지할 수 있습니다.

Oracle HSM 아카이빙 파일 시스템

아카이빙 파일 시스템에는 하나 이상의 QFS *ma* 또는 *ms* 유형 파일 시스템과 아카이브 스토리지 및 Oracle Hierarchical Storage Manager 소프트웨어가 결합되어 있습니다. Oracle HSM 소프트웨어는 파일 시스템 디스크 캐시의 파일을 보조 디스크 스토리지 및/또는 이동식 매체로 복사합니다. 복사본을 파일 시스템의 필수 부분으로 관리합니다. 따라서 파일 시스템은 데이터 보호를 지속적으로 제공할 뿐 아니라 디스크나 반도체 매체에 저장할 경우 과도하게 많은 비용이 드는 아주 큰 파일을 유연하고 효율적으로 저장할 수 있습니다.

Oracle HSM 파일 시스템을 제대로 구성하면 별도의 백업 응용 프로그램 없이도 데이터 보호 기능을 지속적으로 제공할 수 있습니다. 소프트웨어는 사용자 정의 정책에 지정된 대로 파일이 생성되거나 변경될 때 파일 데이터를 자동으로 복사합니다. 디스크 및 테이프 매체 조합에 로컬 및 원격 리소스를 모두 사용하여 복사본을 4개까지 유지할 수 있습니다. 파일 시스템 메타데이터에 파일과 모든 복사본의 위치가 기록됩니다. 소프트웨어에 제공되는 다양한 도구로 복사본을 빠르게 찾을 수 있습니다. 따라서 손실되거나 손상된 파일을 아카이브에서 쉽게 복구할 수 있습니다. 그런데 백업 복사본은 POSIX 호환 표준 *tar*(테이프 아카이브) 형식으로 유지되므로 Oracle HSM 소프트웨어를 사용할 수 없는 경우에도 데이터를 복구할 수 있습니다. Oracle HSM에서는 I/O 오류를 동적으로 검색 및 복구하여 파일 시스템 메타데이터를 항상 일관된 상태로 유지합니다. 따라서 시간이 많이 소요되는 무결성 확인 없이도 파일 시스템을 백업할 수 있습니다. 이 무결성 확인은 수십 만 개 파일 및 페타바이트 데이터를 저장할 때 중요 고려 사항입니다. 파일 시스템 메타데이터를 별도의 장치에 저장하고 데이터 스토리지 디스크만 포함하는 경우 파일 시스템에 교체 디스크를 구성하면 복구가 완료됩니다. 사용자가 실패한 디스크에 있는 파일을 요청하면 Oracle HSM에서는 자동으로 백업 복사본을 테이프에서 교체 디스크로 스테이지합니다. 메타데이터도 손실된 경우 관리자는 *samfsrestore* 명령을 사용하여 *samfsdump* 백업 파일에서 복원할 수 있습니다. 메타데이터가 복원된 경우 사용자가 파일을 요청하면 테이프에서 다시 복원할 수 있습니다. 파일을

요청에 따라 디스크로만 복원하므로 복구 프로세스에서 네트워크 대역폭을 효율적으로 사용하고 일반 작업에 최소로 영향을 미칩니다.

고성능 기본 디스크 또는 반도체 매체와 저비용 고밀도 보조 디스크, 테이프 또는 광 매체에서 파일을 동시에 관리할 수 있는 이 기능 덕분에 Oracle HSM 파일 시스템은 엄청나게 큰 파일이나 거의 사용되지 않는 파일을 경제적으로 저장하는 데 적합합니다. 위성 이미지 및 비디오 파일과 같이 매우 크고 순차적으로 액세스하는 데이터 파일은 단독으로 자기 테이프에 저장할 수 있습니다. 사용자나 응용 프로그램이 파일에 액세스할 때 파일 시스템에서는 선택한 파일 구성에 따라 자동으로 파일을 다시 디스크에 스테이지하거나 테이프에서 직접 메모리로 읽어들입니다. 주로 기록 또는 준수용으로 보관하는 레코드는 파일 수명의 특정 시점에 사용자 액세스 패턴 및 비용 제약 조건과 가장 잘 맞는 매체를 사용하여 계층적으로 저장할 수 있습니다. 사용자가 여전히 파일에 가끔 액세스하는 처음에는 비용이 저렴한 보조 디스크 장치에 파일을 아카이브할 수 있습니다. 수요가 감소하면 테이프나 광 매체에만 복사본을 유지할 수 있습니다. 그러나, 사용자가 법적 개시나 규정 준수 프로세스에 대처하기 위해 데이터가 필요한 경우 파일 시스템에서는 필요한 자료를 자동으로 기본 디스크에 스테이지할 수 있으며, 이때 자료가 내내 거기 있었던 것처럼 지연 시간을 최소화합니다. 법적 규정을 위해 Oracle HSM 파일 시스템에서 WORM을 사용으로 설정할 수 있습니다. WORM 사용 파일 시스템은 기본/사용자 정의 가능 파일 보존 기간, 데이터 및 경로 불변성, 하위 디렉토리의 WORM 설정 상속을 지원합니다. 수동 및/또는 자동 매체 검증을 통해 장기간 데이터 무결성을 모니터링할 수 있습니다.

다음 네 가지 기본 Oracle HSM 프로세스를 통해 아카이빙 파일 시스템을 관리 및 유지합니다.

- 아카이빙
- 스테이징
- 릴리스
- 재활용.

아카이빙

아카이빙 프로세스에서는 활성 파일의 복사본을 저장하기 위해 예약된 아카이브 매체에 파일 시스템의 파일을 복사합니다. 아카이브 매체에는 자기 테이프 카트리지와 같은 이동식 매체 볼륨 및/또는 자기 디스크나 반도체 스토리지 장치에 있는 하나 이상의 파일 시스템이 포함될 수 있습니다. 아카이브 파일 복사본은 활성 파일에 대한 백업 중복성, 비활성 파일에 대한 장기 보존 또는 이 두 가지의 조합을 제공할 수 있습니다.

Oracle HSM 아카이빙 파일 시스템에서는 활성 온라인 파일, 아카이브 복사본 및 관련 스토리지 리소스로 단일한 논리 리소스인 아카이브 세트를 형성합니다. 아카이빙 파일 시스템의 모든 활성 파일은 정확히 하나의 아카이브 세트에 속합니다. 각 아카이브 세트에는 각 파일의 아카이브 복사본 최대 4개와 아카이브 세트의 아카이빙 프로세스를 제어하는 정책이 포함될 수 있습니다.

아카이빙 프로세스는 UNIX 데몬(서비스) *sam-archiverd*에서 관리합니다. 이 데몬은 아카이빙 작업 일정을 잡고 필요한 작업을 수행하는 프로세스 *archiver*, *sam-arfind* 및 *sam-arcopy*를 호출합니다.

archiver 프로세스는 편집 가능한 구성 파일 *archiver.cmd*에서 아카이빙 정책을 읽고 지정된 대로 나머지 아카이빙 프로세스를 설정합니다. 이 파일의 지시어는 아카이빙 프로세스의 일반 동작을 제어하고, 파일 시스템별로 아카이브 세트를 정의하고, 각각에 대해 생성되는 복사본 수와 사용되는 매체를 지정합니다.

그런 다음 *sam-archiverd* 데몬은 현재 마운트된 각 파일 시스템에 대해 *sam-arfind* 프로세스를 시작합니다. *sam-arfind* 프로세스는 지정된 파일 시스템을 검색하여 새 파일, 수정된 파일, 이름이 바뀐 파일 및 다시 아카이브하거나 아카이브 취소할 파일을 찾습니다. 기본적으로 이 프로세스는 파일 및 디렉토리에 대한 변경사항을 지속적으로 검색합니다. 이렇게 해야 전반적으로 최상의 성능을 제공하기 때문입니다. 그러나, 이전 StorageTek Storage Archive Manager 구현과 호환성을 유지해야 하는 등의 경우에는 여러 방법 중 하나를 통해 *archiver.cmd* 파일에서 아카이브 세트 규칙을 편집하여 검색 일정을 잡을 수 있습니다(자세한 내용은 *sam-archiverd* 매뉴얼 페이지 참조).

*sam-arfind*는 후보 파일을 식별하면 이 파일의 아카이빙 정책을 정의하는 아카이브 세트를 식별합니다. *sam-arfind* 프로세스는 파일의 속성을 각 아카이브 세트에 정의된 선택 기준과 비교하여 아카이브 세트를 식별합니다. 이러한 기준에는 다음과 같은 파일 속성이 하나 이상 포함될 수 있습니다.

- 파일에 대한 디렉토리 경로 및 선택적으로 하나 이상의 후보 파일 이름과 일치하는 정규 표현식
- 후보 파일 하나 이상의 소유자와 일치하는 지정된 사용자 이름
- 파일에 연결된 그룹과 일치하는 지정된 그룹 이름
- 후보 파일 크기보다 작거나 같은 지정된 최소 파일 크기
- 후보 파일 크기보다 크거나 같은 지정된 최대 파일 크기

올바른 아카이브 세트와 해당하는 아카이빙 매개변수를 찾았으면 *sam-arfind*는 파일의 아카이브 기간이 아카이브 세트에 지정된 임계값과 동일한지 또는 임계값을 초과하는지 확인합니다. 파일의 아카이브 기간은 파일을 만들거나, 마지막으로 수정하거나(기본값), 마지막으로 액세스한 이후 경과한 시간(초)입니다. 아카이브 기간이 정책에 지정된 기간 기준을 충족하면 *sam-arfind*는 아카이브 세트의 아카이브 요청 대기열에 파일을 추가하고 파일에 우선순위를 지정합니다. 우선 순위는 아카이브 세트에 지정된 규칙 및 이미 있는 아카이브 복사본 수, 파일 크기, 미해결 운영자 요청, 아카이브 복사본 만들기에 따라 달라지는 다른 모든 작업 등과 같은 요소를 기준으로 합니다.

*sam-arfind*는 아카이빙이 필요한 파일을 식별하고 우선 순위를 지정한 다음 각 아카이브 세트의 아카이브 요청에 추가한 이후 요청을 *sam-archiverd* 데몬에 반환합니다. 이 데몬은 각 아카이브 요청을 작성합니다. 그리고 데이터 파일을 적절한 크기의 아카이브 파일로 정렬하여, 매체가 효율적으로 활용되며 파일이 이동식 매체에 효율적으로 기록되고 나중에 매체에서 회수될 수 있게 합니다. 이 데몬은 사용자가 *archiver.cmd* 파일에 설정하는 파일 정렬 매개변수나 매체 제한을 수용하지만(자세한 내용은 *archiver.cmd* 매뉴얼 페이지 참조), 소프트웨어에서 매체를 자유롭게 선택할 수 있는 기능을 제한하면 일반적으로 성능 및 매체 사용률이 떨어진다는 점을 유념하십시오. 아카이브 파일이 어셈블링되면 *sam-archiverd*는 복사 프로세스에서 가장 작은 수의 마운트 작업으로 가장 많은 수의 파일을 전송할 수 있도록 아카이브 요청의 우선 순위를 지정합니다(자세한 내용은 *sam-archiverd*

매뉴얼 페이지의 예약 절 참조). 그런 다음 *sam-archiverd*는 주어진 시간에 복사 작업에 필요한 드라이브 수가 아카이브 세트 정책 및/또는 로봇 라이브러리에서 허용하는 최대 드라이브 수보다 많지 않도록 복사 작업을 예약합니다.

아카이브 요청이 예약되면 *sam-archiverd*는 예약된 각 아카이브 요청 및 드라이브에 대해 *sam-arcopy* 프로세스의 인스턴스를 호출합니다. 그런 다음 *sam-arcopy* 인스턴스는 데이터 파일을 아카이브 매체의 아카이브 파일에 복사하고, 아카이빙 파일 시스템의 메타데이터를 업데이트하여 새 복사본이 있음을 반영하고, 아카이브 로그를 업데이트합니다.

sam-arcopy 프로세스가 종료되면 *sam-archiverd* 데몬은 아카이브 요청을 검사하여 캐시 디스크의 오류, 이동식 매체 볼륨에 대한 쓰기 오류 및 열거나 수정하거나 삭제한 파일로 인한 생략 또는 오류가 있는지 확인합니다. 아카이브되지 않은 파일이 있는 경우 *sam-archiverd*는 아카이브 요청을 다시 작성합니다.

sam-arfind 및 *sam-arcopy* 프로세스에서는 *syslog* 기능과 *archiver.sh*를 사용하여 아카이빙 작업, 경고 및 정보 메시지의 연속 레코드를 만들 수 있습니다. 결과 아카이버 로그에는 아카이브된 각 파일의 모든 복사본 위치와 처리에 대한 자세한 레코드를 포함하여 귀중한 진단 및 내역 정보가 포함됩니다. 따라서 예를 들어 재해 복구 중 아카이브 로그를 사용하여 다른 방법으로 복구할 수 없는 누락된 데이터 파일을 복구할 수 있는 경우가 많습니다(자세한 내용은 고객 설명서 라이브러리의 *Oracle Hierarchical Storage Manager and StorageTek QFS Software* 파일 시스템 복구 설명서 참조). 파일 시스템 관리자는 *archiver.cmd* 파일의 *logfile=* 지시어를 사용하여 아카이버 로깅을 사용으로 설정하고 로그 파일을 정의합니다. 로그 파일에 대한 자세한 내용은 *archiver.cmd* 매뉴얼 페이지를 참조하십시오.

스테이징

스테이징 프로세스에서는 아카이브 스토리지의 파일 데이터를 다시 기본 디스크 캐시에 복사합니다. 응용 프로그램에서 오프라인 파일(기본 스토리지에서 현재 사용할 수 없는 파일)에 액세스하려고 하면 아카이브 복사본이 자동으로 스테이지됩니다. 즉, 기본 디스크에 다시 복사됩니다. 그러면 응용 프로그램은 전체 데이터가 다시 디스크에 기록되기 전에도 파일에 빠르게 액세스할 수 있습니다. 스테이징에 이어 곧바로 읽기 작업이 수행되기 때문입니다. 매체 오류가 발생하는 경우나 특정 매체 볼륨을 사용할 수 없는 경우 스테이징 프로세스에서는 사용 가능한 첫번째 장치를 사용하여 사용 가능한 다음 아카이브 복사본(있는 경우)을 자동으로 로드합니다. 따라서 스테이징에서는 아카이브 스토리지를 사용자와 응용 프로그램에 투명하게 만듭니다. 모든 파일이 디스크에서 항상 사용할 수 있는 것으로 표시됩니다.

기본 스테이징 동작은 대부분의 파일 시스템에 적합합니다. 그러나 구성 파일 */etc/opt/SUNWsamfs/stager.cmd*에서 지시어를 삽입하거나 수정하여 기본값을 변경할 수 있고 명령줄에서 디렉토리별 또는 파일별로 이러한 지시어를 대체할 수 있습니다. 예를 들어 큰 파일의 작은 레코드에 액세스하려면 파일을 스테이지하지 않고 아카이브 매체에서 직접 데이터에 액세스할 수 있습니다. 또는 연관 스테이징 기능을 사용하여 관련 파일 중 하나가 스테이징될 때마다 전체 그룹을 스테이징할 수 있습니다. 자세한 내용은 *stage* 및 *stager.cmd* 매뉴얼 페이지를 참조하십시오.

릴리스

해제 프로세스는 이전에 아카이브한 파일의 온라인 복사본 중 현재 사용되지 않는 복사본을 삭제하여 기본 디스크 캐시 공간을 확보합니다. 파일이 디스크 아카이브나 테이프 볼륨과 같은 아카이브 매체에 복사되면 응용 프로그램에서 파일에 액세스할 때 파일을 스테이지할 수 있습니다. 따라서 디스크 캐시에 파일을 보유할 필요가 없으므로 다른 파일에 필요한 공간을 확보할 수 있습니다. 해제를 통해 필요하지 않은 복사본을 디스크 캐시에서 삭제하면 파일 시스템이 확대에 맞추어 기본 스토리지 용량을 늘리지 않더라도 새로 만들어 활발히 사용되는 파일에 기본 캐시 스토리지를 항상 사용할 수 있게 됩니다.

캐시 사용률이 고수위를 초과하고 저수위 이상으로 유지되는 경우 해제가 자동으로 수행됩니다. 위의 두 가지 구성 가능한 임계값은 아카이빙 파일 시스템을 마운트할 때 설정합니다. 고수위이면 충분한 여유 공간을 항상 사용할 수 있는 반면, 저수위이면 캐시에서 적절한 수의 파일을 항상 사용할 수 있으므로 매체 마운트 작업이 필요한 최소 수준으로 유지됩니다. 일반적으로 값이 80%이면 고수위이고 70%이면 저수위입니다.

대부분의 파일 시스템에서 기본 동작을 사용한 수위별 해제면 적합합니다. 그러나 구성 파일 */etc/opt/SUNWsamfs/releaser.cmd*에서 지시어를 수정하거나 추가하여 기본값을 변경할 수 있고 명령줄에서 디렉토리별 또는 파일별로 이러한 지시어를 대체할 수 있습니다. 예를 들어, 순차적으로 액세스하는 큰 파일을 부분적으로 해제하면, 나머지 부분이 아카이브 매체에서 스테이지되는 동안 응용 프로그램에서 디스크에 항상 보관된 파일의 일부를 우선 읽을 수 있습니다. 자세한 내용은 *release* 및 *releaser.cmd* 매뉴얼 페이지를 참조하십시오.

재활용

재활용 프로세스는 더 이상 사용되지 않는 아카이브 복사본을 삭제하여 아카이브 매체의 공간을 확보합니다. 사용자가 파일을 수정하면 이전 버전의 파일과 연관된 아카이브 복사본은 결국 만료됩니다. 리사이클러는 만료된 아카이브 복사본의 가장 많은 부분을 보유하는 매체 볼륨을 식별합니다. 만료된 파일이 아카이브 디스크 볼륨에 저장되어 있는 경우 리사이클러 프로세스에서는 이 파일을 삭제합니다. 파일이 테이프 볼륨과 같은 이동식 매체에 있으면 리사이클러는 볼륨에 남아 있는 만료되지 않은 복사본을 다른 매체로 다시 아카이브합니다. 그런 다음 편집 가능한 스크립트 */etc/opt/SUNWsamfs/scripts/recycler.sh*를 호출하여 재활용된 볼륨의 레이블을 재지정하거나, 라이브러리에서 볼륨을 내보내거나, 다른 사용자 정의 작업을 수행합니다.

기본적으로 재활용 프로세스는 자동으로 실행되지 않습니다. Solaris *crontab* 파일을 구성하여 편리한 시간에 실행할 수 있습니다. 또는 */opt/SUNWsamfs/sbin/sam-recycler* 명령을 사용하여 명령줄에서 필요에 따라 실행할 수 있습니다. 기본 재활용 매개변수를 수정하려면 */etc/opt/SUNWsamfs/archiver.cmd* 파일을 편집하거나 */etc/opt/SUNWsamfs/recycler.cmd* 파일을 별도로 만듭니다. 자세한 내용은 해당하는 매뉴얼 페이지를 참조하십시오.

2

2장. 호스트 시스템 구성

설치 및 구성을 계속 진행하기 전에 Oracle Hierarchical Storage Manager and StorageTek QFS Software에 대한 호스트 운영체제를 구성하십시오. 이 장에서는 다음 항목을 설명합니다.

Oracle HSM용 Oracle Solaris 구성

Oracle HSM 소프트웨어 및 QFS 파일 시스템에서 사용할 Solaris 호스트를 구성하려면 다음 작업을 수행하십시오.

최신 운영체제 업데이트 설치

가능한 경우 Solaris 운영체제에 대한 최신 패치와 업데이트를 항상 설치하십시오. Oracle Hierarchical Storage Manager and StorageTek QFS Software 릴리스 6.1에서 제공되는 최신 기능을 사용해야 하는 경우 Oracle Solaris 11 운영체제 소프트웨어를 모든 Solaris 호스트에 설치해야 합니다. 소프트웨어와 함께 사용할 최소 권장 운영체제 릴리스에 대한 자세한 내용은 릴리스 노트 및 *support.oracle.com*을 참조하십시오.

선택한 Solaris 버전에 대한 설치 및 업데이트 지침은 *support.oracle.com*에서 해당하는 고객 설명서 라이브러리, OTN(Oracle Technical Network) 및 기술 자료의 설치 및 관리 문서를 참조하십시오. IPS(이미지 패키징 시스템)를 처음 사용하는 경우 자세한 내용은 다음 OTN 문서를 참조하십시오.

- *Introducing the Basics of Image Packaging System (IPS) on Oracle Solaris 11*, Glynn Foster(2011년 11월)
- *How to Update Oracle Solaris 11 Systems From Oracle Support Repositories*, Glynn Foster(2012년 3월)
- *More Tips for Updating Your Oracle Solaris 11 System from the Oracle Support Repository*, Peter Dennis(2012년 5월)

예상 파일 시스템 I/O에 대한 Solaris 시스템 및 드라이버 매개변수 조정

운영체제, 드라이버, 파일 시스템 및 응용 프로그램이 불필요하게 단편화하고 다시 캐시할 필요가 없는 단위로 데이터를 전송할 때 시스템의 종단간 I/O(입력/출력) 성능이 최대화됩니다. 따라서 응용 프로그램과 파일 시스템에서 가능한 최대 데이터 전송 속도로 Solaris를 설정하십시오. 다음과 같이 하십시오.

1. Oracle HSM 파일 시스템 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. */etc/system* 파일의 백업 복사본을 만든 다음 */etc/system*을 텍스트 편집기에서 엽니다.

   예제에서는 *vi* 편집기를 사용합니다.

```
root@solaris:~# cp /etc/system /etc/system.backup
root@solaris:~# vi /etc/system
*ident  "%Z%%M% %I%     %E% SMI" /* SVR4 1.5 */
* SYSTEM SPECIFICATION FILE
...
```

3. */etc/system* 파일에서 *maxphys*(드라이버에서 단일 단위로 처리할 수 있는 최대 물리적 I/O 요청의 크기)를 응용 프로그램 및 파일 시스템에서 가능한 최대 데이터 전송 속도와 동일하게 설정합니다. 라인을 *set maxphys = 0xvalue* 형식으로 입력합니다. 여기서 *value*는 바이트 수를 나타내는 16진수입니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   드라이버에서 *maxphys*를 초과하는 요청을 *maxphys* 크기의 단편으로 분할합니다. 기본값은 운영체제 릴리스에 따라 다를 수 있지만, 일반적으로 약 128KB입니다. 예제에서는 *maxphys*를 *0x800000*(8,388,608바이트 또는 8MB)으로 설정합니다.

```
root@solaris:~# vi /etc/system
*ident  "%Z%%M% %I%     %E% SMI" /* SVR4 1.5 */
* SYSTEM SPECIFICATION FILE
...
set maxphys = 0x800000
:wq
root@solaris:~#
```

4. */kernel/drv/sd.conf* 파일을 텍스트 편집기에서 엽니다.

   이 예에서는 *vi* 편집기를 사용합니다.

```
root@solaris:~# vi /kernel/drv/sd.conf
# Copyright (c) 1991, 2010, Oracle and/or its affiliates. All rights reserved.
```

```
name="sd" class="scsi" target=0 lun=0;
name="sd" class="scsi" target=1 lun=0;
...
# Associate the driver with devid resolution.
ddi-devid-registrant=1;
```

5. */kernel/drv/sd.conf* 파일에서 *sd_max_xfer_size*(SCSI 디스크(*sd*) 드라이버에서 처리할 수 있는 최대 데이터 전송 크기)를 *maxphys*에 대해 설정된 값으로 설정합니다. 라인을 *sd_max_xfer_size=0xvalue;* 형식으로 입력합니다. 여기서 *value*는 바이트 수를 나타내는 16진수입니다. 파일을 저장하고 편집기를 닫습니다.

   기본값은 *0x100000*(1048576바이트 또는 1MB)입니다. 예제에서는 설명을 추가하고 *sd_max_xfer_size*를 *0x800000*(8,388,608바이트 또는 8MB)으로 설정합니다.

```
...
# Associate the driver with devid resolution.
ddi-devid-registrant=1;
# Set SCSI disk maximum transfer size
sd_max_xfer_size=0x800000;
:wq
root@solaris:~#
```

6. */kernel/drv/ssd.conf* 파일을 텍스트 편집기에서 엽니다.

   예제에서는 *vi* 편집기를 사용합니다.

```
root@solaris:~# vi /kernel/drv/ssd.conf
# Copyright 2009 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
name="ssd" parent="sf" target=0;
name="ssd" parent="fp" target=0;
...
name="ssd" parent="ifp" target=127;
```

7. */kernel/drv/ssd.conf* 파일에서 *ssd_max_xfer_size*(광 섬유 채널 디스크(*ssd*) 드라이버에서 처리할 수 있는 최대 데이터 전송 크기)를 *maxphys*에 대해 설정된 값으로 설정합니다. 라인을 *ssd_max_xfer_size=0xvalue;* 형식으로 입력합니다. 여기서 *value*는 바이트 수를 나타내는 16진수입니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   기본값은 *0x100000*(1048576바이트 또는 1MB)입니다. 예제에서는 설명을 추가하고 *ssd_max_xfer_size*를 *0x800000*(8,388,608바이트 또는 8MB)으로 설정합니다.

```
...
name="ssd" parent="ifp" target=127;
```

```
# Set Fibre Channel disk maximum transfer size
ssd_max_xfer_size=0x800000;
:wq
root@solaris:~#
```

8. 시스템을 다시 시작합니다. *init 6* 명령을 사용합니다.

```
root@solaris:~# init 6
```

9. 추가 Solaris 호스트를 포함하는 솔루션을 준비하는 경우 모든 Solaris 호스트가 구성될 때까지 “Oracle HSM용 Oracle Solaris 구성”에 지정된 작업을 반복합니다.
10. 하나 이상의 Linux 클라이언트를 포함하는 솔루션을 준비하는 경우 "Oracle HSM 클라이언트용 Linux 구성"으로 이동합니다.
11. 그렇지 않은 경우 3장. 스토리지 호스트 및 장치 구성으로 이동합니다.

Oracle HSM 클라이언트용 Linux 구성

Oracle HSM 클라이언트 소프트웨어를 설치하기 전에 다음과 같이 Linux 운영체제를 준비해야 합니다.

- 호환되지 않는 운영체제 기능 사용 안함
- 필요한 커널 개발 및 유틸리티 패키지 설치

호환되지 않는 운영체제 기능 사용 안함

1. Oracle HSM 클라이언트 호스트에 *root*로 로그인합니다.

```
[root@linux ~]#
```

2. SELinux(Secure Linux)가 설치되어 있는 경우 사용 안함으로 설정합니다. */etc/selinux/config* 파일을 텍스트 편집기에서 열고 *SELINUX* 플래그를 *disabled*로 설정한 후 파일을 저장한 다음 편집기를 닫고 재부트합니다.

   SELinux는 기본적으로 Oracle Linux 및 Red Hat Enterprise Linux에서 사용으로 설정되지만 Oracle HSM에서는 지원되지 않습니다. 예제에서는 *vi* 편집기에서 파일을 엽니다.

```
[root@linux ~]# vi /etc/selinux/config
# This file controls the state of SELinux on the system.
...
#SELINUX=enforcing
#SELINUX=permissive
SELINUX=disabled
SELINUXTYPE=targeted
:wq
```

```
[root@linux ~]# reboot
```

3. AppArmor가 설치되어 있는 경우 Linux 배포 설명서에서 권장되는 절차에 따라 사용 안 함으로 설정합니다.

   AppArmor가 SELinux 대신 사용되는 경우도 있습니다. Oracle HSM에서는 AppArmor를 지원하지 않습니다.
4. 이제 필요한 커널 개발 및 유틸리티 패키지 설치를 수행합니다.

필요한 커널 개발 및 유틸리티 패키지 설치

Oracle HSM 클라이언트 소프트웨어를 설치하기 전에 Linux 커널 개발 패키지를 일부 지정된 유틸리티 패키지와 함께 설치해야 합니다. 필요한 패키지를 식별하여 설치하려면 다음 절차를 수행합니다.

1. Linux 클라이언트 호스트에 *root*로 로그인합니다.

   예제에서는 클라이언트가 Oracle Linux에서 호스팅됩니다.

```
[root@linux ~]#
```

2. 클라이언트에 설치된 커널 버전을 식별합니다. *uname -r* 명령을 사용합니다.

   예제에서는 커널 버전이 *2.6.9-89.0.0.0.1.EL*입니다.

```
[root@linux ~]# uname -r
2.6.9-89.0.0.0.1.EL
[root@linux ~]#
```

3. *kernel-devel-kernel-version* 커널 개발 키트를 설치합니다. 여기서 *kernel-version*은 이전 단계에서 식별된 버전 문자열입니다.

   Oracle HSM 클라이언트를 설치하려면 이 패키지의 일부인 *Module.symvers*가 필요합니다. 예제에서는 Oracle Linux 명령 *yum*을 *-y install* 매개변수와 함께 사용합니다. 여기서 *-y*는 모든 프롬프트에 자동으로 "yes"로 대답하도록 합니다.

```
[root@linux ~]# yum -y install / kernel-devel-2.6.9-89.0.0.0.1.EL.i686.rpm
[root@linux ~]#
```

4. Korn 셸 *ksh*가 설치되어 있는지 확인합니다. 설치되어 있지 않은 경우 설치합니다.

   예제에서는 Oracle Linux 명령 *rpm -qa*의 출력을 *grep* 명령에 연결하고 *ksh* 문자열을 검색합니다. 이 명령은 출력을 반환하지 않으므로, *ksh*가 설치되어 있지 않은 것입니다. 따라서 *yum install ksh* 명령을 사용하여 이 셸을 설치합니다.

```
[root@linux ~]# rpm -qa | grep ksh
```

```
[root@linux ~]#
[root@linux ~]# yum install ksh
...
--> Running transaction check
---> Package ksh-20100621-19.e16.x86_64 set to be installed

================================================================================
Package            Arch          Version                  Repository      Size
================================================================================
Installing:
 ksh               i686          2.6.9-89.0.0.0.1.EL       updates        506 k
...
Installed:
  ksh-2.6.9-89.0.0.0.1.EL.i686
Complete!
[root@linux ~]#
```

5. *cpio* 유틸리티가 설치되어 있는지 확인합니다. 설치되어 있지 않은 경우 설치합니다.

   예제에서는 Oracle Linux 명령 *rpm -qa*의 출력을 *grep* 명령에 연결하고 *cpio* 문자열을 검색합니다. 이 명령은 버전 정보를 반환하므로, *cpio* 유틸리티가 설치되어 있는 것입니다.

```
[root@linux ~]#  rpm -qa | grep cpio
cpio-2.10-10.e16.x86_64
[root@linux ~]#
```

6. *find* 유틸리티가 설치되어 있는지 확인합니다. 설치되어 있지 않은 경우 설치합니다.

   예제에서는 Oracle Linux 명령 *rpm -qa*의 출력을 *grep* 명령에 연결하고 *findutils* 문자열을 검색합니다. 이 명령은 버전 정보를 반환하므로, *findutils* 패키지가 설치되어 있는 것입니다.

```
[root@linux ~]#  rpm -qa | grep findutils
findutils-4.4.2-6.e16.x86_64
[root@linux ~]#
```

7. *gcc* 컴파일러가 설치되어 있는지 확인합니다. 설치되어 있지 않은 경우 설치합니다.

   예제에서는 Oracle Linux 명령 *rpm -qa*의 출력을 *grep* 명령에 연결하고 *gcc* 문자열을 검색합니다. 이 명령은 버전 정보를 반환하므로, *gcc* 컴파일러가 설치되어 있는 것입니다.

```
[root@linux ~]#  rpm -qa | grep gcc
gcc-4.4.7-3.e16.x86_64
```

```
libgcc-4.4.7-3.e16.x86_64
[root@linux ~]#
```

8. *make* 유틸리티가 설치되어 있는지 확인합니다. 설치되어 있지 않은 경우 설치합니다.

   예제에서는 Oracle Linux 명령 *rpm -qa*의 출력을 *grep* 명령에 연결하고 *make* 문자열을 검색합니다. 이 명령은 버전 정보를 반환하므로, *make* 유틸리티가 설치되어 있는 것입니다.

```
[root@linux ~]#  rpm -qa | grep make
make-4.4.7-3.e16.x86_64
libmake-3.81.20.e16.x86_64
[root@linux ~]#
```

9. *binutils* 패키지가 설치되어 있는지 확인합니다. 설치되어 있지 않은 경우 설치합니다.

   Oracle HSM 설치 소프트웨어에서 Linux 커널을 구성해야 하는 경우 이 패키지의 일부인 *nm* 유틸리티가 필요합니다. 예제에서는 Oracle Linux 명령 *rpm -qa*의 출력을 *grep* 명령에 연결하고 *nm* 문자열을 검색합니다. 이 명령은 버전 정보를 반환하므로, *nm* 유틸리티가 설치되어 있는 것입니다.

```
[root@linux ~]#  rpm -qa | grep nm
binutils-2.20.51.0.2-5.34.e16.x86_64
[root@linux ~]#
```

10. *rpmbuild* 패키지가 설치되어 있는지 확인합니다. 설치되어 있지 않은 경우 설치합니다.

    예제에서는 Oracle Linux 명령 *rpm -qa*의 출력을 *grep* 명령에 연결하고 *rpmbuild* 문자열을 검색합니다. 이 명령은 버전 정보를 반환하므로, *rpmbuild* 패키지가 설치되어 있는 것입니다.

```
[root@linux ~]#  rpm -qa | grep rpmbuild
rpm-build-4.8.0-37.el6.x86_64
[root@linux ~]#
```

11. *rpm* 패키지가 설치되어 있는지 확인합니다. 설치되어 있지 않은 경우 설치합니다.

    Oracle HSM 설치 소프트웨어에서 Linux 커널을 구성해야 하는 경우 이 패키지의 일부인 *rpm2cpio* 유틸리티가 필요합니다. 예제에서는 Oracle Linux 명령 *rpm -qa*의 출력을 *grep* 명령에 연결하고 *rpm* 문자열을 검색합니다. 이 명령은 버전 정보를 반환하므로, 유틸리티가 설치되어 있는 것입니다.

```
[root@linux ~]#  rpm -qa | grep rpm
rpm-4.8.0-27.e16.x86_64
rpm-libs-4.8.0-27.e16.x86_64
```

```
rpm-python-4.8.0-27.e16.x86_64
[root@linux ~]#
```

12. 추가 Linux 클라이언트를 포함하는 솔루션을 준비하는 경우 모든 Linux 클라이언트가 구성될 때까지 “Oracle HSM 클라이언트용 Linux 구성”에 지정된 작업을 반복합니다.
13. 그런 다음 3장. 스토리지 호스트 및 장치 구성으로 이동합니다.

3

3장. 스토리지 호스트 및 장치 구성

Oracle HSM 설치 및 구성을 계속하기 전에 이 장에 요약된 스토리지 구성 작업을 수행합니다. 이 장에서는 다음 항목을 설명합니다.

기본 스토리지 구성

Oracle HSM 파일 시스템에서 기본 디스크 또는 SSD(Solid State Disk) 장치에는 자주 사용 및 수정하는 파일을 저장합니다. 캐시에 대해 디스크 또는 SSD(Solid State Disk) 장치를 구성할 경우 아래 지침을 따르십시오.

기본 캐시에 대한 장치 구성

1. 주 캐시의 시작 용량을 추정하려면 가득 찼을 때 각 파일 시스템에 보관되는 데이터 양을 결정합니다.
2. 파일 시스템 메타데이터를 허용하려면 이 시작 용량을 10% 늘립니다.
3. 고성능 *ma* 유형 파일 시스템을 준비할 경우 *mm* 메타데이터 장치에 대해 하드웨어를 구성합니다. *mm* 메타데이터 장치당 하나의 하드웨어 제어 4디스크 RAID 10(1+0) 볼륨 그룹을 구성하는 것이 좋습니다. 최대 성능을 얻으려면 SSD(Solid State Disk) 사용을 고려합니다.

   스트라이프 미러 RAID 10 어레이의 특성은 Oracle HSM 메타데이터를 저장하는 데 적합합니다. RAID 10 스토리지 하드웨어는 중복도가 높으므로, 중요 메타데이터가 보호됩니다. 대부분의 다른 RAID 구성보다 처리량은 더 높고 지연 시간은 더 짧습니다.

   전용 컨트롤러 하드웨어에 의해 제어되는 어레이는 범용 공유 프로세서에서 실행 중인 소프트웨어에 의해 제어되는 어레이보다 일반적으로 더 높은 성능을 제공합니다.

   솔리드 상태 장치는 본질적으로 자주 업데이트되고 자주 읽는 메타데이터를 저장하는 데 특히 유용합니다.
4. 기본 캐시 스토리지에 대해 외부 디스크 어레이를 사용 중인 경우 파일 시스템 구성에서 각 *md* 또는 *mr* 장치에 대해 3+1 또는 4+1 RAID 5 볼륨 그룹을 구성합니다. 각 볼륨 그룹에 하나의 논리 볼륨(LUN)을 구성합니다.

지정된 수의 디스크에 대해 소규모 3+1 및 4+1 RAID 5 볼륨 그룹은 병렬 처리가 향상되므로 대규모 볼륨 그룹보다 더 우수한 입력/출력(I/O) 성능을 제공합니다. RAID 5 볼륨 그룹의 개별 디스크 장치는 독립적으로 작동하지 않습니다. I/O의 관점에서 각 볼륨 그룹은 단일 장치처럼 작동합니다. 따라서 지정된 수의 디스크를 3+1 및 4+1 볼륨 그룹으로 분할하면 분할하지 않은 대규모 구성보다 더 많은 독립 장치가 생성되고, 병렬 처리 기능이 개선되고, I/O 경합이 감소됩니다.

소규모 RAID 그룹은 패리티 대 스토리지 비율이 더 높기 때문에 더 적은 용량을 제공합니다. 하지만 대부분의 사용자의 경우 성능 향상보다 오프셋이 더 큽니다. 아카이빙 파일 시스템에서 디스크 캐시 용량의 소폭 감소는 아카이브에서 사용 가능한 무제한 용량으로 완벽하게 오프셋됩니다.

볼륨 그룹에서 여러 논리 그룹(LUN)을 구성하면 I/O에 대한 볼륨 간 리소스 경합을 논리적으로 구분하여 한 번에 하나의 I/O만 처리할 수 있습니다. 따라서 I/O 관련 오버헤드가 증가하고 처리량이 감소됩니다.

5. 이제 아카이브 스토리지 구성을 시작합니다.

아카이브 스토리지 구성

다음 작업을 수행합니다.

- SAN 연결 장치 영역 분할
- 아카이브 디스크 스토리지 구성
- 아카이브 테이프 스토리지 구성

SAN 연결 장치 영역 분할

SAN(storage area network)이 Oracle HSM 호스트에서 드라이브와 호스트 버스 어댑터 사이에 통신할 수 있도록 영역이 분할되었는지 확인합니다. 영역 분할을 확인하려면 다음과 같이 하십시오.

Oracle HSM 구성에서 올바르게 모든 장치 영역 분할

1. 호스트에 SAN의 장치가 표시되는지 확인합니다. Solaris 구성 관리 명령 *cfgadm*을 *-al*(연결 지점 목록) 및 *-o show_SCSI_LUN* 옵션과 함께 입력합니다. 출력에서 드라이브 포트의 WWN(World Wide Name)을 검사합니다.

   출력의 첫번째 열에는 연결 지점 ID(*Ap_id*)가 표시됩니다. 연결 지점 ID는 호스트 버스 어댑터의 컨트롤러 번호와 WWN이 콜론으로 구분되어 표시됩니다. 노드가 ADI 인터페이스를 통해 매체 교환기를 제어하는 브리지된 드라이브인 경우 *-o show_SCSI_LUN* 옵션은 노드에 있는 모든 LUN을 표시합니다.

```
root@solaris:~# cfgadm -al -o show_SCSI_LUN
Ap_Id      Type Receptacle Occupant   Condition
c2::500104f000937528   tape connected  configured   unknown
```

```
c3::50060160082006e2,0 tape connected  unconfigured unknown
```

2. 드라이브의 WWN이 *cfgadm -al -o show_SCSI_LUN*의 출력에 나열되지 않는 경우 드라이브가 표시되지 않습니다. SAN 구성에 잘못된 사항이 있습니다. 따라서 SAN 연결과 영역 분할 구성을 다시 확인하십시오. 그런 다음 이전 단계를 반복합니다.
3. *cfgadm -al* 명령의 출력에 드라이브가 구성되어 있지 않은 것으로 표시되는 경우 이번에는 *-c*(구성) 스위치를 사용하여 명령을 다시 실행합니다.

   이 명령은 */dev/rmt*에서 필요한 장치 파일을 구성합니다.

```
root@solaris:~# cfgadm -al
Ap_Id     Type Receptacle Occupant   Condition
c2::500104f000937528   tape connected  configured   unknown
c3::50060160082006e2,0 tape connected  unconfigured unknown
root@solaris:~# cfgadm -c configure 50060160082006e2,0
```

4. 장치 이름과 WWN(World Wide Name) 사이의 연관성을 확인합니다. *ls -al /dev/rmt | grep WWN* 명령을 사용합니다. 여기서 *WWN*은 World Wide Name입니다.

```
root@solaris:~# ls -al /dev/rmt | grep 50060160082006e2,0
lrwxrwxrwx 1 root root 94 May 20 05:05 3un -> /
../../devices/pci@1f,700000/SUNW,qlc@2/fp@0,0/st@w50060160082006e2,0:
```

5. 권장되는 최소 Solaris 패치 레벨인 경우 지금 디스크 스토리지 구성을 수행합니다.
6. 그렇지 않으면 장치에 대한 대상 ID를 가져옵니다.
7. */kernel/drv/st.conf*를 편집합니다. 위에서 확인한 대상 ID를 지정하여 *tape-config-list*에 공급업체에서 지정한 항목을 추가합니다.
8. *st* 모듈을 강제로 다시 로드합니다. *update_drv -f st* 명령을 사용합니다.

```
root@solaris:~# update_drv -f st
root@solaris:~#
```

9. 이제 디스크 스토리지 구성을 수행합니다.

아카이브 디스크 스토리지 구성

Oracle HSM 아카이빙 파일 시스템은 디스크 및 테이프 매체에 파일을 아카이브할 수 있습니다. 디스크 파일 시스템이 디스크 아카이브로 구성된 경우 소프트웨어에서는 파일 시스템을 테이프 카트리지인 것처럼 사용합니다. VSN(볼륨 일련 번호)로 파일 시스템을 찾고 파일 복사본을 테이프 아카이브(*tar*) 파일로 저장합니다.

디스크 기반 아카이브 스토리지는 아카이빙 솔루션의 유연성과 중복성을 높일 수 있습니다. 랜덤 액세스 디스크 장치는 순차 액세스 테이프 장치와 연관된 오버헤드 마운트 및 위치 지정을 유발하지 않습니다. 따라서 매우 많은 수의 작은 파일을 아카이브하고 검색하는 솔루션은 각 파일의 첫번째 복사본을 디스크에 저장할 때 더욱 빠르고 안정적으로 수행할 수 있습니다.

오프사이트 매체에서 복사본을 유지 관리해야 하는 아카이빙 솔루션은 원격 호스트의 NFS로 마운트된 디스크 상주 파일 시스템에 복사본을 기록하여 간단히 수행할 수 있습니다.

Oracle Storage Cloud Software Appliance(OSCSA)는 원격 호스트의 제한된 로컬 디스크 공간을 거의 무제한인 클라우드 기반 스토리지에 대한 프론트엔드 캐시로 사용하여 이러한 NFS 마운트 아카이브 스토리지의 유용성을 더욱 확장할 수 있습니다. 어플라이언스는 NFSv4(Network File System 버전 4)로 구성된 Oracle Linux 7 이상의 호스트, 오픈 소스 Docker Engine 1.6 이상의 컨테이너 관리 소프트웨어 및 Oracle Storage Cloud Software Appliance Docker 이미지로 구성됩니다.

아카이브 디스크 스토리지를 사용할 계획인 경우 먼저 필요한 총 용량, 아카이브 볼륨 수 및 파일 시스템 수 결정을 수행합니다. 그런 다음 Oracle Storage Cloud에서 아카이브 디스크 스토리지를 구성할 계획인 경우 필요한 Oracle Storage Cloud Software Appliances 프로비전을 수행합니다.

용량, 볼륨 및 파일 시스템 요구 사항 결정

예상되는 작업 로드를 처리할 충분한 하드웨어 리소스를 계획합니다. 동시 Oracle HSM 아카이브 및 스테이지 작업이 동일한 물리적 장치 세트에 대해 서로 또는 다른 응용 프로그램과 경합해야 하는 경우 성능이 저하됩니다. 따라서 아래 나열된 지침을 따르십시오.

1. 각 Oracle HSM 작업 및 각 10 ~ 20TB의 아카이브 데이터에 대해 하나의 디스크 볼륨(디스크 또는 RAID 그룹)을 허용합니다.

   디스크 볼륨이 디스크 장치 풀에서 동적으로 할당되는 스토리지인 경우 할당량을 설정합니다. 기본 물리적 스토리지가 초과 구독되지 않도록 합니다.
2. 디스크 볼륨당 하나의 파일 시스템을 허용합니다.

   동일한 물리적 드라이브 또는 RAID 그룹에 있는 LUN에 두 개 이상의 파일 시스템을 구성하지 마십시오.
3. 각 파일 시스템을 단일 디스크 아카이브로 사용하도록 계획합니다.

   하위 디렉토리를 별도의 아카이브 볼륨으로 사용하지 않습니다.
4. 각 파일 시스템을 아카이브 전용으로 사용하도록 계획합니다.

   범용 파일 시스템을 디스크 아카이브로 사용하지 않습니다.
5. 이제 NFS 마운트 디스크 아카이브로 사용할 원격 파일 시스템을 만듭니다.

NFS 마운트 디스크 아카이브로 사용할 원격 파일 시스템 만들기

1. 디스크 아카이브로 사용할 원격 파일 시스템을 만듭니다.

   새 전용 파일 시스템을 만듭니다. 다른 응용 프로그램과 공유해야 하는 기존 범용 파일 시스템은 사용하지 마십시오.

   Oracle HSM 서버를 구성할 때 로컬에 마운트된 디스크 아카이브 파일 시스템은 나중에 만들게 됩니다.

2. Oracle Storage Cloud를 디스크 아카이빙 솔루션의 일부로 사용할 계획인 경우 지금 Oracle Storage Cloud Software Appliance 호스트 구성을 수행합니다.
3. 그렇지 않은 경우 아카이브 테이프 스토리지 구성을 수행합니다.

Oracle Storage Cloud Software Appliance 호스트 구성

1. Oracle Cloud > Public > Infrastructure > Storage > Storage Cloud Software Appliance(*http://docs.oracle.com/cloud/latest/storagecs_common/CSSGU/*)에서 최신 OSCSA 설명서를 다운로드합니다.

   편의를 위해 이 절차에서는 구성 프로세스 및 시스템 요구 사항을 요약합니다. 그러나 항상 OSCSA 제품 설명서 및 *README* 파일에서 전체적인 최신 정보를 확인하십시오.
2. Oracle 영업팀에 문의하십시오. Oracle Storage Cloud Service 구독을 구매하고 Oracle Storage Cloud Software Appliance 이미지를 요청합니다.
3. 각 어플라이언스 호스트에 대해 범용 x86 서버에 최소한 2개의 듀얼 코어 CPU(central processors) 및 4GB의 RAM(random access memory)을 제공합니다.
4. 각 OSCSA 호스트에 Oracle Linux 7(커널 버전 3.10 이상)을 설치합니다.

   Oracle Linux는 Oracle Software Delivery Cloud(*https://edelivery.oracle.com/*)에서 얻을 수 있습니다.
5. 각 OSCSA 호스트에 Docker 1.6.1 이상을 설치합니다.

   Docker는 소프트웨어 컨테이너를 위한 오픈 소스 배포 플랫폼입니다. Docker는 Docker(*https://www.docker.com*)에서 얻을 수 있습니다.
6. 각 OSCSA 호스트에 NFSv4(Network File System 버전 4) 서비스를 설치합니다.

   Oracle HSM 호스트는 NFSv4를 사용하여 OSCSA 프론트엔드 캐시를 구성하는 Linux 파일 시스템을 원격으로 마운트합니다.
7. OSCSA 문서 *Using Oracle Storage Cloud Software Appliance*(*http://docs.oracle.com/cloud/latest/storagecs_common/CSSGU/*)의 지침에 따라 Oracle Storage Cloud Software Appliance를 설치하고 구성합니다.
8. OSCSA 설명서에 나온 대로 OSCSA 캐시 파일 시스템을 만듭니다.
9. 이제 테이프 스토리지를 구성합니다.

아카이브 테이프 스토리지 구성

다음 작업을 수행합니다.

- 라이브러리에 드라이브가 설치되는 순서 결정
- 직접 연결 라이브러리 구성(있는 경우).

라이브러리에 드라이브가 설치되는 순서 결정

자동화된 라이브러리에 여러 개의 드라이브가 있는 경우 Oracle HSM 마스터 구성 파일(*mcf*)의 드라이브 순서가 라이브러리 컨트롤러에서 드라이브가 표시되는 순서와 같아야 합

니다. 이 순서는 드라이브가 호스트에 표시되고 호스트 */var/adm/messages* 파일에 보고되는 순서와 다를 수 있습니다.

각 Oracle HSM 메타데이터 서버 및 datamover 호스트에 대해 아래에 나열된 작업을 수행하여 드라이브 순서를 결정합니다.

- 라이브러리 및 Solaris 호스트 모두에서 드라이브에 대한 식별 정보 수집을 수행합니다.
- 그런 다음 직접 또는 ACSLS 연결 라이브러리에 알맞은 절차에 따라 드라이브를 Solaris 장치 이름에 매핑합니다.

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. 출력을 텍스트 파일로 재지정하여 */dev/scsi/changer/*에 Solaris 논리 장치 이름을 나열합니다.

   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 Solaris 장치 이름을 직접 또는 ACSLS 연결 라이브러리의 드라이브에 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. 텍스트 편집기에서 “라이브러리 및 Solaris 호스트에 대한 드라이브 정보 수집” 절차로 만든 장치 매핑 파일을 엽니다. 정보를 간단한 테이블로 구성합니다.

후속 단계에서 이 정보를 참조해야 합니다. 예제에서는 라이브러리 장치 정보를 위한 헤더 및 공간을 추가하는 동안, *vi* 편집기를 사용하여 */dev/rmt/* 목록에서 권한, 소유권, 날짜 속성을 삭제합니다.

```
root@solaris:~# vi /root/device-mappings.txt
LIBRARY SOLARIS      SOLARIS
DEVICE  LOGICAL      PHYSICAL
NUMBER  DEVICE       DEVICE
------- ----------   ----------------------------------------
        /dev/rmt/0 -> ../../devices/pci@1f,4000/scsi@2,1/st@2,0:
        /dev/rmt/1 -> ../../devices/pci@1f,4000/scsi@4,1/st@5,0:
        /dev/rmt/2 -> ../../devices/pci@1f,4000/scsi@4,1/st@6,0:
        /dev/rmt/3 -> ../../devices/pci@1f,4000/scsi@4/st@1,0:
lrwxrwxrwx 1 root root 40 Mar 18 2014 /dev/rmt/4 -> ../../devices/pci@1f,4000/scsi@4/st@2,0:
```

3. 라이브러리에서 모든 드라이브가 비어 있는지 확인합니다.
4. 아직 Solaris 논리 장치 이름에 매핑되지 않은 라이브러리의 첫번째 드라이브로 테이프를 로드합니다.

   아래 예제의 목적상 HP Ultrium LTO4 테이프 드라이브로 LTO4 테이프를 로드합니다.
5. 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
```

6. 장치 매핑 파일에서 테이프를 보유한 Solaris 장치에 대한 항목을 찾고 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/0 -> ../../devices/pci@1f,4000/scsi@2,1/st@2,0:
   1    /dev/rmt/1 -> ../../devices/pci@1f,4000/scsi@4,1/st@5,0:
        /dev/rmt/2 -> ../../devices/pci@1f,4000/scsi@4,1/st@6,0:
        /dev/rmt/3 -> ../../devices/pci@1f,4000/scsi@4/st@1,0:
:w
```

7. 테이프를 언로드합니다.
8. 장치 매핑 파일에 라이브러리가 Oracle HSM 호스트에 지정하는 모든 장치의 Solaris 논리 장치 이름이 보관될 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/0 -> ../../devices/pci@1f,4000/scsi@2,1/st@2,0:
   1    /dev/rmt/1 -> ../../devices/pci@1f,4000/scsi@4,1/st@5,0:
   3    /dev/rmt/2 -> ../../devices/pci@1f,4000/scsi@4,1/st@6,0:
   4    /dev/rmt/3 -> ../../devices/pci@1f,4000/scsi@4/st@1,0:
:wq
root@solaris:~#
```

9. 매핑 파일을 보관합니다.

   파일 시스템 구성을 수행할 때 이 정보가 필요하며, 완료된 Oracle HSM 구성 백업을 수행할 때 이 정보를 포함시킬 수 있습니다.
10. 다음에는 “직접 연결 라이브러리 구성”으로 이동합니다.

ACSLS 연결 라이브러리의 드라이브를 Solaris 장치 이름에 매핑

1. Oracle HSM Solaris 호스트에 아직 로그인하지 않은 경우 *root*로 로그인합니다.

```
root@solaris:~#
```

2. 텍스트 편집기에서 “라이브러리 및 Solaris 호스트에 대한 드라이브 정보 수집” 절차로 만든 장치 매핑 파일을 열고 간단한 테이블로 구성합니다.

   후속 단계에서 이 정보를 참조해야 합니다. 예제에서는 라이브러리 장치 정보를 위한 헤더 및 공간을 추가하는 동안, *vi* 편집기를 사용하여 */dev/rmt/* 목록에서 권한, 소유권, 날짜 속성을 삭제합니다.

```
root@solaris:~# vi /root/device-mappings.txt
```

```
LOGICAL DEVICE  DEVICE SERIAL NUMBER  ACSLS DEVICE ADDRESS
--------------  --------------------  --------------------------------
/dev/rmt/0
/dev/rmt/1
/dev/rmt/2
/dev/rmt/3
```

3. */dev/rmt/*에 나열된 각 논리 장치 이름에 대해 장치 일련 번호를 표시합니다. *luxadm display /dev/rmt/number* 명령을 사용합니다. 여기서 *number*는 */dev/rmt/*에서 드라이브를 식별합니다.

   예제에서는 */dev/rmt/0* 장치에 대한 일련 번호 *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:~#
```

4. *device-mappings.txt* 파일의 해당 행에 일련 번호를 입력합니다.

   예제에서는 논리 장치 */dev/rmt/0*에 대한 행에 */dev/rmt/0* 장치의 일련 번호 *HU92K00200*을 기록합니다.

```
root@solaris:~# vi /root/device-mappings.txt
LOGICAL DEVICE  DEVICE SERIAL NUMBER  ACSLS DEVICE ADDRESS
--------------  --------------------  --------------------------------
/dev/rmt/0      HU92K00200
/dev/rmt/1
/dev/rmt/2
/dev/rmt/3
:wq
root@solaris:~#
```

5. */dev/rmt/*에 나열된 모든 논리 장치에 대한 장치 일련 번호를 식별하고 *device-mappings.txt* 파일에 결과를 기록할 때까지 이전의 두 단계를 반복합니다.

   예제에는 네 개의 논리 장치가 있습니다.

```
root@solaris:~# vi /root/device-mappings.txt
LOGICAL DEVICE  DEVICE SERIAL NUMBER  ACSLS DEVICE ADDRESS
```

```
-------------  -------------------  --------------------------------
/dev/rmt/0      HU92K00200
/dev/rmt/1      HU92K00208
/dev/rmt/2      HU92K00339
/dev/rmt/3      HU92K00289
:w
root@solaris:~#
```

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    12    331000049255
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
```

7. *device-mappings.txt* 파일의 해당 행에 각 ACSLS 드라이브 주소를 기록합니다. 파일을 저장하고 텍스트 편집기를 닫습니다.

```
root@solaris:~# vi /root/device-mappings.txt
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
```

8. 매핑 파일을 보관합니다.

   파일 시스템 구성을 수행할 때 이 정보가 필요하며, 완료된 Oracle HSM 구성 백업을 수행할 때 이 정보를 포함시킬 수 있습니다.

9. 아카이빙 파일 시스템을 구성할 때 Oracle StorageTek ACSLS 네트워크 연결 라이브러리를 구성합니다. 따라서 고가용성 파일 시스템을 계획할 경우 “고가용성 파일 시스템에 대한 스토리지 구성”으로 이동합니다. 그렇지 않은 경우 4장. *Oracle HSM* 및 *QFS* 소프트웨어 설치 로 이동합니다.

직접 연결 라이브러리 구성

직접 연결 테이프 라이브러리를 구성하려면 하드웨어를 물리적으로 연결하고, 경우에 따라 SCSI 드라이버를 구성해야 합니다(Oracle HSM은 릴리스 5.4 이전에 SAM-QFS가 사용한 *samst* 드라이버 대신 일반 *sgen* 드라이버를 통해 라이브러리 로봇을 제어함). 다음과 같이 하십시오.

1. 라이브러리와 드라이브를 Oracle HSM 서버 호스트에 물리적으로 연결합니다.
2. Oracle HSM을 처음으로 설치하거나 Solaris 11에서 Oracle HSM 또는 SAM-QFS 5 .4 구성을 업그레이드하는 경우 하드웨어가 물리적으로 연결되면 중지합니다.

   Solaris 11에서는 *sgen*이 기본 SCSI 드라이버이므로 Oracle HSM 설치 소프트웨어가 드라이버 별칭과 구성 파일을 자동으로 업데이트할 수 있습니다.
3. Solaris 10 시스템에 Oracle HSM을 설치하는 경우 아래 목록에 있는 드라이버 별칭 중 하나가 *sgen* 드라이버에 지정되었는지 확인합니다. *grep scs.*,08 /etc/driver_aliases* 명령을 사용합니다.

   sgen 드라이버에 다음 별칭 중 하나가 지정될 수 있습니다.
   - *scsa,08.bfcp"* 및/또는 *scsa,08.bvhci*
   - *scsiclass,08*

   예제의 Solaris에서는 *sgen* 드라이버에 대해 *scsiclass,08* 별칭을 사용합니다.

```
root@solaris:~# grep scs.*,08 /etc/driver_aliases
sgen "scsiclass,08"
root@solaris:~#
```

4. *grep* 명령이 *sgen "alias"*를 반환하는 경우 *sgen* 드라이버가 설치되고 별칭에 올바르게 지정됩니다. 여기서 *alias*는 위 목록에 있는 별칭 중 하나입니다. 따라서 다음과 같이 하십시오.
   - 고가용성 파일 시스템을 구성할 경우 "고가용성 파일 시스템에 대한 스토리지 구성"을 참조하십시오.
   - 그렇지 않은 경우 4장. *Oracle HSM* 및 *QFS* 소프트웨어 설치 로 이동합니다.
5. *grep* 명령이 *some-driver "alias"*를 반환하는 경우 해당 별칭이 다른 드라이버에 이미 지정된 것입니다. 여기서 *some-driver*는 *sgen* 이외의 일부 드라이버이고, *alias*는 위에 나열된 별칭 중 하나입니다. 따라서 *sgen* 드라이버에 대해 경로 지향 별칭 만들기를 수행합니다.
6. *grep scs.*,08 /etc/driver_aliases* 명령이 출력을 반환하지 않는 경우 *sgen* 드라이버가 설치되지 않은 것입니다. 따라서 드라이버를 설치하십시오. *add_drv -i scsiclass,08 sgen* 명령을 사용합니다.

   예제에서는 *grep* 명령이 아무것도 반환하지 않습니다. 따라서 *sgen* 드라이버를 설치합니다.

```
root@solaris:~# grep scs.*,08 /etc/driver_aliases
```

```
root@solaris:~# add_drv -i scsiclass,08 sgen
```

7. *add_drv -i scsiclass,08 sgen* 명령이 *Driver (sgen) is already installed* 메시지를 반환하는 경우 드라이버가 이미 설치되어 있지만 연결되지 않은 것입니다. 따라서 지금 연결합니다. *update_drv -a -i scsiclass,08 sgen* 명령을 사용합니다.

   예제에서 *add_drv* 명령은 드라이버가 이미 설치되어 있다는 메시지를 표시합니다. 따라서 드라이버를 연결합니다.

```
root@solaris:~# add_drv -i scsiclass,08 sgen
Driver (sgen) is already installed.
root@solaris:~# update_drv -a -i scsiclass,08 sgen
```

8. *grep scs.*,08 /etc/driver_aliases* 명령에서 *scsiclass,08* 별칭이 *sgen* 드라이버에 지정되었다고 표시하는 경우 드라이버가 제대로 구성된 것입니다.

```
root@solaris:~# grep scs.*,08 /etc/driver_aliases
sgen "scsiclass,08"
root@solaris:~#
```

9. 고가용성 파일 시스템을 구성할 경우 "고가용성 파일 시스템에 대한 스토리지 구성"을 참조하십시오.
10. 그렇지 않은 경우 4장. *Oracle HSM* 및 *QFS* 소프트웨어 설치 로 이동합니다.

sgen 드라이버에 대해 경로 지향 별칭 만들기

예상한 *sgen* 별칭이 다른 드라이버에 이미 지정되어 있는 경우 기존 드라이버 지정에 영향을 주지 않고 *sgen*을 사용하여 지정된 라이브러리를 연결하는 경로 지향 별칭을 만들어야 합니다. 다음과 같이 하십시오.

1. Oracle HSM 서버 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. 시스템 구성을 표시합니다. *cfgadm -vl* 명령을 사용합니다.

   cfgadm 출력은 2행 헤더와 레코드당 두 개의 행을 사용하여 형식이 지정됩니다.

```
root@solaris:~# cfgadm -vl
Ap_Id                Receptacle  Occupant     Condition Information  When
Type        Busy  Phys_Id
c3                      connected   configured   unknown   unavailable
scsi-sas    n     /devices/pci@0/pci@0/pci@2/scsi@0:scsi
c5::500104f0008e6d78 connected   configured   unknown   unavailable
med-changer y     /devices/pci@0/.../SUNW,qlc@0,1/fp@0,0:fc::500104f0008e6d78
...
```

```
root@solaris:~#
```

3. *cfgadm -vl*의 출력에서 라이브러리에 대한 레코드를 찾습니다. 각 레코드의 두번째 행의 *Type* 열에서 *med-changer*를 찾습니다.

   예제에서는 두번째 레코드에서 라이브러리를 찾습니다.

```
root@solaris:~# cfgadm -vl
Ap_Id                Receptacle  Occupant     Condition Information  When
Type        Busy  Phys_Id
c3                   connected   configured   unknown   unavailable
scsi-sas    n     /devices/pci@0/pci@0/pci@2/scsi@0:scsi
c5::500104f0008e6d78 connected   configured   unknown   unavailable
med-changer y     /devices/pci@0/.../SUNW,qlc@0,1/fp@0,0:fc::500104f0008e6d78
...
root@solaris:~#
```

4. 새로운 경로 지향 별칭 역할을 하는 물리적 경로를 가져옵니다. *cfgadm -vl* 출력의 *Phys_Id* 열에 있는 항목에서 */devices* 하위 문자열을 제거합니다.

   예제에서 매체 교환기 레코드의 *Phys_Id* 열에는 */devices/pci@0/pci@0/pci@9/SUNW,qlc@0,1/fp@0,0:fc::500104f0008e6d78* 경로가 포함되어 있으므로, */devices/* 뒤에 오는 문자열의 일부를 별칭으로 선택합니다. 이 물리적 경로는 아래에 사용 가능한 공간에 맞게 약어로 표시됩니다.

```
root@solaris:~# grep scsiclass,08 /etc/driver_aliases
sdrv "scsiclass,08"
root@solaris:~# cfgadm -vl
Ap_Id                Receptacle  Occupant     Condition Information  When
Type        Busy  Phys_Id
c3                   connected   configured   unknown   unavailable
scsi-sas    n     /devices/pci@0/pci@0/pci@2/scsi@0:scsi
c5::500104f0008e6d78 connected   configured   unknown   unavailable
med-changer y     /devices/pci@0/.../SUNW,qlc@0,1/fp@0,0:fc::500104f0008e6d78
...
root@solaris:~#
```

5. 경로 지향 별칭을 만든 다음 *sgen* 드라이버에 지정합니다. *update_drv -d -i '"/path-to-library"' sgen* 명령을 사용합니다. 여기서 *path-to-library*는 이전 단계에서 식별한 경로입니다.

   예제에서는 라이브러리 경로를 사용하여 경로 지향 별칭 *'"/pci@0/pci@0/pci@9/SUNW,qlc@0,1/fp@0,0:fc::500104f0008e6d78"'*을 만듭니다(작은 따옴표와 큰 따옴표에 유의). 명령은 한 행이지만 페이지 레이아웃에 맞도록 두 행으로 형식이 지정되었습니다.

```
root@solaris:~# update_drv -d -i / '"/pci@0/pci@0/pci@9/SUNW,qlc@0,1/fp@0,0:fc::500104f0008e6d78"' sgen
root@solaris:~#
```

이제 라이브러리가 *sgen* 드라이버를 사용하여 구성되었습니다.

6. 고가용성 파일 시스템을 구성할 경우 "고가용성 파일 시스템에 대한 스토리지 구성"으로 이동합니다.
7. 그렇지 않은 경우 4장. *Oracle HSM* 및 *QFS* 소프트웨어 설치 로 이동합니다.

고가용성 파일 시스템에 대한 스토리지 구성

단일 지점 하드웨어 오류가 발생하더라도 계속해서 파일 시스템에 접근할 수 있도록 고가용성 파일 시스템에는 중복된 하드웨어 및 여러 독립된 I/O 경로가 필요합니다. 다음 작업을 수행합니다.

- 다중 경로 I/O에 대해 Solaris 클러스터 노드 구성
- 다중 경로 I/O에 대해 Linux 클라이언트 구성

다중 경로 I/O에 대해 Solaris 클러스터 노드 구성

고가용성 공유 파일 시스템을 구성하려면 사용하는 Solaris Cluster 소프트웨어에 대한 하드웨어 관리 매뉴얼의 권장 사항을 따라야 합니다. 중복된 주 스토리지 장치 및 중복된 I/O 경로를 구성합니다.

파일 시스템 데이터 및 메타데이터를 하드웨어 RAID 장치 또는 SVM(Solaris Volume Manager) 소프트웨어 RAID 볼륨에 저장합니다. Oracle HSM 메타데이터 및 구성 파일을 RAID-10 볼륨 그룹 또는 미러링된 SVM 볼륨에 둡니다. 파일 시스템 데이터를 하드웨어 제어 RAID-10 또는 RAID-5 볼륨 그룹이나 미러링된 SVM 볼륨에 둡니다. 현재 Solaris 릴리스에는 더 이상 SVM이 포함되지 않습니다. Solaris 10 9/10 릴리스에 포함된 소프트웨어 버전을 다운로드하여 설치해야 합니다.

SAN(Storage Area Network) 연결이 단일 지점 실패를 견딜 수 없는지 확인합니다. 각 클러스터 노드에 여러 HBA(호스트 버스 어댑터)를 설치합니다. SAN(Storage Area Network)을 여러 상호 연결 및 중복된 스위치로 구성합니다. Oracle Solaris I/O 다중 경로 소프트웨어로 경로 페일오버를 관리합니다(자세한 내용은 Oracle Solaris 고객 문서 라이브러리의 *Oracle Solaris SAN Configuration and Multipathing Guide* 및 `stmsboot` 매뉴얼 페이지 참조).

다중 경로 I/O에 대해 Linux 클라이언트 구성

Linux 클라이언트에서 DMM(Device Mapper Multipath) 소프트웨어 패키지를 사용하여 경로 페일오버에 대한 중복 스토리지 장치를 구성합니다. DMM 소프트웨어는 호스트와 스토리지 장치를 단일 가상 I/O 장치인 다중 경로로 연결하는 호스트 버스 어댑터, 케이블, 스위치 및 컨트롤러를 모두 관리합니다.

호스트와 스토리지 장치 등을 단일 가상 장치로 연결하는 모든 I/O 경로. 별도의 케이블, 스위치 및 컨트롤러. 다중 경로 지정은 I/O 경로를 통합하여 통합된 경로로 구성된 새로운 장치를 만듭니다. 다중 경로 지정을 사용으로 설정하려면 다음을 수행합니다.

- Device Mapper Multipath 소프트웨어 패키지 설치
- Device Mapper Multipath 소프트웨어 구성

Device Mapper Multipath 소프트웨어 패키지 설치

아래 지침에 따라 Oracle Linux 6.*x*를 실행하는 클라이언트를 구성합니다(다른 Linux 버전의 경우 해당 공급업체의 설명서 참조).

1. Linux 호스트에 *root*로 로그인합니다.

   ```
   [root@linux ~]#
   ```

2. */etc/yum.repos.d* 하위 디렉토리로 변경하고 디렉토리 내용을 나열합니다.

   ```
   [root@linux ~]# cd /etc/yum.repos.d
   [root@linux ~]# ls -l
   total 4
   -rw-r--r--. 1 root root 1707 Jun 25  2012 public-yum-ol6.repo
   [root@linux ~]#
   ```

3. */etc/yum.repos.d* 하위 디렉토리에 *public-yum-ol6.repo* 파일이 포함되어 있지 않을 경우, *wget* 명령을 사용하여 Oracle YUM 저장소에서 다운로드합니다.

   ```
   [root@linux ~]# wget http://public-yum.oracle.com/public-yum-ol6.repo
   -- 2013-02-25 12:50:32 --   http://public-yum.oracle.com/public-yum-ol6.repo
   Resolving public-yum.oracle.com... 14 1.146.44.34
   Connecting to public-yum.oracle.com|141.146.44.34|:80... connected.
   HTTP request sent, awaiting response... 200 OK
   Length: 2411 (2.4K) [text/plain]
   Saving to: ”public-yum-ol6.repo”
   100%[======================================>] 2,411   -- . - K/s   in 0.001s
   2013-02-25 12:50:32 (3.80 MB/s) - ”public-yum-ol6.repo” saved
   [2411/2411]
   [root@linux ~]#
   ```

4. 텍스트 편집기를 사용하여 *public-yum-ol6.repo* 파일을 엽니다. 첫번째 항목 *[ol6_latest]*에 *enabled=1* 라인이 포함되어 있는지 확인합니다.

   예제에서는 *vi* 편집기를 사용합니다. 필요한 라인이 존재하므로 파일을 닫습니다.

   ```
   [root@linux ~]# vi public-yum-ol6.repo
   [ol6_latest]
   ```

```
name=Oracle Linux $releasever Latest ($basearch)
baseurl=http://public-yum.oracle.com/repo/OracleLinux/OL6/latest/$basearch/
gpgkey=http://public-yum.oracle.com/RPM-GPG-KEY-oracle-ol6
gpgcheck=1
enabled=1
...
:q
[root@linux ~]#
```

5. Device Mapper Multipath 소프트웨어 패키지를 찾습니다. *yum search multipath* 명령을 사용합니다.

```
[root@linux ~]# yum search multipath
Loaded plugins: refresh-packagekit, security
============================ N/S Matched: multipath ============================
device-mapper-multipath.x86_64 : Tools to manage multipath devices using
                               : device-mapper
device-mapper-multipath-libs.x86_64 : The device-mapper-multipath modules and
                                    : shared library
  Name and summary matches only, use "search all" for everything.
[root@linux ~]#
```

6. Device Mapper Multipath 소프트웨어를 설치합니다. *yum install device-mapper-multipath* 명령을 사용합니다. 프롬프트가 표시되면 *y*(예)를 입력하여 나열된 패키지 및 종속성을 수락합니다.

```
[root@linux ~]# yum install device-mapper-multipath
Loaded plugins: refresh-packagekit, security
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package device-mapper-multipath.x86_64 0:0.4.9-56.el6_3.1 will be
installed
--> Processing Dependency: device-mapper-multipath-libs = 0.4.9-56.el6_3.1
for package: device-mapper-multipath-0.4.9-56.el6_3.1.x86_64
--> Processing Dependency: libmultipath.so()(64bit)
for package: device-mapper-multipath-0.4.9-56.el6_3.1.x86_64
--> Running transaction check
---> Package device-mapper-multipath-libs.x86_64 0:0.4.9-56.el6_3.1 will be
installed
--> Finished Dependency Resolution
Dependencies Resolved
================================================================================
 Package                     Arch   Version           Repository   Size
================================================================================
```

```
Installing:
 device-mapper-multipath      x86_64 0.4.9-56.el6_3.1  ol6_latest    96 k
Installing for dependencies:
 device-mapper-multipath-libs x86_64 0.4.9-56.el6_3.1  ol6_latest   158 k
Transaction Summary
================================================================================
Install       2 Package(s)
Total download size: 254 k
Installed size: 576 k
Is this ok [y/N]: y
Downloading Packages:
(1/2): device-mapper-multipath-0.4.9-56.el6_3.1.x86_64.r |  96 kB     00:00
(2/2): device-map
per-multipath-libs-0.4.9-56.el6_3.1.x86 | 158 kB     00:00
--------------------------------------------------------------------------------
Total                                           104 kB/s | 254 kB     00:02
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : device-mapper-multipath-libs-0.4.9-56.el6_3.1.x86_64     1/2
  Installing : device-mapper-multipath-0.4.9-56.el6_3.1.x86_64          2/2
  Verifying  : device-mapper-multipath-0.4.9-56.el6_3.1.x86_64          1/2
  Verifying  : device-mapper-multipath-libs-0.4.9-56.el6_3.1.x86_64     2/2
Installed:
  device-mapper-multipath.x86_64 0:0.4.9-56.el6_3.1
Dependency Installed:
  device-mapper-multipath-libs.x86_64 0:0.4.9-56.el6_3.1
Complete!
[root@linux ~]#
```

7. 다중 경로 데몬을 시작합니다. *chkconfig multipathd on* 명령을 사용합니다.

```
[root@linux ~]# chkconfig multipathd on
[root@linux ~]#
```

8. 이제 Device Mapper Multipath 소프트웨어 구성을 수행합니다.

Device Mapper Multipath 소프트웨어 구성

/etc/multipath.conf 파일을 편집하여 Device Mapper Multipath 소프트웨어를 구성합니다. 파일은 일련의 섹션으로 구성되어 있고, 각 섹션에는 관련 속성, 값 및 하위 섹션 세트가 포함되어 있습니다.

- *default* 섹션에서는 다중 경로 소프트웨어 자체를 구성합니다. 기록되는 세부정보 레벨을 지정하고, 페일오버 동작을 정의하며, 필요한 운영체제 명령 및 디렉토리의 위치를 지정합니다.
- *blacklist* 섹션에서는 로컬 시스템 디스크와 같이 다중 경로 구성에서 제외시켜야 하는 장치를 식별합니다. 장치는 WWN/WID(World Wide Name/World Wide Identifier)로 식별하거나 장치 노드 이름 또는 공급업체 및 제품 장치 문자열을 지정하는 정규 표현식으로 식별할 수 있습니다.
- *blacklist_exceptions* 섹션에서는 *blacklist* 섹션의 일반 규칙에서 의도치 않게 제외시키는 경우 다중 경로 구성에 장치를 명시적으로 포함시킬 수 있습니다.
- *multipaths* 섹션에서는 하나 이상의 *multipath* 하위 섹션을 정의할 수 있습니다. 이러한 각 하위 섹션은 World Wide Name으로 지정하는 다중 경로에 특수한 구성을 적용합니다.
- *devices* 섹션에서는 하나 이상의 *device* 하위 섹션을 정의할 수 있습니다. 이러한 각 하위 섹션은 장치에 특수한 다중 경로 구성을 적용합니다.

개별 기본값에 대한 자세한 설명은 주석을 단 샘플 파일 */usr/share/doc/device-mapper-multipath-0.4.9/multipath.conf.annotated*을 참조하십시오.

blacklist_exceptions는 blacklist에서 해당 장치를 식별하더라도 사용되어야 하는 장치를 나열합니다. defaults: DM-Multipath에 대한 일반적인 기본 설정입니다. multipaths: 개별 다중 경로 장치의 특성에 대한 설정입니다. 이러한 값은 구성 파일의 defaults 및 devices 섹션에 지정된 값을 겹쳐씁니다. devices: 개별 스토리지 컨트롤러에 대한 설정입니다. 이러한 값은 구성 파일의 defaults 섹션에 지정된 값을 겹쳐씁니다. 기본적으로 지원되지 않는 스토리지 어레이를 사용 중인 경우 해당 어레이에 대한 devices 하위 섹션을 만들어야 할 수 있습니다. 시스템에서 다중 경로 장치의 속성을 결정할 때 먼저 다중 경로 설정을 확인하고 장치별 설정을 확인한 다음 다중 경로 시스템 기본값을 확인합니다.

4

4장. Oracle HSM 및 QFS 소프트웨어 설치

Oracle HSM은 Oracle Solaris 11에서 표준이 된 IPS(이미지 패키징 시스템)를 사용합니다. IPS는 소프트웨어 패키지의 설치, 업그레이드 및 제거를 간소화하고 조정하는 네트워크 중심 패키지 관리 시스템입니다. 패치 관리를 크게 간소화하고 운영 환경으로의 배치를 쉽게 합니다.

관리자가 Solaris Package Manager 그래픽 데스크탑 응용 프로그램이나 IPS 단말기 명령을 사용하여 Oracle Solaris 소프트웨어 저장소에 액세스하고 필요한 소프트웨어 패키지를 찾고, 다운로드 및 설치하는 동안에 IPS에서는 자동으로 종속성 검사와 패키지 검증을 처리합니다. IPS에서는 유지 관리 기간 동안 새 소프트웨어를 중단 없이 배치할 수 있도록 시스템의 스냅샷을 변경합니다. 따라서 필요한 경우 변경사항을 롤백할 수 있습니다. 따라서 설치 및 업데이트를 실행 중인 운용 시스템에 안전하게 적용할 수 있습니다.

Oracle HSM 소프트웨어를 설치하려면 다음 작업을 수행합니다.

- 소프트웨어 얻기
- Solaris Cluster Software 설치(고가용성 구성에만 해당)
- Oracle HSM 공유 파일 시스템 업그레이드(해당하는 경우)
- 호스트에서 Oracle HSM 소프트웨어 설치, 업그레이드 또는 다운그레이드

이 장은 Oracle HSM 소프트웨어 설치 제거에 대한 간략한 설명으로 마무리합니다.

소프트웨어 얻기

이 절에서는 필요한 설치 소프트웨어 및 소프트웨어 업데이트를 얻는 프로세스를 설명합니다. 다음 절을 참조하십시오.

- 설치 요구 사항 확인
- 소프트웨어 설치 패키지 다운로드

설치 요구 사항 확인

지원되는 Oracle Solaris 및 Linux 운영체제 버전, Oracle 클러스터 소프트웨어 및 기타 필요하거나 지원되는 소프트웨어 패키지를 포함하여 설치 요구 사항에 대한 최신 정보는 Oracle HSM 릴리스 노트, 오라클 고객지원센터(*support.oracle.com*) 및 Oracle HSM wiki 페이지(*wikis.oracle.com/display/hsmqfs/Home*)를 참조하십시오.

소프트웨어 설치 패키지 다운로드

Oracle Software Delivery Cloud에서 Oracle 소프트웨어 제품의 설치 패키지를 다운로드합니다. 기본 절차는 모든 Oracle 제품에 대해 유사합니다.

Oracle HSM 릴리스 6.1 패키지를 다운로드하려면 다음과 같이 하십시오.

1. 웹 브라우저 창에서 *edelivery.oracle.com*을 엽니다.
2. 이 사이트를 아직 사용하지 않았다면 등록합니다.
3. 등록 자격 증명을 사용하여 로그인합니다.
4. 해당되는 소프트웨어 라이센스를 확인하는 확인란을 선택합니다.
5. 소프트웨어에 적용되는 수출 제한에 동의하는 확인란을 선택합니다.
6. Media Pack Search 페이지의 Select a Product Pack 컨트롤 목록에서 Oracle StorageTek Products를 선택합니다.
7. Platform 목록에서 Oracle HSM 소프트웨어를 호스트할 플랫폼 아키텍처의 Oracle Solaris를 선택합니다.
8. Go 버튼을 누릅니다.
9. 결과 목록이 나타나면 Oracle Hierarchical Storage Manager 미디어 팩에 해당하는 라디오 버튼을 누르고 Continue를 누릅니다.
10. Oracle Hierarchical Storage Manager and StorageTek QFS Software Media Pack for Oracle Solaris 페이지가 나타나면 Readme 버튼을 누르고 다운로드 지침을 읽습니다.
11. 계속 Oracle Hierarchical Storage Manager and StorageTek QFS Software Media Pack for Oracle Solaris 페이지에서 View Digest 버튼을 누르고 다이제스트 값을 저장합니다.

    다이제스트는 암호화 해시 함수에서 만든 체크섬입니다. 게시된 다이제스트를 다운로드된 파일에서 로컬로 계산된 다이제스트와 비교하면 다운로드된 파일이 완전하고 그대로 유지되었는지 확인할 수 있습니다. 파일에서 체크섬을 계산하는 방법에 대한 지침은 Solaris *dgst* 및 *md5* 매뉴얼 페이지를 참조하십시오.
12. 계속 Oracle Hierarchical Storage Manager and StorageTek QFS Software Media Pack for Oracle Solaris 페이지에서 라이센스를 받은 제품에 해당하는 Download 버튼을 누릅니다.

    목록에는 Oracle Hierarchical Storage Manager and StorageTek QFS Software에 대한 개별 항목이 포함되어 있습니다. Oracle Hierarchical Storage Manager 미디어 팩에는 아카이빙 소프트웨어와 파일 시스템 소프트웨어가 모두 포함됩니다. Oracle StorageTek QFS 소프트웨어 미디어 팩에는 파일 시스템 소프트웨어만 포함됩니다.
13. Readme 페이지에 표시된 대로 프롬프트가 표시되면 ZIP 아카이브를 로컬 디렉토리에 저장합니다.

    선택한 디렉토리는 모든 Oracle HSM 호스트에서 로컬 네트워크를 통해 액세스할 수 있어야 합니다. 이 장의 예제에서는 *sw_install*이라는 네트워크 파일 서버의 */hsmqfs* 디렉토리에 파일을 다운로드합니다.

14. 필요한 파일을 여러 번 시도해도 다운로드할 수 없는 경우 *edelivery_ww@oracle.com*으로 Software Delivery 고객 서비스에 문의하여 도움을 받으십시오.
15. ZIP 파일을 다운로드했으면 로컬 디렉토리에 파일의 압축을 풉니다.

    아래 예제에서는 */hsmqfs* 하위 디렉토리에 Oracle Hierarchical Storage Manager and StorageTek QFS Software 파일 *Q12345-01.zip*의 압축을 푼 다음 내용을 나열합니다.

```
[sw_install]root@solaris:~# cd /hsmqfs
[sw_install]root@solaris:~# unzip Q12345-01.zip
[sw_install]root@solaris:~# ls Q12345-01/
./       COPYRIGHT.txt       linux.iso             README.txt
../      iso.md5             Oracle-HSM_6.0/
[sw_install]root@solaris:~# ls Oracle-HSM_6.0/
total 42
./        COPYRIGHT.txt      linux1/       solaris_sparc/../       README.txt         linux2/
 solaris_x64/
```

16. 고가용성 파일 시스템을 준비하는 경우 "Solaris Cluster Software 설치(고가용성 구성에만 해당)"로 이동합니다.
17. 다중 호스트 공유 파일 시스템을 업그레이드하는 경우 "Oracle HSM 공유 파일 시스템 업그레이드"로 이동합니다.
18. 그렇지 않은 경우 “호스트에서 Oracle HSM 소프트웨어 설치, 업그레이드 또는 다운그레이드”로 바로 이동합니다.

Solaris Cluster Software 설치(고가용성 구성에만 해당)

고가용성 Oracle HSM 구성을 준비하는 경우 다음과 같이 하십시오.

1. Solaris Cluster Software에 대한 온라인 정보 라이브러리의 설치 및 데이터 서비스 관리 문서에 설명된 대로 각 호스트에서 Oracle Solaris Cluster 및 *SUNW.HAStoragePlus* 데이터 서비스 소프트웨어를 설치합니다.
2. 그런 다음 “호스트에서 Oracle HSM 소프트웨어 설치, 업그레이드 또는 다운그레이드”로 이동합니다.

Oracle HSM 공유 파일 시스템 업그레이드

업그레이드 프로세스 중에도 계속 사용할 수 있어야 하는 공유 파일 시스템에 대한 소프트웨어를 업그레이드하는 경우 롤링 업그레이드를 고려합니다. 활성 서버 외에 하나 이상의 잠재적 메타데이터 서버가 구성되어 있는 경우 비활성 서버를 업데이트하고 업데이트된 서버를 활성화한 다음 기본 서버를 구성 및 다시 활성화한 후 나머지 잠재적 메타데이터 서버와 클라이언트를 업그레이드합니다. 이 롤링 업그레이드 프로세스는 활성 Oracle HSM 메타데이터 서버를 항상 사용할 수 있게 유지하므로 클라이언트가 계속 파일 시스템에 액세스할 수 있습니다.

롤링 업그레이드를 수행하려면 다음 작업을 수행합니다.

- 상당히 오래된 Oracle HSM 릴리스 업그레이드
- 롤링 업그레이드 수행

상당히 오래된 Oracle HSM 릴리스 업그레이드

특정 시점에 공유 파일 시스템의 메타데이터 서버와 클라이언트에 있는 Oracle HSM 소프트웨어는 릴리스 차이가 한 수준 이하여야 합니다. 목표로 하는 업그레이드 릴리스보다 두 수준 이상 이전의 Oracle HSM(또는 SAM-QFS) 소프트웨어를 실행 중인 호스트가 공유 파일 시스템 구성에 포함된 경우 수정 작업을 수행하기 전에는 원하는 릴리스로 업그레이드할 수 없습니다.

다음과 같이 하십시오.

1. 메타데이터 서버와 동일한 릴리스의 Oracle HSM(또는 SAM-QFS) 소프트웨어를 실행 중이지 않는 클라이언트 호스트가 있는 경우 진행하기 전에 서버에서 사용되는 릴리스로 업그레이드합니다.
2. 활성 메타데이터 서버의 Oracle HSM(또는 SAM-QFS) 소프트웨어가 목표로 하는 업그레이드 릴리스보다 두 수준 이상 이전이고 업그레이드 중 파일 시스템을 마운트된 상태로 유지해야 하는 경우, 모든 호스트가 완전히 최신 상태가 될 때까지 한 번에 한 릴리스 레벨씩 롤링 업그레이드 수행 작업을 반복합니다.
3. 활성 메타데이터 서버의 Oracle HSM(또는 SAM-QFS) 소프트웨어가 목표로 하는 업그레이드 릴리스보다 두 수준 이상 이전이고 업그레이드 중 파일 시스템을 마운트된 상태로 유지할 필요가 없는 경우 롤링 업그레이드를 시도하지 마십시오. “호스트에서 Oracle HSM 소프트웨어 설치, 업그레이드 또는 다운그레이드”에 설명된 대로 아카이빙 및 스테이징 프로세스를 중지하고, 파일 시스템을 마운트 해제하고, 각 호스트를 개별적으로 업그레이드합니다.

롤링 업그레이드 수행

1. 계속하기 전에 상당히 오래된 Oracle HSM 릴리스 업그레이드를 수행했는지 확인하십시오!

   롤링 업그레이드를 할 때 호스트의 릴리스가 목표로 하는 업그레이드 릴리스보다 두 수준 이상 이전인 경우 업그레이드가 실패하고 기껏해야 파일 시스템이 일관되지 않은 상태로 유지됩니다.
2. 현재 활성(첫번째) 메타데이터 서버에 *root*로 로그인합니다. 그런 다음 현재 잠재적(두번째) 메타데이터 서버에 역시 *root*로 로그인합니다.

   다음 예제에서는 활성 메타데이터 서버 *first-mds*에 로그인합니다. 그런 다음 두번째 단말기 창에서 보안 셸(*ssh*)을 사용하여 비활성 잠재적 메타데이터 서버 *second-mds*에 로그인합니다.

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

```
[first-mds]root@solaris:~# ssh root@second-mds
Password:
[second-mds]root@solaris:~#
```

3. 현재 비활성 상태인 두번째 메타데이터 서버를 업그레이드합니다. "호스트에서 Oracle HSM 소프트웨어 설치, 업그레이드 또는 다운그레이드"의 절차에 따라 업데이트된 Oracle HSM 소프트웨어를 설치합니다.
4. 업그레이드 단계가 완료되면 두번째 서버를 활성화할 준비를 합니다. 첫번째 활성 메타데이터 서버가 Oracle HSM 또는 SAM-QFS 아카이빙 파일 시스템을 마운트하는 경우 새 아카이빙 및 스테이징 작업을 중지하고, 매체 드라이버를 유휴 설정하고, 현재 작업이 완료될 때까지 기다립니다. 그런 다음 라이브러리 제어 데몬을 중지합니다.

   아카이빙 작업을 중지하는 방법에 대한 자세한 설명은 Oracle Hierarchical Storage Manager and StorageTek QFS Software 유지 관리 및 관리 설명서를 참조하십시오.

```
[first-mds]root@solaris:~# samcmd aridle
[first-mds]root@solaris:~# samcmd stidle
[first-mds]root@solaris:~# samcmd 901 idle
...
[first-mds]root@solaris:~# samcmd a
 ...
Waiting for :arrun
[first-mds]root@solaris:~# samcmd r
...
ty   eq status      act  use  state  vsn
li  801  ---------p    0   0%  off
         empty
...
[first-mds]root@solaris:~# samd stop
[first-mds]root@solaris:~#
```

5. 두번째 메타데이터 서버에서 Oracle HSM 구성 파일을 로드하고 Oracle HSM 프로세스를 시작합니다. *samd config* 명령을 사용합니다.

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

6. 두번째 메타데이터 서버에서 Oracle HSM 파일 시스템을 마운트합니다.

```
[second-mds]root@solaris:~# mount sharefs1
[second-mds]root@solaris:~#
```

7. 새로 업데이트된 두번째 메타데이터 서버를 활성화합니다. 두번째 메타데이터 서버에서 *samsharefs -s server file-system* 명령을 실행합니다. 여기서 *server*는 새로 업

데이트된 메타데이터 서버의 호스트 이름이고 *file-system*은 Oracle HSM 공유 파일 시스템의 이름입니다.

다음 예제에서 잠재적 메타데이터 서버는 *second-mds*이고 파일 시스템 이름은 *sharefs1*입니다.

```
[second-mds]root@solaris:~# samsharefs -s second-mds sharefs1
[second-mds]root@solaris:~#
```

8. 이제 비활성 상태인 첫번째 메타데이터 서버를 업그레이드합니다. "호스트에서 Oracle HSM 소프트웨어 설치, 업그레이드 또는 다운그레이드"의 절차에 따라 업데이트된 Oracle HSM 소프트웨어를 설치합니다.
9. 업그레이드 단계가 완료되면 첫번째 메타데이터 서버를 다시 활성화할 준비를 합니다. 현재 활성 상태인 두번째 메타데이터 서버가 Oracle HSM 아카이빙 파일 시스템을 마운트하는 경우 새 아카이빙 및 스테이징 작업을 중지하고, 매체 드라이버를 유휴 설정하고, 현재 작업이 완료될 때까지 기다립니다. 그런 다음 라이브러리 제어 데몬을 중지합니다.

```
[second-mds]root@solaris:~# samcmd aridle
[second-mds]root@solaris:~# samcmd stidle
...
[second-mds]root@solaris:~# samd stop
[second-mds]root@solaris:~#
```

10. 첫번째 메타데이터 서버에서 Oracle HSM 구성 파일을 로드하고 Oracle HSM 프로세스를 시작합니다. *samd config* 명령을 사용합니다.

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

11. 첫번째 메타데이터 서버에서 Oracle HSM 파일 시스템을 마운트합니다.

```
[first-mds]root@solaris:~# mount sharefs1
[first-mds]root@solaris:~#
```

12. 첫번째 메타데이터 서버를 다시 활성화합니다. 첫번째 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.

    다음 예제에서 잠재적 메타데이터 서버는 *first-mds*이고 파일 시스템 이름은 *sharefs1*입니다.

```
[first-mds]root@solaris:~# samsharefs -s first-mds sharefs1
[first-mds]root@solaris:~#
```

13. 나머지 클라이언트를 업데이트합니다. “호스트에서 Oracle HSM 소프트웨어 설치, 업그레이드 또는 다운그레이드”의 절차에 따라 업데이트된 Oracle HSM 소프트웨어를 설치합니다.
14. 여기서 중지합니다. 업그레이드가 완료되었습니다.

호스트에서 Oracle HSM 소프트웨어 설치, 업그레이드 또는 다운그레이드

개별 호스트에서 Oracle HSM 소프트웨어를 설치, 업그레이드 또는 다운그레이드하려면 다음 작업을 수행합니다.

- Oracle Solaris 호스트에서 Oracle HSM 소프트웨어 설치, 업그레이드 또는 다운그레이드.
- Linux 호스트에서 Oracle HSM 클라이언트 소프트웨어 설치 또는 업데이트(있는 경우)

Oracle Solaris 호스트에서 Oracle HSM 소프트웨어 설치, 업그레이드 또는 다운그레이드

Solaris 호스트에서 Oracle HSM 패키지를 설치, 업그레이드 또는 다운그레이드하려면 먼저 다음 작업을 수행합니다.

- 소프트웨어 변경에 대해 호스트 준비.
- 호스트 아키텍처에 맞는 패키지 찾기.

그런 다음 상황에 가장 맞는 설치 작업을 수행합니다.

- 새 소프트웨어를 설치하고 호스트 운영체제가 Solaris 11 이상인 경우 Solaris IPS(이미지 패키징 시스템) 명령 *pkg install* 사용을 수행합니다.
- IPS 명령 *pkg install*을 사용하여 설치된 소프트웨어를 업그레이드하거나 다운그레이드하는 경우 IPS(이미지 패키징 시스템) 명령 *pkg update* 사용을 수행합니다.
- Solaris 10 호스트에 새 소프트웨어를 설치하는 경우 SVR4 *pkgrm* 및 *pkgadd* 명령 사용을 수행합니다.
- SVR4 명령 *pkgadd*를 사용하여 설치된 소프트웨어를 업그레이드하는 경우 SVR4 *pkgrm* 및 *pkgadd* 명령 사용을 수행합니다.

소프트웨어 변경에 대해 호스트 준비

1. Oracle HSM 소프트웨어가 현재 호스트 시스템에 설치되어 있지 않으면 "호스트 아키텍처에 맞는 패키지 찾기"로 이동합니다.
2. 그렇지 않은 경우 Oracle HSM 서버에 *root*로 로그인합니다.

```
[samqfs1host]root@solaris:~#
```

3. Oracle HSM 소프트웨어가 현재 호스트 시스템에 설치되어 있으면 모든 아카이빙 프로세스를 유휴 설정합니다. *samcmd aridle* 명령을 사용합니다.

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

```
[samqfs1host]root@solaris:~# samcmd aridle
[samqfs1host]root@solaris:~#
```

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

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

```
[samqfs1host]root@solaris:~# samcmd stidle
[samqfs1host]root@solaris:~#
```

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

   아카이빙 프로세스가 *Waiting for :arrun*이면 아카이빙 프로세스가 유휴 상태임을 나타냅니다.

```
[samqfs1host]root@solaris:~# samcmd a
Archiver status samcmd     6.0 10:20:34 Feb 20 2015
samcmd on samqfs1host
sam-archiverd:  Waiting for :arrun
sam-arfind: ...
Waiting for :arrun
```

6. 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.

```
[samqfs1host]root@solaris:~# samcmd u
Staging queue samcmd      6.0  10:20:34 Feb 20 2015
samcmd on solaris.demo.lan
Staging queue by media type: all
sam-stagerd:  Waiting for :strun
[samqfs1host]root@solaris:~#
```

7. 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.

```
[samqfs1host]root@solaris:~# samcmd 801 idle
[samqfs1host]root@solaris:~# samcmd 802 idle
[samqfs1host]root@solaris:~# samcmd 803 idle
[samqfs1host]root@solaris:~# samcmd 804 idle
[samqfs1host]root@solaris:~#
```

8. 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.

```
[samqfs1host]root@solaris:~# samcmd r
Removable media samcmd     6.0 10:37:09 Feb 20 2014
samcmd on samqfs1host
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
[samqfs1host]root@solaris:~#
```

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

```
[samqfs1host]root@solaris:~# samd stop
[samqfs1host]root@solaris:~#
```

10. 파일 시스템이 NFS 또는 SMB/CIFS를 통해 공유되는 경우 파일 시스템 공유를 취소합니다. 메타데이터 서버에서 *unshare mount-point* 명령을 사용합니다. 여기서 *mount-point*는 Oracle HSM 파일 시스템의 마운트 지점 디렉토리입니다.

    첫번째 예제에서는 Oracle HSM 독립형 파일 시스템 *samqfs1*의 NFS 공유를 중지합니다.

```
[samqfs1host]root@solaris:~# unshare /hsmqfs1
[samqfs1host]root@solaris:~#
```

두번째 예제에서는 Oracle HSM 공유 파일 시스템 *samqfs2*의 NFS 공유를 중지합니다.

```
[samqfs2server]root@solaris:~# unshare /hsmqfs2
[samqfs2server]root@solaris:~#
```

11. 모든 Oracle HSM 파일 시스템을 마운트 해제합니다.

    첫번째 예제에서는 비공유 독립형 파일 시스템 *samqfs1*을 마운트 해제합니다.

```
[samqfs1host]root@solaris:~# umount samqfs1
```

    두번째 예제에서는 공유 파일 시스템 *samqfs1*을 먼저 클라이언트에서 마운트 해제한 다음 서버에서 마운트 해제합니다. 클라이언트가 마운트 해제되는 데 *60*초 정도 걸립니다.

```
[samqfs2server]root@solaris:~# ssh root@samqfs2client1
Password:
[samqfs2client1]root@solaris:~# umount /hsmqfs2
[samqfs2client1]root@solaris:~# exit
[samqfs2server]root@solaris:~#

[samqfs2server]root@solaris:~# ssh root@samqfs2client1
Password:
[samqfs2client2]root@solaris:~# umount /hsmqfs2
[samqfs2client2]root@solaris:~# exit
[samqfs2server]root@solaris:~# umount -o await_clients=60 /sharefs2
```

12. 현재 SAM-QFS 5.3 이전 버전이 설치되어 있는 경우 모든 패키지를 제거합니다. *pkgrm SUNWsamfsu SUNWsamfsr* 명령(QFS만 설치된 경우 *pkgrm SUNWqfsu SUNWqfsr* 명령)을 사용합니다.

    지정된 순서로 패키지를 제거합니다. *SUNWsamfsu*로 시작하여 *SUNWsamfsr*로 끝냅니다. 아래 예제에서는 *yes* 응답을 명령으로 파이프하여 모든 질문에 자동으로 답하도록 합니다.

```
[host1]root@solaris:~# yes | pkgrm SUNWsamfsu SUNWsamfsr
```

13. 이제 호스트 아키텍처에 맞는 Oracle HSM 패키지 찾기를 수행합니다.

호스트 아키텍처에 맞는 패키지 찾기

1. Oracle HSM 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. Oracle HSM 다운로드 파일의 압축을 푼 디렉토리로 변경한 다음 원하는 버전의 패키지가 저장된 하위 디렉토리를 찾습니다.

   처음에 릴리스된 패키지는 *Oracle_HSM_release-number*(또는 *STK_QFS_release-number*) 하위 디렉토리에 저장됩니다. 여기서 *release-number*는 주 릴리스 번호와 부 릴리스 번호가 점으로 연결된 것입니다(*Oracle_HSM_6.0+/*). 패치 릴리스(있는 경우)는 추가 *-patch-number* 접미어를 가진 유사한 하위 디렉토리에 있습니다. 여기서 *patch-number*는 2자리 패치 시퀀스 번호입니다(*Oracle_HSM_6.0-01/*).

   예제에서는 초기 소프트웨어 릴리스의 다운로드 디렉토리인 *Oracle_HSM_6.0/*으로 변경하고 내용을 나열합니다.

```
root@solaris:~# cd /net/sw-install/hsmqfs/Oracle_HSM_6.0/
root@solaris:~# ls -1
./
../
linux1/
linux2/
Notices/
README.txt
solaris_sparc/
solaris_x64/
```

3. 호스트 아키텍처에 해당하는 하위 디렉토리(*solaris_sparc/* 또는 *solaris_x64/*)로 변경하고 내용을 나열합니다.

   다음 예제에서는 *solaris_sparc/* 하위 디렉토리로 변경합니다.

```
root@solaris:~# cd solaris_sparc/
root@solaris:~# ls -1
./
../
S10/
S11/
S11_ips/
fsmgr_6.1.zip
fsmgr_setup*
```

4. Solaris 11 이상이 호스트에 설치된 경우 이미지 패키징 시스템을 사용하여 소프트웨어를 설치, 업그레이드 또는 다운그레이드할 수 있습니다. 다음 중 하나로 이동합니다.
   - "IPS(이미지 패키징 시스템)를 사용하여 소프트웨어 설치".
   - “IPS(이미지 패키징 시스템)를 사용하여 소프트웨어 업그레이드 또는 다운그레이드”.

5. Solaris 11 이상이 호스트에 설치된 경우 *pkgadd* 방법을 사용하여 소프트웨어를 설치, 업그레이드 또는 다운그레이드할 수도 있습니다. "SVR4 *pkgrm* 및 *pkgadd* 명령을 사용하여 소프트웨어 업그레이드 또는 다운그레이드"를 참조하십시오.
6. Solaris 10이 호스트에 설치된 경우 *pkgadd* 방법을 사용하여 소프트웨어를 설치, 업그레이드 또는 다운그레이드할 수 있습니다. "SVR4 *pkgrm* 및 *pkgadd* 명령을 사용하여 소프트웨어 업그레이드 또는 다운그레이드"으로 이동합니다.

IPS(이미지 패키징 시스템)를 사용하여 소프트웨어 설치

일반적으로 Solaris 11 이상을 실행하는 호스트에서 Oracle HSM 소프트웨어를 설치, 업그레이드 또는 다운그레이드하려면 IPS(이미지 패키징 시스템) 명령을 사용해야 합니다. 메타데이터 서버 및 공유 파일 시스템 클라이언트(있는 경우)를 비롯한 각 호스트에 대해 다음과 같이 하십시오.

1. 아직 수행하지 않은 경우 호스트 아키텍처에 맞는 Oracle HSM 패키지 찾기를 수행합니다.
2. Solaris 11 IPS 패키지에 대한 저장소 디렉토리인 *repo.samqfs/*로 변경합니다.

   예제에서는 Oracle HSM 6.0에 대한 저장소 디렉토리인 *Oracle_HSM_6.0/solaris_sparc/S11_ips/repo.samqfs*로 변경합니다.

```
root@solaris:~# cd repo.samqfs/
root@solaris:~#
```

3. Oracle Hierarchical Storage Manager and StorageTek QFS Software 패키지를 모두 설치하려면 *pkg install -g . --accept SUNWsamfs SUNWsamqassy*를 사용합니다. 여기서 *.*은 현재 디렉토리(저장소)이고 *SUNWsamfs* 및 *SUNWsamqassy*는 Oracle HSM 이미지 패키징 시스템 패키지 이름입니다.

```
root@solaris:~# pkg install -g . --accept SUNWsamfs SUNWsamqassy
Creating plan
...
* The licence and distribution terms for any publically available version or
 * derivative of this code cannot be changed.  i.e. this code cannot simply be
 * copied and put under another distribution licence
 * [including the GNU Public Licence.]
 */
           Packages to install:   2
       Create boot environment:  No
Create backup boot environment: Yes
DOWNLOAD                                  PKGS         FILES    XFER (MB)   SPEED
Completed                                  2/2       520/520    21.4/21.4    0B/s
PHASE                                        ITEMS
Installing new actions                     693/693
Updating package state database              Done
```

```
Updating image state                            Done
Creating fast lookup database                   Done
```

4. QFS 소프트웨어 패키지만 설치하려면 *pkg install -g . --accept SUNWqfs SUNWsamqassy* 명령을 사용합니다. 여기서 .은 현재 디렉토리(저장소)이고 *SUNWqfs* 및 *SUNWsamqassy*는 Oracle HSM 이미지 패키징 시스템 패키지 이름입니다.

```
root@solaris:~# pkg install -g . --accept SUNWqfs SUNWsamqassy
Creating plan
...
* The licence and distribution terms for any publically available version or
 * derivative of this code cannot be changed.  i.e. this code cannot simply be
 * copied and put under another distribution licence
 * [including the GNU Public Licence.]
 */
           Packages to install:   2
       Create boot environment:  No
Create backup boot environment: Yes
DOWNLOAD                                  PKGS         FILES    XFER (MB)   SPEED
Completed                                  2/2       520/520    21.4/21.4    0B/s
PHASE                                        ITEMS
Installing new actions                     693/693
Updating package state database              Done
Updating image state                         Done
Creating fast lookup database                Done
```

5. 패키지 설치가 완료되면 사후 설치 스크립트 *sam-qfs-post-install*을 실행합니다. 이 스크립트는 Oracle HSM 설치 디렉토리(*/opt/SUNWsamfs/* 또는 */opt/SUNWqfs/)*의 *util/* 하위 디렉토리에 있습니다.

   다음 예제에서는 */opt/SUNWsamfs/util/sam-qfs-post-install*을 실행합니다.

```
root@solaris:~# /opt/SUNWsamfs/util/sam-qfs-post-install
SUNWsamfs IPS package installed.

inquiry.conf may have been updated for this release.
...
root@solaris:~#
```

6. Oracle HSM 디렉토리 */opt/SUNWsamfs/bin* 및 */opt/SUNWsamfs/sbin*(또는 */opt/SUNWqfs/bin* 및 */opt/SUNWqfs/sbin*)을 시스템 *PATH* 변수에 추가합니다(해당 디렉토리가 경로에 아직 없는 경우).
7. Oracle HSM 디렉토리 */opt/SUNWsamfs/man*(또는 */opt/SUNWqfs/man*)을 시스템 *MANPATH* 변수에 추가합니다(해당 디렉토리가 man 경로에 아직 없는 경우).

8. 계획된 Oracle HSM 구성에 추가 Solaris 호스트가 포함된 경우 모든 호스트에 소프트웨어가 설치될 때까지 이 절차를 처음부터 반복합니다.
9. 계획된 Oracle HSM 구성에 Linux 호스트가 공유 파일 시스템 클라이언트로 포함된 경우 “Linux 호스트에서 Oracle HSM 클라이언트 소프트웨어 설치 또는 업데이트”로 이동합니다.
10. 그렇지 않으면 5장. *samsetup* 구성 마법사 사용 또는 6장. 기본 파일 시스템 구성 으로 이동합니다.

IPS(이미지 패키징 시스템)를 사용하여 소프트웨어 업그레이드 또는 다운그레이드

원래 IPS를 사용하여 설치된 Oracle HSM 소프트웨어를 업그레이드 또는 다운그레이드하려면 IPS(이미지 패키징 시스템) 명령을 사용합니다.

메타데이터 서버 및 공유 파일 시스템 클라이언트(있는 경우)를 비롯한 각 호스트에 대해 다음과 같이 하십시오.

1. 아직 수행하지 않은 경우 호스트 아키텍처에 맞는 Oracle HSM 패키지 찾기를 수행합니다.
2. Oracle Hierarchical Storage Manager and StorageTek QFS Software 패키지를 저장소의 최신 버전으로 업그레이드하려면 *pkg update -g . --accept SUNWsamfs SUNWsamqassy*를 사용합니다. 여기서 *.*은 현재 디렉토리(저장소)이고 *SUNWsamfs* 및 *SUNWsamqassy*는 Oracle HSM 이미지 패키징 시스템 패키지 이름입니다.

```
root@solaris:~# pkg update -g . --accept SUNWsamfs SUNWsamqassy
...
root@solaris:~#
```

3. QFS 소프트웨어 패키지만 저장소의 최신 버전으로 업그레이드하려면 *pkg update -g . --accept SUNWqfs SUNWsamqassy*를 사용합니다. 여기서 *.*은 현재 디렉토리(저장소)이고 *SUNWqfs* 및 *SUNWsamqassy*는 Oracle HSM 이미지 패키징 시스템 패키지 이름입니다.

```
[host1]root@solaris:~# pkg update -g . --accept SUNWqfs SUNWsamqassy
...
root@solaris:~#
```

4. Oracle HSM 패키지를 다운그레이드하거나 지정한 버전으로 업그레이드하려면 먼저 원하는 패키지에 대한 FMRI(결함 관리 리소스 식별자)를 가져옵니다. *pkg info -r -g . package-name* 명령을 사용합니다. 여기서 *.*은 현재 디렉토리를 지정하고 *package-name*은 Oracle HSM 패키지의 이름입니다.

   예제에서는 Oracle HSM 버전 6.0.0이 호스트에 설치되어 있습니다.

```
root@solaris:~# samcmd l
```

```
Usage information samcmd      6.0.0 14:06:20 Feb  20 2015 ...
root@solaris:~#
```

SAM-QFS 5.4.6으로 다운그레이드해야 합니다. 따라서 버전 5.4.6에 대한 IPS 저장소 *Oracle_HSM_6.0/solaris_sparc/S11_ips/repo.samqfs*에서 *SUNWsamfs* 및 *SUNWsamqassy*에 대해 *pkg info* 명령을 실행합니다.

```
root@solaris:~# pwd
/net/Oracle_HSM_6.0/solaris_sparc/S11_ips/repo.samqfs
root@solaris:~# pkg info -r -g . SUNWsamfs
          Name: SUNWsamfs
       Summary: StorageTek SAM and StorageTek SAM-QFS software
   Description: StorageTek Storage and Archive Manager File System
      Category: System/File System
         State: Not installed
     Publisher: samqfs
       Version: 5.4
 Build Release: 5.11
        Branch: None
Packaging Date: Tue Jul 08 22:56:56 2014
          Size: 88.64 MB
          FMRI: pkg://hsmqfs/SUNWsamfs@5.4,5.11:20140708T225656Z

root@solaris:~# pkg info -r -g . SUNWsamqassy
          Name: SUNWsamqassy
       Summary: StorageTek QFS and Storage Archive Manager SAM-QFS IPS assembly
 services
   Description: SAM-QFS IPS Assembly Services
      Category: System/File System
         State: Installed
     Publisher: samqfs
       Version: 5.4
 Build Release: 5.11
        Branch: None
Packaging Date: Fri Sep 26 17:21:35 2014
          Size: 15.15 kB
          FMRI: pkg://hsmqfs/SUNWsamqassy@5.4,5.11:20140926T172135Z
root@solaris:~#
```

5. 그런 다음 Oracle HSM 패키지를 다운그레이드하거나 지정한 버전으로 업그레이드하려면 *pkg update -g . fmri* 명령을 실행합니다. 여기서 *.*은 현재 디렉토리를 지정하고 *fmri*는 원하는 소프트웨어 버전의 결함 관리 리소스 식별자를 지정합니다.

   예제에서는 *SUNWsamfs* 및 *SUNWsamqassy* 패키지 5.4.6 버전의 FMRI를 지정합니다.

```
root@solaris:~# pkg update -g . SUNWsamfs@5.4,5.11:20140708T225656Z
           Packages to update:   1
       Create boot environment:  No
Create backup boot environment: Yes
DOWNLOAD                                  PKGS         FILES    XFER (MB)
 SPEEDCompleted                             1/1       160/160    19.2/19.2  3.4M/s
PHASE                                       ITEMS
Updating modified actions                 172/172
Updating package state database              Done
Updating package cache                        1/1
Updating image state                         Done
Creating fast lookup database                Done
Updating package cache                        3/3
root@solaris:~# pkg update -g . SUNWsamqassy@5.4,5.11:20140926T172135Z
...
root@solaris:~#
```

6. *pkg update* 명령이 완료되었으면 시스템을 다시 시작합니다. Solaris *reboot* 명령을 사용합니다.

   ```
   root@solaris:~# reboot
   ```

7. 계획된 Oracle HSM 구성에 추가 Solaris 호스트가 포함된 경우 모든 호스트에서 소프트웨어가 업데이트 또는 다운그레이드될 때까지 이 절차를 처음부터 반복합니다.
8. 계획된 Oracle HSM 구성에 Linux 호스트가 공유 파일 시스템 클라이언트로 포함된 경우 “Linux 호스트에서 Oracle HSM 클라이언트 소프트웨어 설치 또는 업데이트”로 이동합니다.

SVR4 *pkgrm* 및 *pkgadd* 명령을 사용하여 소프트웨어 설치

Solaris 10을 실행하는 호스트에 Oracle HSM 소프트웨어를 설치하는 경우 및 원래 SVR4 명령을 사용하여 설치된 소프트웨어를 업그레이드하는 경우 SVR4 패키지 명령을 사용합니다.

메타데이터 서버 및 공유 파일 시스템 클라이언트(있는 경우)를 비롯한 각 Oracle HSM Solaris 호스트에 대해 다음과 같이 하십시오.

1. 아직 수행하지 않은 경우 호스트 아키텍처에 맞는 Oracle HSM 패키지 찾기를 수행합니다.
2. Oracle Hierarchical Storage Manager and StorageTek QFS Software 패키지를 모두 설치하려면 *pkgadd -d . SUNWsamfsr SUNWsamfsu* 명령을 사용하고 기본값을 모두 적용합니다.

   SUNWsamfsu 패키지를 설치하려면 먼저 *SUNWsamfsr* 패키지를 설치해야 합니다. 예제에서는 사용하는 운영체제의 디렉토리인 *Oracle_HSM_6.0/solaris_sparc/S10*에

있는지 확인합니다. 그런 다음 *yes* 응답을 명령으로 파이프하여 모든 질문에 자동으로 답하도록 합니다.

```
root@solaris:~# pwd
/net/Oracle_HSM_6.0/solaris_sparc/s10
root@solaris:~# yes | pkgadd -d . SUNWsamfsr SUNWsamfsu
```

3. QFS 소프트웨어 패키지만 설치하려면 *pkgadd -d . SUNWqfsr SUNWqfsu* 명령을 사용하고 기본값을 모두 적용합니다.

   SUNWqfsu 패키지를 설치하려면 먼저 *SUNWqfsr* 패키지를 설치해야 합니다. 아래 예제에서는 *yes* 응답을 명령으로 파이프하여 모든 질문에 자동으로 답하도록 합니다.

```
root@solaris:~# yes | pkgadd -d . SUNWqfsr SUNWqfsu
```

4. 계획된 Oracle HSM 구성에 Linux 호스트가 공유 파일 시스템 클라이언트로 포함된 경우 "Linux 호스트에서 Oracle HSM 클라이언트 소프트웨어 설치 또는 업데이트"로 이동합니다.
5. 그렇지 않으면 5장. ***samsetup*** 구성 마법사 사용 또는 6장. 기본 파일 시스템 구성 으로 이동합니다.

SVR4 *pkgrm* 및 *pkgadd* 명령을 사용하여 소프트웨어 업그레이드 또는 다운그레이드

Solaris 10을 실행하는 호스트에서 Oracle HSM 소프트웨어를 업그레이드 또는 다운그레이드하는 경우 및 원래 SVR4 명령을 사용하여 설치된 소프트웨어를 업그레이드 또는 다운그레이드하는 경우 SVR4 패키지 명령을 사용합니다.

메타데이터 서버 및 공유 파일 시스템 클라이언트(있는 경우)를 비롯한 각 Oracle HSM Solaris 호스트에 대해 다음과 같이 하십시오.

1. Oracle HSM 소프트웨어를 SAM-QFS 5.3으로 다운그레이드하는 경우 먼저 구성 파일을 이전 소프트웨어에서 지정된 위치로 복원합니다. */opt/SUNWsamfs/sbin/backto 5.3* 명령을 사용합니다.

   backto 명령은 파일을 이전 위치 및 형식으로 복원합니다. 자세한 내용은 *backto* 매뉴얼 페이지를 참조하십시오.

   예제에서는 Oracle SAM 5.3에서 사용하기 위해 Oracle HSM 6.0 구성 파일을 변환합니다.

```
root@solaris:~# /opt/SUNWsamfs/sbin/backto 5.3 ...
root@solaris:~#
```

2. 현재 설치된 Oracle HSM 패키지를 모두 제거합니다. *pkgrm SUNWsamfsu SUNWsamfsr* 명령(QFS만 설치된 경우 *pkgrm SUNWqfsu SUNWqfsr* 명령)을 사용합니다.

   지정된 순서로 패키지를 제거합니다. *SUNWsamfsu*로 시작하여 *SUNWsamfsr*로 끝냅니다. 아래 예제에서는 *yes* 응답을 명령으로 파이프하여 모든 질문에 자동으로 답하도록 합니다.

```
root@solaris:~# yes | pkgrm SUNWsamfsu SUNWsamfsr
```

3. 아직 수행하지 않은 경우 호스트 아키텍처에 맞는 Oracle HSM 패키지 찾기를 수행합니다.
4. Oracle Hierarchical Storage Manager and StorageTek QFS Software 패키지를 모두 설치하려면 *pkgadd -d . SUNWsamfsr SUNWsamfsu* 명령을 사용하고 기본값을 모두 적용합니다.

   SUNWsamfsu 패키지를 설치하려면 먼저 *SUNWsamfsr* 패키지를 설치해야 합니다. 예제에서는 사용하는 운영체제의 올바른 디렉토리인 *Oracle_HSM_6.0/solaris_sparc/S10*에 있는지 확인합니다. 그런 다음 *yes* 응답을 명령으로 파이프하여 모든 질문에 자동으로 답하도록 합니다.

```
root@solaris:~# pwd
/net/Oracle_HSM_6.0/solaris_sparc/s10
root@solaris:~# yes | pkgadd -d . SUNWsamfsr SUNWsamfsu
```

5. QFS 소프트웨어 패키지만 설치하려면 *pkgadd -d . SUNWqfsr SUNWqfsu* 명령을 사용하고 기본값을 모두 적용합니다.

   SUNWqfsu 패키지를 설치하려면 먼저 *SUNWqfsr* 패키지를 설치해야 합니다. 아래 예제에서는 *yes* 응답을 명령으로 파이프하여 모든 질문에 자동으로 답하도록 합니다.

```
root@solaris:~# pwd
/net/Oracle_HSM_6.0/solaris_sparc/s10
root@solaris:~# yes | pkgadd -d . SUNWqfsr SUNWqfsu
```

6. 계획된 Oracle HSM 구성에 Linux 호스트가 공유 파일 시스템 클라이언트로 포함된 경우 "Linux 호스트에서 Oracle HSM 클라이언트 소프트웨어 설치 또는 업데이트"로 이동합니다.
7. 그렇지 않으면 5장. *samsetup* 구성 마법사 사용 또는 6장. 기본 파일 시스템 구성 으로 이동합니다.

Linux 호스트에서 Oracle HSM 클라이언트 소프트웨어 설치 또는 업데이트

Oracle HSM 공유 파일 시스템의 각 Linux 클라이언트에 대해 다음과 같이 하십시오.

1. Linux 클라이언트에 *root*로 로그인합니다.

```
[root@linux ~]#
```

2. 마운트된 Oracle HSM 파일 시스템을 모두 마운트 해제합니다.
3. 이전 Oracle HSM 패키지를 제거합니다. */var/opt/SUNWsamfs/Uninstall* 스크립트를 실행합니다.

```
[root@linux ~]# /var/opt/SUNWsamfs/Uninstall
```

4. Linux 클라이언트 ISO 이미지를 찾습니다. ISO 이미지는 Oracle HSM 설치 소프트웨어를 다운로드한 디렉토리에 있습니다("소프트웨어 얻기" 참조).

   예제에서는 *ssh*를 사용하여 저장소 호스트 *sw-install*(IP 주소 *192.168.0.2*)에 로그인합니다. */hsmqfs* 디렉토리에서 소프트웨어를 찾습니다.

```
[root@linux ~]# ssh root@sw-install
Password:
[sw_install]root@solaris:~# ls -1 /hsmqfs
./       COPYRIGHT.txt       linux.iso             README.txt
../      iso.md5             Oracle-HSM_6.0/
```

5. Linux 호스트에서 임시 디렉토리를 만듭니다.

   예제에서는 */hsmtemp* 디렉토리를 만듭니다.

```
[root@linux ~]# mkdir /hsmtemp
[root@linux ~]#
```

6. Linux 호스트가 *linux.iso* 이미지를 사용할 수 있도록 설정합니다. NFS가 이미지를 보관하는 원격 디렉토리를 방금 만든 임시 디렉토리에 마운트합니다. *mount -t nfs repository-host-IP:hsm-repository-dir temp-dir* 명령을 사용합니다. 설명:
   - *-t nfs*는 마운트되는 파일 시스템 유형을 지정합니다.
   - *repository-host-IP*는 설치 소프트웨어를 호스트하는 서버의 IP 주소입니다.
   - *hsm-repository-dir*은 Oracle HSM 설치 소프트웨어를 보관하는 디렉토리입니다.
   - *temp-dir*은 Linux 호스트에 만든 임시 디렉토리입니다.

   예제에서는 *sw-install* 호스트(*192.168.0.2*)의 NFS 마운트 디렉토리 */hsmqfs*를 마운트 지점 디렉토리 */hsmtemp*에 만듭니다.

```
[root@linux ~]# mount -t nfs 192.168.0.2:/hsmqfs /hsmtemp
[root@linux ~]#
```

7. *linux.iso* 이미지를 Linux 호스트에 마운트합니다. *mount -o ro,loop -t iso9660 temp-dir/linux.iso /mnt* 명령을 사용합니다. 설명:
   - *-o*는 마운트 옵션 목록을 지정합니다.
   - *ro*는 이미지를 읽기 전용으로 마운트합니다.
   - *loop*는 이미지를 루프 장치로 마운트합니다.
   - *-t iso9660*은 마운트되는 파일 시스템 유형을 지정합니다.
   - *temp-dir*은 원격 이미지 저장소 디렉토리가 마운트되는 임시 디렉토리입니다.
   - */mnt*는 Linux 시스템의 표준 임시 마운트 지점 디렉토리입니다.

   예를 들어, ISO 이미지는 */hsmtemp*에 있습니다.

```
[root@linux ~]# mount -o ro,loop -t iso9660 /hsmtemp/linux.iso /mnt
[root@linux ~]#
```

8. 설치 프로그램을 실행합니다. */mnt/linux1/Install* 명령을 사용합니다.

```
[root@linux ~]# /mnt/linux1/Install
```

9. 설치 프로그램에서 설치된 버전의 Linux 커널을 인식하지 못하는 경우 사용자 정의 커널을 만들라는 메시지를 표시합니다. *yes*를 입력합니다.

```
[root@linux ~]# ./Install
...
A direct match for your kernel wasn't found. Attempt creating a custom rpm for your kernel (yes/
no)? yes
```

   Linux 커널의 변형은 많습니다. Oracle HSM 설치 프로그램에서는 최대한 많은 수의 변형을 지원할 수 있도록 사용자 정의 커널 모듈을 컴파일합니다.

10. 화면에 나타난 지침을 따릅니다.
11. SuSE Linux 클라이언트를 설치하는 경우 매뉴얼 페이지를 인식하도록 시스템을 구성합니다. 텍스트 편집기에서 */etc/manpath.config* 파일을 열고 *SECTION* 매개변수 값에 *1m*를 추가합니다.

    이 예에서는 *vi* 편집기를 사용합니다.

```
[root@linux ~]# vi /etc/manpath.config
...
#-------------------------------------------------------------
# Section names. Manual sections will be searched in the order listed here;
# the default is 1, n, l, 8, 3, 2, 5, 4, 9, 6, 7. Multiple SECTION
# directives may be given for clarity, and will be concatenated together in
# the expected way.
# If a particular extension is not in this list (say, 1mh), it will be
```

```
# displayed with the rest of the section it belongs to. The effect of this
# is that you only need to explicitly list extensions if you want to force a
# particular order. Sections with extensions should usually be adjacent to
# their main section (e.g. "1 1mh 8 ...").
SECTION 1 1m n l 8 3 2 3posix 3pm 3perl 5 4 9 6 7
```

12. 계획된 Oracle HSM 구성에 추가 Linux 클라이언트 호스트가 포함된 경우 모든 호스트에 클라이언트 소프트웨어가 설치될 때까지 이 절차를 처음부터 반복합니다.
13. 그렇지 않으면 5장. ***samsetup*** 구성 마법사 사용 또는 6장. 기본 파일 시스템 구성 으로 이동합니다.

Oracle HSM 소프트웨어 제거

이 절에서는 다음 절차를 설명합니다.

- Solaris 호스트에서 Oracle HSM 제거
- Linux 호스트에서 Oracle HSM 클라이언트 제거.

> **주의:**
>
> 기존 구성을 사용하여 Oracle HSM을 업그레이드 또는 다시 설치하려는 경우 소프트웨어를 제거하지 마십시오. 제거하면 모든 구성 파일이 제거됩니다. 대신, "Oracle Solaris 호스트에서 Oracle HSM 소프트웨어 설치, 업그레이드 또는 다운그레이드"에 설명된 업그레이드 방법 중 하나를 사용합니다.

Solaris 호스트에서 Oracle HSM 제거

소프트웨어를 완전히 제거하고 구성 파일을 제거하려면 다음과 같이 하십시오.

1. 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. 소프트웨어가 Solaris IPS(이미지 패키징 시스템)를 사용하여 Solaris 11 이상에 설치된 경우 *pkg uninstall SUNWsamfs SUNWsamqassy*(또는 QFS 소프트웨어가 설치된 경우에만 *pkg uninstall SUNWqfs SUNWsamqassy*) 명령을 사용하여 소프트웨어를 제거합니다.

```
root@solaris:~#  pkg uninstall SUNWsamfs SUNWsamqassy
```

3. 소프트웨어가 SVR4 *pkginstall* 방법을 사용하여 Solaris 10 또는 Solaris 11에 설치된 경우 *pkgrm SUNWsamfsu SUNWsamfsr*(QFS 소프트웨어가 설치된 경우에만 *pkgrm SUNWqfsu SUNWqfsr*) 명령을 사용하여 소프트웨어를 제거합니다.

   지정된 순서로 패키지를 제거합니다. *SUNWsamfsu*로 시작하여 *SUNWsamfsr*로 끝냅니다. 아래 예제에서는 *yes* 응답을 명령으로 파이프하여 모든 질문에 자동으로 답하도록 합니다.

```
root@solaris:~# yes | pkgrm SUNWsamfsu SUNWsamfsr
```

4. 소프트웨어가 SVR4 *pkginstall* 방법을 사용하여 Solaris 10 또는 Solaris 11에 설치된 경우 더 이상 필요하지 않은 구성 및 로그 파일을 삭제합니다.

```
root@solaris:~# rm -R /var/opt/SUNWsamfs/
root@solaris:~# rm -R /etc/opt/SUNWsamfs/
root@solaris:~# rm -R /var/adm/sam-log/
root@solaris:~#
```

5. 호스트를 재부트합니다.

```
root@solaris:~#  reboot
```

6. 여기서 중지합니다.

Linux 호스트에서 Oracle HSM 클라이언트 제거

Linux 클라이언트 소프트웨어를 완전히 제거하려면 다음과 같이 하십시오.

1. Linux 클라이언트 호스트에 *root*로 로그인합니다.

```
[root@linux ~]#
```

2. Oracle HSM 스크립트 */var/opt/SUNWsamfs/Uninstall*(QFS가 설치된 경우에만 */var/opt/SUNWqfs/Uninstall*)을 실행합니다.

   다른 방법은 사용하지 마십시오. *rpm -e*와 같은 다른 방법을 사용하면 소프트웨어를 제거하거나 다시 설치할 때 예기치 않은 결과와 문제가 발생할 수 있습니다. 따라서 항상 다음 스크립트를 사용합니다.

```
[root@linux ~]#  /var/opt/SUNWsamfs/Uninstall
```

5

5장. samsetup 구성 마법사 사용

samsetup 마법사는 간단한 텍스트 기반의 메뉴 방식 유틸리티로서, 가장 일반적으로 처리할 요구 사항에 맞추어 Oracle HSM 파일 시스템을 빠르게 만들고 구성할 수 있습니다. 이 마법사는 다음과 같은 모든 기본 작업을 안내합니다.

- 단일 호스트에 마운트된 QFS 독립형 파일 시스템 만들기
- 여러 호스트에 마운트된 QFS 공유 파일 시스템 만들기
- QFS 파일 시스템에 대한 Oracle HSM 아카이빙 구성
- 기본(캐시) 디스크 스토리지, 아카이브 디스크 스토리지, 이동식 매체 라이브러리, 드라이브, 미디어 등을 비롯한 스토리지 하드웨어 구성

마법사의 출력(유효한 Oracle HSM 구성 스크립트)을 사용하여 보다 전문적인 솔루션을 만들 수도 있습니다.

samsetup 마법사에는 기본적으로 설명이 포함되어 있어 메뉴 및 프롬프트가 프로세스를 안내하고 상황에 맞는 온라인 도움말을 바로 사용할 수 있습니다. 따라서 이 장에서는 도구에 제공된 정보를 중복하여 설명하지 않습니다.

그러나 특히, Oracle HSM를 처음 사용하는 경우 이 설명서의 나머지 절을 살펴본 후에 마법사를 사용하는 것이 좋습니다.

- 6장. 기본 파일 시스템 구성 에서는 Oracle HSM의 작동 방식에 대한 중요 정보를 제공하고 구성 파일 및 파일 시스템을 만드는 프로세스에 대해 설명합니다. 구성 파일을 직접 만들거나 편집할 필요가 없어도 마법사에 제공되는 옵션을 완전히 이해하려면 이 정보가 필요합니다.
- Oracle HSM 아카이빙 파일 시스템이 필요할 경우 “파일 시스템 보호 구성”의 정보가 필요합니다. *samsetup* 마법사에서는 중요 파일 시스템 메타데이터 및 로그에 대한 예약된 백업을 구성하지 않습니다.
- Oracle HSM 공유 파일 시스템이 필요할 경우 7장. 여러 호스트에서 파일 시스템 액세스도 검토하십시오. 특히 “Oracle HSM 소프트웨어를 사용하여 여러 호스트에서 파일 시스템 액세스” 및 “Oracle HSM 공유 파일 시스템 구성” 절이 관련성이 높습니다.
- Oracle HSM의 추가 기능을 사용해야 할 경우 이 설명서의 관련 절에서 추가 구성 지침을 참조해야 합니다. 예를 들어 “아카이브 매체 검증 구성”, “WORM(Write Once Read Many) 파일 지원을 사용으로 설정”, “LTFS(Linear Tape File System)에 대한 지원을 사용으로 설정”, 9장. 고가용성 솔루션 준비 , 8장. *SAM-Remote* 구성 및 10장. 보고 데이터베이스 구성을 참조하십시오.

- 마지막으로 파일 시스템을 구성할 때 *samsetup*, 명령줄 또는 Oracle HSM Manager 사용자 인터페이스 중 무엇을 사용하더라도 13장. *Oracle HSM* 구성 백업에 설명된 대로 작업을 보호해야 합니다.

6

6장. 기본 파일 시스템 구성

QFS 파일 시스템은 모든 Oracle HSM 솔루션의 기본 구성 요소입니다. QFS 파일 시스템은 단독으로 사용되어 고성능, 사실상 무제한적인 용량 및 매우 큰 파일에 대한 지원을 제공합니다. Oracle Hierarchical Storage Manager 및 적절하게 구성된 아카이브 스토리지와 함께 사용될 경우 QFS 파일 시스템은 Oracle HSM 아카이빙 파일 시스템이 됩니다. 그런 다음 아카이빙 및 비아카이빙 QFS 파일 시스템이 둘 다 사용되어 더 복잡한 다중 호스트 고가용성 구성의 기초가 됩니다. 따라서 이 장에서는 이러한 파일 시스템을 만들고 구성하기 위한 기본 작업에 대해 설명합니다.

QFS 파일 시스템 구성

기본 QFS 파일 시스템을 만들고 구성하는 것은 간단합니다. 각 경우에 다음 작업을 수행합니다.

- 파일 시스템을 지원할 디스크 장치를 준비합니다.
- 마스터 구성 파일(*mcf*)을 만듭니다.
- */opt/SUNWsamfs/sbin/sammkfs* 명령을 사용하여 파일 시스템을 만듭니다.
- */etc/vfstab* 파일을 편집하여 새 파일 시스템을 호스트의 가상 파일 시스템 구성에 추가합니다.
- 새 파일 시스템을 마운트합니다.

그래픽 Oracle HSM Manager 인터페이스나 텍스트 편집기 및 명령줄 터미널을 사용하여 이 프로세스를 수행할 수 있습니다. 그러나 예제에서는 프로세스의 일부를 명확하게 만들어 더 쉽게 이해할 수 있도록 편집기 및 명령줄 방법이 사용됩니다.

초기 Oracle HSM 구성을 간단하고 편리하게 수행할 수 있도록 이 절의 절차에서는 Solaris 가상 파일 시스템 */etc/vfstab*의 구성 파일에서 파일 시스템 마운트 옵션을 설정합니다. 그러나 대부분의 옵션은 선택적 */etc/opt/SUNWsamfs/samfs.cmd* 파일이나 명령줄에서 설정할 수도 있습니다. 자세한 내용은 *samfs.cmd* 및 *mount_samfs* 매뉴얼 페이지를 참조하십시오.

QFS 파일 시스템을 위한 디스크 스토리지 준비

구성 프로세스를 시작하기 전에 계획된 구성에 필요한 디스크 리소스를 선택합니다. 원시 장치 슬라이스, ZFS *zvol* 볼륨 또는 Solaris Volume Manager 볼륨을 사용할 수 있습니다.

범용 ms 파일 시스템 구성

1. 파일 시스템 호스트에 *root*로 로그인합니다. 호스트에 영역이 구성된 경우 전역 영역에 로그인합니다.

```
root@solaris:~#
```

2. */etc/opt/SUNWsamfs/mcf* 파일을 만듭니다.

   mcf(마스터 구성 파일)는 공백으로 구분된 6개의 열이 있는 테이블이며 각 열은 QFS 파일 시스템을 정의하는 매개변수인 *Equipment Identifier*, *Equipment Ordinal*, *Equipment Type*, *Family Set*, *Device State* 및 *Additional Parameters* 중 하나를 나타냅니다. 테이블의 행은 스토리지 장치 및 장치 그룹(패밀리 세트)을 모두 포함하는 파일 시스템 장비를 나타냅니다.

   Oracle HSM Manager 그래픽 사용자 인터페이스에서 옵션을 선택하거나 텍스트 편집기를 사용하여 *mcf* 파일을 만들 수 있습니다. 아래 예제에서는 *vi* 텍스트 편집기를 사용합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
~
~
"/etc/opt/SUNWsamfs/mcf"  [New File]
```

3. 명확하게 알 수 있도록 열 머리글을 주석으로 입력합니다.

   주석 행은 해시 기호(#)로 시작됩니다.

```
# Equipment          Equipment  Equipment  Family     Device   Additional
# Identifier         Ordinal    Type       Set        State    Parameters
#-----------------   ---------  ---------  ---------  ------   ---------------
```

4. 첫번째 행의 *Equipment Identifier* 필드(첫번째 열)에 새 파일 시스템의 이름을 입력합니다.

   이 예제에서는 파일 시스템의 이름이 *qfsms*로 지정됩니다.

```
# Equipment          Equipment  Equipment  Family     Device   Additional
# Identifier         Ordinal    Type       Set        State    Parameters
#-----------------   ---------  ---------  ---------  ------   ---------------
qfsms
```

5. *Equipment Ordinal* 필드(두번째 열)에 파일 시스템을 고유하게 식별하는 숫자를 입력합니다.

장비 순서 번호는 Oracle HSM에 의해 제어되는 모든 장비를 고유하게 식별합니다. 이 예제에서는 *qfsms* 파일 시스템에 *100*을 사용합니다.

```
# Equipment          Equipment  Equipment  Family     Device   Additional
# Identifier         Ordinal    Type       Set        State    Parameters
#-----------------   ---------  ---------  ---------  ------   ----------------
qfsms                100
```

6. *Equipment Type* 필드(세번째 열)에 범용 QFS 파일 시스템의 장비 유형 *ms*를 입력합니다.

```
# Equipment          Equipment  Equipment  Family     Device   Additional
# Identifier         Ordinal    Type       Set        State    Parameters
#-----------------   ---------  ---------  ---------  ------   ----------------
qfsms                100        ms
```

7. *Family Set* 필드(네번째 열)에 파일 시스템의 이름을 입력합니다.

   Family Set 매개변수는 로봇 테이프 라이브러리 및 해당 상주 테이프 드라이브나 파일 시스템 및 해당 구성 요소 디스크 장치와 같은 단위를 이루기 위해 함께 구성되는 장비 그룹을 정의합니다.

```
# Equipment          Equipment  Equipment  Family     Device   Additional
# Identifier         Ordinal    Type       Set        State    Parameters
#-----------------   ---------  ---------  ---------  ------   ----------------
qfsms                100        ms         qfsms
```

8. *Device State* 열에 *on*을 입력하고 *Additional Parameters* 열은 비워 둡니다.

   이 행이 완료되었습니다.

```
# Equipment          Equipment  Equipment  Family     Device   Additional
# Identifier         Ordinal    Type       Set        State    Parameters
#-----------------   ---------  ---------  ---------  ------   ----------------
qfsms                100        ms         qfsms      on
```

9. 새 행을 시작합니다. *Equipment Identifier* 필드(첫번째 열)에서 선택한 디스크 장치 중 하나에 대한 식별자를 입력하고 *Equipment Ordinal* 필드(두번째 열)에 고유한 숫자를 입력합니다.

   예제에서는 장치가 *qfsms* 파일 시스템 패밀리 세트의 일부라는 것을 강조 표시하기 위해 장치 라인이 들여쓰기되고 패밀리 세트의 장비 번호를 증분하여 장치 번호를 만듭니다(이 경우에는 *101*).

```
# Equipment          Equipment  Equipment  Family     Device   Additional
```

```
# Identifier         Ordinal    Type       Set        State    Parameters
#----------------   ---------  ---------  ---------  ------   ---------------
qfsms                100        ms         qfsms      on
  /dev/dsk/c1t3d0s3   101
```

10. 디스크 장치 행의 *Equipment Type* 필드(세번째 열)에 디스크 장치의 장비 유형 *md*를 입력합니다.

```
# Equipment          Equipment  Equipment  Family     Device   Additional
# Identifier         Ordinal    Type       Set        State    Parameters
#----------------   ---------  ---------  ---------  ------   ---------------
qfsms                100        ms         qfsms      on
  /dev/dsk/c1t3d0s3   101        md
```

11. 파일 시스템의 패밀리 세트 이름을 디스크 장치 행의 *Family Set* 필드(네번째 열)에 입력하고 *on*을 *Device State* 필드(다섯번째 열)에 입력한 다음 *Additional Parameters* 필드(여섯번째 열)는 비워 둡니다.

    패밀리 세트 이름 *qfsms*는 파일 시스템을 위한 하드웨어의 일부로 디스크 장비를 식별합니다.

```
# Equipment          Equipment  Equipment  Family     Device   Additional
# Identifier         Ordinal    Type       Set        State    Parameters
#----------------   ---------  ---------  ---------  ------   ---------------
qfsms                100        ms         qfsms      on
  /dev/dsk/c1t3d0s3   101        md         qfsms      on
```

12. 이제 나머지 디스크 장치에 대한 항목을 추가하고 파일을 저장한 다음 편집기를 종료합니다.

```
# Equipment          Equipment  Equipment  Family     Device   Additional
# Identifier         Ordinal    Type       Set        State    Parameters
#----------------   ---------  ---------  ---------  ------   ---------------
qfsms                100        ms         qfsms      on
  /dev/dsk/c1t3d0s3   101        md         qfsms      on
  /dev/dsk/c1t4d0s5   102        md         qfsms      on
:wq
root@solaris:~#
```

13. *sam-fsd* 명령을 실행하여 *mcf* 파일에서 오류를 확인합니다.

    sam-fsd 명령은 Oracle HSM 구성 파일을 읽고 파일 시스템을 초기화합니다. 오류가 발견되면 실행을 중지합니다.

```
root@solaris:~# sam-fsd
```

14. *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.

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

    대개 이러한 오류는 부주의한 타이핑 실수의 결과입니다. 여기에서는 *mcf* 파일을 편집기에서 열면 두번째 *md* 장치인 장치 *102*에 대한 장비 이름의 슬라이스 번호 부분에서 0 대신에 문자 *o*를 입력했다는 것을 알 수 있습니다.

```
qfsms                100        ms         qfsms      on
  /dev/dsk/c0t0d0s0   101        md         qfsms      on
  /dev/dsk/c0t3d0so   102        md         qfsms      on
```

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

    이 예는 오류가 없는 출력의 일부입니다.

```
root@solaris:~# sam-fsd
Trace file controls:
sam-amld      /var/opt/SUNWsamfs/trace/sam-amld
              cust err fatal ipc misc proc date
              size    10M  age 0
sam-archiverd /var/opt/SUNWsamfs/trace/sam-archiverd
              cust err fatal ipc misc proc date module
              size    10M  age 0
sam-catserverd /var/opt/SUNWsamfs/trace/sam-catserverd
              cust err fatal ipc misc proc date module
              size    10M  age 0
...
Would start sam-archiverd()
Would start sam-stagealld()
Would start sam-stagerd()
Would start sam-amld()
```

16. 새 파일 시스템의 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 */qfsms* 마운트 지점 디렉토리를 만들고 *755*(*-rwxr-xr-x*)로 권한을 설정합니다.

```
root@solaris:~# mkdir /qfsms
root@solaris:~# chmod 755 /qfsms
```

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

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

18. *samd config* 명령이 실패하고 *You need to run /opt/SUNWsamfs/util/SAM-QFS-post-install* 메시지가 표시될 경우 소프트웨어를 설치할 때 사후 설치 스크립트를 잊어버리고 실행하지 않은 것입니다. 지금 실행합니다.

```
root@solaris:~# /opt/SUNWsamfs/util/SAM-QFS-post-install
- The administrator commands will be executable by root only (group bin).
If this is the desired value, enter "y".  If you want to change
the specified value enter "c".
...
root@solaris:~#
```

19. */opt/SUNWsamfs/sbin/sammkfs* 명령과 파일 시스템의 패밀리 세트 이름을 사용하여 파일 시스템을 만듭니다.

    Oracle HSM 소프트웨어는 이중 할당 및 기본 DAU(디스크 할당 단위) 크기를 *md* 장치에 사용합니다. 크고 작은 파일과 I/O 요청을 모두 수용할 수 있다는 점에서 범용 파일 시스템에는 이러한 크기가 적합합니다. 예제에서는 기본값을 사용합니다.

```
root@solaris:~# sammkfs qfsms
Building 'qfsms' will destroy the contents of devices:
  /dev/dsk/c1t3d0s3
  /dev/dsk/c1t4d0s5
Do you wish to continue? [y/N]yes
total data kilobytes       = ...
```

    I/O 요구 사항에 더 적합한 기본값이 아닌 DAU 크기를 지정해야 하는 *mr* 장치를 사용 중인 경우 *sammkfs* 명령을 *-a* 옵션과 함께 사용합니다.

```
root@solaris:~# sammkfs -a 16 qfs2ma
```

추가 정보는 *sammkfs* 매뉴얼 페이지를 참조하십시오.

20. 운영체제의 */etc/vfstab* 파일을 백업합니다.

```
root@solaris:~# cp /etc/vfstab /etc/vfstab.backup
```

21. 새 파일 시스템을 운영체제의 가상 파일 시스템 구성에 추가합니다. 텍스트 편집기에서 파일을 열고 *qfsms* 패밀리 세트 장치에 대한 라인을 시작합니다.

```
#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       -
...
qfsms      -        /qfsms    samfs   -
```

22. */etc/vfstab* 파일의 여섯번째 열인 *Mount at Boot*에 대부분의 경우 *no*를 입력합니다.

```
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       -
...
qfsms      -        /qfsms    samfs   -     no
```

23. 라운드 로빈 할당을 지정하려면 *stripe=0* 마운트 옵션을 추가합니다.

```
#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       -
...
qfsms      -        /qfsms    samfs   -     no       stripe=0
```

24. 스트라이프 할당을 지정하려면 *stripe=stripe-width* 마운트 옵션을 추가합니다. 여기서 *stripe-width*는 스트라이프의 각 디스크에 기록해야 하는 DAU(디스크 할당 단위) 수입니다.

예제에서는 스트라이프 너비를 DAU 1개로 설정합니다.

```
#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       -
...
qfsms      -        /qfsms    samfs   -     no       stripe=1
```

여기에서 *stripe=1* 옵션은 스트라이프 너비로 DAU 1개를 지정하고 쓰기 크기로 DAU 2개를 지정합니다. 따라서 파일 시스템은 한 번에 DAU 2개를 쓸 경우 *qfsms* 패밀리 세트에 있는 2개의 각 *md* 디스크 장치에 하나씩 씁니다.

25. */etc/vfstab* 파일에 다른 원하는 변경을 수행합니다.

    예를 들어, 메타데이터 서버가 응답하지 않을 경우 파일 시스템을 백그라운드로 마운트하려면 *bg* 마운트 옵션을 *Mount Options* 필드에 추가합니다.

```
#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       -
...
qfsms      -        /qfsms    samfs   -     no       stripe=1,bg
```

26. *vfstab* 파일을 저장하고 편집기를 닫습니다.

```
...
qfsms      -        /qfsms    samfs   -     no       stripe=1
:wq
root@solaris:~#
```

27. 새 파일 시스템을 마운트합니다.

```
root@solaris:~# mount /qfsms
```

28. 이제 파일 시스템이 완료되었으며 사용할 준비가 되었습니다.

    다음 단계:

- Oracle Hierarchical Storage Manager를 사용하여 아카이빙 파일 시스템을 설정하는 중이면 “Oracle HSM 아카이빙 파일 시스템 구성”을 참조하십시오.
- 파일 시스템에서 WORM(Write Once Read Many) 기능을 사용으로 설정해야 할 경우 “WORM(Write Once Read Many) 파일 지원을 사용으로 설정”을 참조하십시오.
- LTFS를 사용하는 시스템과 상호 작용해야 하거나 원격 사이트 간에 많은 양의 데이터를 전송해야 할 경우 “LTFS(Linear Tape File System)에 대한 지원을 사용으로 설정”을 참조하십시오.
- 다중 호스트 파일 시스템 액세스 또는 고가용성 구성과 같은 추가 요구 사항이 있는 경우 “기본 과정 이후”를 참조하십시오.

고성능 ma 파일 시스템 구성

Oracle HSM 소프트웨어가 파일 시스템 호스트에 설치되고 나면 아래 설명된 대로 *ma* 파일 시스템을 구성합니다.

1. 파일 시스템 호스트에 *root*로 로그인합니다. 호스트에 영역이 구성된 경우 전역 영역에 로그인합니다.

```
root@solaris:~#
```

2. 메타데이터를 보유할 디스크 장치를 선택합니다.
3. 데이터를 보유할 디스크 장치를 선택합니다.
4. *mcf* 파일을 만듭니다.

   Oracle HSM Manager 그래픽 사용자 인터페이스에서 옵션을 선택하거나 텍스트 편집기를 사용하여 *mcf* 파일을 만들 수 있습니다. 아래 예제에서는 *vi* 텍스트 편집기를 사용합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
~
"/etc/opt/SUNWsamfs/mcf"  [New File]
```

5. 명확하게 알 수 있도록 열 머리글을 주석으로 입력합니다.

   주석 행은 해시 기호(#)로 시작됩니다.

```
# Equipment           Equipment  Equipment  Family  Device   Additional
# Identifier          Ordinal    Type       Set     State    Parameters
#------------------   ---------  ---------  ------  ------   -----------------
```

6. 파일 시스템 패밀리 세트에 대한 항목을 만듭니다.

   이 예제에서는 파일 시스템을 *qfsma*로 식별하고 장비 순서를 *200*으로 증분하며 장비 유형을 *ma*로 설정하고 패밀리 세트 이름을 *qfsma*로 설정한 다음 장치 상태를 *on*으로 설정합니다.

```
# Equipment           Equipment  Equipment  Family  Device   Additional
# Identifier          Ordinal    Type       Set     State    Parameters
#------------------   ---------  ---------  ------  ------   -----------------
qfsma          200       ma        qfsma   on
```

7. 각 메타데이터 장치에 대한 항목을 추가합니다. 선택한 디스크 장치에 대한 식별자를 장비 식별자 열에 입력하고 장비 순서를 설정한 다음 장비 유형을 *mm*으로 설정합니다.

   파일 시스템의 크기에 필요한 메타데이터를 보유할 수 있도록 충분한 메타데이터 장치를 추가합니다. 예제에서는 단일 메타데이터 장치를 추가합니다.

```
# Equipment           Equipment  Equipment  Family  Device   Additional
# Identifier          Ordinal    Type       Set     State    Parameters
#------------------   ---------  ---------  ------  ------   -----------------
qfsma                 200        ma         qfsma   on
  /dev/dsk/c0t0d0s0    201        mm         qfsma   on
```

8. 이제 데이터 장치에 대한 항목을 추가하고 파일을 저장한 다음 편집기를 종료합니다.

   md, *mr* 또는 스트라이프 그룹(*gXXX*) 장치가 될 수 있습니다. 이 예제에서는 *md* 장치를 지정합니다.

```
# Equipment           Equipment  Equipment  Family  Device   Additional
# Identifier          Ordinal    Type       Set     State    Parameters
#------------------   ---------  ---------  ------  ------   -----------------
qfsma                 200        ma         qfsma   on
  /dev/dsk/c0t0d0s0    201        mm         qfsma   on
  /dev/dsk/c0t3d0s0   202        md         qfsma   on
  /dev/dsk/c0t3d0s1   203        md         qfsma   on
:wq
root@solaris:~#
```

9. *sam-fsd* 명령을 실행하여 *mcf* 파일에서 오류를 확인합니다.

   sam-fsd 명령은 Oracle HSM 구성 파일을 읽고 파일 시스템을 초기화합니다. 오류가 발견되면 실행을 중지합니다.

```
root@solaris:~# sam-fsd
```

10. *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.

```
root@solaris:~# sam-fsd
```

```
Problem in mcf file /etc/opt/SUNWsamfs/mcf for filesystem qfsma
sam-fsd: Problem with file system devices.
```

대개 이러한 오류는 부주의한 타이핑 실수의 결과입니다. 여기에서는 *mcf* 파일을 편집기에서 열면 첫번째 *md* 장치인 장치 *202*에 대한 장비 이름의 슬라이스 번호 부분에서 1 대신에 느낌표(*!*)를 입력했다는 것을 알 수 있습니다.

```
sharefs1              200         ma         qfsma   on
  /dev/dsk/c0t0d0s0    201         mm         qfsma   on
  /dev/dsk/c0t0d0s!    202         md         qfsma   on
  /dev/dsk/c0t3d0s0    203         md         qfsma   on
```

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

    이 예는 오류가 없는 출력의 일부입니다.

```
root@solaris:~# sam-fsd
Trace file controls:
sam-amld      /var/opt/SUNWsamfs/trace/sam-amld
              cust err fatal ipc misc proc date
              size    10M  age 0
sam-archiverd /var/opt/SUNWsamfs/trace/sam-archiverd
              cust err fatal ipc misc proc date module
              size    10M  age 0
sam-catserverd /var/opt/SUNWsamfs/trace/sam-catserverd
              cust err fatal ipc misc proc date module
              size    10M  age 0
...
Would start sam-archiverd()
Would start sam-stagealld()
Would start sam-stagerd()
Would start sam-amld()
```

12. */opt/SUNWsamfs/sbin/sammkfs* 명령과 파일 시스템의 패밀리 세트 이름을 사용하여 파일 시스템을 만듭니다.

    예제에서는 *md* 장치가 있는 *ma* 파일 시스템에 대한 기본 DAU(디스크 할당 단위) 크기인 *64*KB를 사용하여 파일 시스템을 만듭니다.

```
root@solaris:~# sammkfs qfsma
Building 'qfsma' will destroy the contents of devices:
  /dev/dsk/c0t0d0s0
  /dev/dsk/c0t3d0s0
```

```
  /dev/dsk/c0t3d0s1
Do you wish to continue? [y/N]yes
total data kilobytes       = ...
```

일반적으로 기본값을 선택하는 것이 좋습니다. 그러나 파일 시스템이 작은 양의 데이터를 읽고 쓰는 작은 파일 또는 응용 프로그램을 주로 지원할 경우 *16* 또는 *32*KB의 DAU 크기를 지정할 수도 있습니다. 16KB DAU를 지정하려면 *sammkfs* 명령과 함께 *-a* 옵션을 사용합니다.

```
root@solaris:~# sammkfs -a 16 qfsma
```

mr 장치 및 *gXXX* 스트라이프 그룹에 대한 DAU는 *8-65528*KB 범위에서 8KB 증분 단위로 완전히 조정할 수 있습니다. 기본값은 *mr* 장치의 경우 *64*KB이고 *gXXX* 스트라이프 그룹의 경우 *256*KB입니다. 자세한 내용은 *sammkfs* 매뉴얼 페이지를 참조하십시오.

13. 운영체제의 */etc/vfstab* 파일을 백업합니다.

```
root@solaris:~# cp /etc/vfstab /etc/vfstab.backup
```

14. 새 파일 시스템을 운영체제의 가상 파일 시스템 구성에 추가합니다. 텍스트 편집기에서 */etc/vfstab* 파일을 열고 *qfsma* 패밀리 세트에 대한 라인을 시작합니다.

```
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       -
...
qfsma      -        /qfsma    samfs   -
```

15. */etc/vfstab* 파일의 여섯번째 열인 *Mount at Boot*에 *no*를 입력합니다.

```
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       -
...
qfsma      -        /qfsma    samfs   -     no
```

16. 라운드 로빈 할당을 지정하려면 *stripe=0* 마운트 옵션을 추가합니다.

```
#File
#Device    Device   Mount     System  fsck  Mount    Mount
#to Mount  to fsck  Point     Type    Pass  at Boot  Options
#--------  -------  --------  ------  ----  -------  -------------------------
/devices   -        /devices  devfs   -     no       -
...
qfsma      -        /qfsma    samfs   -     no       stripe=0
```

17. 스트라이프 할당을 지정하려면 *stripe=stripe-width* 마운트 옵션을 추가합니다. 여기서 *stripe-width*는 스트라이프의 각 디스크에 기록해야 하는 DAU(디스크 할당 단위) 수를 나타내는 *[1-255]* 범위의 정수입니다.

    스트라이프 할당이 지정된 경우 데이터가 장치에 병렬로 기록됩니다. 따라서 최상의 성능을 위해 스토리지 하드웨어에서 사용할 수 있는 대역폭을 최대한 활용하는 스트라이프 너비를 선택합니다. 지정된 스트라이프 너비 동안 전송되는 데이터 양은 하드웨어가 구성된 방법에 따라 다릅니다. 단일 디스크 볼륨에서 구현된 *md* 장치의 경우 스트라이프 너비 **1**은 디스크 2개 각각에 64KB DAU 1개를 기록하므로 총 128KB에 해당합니다. 3+1 RAID 5 볼륨 그룹에서 구현된 *md* 장치의 경우 동일한 스트라이프 너비는 2개의 각 장치에 있는 데이터 디스크 3개 각각에 64KB DAU 1개를 전송하므로 전송별로 총 DAU 6개 또는 384KB에 해당합니다. 예제에서는 스트라이프 너비를 DAU 1개로 설정합니다.

```
#File
#Device    Device   Mount     System  fsck  Mount    Mount
#to Mount  to fsck  Point     Type    Pass  at Boot  Options
#--------  -------  --------  ------  ----  -------  -------------------------
/devices   -        /devices  devfs   -     no       -
...
qfsma      -        /qfsma    samfs   -     no       stripe=1
```

18. 사용 가능한 하드웨어를 더 잘 활용하기 위해 스트라이프 너비를 조정해 볼 수 있습니다. 파일 시스템의 *Mount Options* 필드에서 *stripe=n* 마운트 옵션을 설정합니다. 여기서 *n*은 파일 시스템에 지정된 DAU 크기의 배수입니다. 파일 시스템의 I/O 성능을 테스트하고 필요에 따라 설정을 재조정합니다.

    *stripe=0*을 설정할 경우 Oracle HSM는 라운드 로빈 할당을 사용하여 파일을 장치에 기록합니다. 해당 장치가 가득 찰 때까지 각 파일이 하나의 장치에 완전히 할당됩니다. 공유 파일 시스템 및 멀티스트림 환경의 경우 라운드 로빈이 선호됩니다.

    예제에서는 RAID-5 볼륨 그룹의 대역폭이 스트라이프 너비 1에서 제대로 활용되지 않는 것으로 확인되었기 때문에 *stripe=2*를 사용해 봅니다.

```
#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       -
...
qfsma      -        /qfsma    samfs   -     no       ...,stripe=2
```

19. 그렇지 않은 경우 *vfstab* 파일을 저장합니다.

```
...
qfsma      -        /qfsma    samfs   -     no       stripe=1
:wq
root@solaris:~#
```

20. 새 파일 시스템을 마운트합니다.

```
root@solaris:~# mount /qfsms
```

이제 기본 파일 시스템이 완료되었으며 사용할 준비가 되었습니다.

21. Oracle Hierarchical Storage Manager를 사용하여 아카이빙 파일 시스템을 설정하는 중이면 "Oracle HSM 아카이빙 파일 시스템 구성"을 참조하십시오.
22. 파일 시스템에서 WORM(Write Once Read Many) 기능을 사용으로 설정해야 할 경우 "WORM(Write Once Read Many) 파일 지원을 사용으로 설정"을 참조하십시오.
23. LTFS를 사용하는 시스템과 상호 작용해야 하거나 원격 사이트 간에 많은 양의 데이터를 전송해야 할 경우 "LTFS(Linear Tape File System)에 대한 지원을 사용으로 설정"을 참조하십시오.
24. 다중 호스트 파일 시스템 액세스 또는 고가용성 구성과 같은 추가 요구 사항이 있는 경우 "기본 과정 이후"를 참조하십시오.
25. 그렇지 않은 경우 11장. 알림 및 로깅 구성으로 이동합니다.

Oracle HSM 아카이빙 파일 시스템 구성

아카이빙 파일 시스템에는 하나 이상의 QFS *ma* 또는 *ms* 유형 파일 시스템과 아카이브 스토리지 및 Oracle HSM 소프트웨어가 결합되어 있습니다. Oracle HSM 소프트웨어는 보조 디스크 스토리지 및/또는 이동식 매체를 기본 파일 시스템 작업에 통합하므로 파일이 다양한 매체의 여러 복사본에서 유지 관리됩니다. 이 중복성은 연속 데이터 보호를 제공하고 매우 큰 파일의 정책 구동 보존 및 효율적인 스토리지를 지원합니다.

- Oracle HSM 호스트 구성에 디스크 아카이브 파일 시스템 추가
- 아카이빙 파일 시스템 구성
- 아카이빙 파일 시스템 마운트
- 아카이빙 프로세스 구성
- 네트워크 연결 테이프 라이브러리에 저장된 아카이브 매체 카탈로그화

- 파일 시스템 보호 구성
- 아카이브 매체 검증 구성
- Oracle HSM 파일 시스템에서 WORM 지원을 사용으로 설정

Oracle HSM 호스트 구성에 디스크 아카이브 파일 시스템 추가

Oracle HSM 호스트에서 필요한 디스크 아카이브 파일 시스템을 만들고 로컬 및 원격 디스크 아카이브 파일 시스템을 모두 호스트 구성에 추가합니다. 아래에 나열된 절차를 사용하십시오.

- 디스크 아카이브로 사용할 로컬 파일 시스템 만들기
- Oracle HSM 호스트 구성에 디스크 아카이브 추가

디스크 아카이브로 사용할 로컬 파일 시스템 만들기

Oracle HSM 서버에서 파일 시스템을 디스크 아카이브로 사용하려는 경우 다음과 같이 하십시오.

1. Oracle HSM 서버에서 로컬로 마운트되는 각 디스크 아카이브 볼륨에 대해 QFS, ZFS 또는 UFS 파일 시스템을 만듭니다.

   다른 응용 프로그램과 공유해야 하는 기존 범용 파일 시스템은 사용하지 마십시오.
2. 하나 이상의 QFS 파일 시스템을 디스크 아카이브 볼륨으로 구성할 경우 해당 파일 시스템을 아카이브 스토리지 볼륨으로 명확하게 식별하는 패밀리 세트 이름 및 장비 순서 번호 범위를 각각에 대해 지정합니다.

   QFS 아카이브 스토리지 파일 시스템을 다른 Oracle HSM 주 파일 시스템과 명확하게 구별하면 구성을 더 쉽게 이해하고 유지 관리할 수 있습니다. 이 예제에서는 새 파일 시스템의 이름인 *DISKVOL1*이 해당 기능을 나타냅니다. *mcf* 파일에서 이 이름 및 장비 순서 *800*으로 디스크 아카이브가 *samms* 및 *100*(후속 예제에서 아카이빙 Oracle HSM 파일 시스템을 만들 때 사용할 패밀리 세트 이름 및 순서 번호)과 구별됩니다.

```
# Archiving file systems:
#
# Equipment              Equipment Equipment Family    Device Additional
# Identifier             Ordinal   Type      Set       State  Parameters
#----------------------- --------- --------- --------- ------ -----------------

# Archival storage for copies:
#
# Equipment              Equipment Equipment Family    Device Additional
# Identifier             Ordinal   Type      Set       State  Parameters
#----------------------- --------- --------- --------- ------ -----------------
DISKVOL1                 800       ms        DISKVOL1  on
 /dev/dsk/c6t0d1s7         801       md        DISKVOL1  on
 /dev/dsk/c4t0d2s7         802       md        DISKVOL1  on
```

3. 그런 다음 디스크 아카이브를 Oracle HSM 호스트 시스템 구성에 추가합니다.

Oracle HSM 호스트 구성에 디스크 아카이브 추가

1. 물리적 테이프 라이브러리가 아카이브 테이프 볼륨을 보유하는 것처럼 Oracle HSM 호스트에서 디스크 아카이브 볼륨에 대한 마운트 지점을 보유하기 위한 단일 상위 디렉토리를 만듭니다.

   예제에서는 */diskvols* 디렉토리를 만듭니다.

```
root@solaris:~# mkdir /diskvols
```

2. 상위 디렉토리에서 각 아카이브 파일 시스템에 대한 마운트 지점 디렉토리를 만듭니다.

   예제에서는 마운트 지점 디렉토리 *DISKVOL1* 및 *DISKVOL2* - *DISKVOL15*를 만듭니다.

```
root@solaris:~# mkdir /diskvols/DISKVOL1
root@solaris:~# mkdir /diskvols/DISKVOL2
...
root@solaris:~# mkdir /diskvols/DISKVOL15
```

3. Oracle HSM 호스트에서 */etc/vfstab* 파일을 백업합니다.

```
root@solaris:~# cp /etc/vfstab /etc/vfstab.backup
```

4. 편집기에서 */etc/vfstab* 파일을 엽니다.

```
root@solaris:~# vi /etc/vfstab
```

5. 디스크 아카이브로 작동하는 각 로컬 QFS 파일 시스템에 대한 항목을 추가합니다. *samfs* 파일 시스템 유형의 각 항목을 식별하고 마운트 옵션 *nosam*을 추가합니다 (Oracle HSM에서 아카이브를 아카이브하지 않도록 하려는 경우).

   nosam 마운트 옵션은 QFS 파일 시스템에 저장된 아카이브 복사본 자체가 아카이브되지 않게 합니다.

   예제에서는 *vi* 편집기를 사용하여 한 로컬 QFS 파일 시스템 *DISKVOL1*에 대한 항목을 추가합니다.

```
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       -
```

```
...
DISKVOL1          -        /diskvols/DISKVOL1  samfs   -     no       nosam
```

6. 디스크 아카이브로 작동하는 각 NFS 파일 시스템에 대한 항목을 추가합니다. *nfs* 파일 시스템 유형의 각 항목을 식별합니다.

   예제에서는 *vi* 편집기를 사용하여 *DISKVOL2*부터 *DISKVOL15*까지 항목을 추가합니다. 여기서 *nfs1*은 디스크 아카이브 *DISKVOL2*부터 *DISKVOL13*까지 호스트하는 NFS 서버의 이름이고, *oscsa1*은 나머지 디스크 아카이브를 호스트하는 Oracle Storage Cloud Software Appliance의 이름입니다.

```
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       -
...
DISKVOL1          -        /diskvols/DISKVOL1  samfs   -     no       nosam
nfs1:/DISKVOL2    -        /diskvols/DISKVOL2  nfs     -     yes      -
nfs1:/DISKVOL3    -        /diskvols/DISKVOL3  nfs     -     yes      -
...
oscsa1:/DISKVOL14 -        /diskvols/DISKVOL3  nfs     -     yes      -
oscsa1:/DISKVOL15 -        /diskvols/DISKVOL15 nfs     -     yes      -
```

7. */etc/vfstab* 파일을 저장하고 편집기를 닫습니다.

```
...
oscsa1:/DISKVOL14 -        /diskvols/DISKVOL3  nfs     -     yes      -
oscsa1:/DISKVOL15 -        /diskvols/DISKVOL15 nfs     -     yes      -
:wq
root@solaris:~#
```

8. Oracle HSM 호스트에서 디스크 아카이브 파일 시스템을 마운트합니다.

   예제에서는 *DISKVOL1* 및 *DISKVOL2* - *DISKVOL15*를 마운트합니다.

```
root@solaris:~# mount /diskvols/DISKVOL1
root@solaris:~# mount /diskvols/DISKVOL2
...
root@solaris:~# mount /diskvols/DISKVOL14
root@solaris:~# mount /diskvols/DISKVOL15
```

9. 이제 이동식 매체 라이브러리 및 드라이브 준비를 수행합니다.

이동식 매체 라이브러리 및 드라이브 준비

이 절에서는 다음 작업을 다룹니다.

- Oracle StorageTek ACSLS 네트워크 연결 자동화 라이브러리 구성
- 바코드가 붙은 이동식 매체에 대한 레이블 지정 동작 구성
- 드라이브 타이밍 값 설정

Oracle StorageTek ACSLS 네트워크 연결 자동화 라이브러리 구성

Oracle StorageTek ACSLS 네트워크 연결 라이브러리가 있는 경우 다음과 같이 구성하거나 Oracle HSM Manager 그래픽 사용자 인터페이스를 사용하여 라이브러리를 자동으로 검색 및 구성할 수 있습니다(Oracle HSM Manager 사용을 위한 지침은 온라인 도움말 참조).

다음과 같이 하십시오.

1. Oracle HSM 서버 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. */etc/opt/SUNWsamfs* 디렉토리로 변경합니다.

```
root@solaris:~# cd /etc/opt/SUNWsamfs
```

3. 텍스트 편집기에서 구성 중인 네트워크 연결 라이브러리의 유형에 해당하는 이름으로 새 파일을 시작합니다.

   예제에서는 Oracle StorageTek ACSLS 네트워크 연결 라이브러리에 대한 매개변수 파일을 시작합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/acsls1params
# Configuration File for an ACSLS Network-Attached Tape Library 1
```

4. ACSLS 연결 라이브러리와 통신할 때 Oracle HSM 소프트웨어가 사용할 매개변수와 값을 입력합니다.

   Oracle HSM 소프트웨어는 다음 Oracle StorageTek ACSAPI(Automated Cartridge System Application Programming Interface) 매개변수를 사용하여 ACSLS 관리 라이브러리를 제어합니다(자세한 내용은 *stk* 매뉴얼 페이지 참조).

   - *access=user-id*는 액세스 제어에 대한 선택적 사용자 식별 값을 지정합니다. 기본적으로 사용자 식별 기반 액세스 제어는 없습니다.
   - *hostname=hostname*은 StorageTek ACSLS 인터페이스를 실행하는 서버의 호스트 이름을 지정합니다.
   - *portnum=portname*은 ACSLS 및 Oracle HSM 소프트웨어 간의 통신에 사용되는 포트 번호를 지정합니다.

- *ssihost=hostname*은 ACSLS 호스트에 연결되는 네트워크에 대한 다중 홈 Oracle HSM 서버를 식별하는 호스트 이름을 지정합니다. 기본값은 로컬 호스트의 이름입니다.
- *ssi_inet_port=ssi-inet-port*는 ACSLS 서버 시스템 인터페이스가 수신 ACSLS 응답에 사용해야 하는 고정 방화벽 포트를 지정합니다. *0* 또는 [*1024-65535*] 범위의 값을 지정합니다. 기본값 *0*을 사용하면 동적 포트 할당이 허용됩니다.
- *csi_hostport=csi-port*는 Oracle HSM가 해당 ACSLS 요청을 보내는 ACSLS 서버의 클라이언트 시스템 인터페이스 포트 번호를 지정합니다. *0* 또는 [*1024-65535*] 범위의 값을 지정합니다. 기본값 *0*을 사용하면 시스템은 ACSLS 서버의 포트 매퍼에 포트를 쿼리합니다.
- *capid=(acs=acsnum, lsm=lsmnum, cap=capnum)*은 CAP(카트리지 액세스 포트)의 ACSLS 주소를 지정합니다. 여기서 *acsnum*은 라이브러리에 대한 ACS(Automated Cartridge System) 번호이고 *lsmnum*은 CAP를 보유하는 LSM(Library Storage Module) 번호이며 *capnum*은 원하는 CAP에 대한 식별 번호입니다. 전체 주소를 괄호로 묶습니다.
- *capacity=(index-value-list)*는 이동식 매체 카트리지의 용량을 지정합니다. 여기서 *index-value-list*는 *index=value* 쌍의 콤마로 구분된 목록입니다. 목록의 각 *index*는 ACSLS 정의 매체 유형의 인덱스이고 각 *value*는 1024바이트 단위의 해당 볼륨 용량입니다.

  ACSLS 파일 */export/home/ACSSS/data/internal/mixed_media/media_types.dat*는 매체 유형 인덱스를 정의합니다. 일반적으로 새 카트리지 유형인 경우 또는 지원되는 용량을 대체해야 하는 경우에만 용량 항목을 제공하면 됩니다.
- *device-path-name=(acs=ACSnumber, lsm=LSMnumber, panel=Panelnumber, drive=Drivenumber)[shared]*는 클라이언트에 연결된 드라이브의 ACSLS 주소를 지정합니다. 여기서 *device-path-name*은 Oracle HSM 서버의 장치를 식별하고 *acsnum*은 라이브러리에 대한 ACS(Automated Cartridge System) 번호, *lsmnum*은 드라이브를 제어하는 모듈에 대한 LSM(Library Storage Module) 번호, *Panelnumber*는 드라이브가 설치된 패널에 대한 식별 번호, *Drivenumber*는 드라이브의 식별 번호입니다. 전체 주소를 괄호로 묶습니다.

  ACSLS 주소 뒤에 선택적 *shared* 키워드를 추가하면 2개 이상의 Oracle HSM 서버가 각각 고유한 매체를 배타적으로 제어하는 경우에 한하여 드라이브를 공유할 수 있습니다. 기본적으로 공유 드라이브의 카트리지는 언로드되기 전에 60초 동안 유휴 상태일 수 있습니다.

예제에서는 *acslserver1*을 ACSLS 호스트로 식별하고 *sam_user*에 대한 액세스를 제한하며 동적 할당을 지정하고 카트리지 액세스 포트 및 2개의 드라이브를 매핑합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/acsls1params
# Configuration File for an ACSLS Network-Attached Tape Library 1
hostname = acslserver1
portnum = 50014
access = sam_user
```

```
ssi_inet_port = 0
csi_hostport = 0
capid = (acs=0, lsm=1, cap=0)
/dev/rmt/0cbn = (acs=0, lsm=1, panel=0, drive=1)
/dev/rmt/1cbn = (acs=0, lsm=1, panel=0, drive=2)
```

5. 파일을 저장하고 편집기를 닫습니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/acsls1params
# /etc/opt/SUNWsamfs/acslibrary1
# Configuration File for an ACSLS Network-Attached Tape Library
...
/dev/rmt/0cbn = (acs=0, lsm=1, panel=0, drive=1)
/dev/rmt/1cbn = (acs=0, lsm=1, panel=0, drive=2)
:wq
root@solaris:~#
```

6. 라이브러리 또는 응용 프로그램 소프트웨어에서 바코드가 붙은 이동식 매체에 대해 비표준 레이블을 사용하는 경우 지금 레이블 지정 동작 구성을 수행합니다.
7. 드라이브 또는 응용 프로그램 소프트웨어가 Oracle HSM 기본값과 호환되지 않는 것으로 알려진 경우 지금 드라이브 타이밍 값 설정을 수행합니다.
8. 그렇지 않은 경우 “아카이빙 파일 시스템 구성”으로 이동합니다.

바코드가 붙은 이동식 매체에 대한 레이블 지정 동작 구성

기본적으로 라이브러리에 바코드 읽기 장치 및 바코드가 붙은 매체가 있는 경우 Oracle HSM 소프트웨어는 바코드의 처음 6자를 사용하여 볼륨의 레이블을 자동으로 지정합니다. 하지만 Oracle HSM에서 볼륨 레이블을 바코드의 대체 읽기를 기반으로 하도록 구성할 수 있습니다. 그렇게 하려면 다음과 같이 하십시오.

1. Oracle HSM 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. 기본값이 아닌 동작이 필요하거나 이전에 기본값을 대체하고 재설정해야 하는 경우 텍스트 편집기에서 */etc/opt/SUNWsamfs/defaults.conf* 파일을 엽니다.

   예제에서는 *vi* 편집기에서 파일을 엽니다.

```
root@solaris:~# vi /opt/SUNWsamfs/examples/defaults.conf
...
```

3. *labels* = 라인을 찾습니다(없을 경우 추가).

   예제에서는 이 지시어를 추가합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults.
...
labels =
```

4. 바코드의 처음 6자에 기초하여 자동으로 레이블을 지정하는 기본값을 다시 사용으로 설정하려면 *labels* 지시어의 값을 *barcodes*로 설정합니다. 파일을 저장하고 편집기를 닫습니다.

   이제 Oracle HSM 소프트웨어는 테이프 바코드의 처음 6자를 레이블로 사용하여 레이블이 지정되지 않은 테이프의 레이블을 자동으로 지정합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
...
labels = barcodes
:wq
root@solaris:~#
```

5. 테이프 바코드의 마지막 6자에 기초하여 자동으로 레이블 지정을 사용으로 설정하려면 *labels* 지시어의 값을 *barcodes_low*로 설정합니다. 파일을 저장하고 편집기를 닫습니다.

   labels 지시어가 *barcodes_low*로 설정된 경우 Oracle HSM 소프트웨어는 테이프 바코드의 마지막 6자를 레이블로 사용하여 레이블이 지정되지 않은 테이프의 레이블을 자동으로 다시 지정합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
...
labels = barcodes_low
:wq
root@solaris:~#
```

6. 자동 레이블 지정을 사용 안함으로 설정하고 테이프에서 레이블을 읽도록 Oracle HSM를 구성하려면 *labels* 지시어의 값을 *read*로 설정합니다. 파일을 저장하고 편집기를 닫습니다.

   labels 지시어가 *read* 값으로 설정된 경우 Oracle HSM 소프트웨어는 테이프의 레이블을 자동으로 재지정할 수 없습니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
...
labels = read
idle_unload = 0
...
```

```
:wq
root@solaris:~#
```

7. 드라이브 또는 응용 프로그램 소프트웨어가 Oracle HSM 기본값과 호환되지 않는 것으로 알려진 경우 지금 드라이브 타이밍 값 설정을 수행합니다.
8. 그렇지 않은 경우 "아카이빙 파일 시스템 구성"으로 이동합니다.

드라이브 타이밍 값 설정

기본적으로 Oracle HSM 소프트웨어는 드라이브 타이밍 매개변수를 다음과 같이 설정합니다.

- 지정된 장치 유형이 매체를 마운트 해제할 수 있기 전까지 경과해야 하는 최소 시간은 *60*초입니다.
- Oracle HSM 소프트웨어가 SCSI *unload* 명령에 응답하는 중인 라이브러리에 대해 새 명령을 실행하기 전까지 기다리는 시간은 *15*초입니다.
- Oracle HSM 소프트웨어가 유휴 드라이브를 언로드하기 전까지 기다리는 시간은 *600*초(10분)입니다.
- Oracle HSM 소프트웨어가 둘 이상의 Oracle HSM 서버에서 공유하는 유휴 드라이브를 언로드하기 전까지 기다리는 시간은 *600*초(10분)입니다.

기본 타이밍 값을 변경하려면 다음과 같이 하십시오.

1. 로그인하지 않은 경우 Oracle HSM 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. 텍스트 편집기에서 */etc/opt/SUNWsamfs/defaults.conf* 파일을 엽니다.

   이 예에서는 *vi* 편집기를 사용합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults.  To change the default behavior, uncomment the
# appropriate line (remove the '#' character and change the value.
...
```

3. 필요한 경우 지정된 장치 유형이 매체를 마운트 해제할 수 있기 전까지 경과해야 하는 최소 시간을 지정합니다. *defaults.conf* 파일에서 *equipment-type_delay = number-of-seconds* 형태의 지시어를 추가합니다. 여기서 *equipment-type*은 구성 중인 드라이브 유형을 식별하는 2자의 Oracle HSM 코드이고 *number-of-seconds*는 이 장치 유형에 대한 기본 초 수를 나타내는 정수입니다.

   장비 유형 코드 및 해당 장비 목록은 부록 A. 장비 유형 용어집을 참조하십시오. 예제에서는 LTO 드라이브에 대한 언로드 지연(장비 유형 *li*)을 기본값(*60*초)에서 *90*초로 변경합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults.  To change the default behavior, uncomment the
# appropriate line (remove the '#' character and change the value.
...
li_delay = 90
```

4. 필요한 경우 Oracle HSM 소프트웨어가 SCSI *unload* 명령에 응답하는 중인 라이브러리에 대해 새 명령을 실행하기 전까지 기다리는 시간을 지정합니다. *defaults.conf* 파일에서 *equipment-type_unload* = *number-of-seconds* 형태의 지시어를 추가합니다. 여기서 *equipment-type*은 구성 중인 드라이브 유형을 식별하는 2자의 Oracle HSM 코드이고 *number-of-seconds*는 이 장치 유형에 대한 기본 초 수를 나타내는 정수입니다.

   장비 유형 코드 및 해당 장비 목록은 부록 A. 장비 유형 용어집을 참조하십시오. *unload* 명령에 응답할 때 라이브러리에 최악의 경우 필요할 수 있는 가장 긴 시간을 설정합니다. 예제에서는 LTO 드라이브에 대한 언로드 지연(장비 유형 *li*)을 기본값(*15*초)에서 *35*초로 변경합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults.  To change the default behavior, uncomment the
# appropriate line (remove the '#' character and change the value.
...
li_delay = 90
li_unload = 35
```

5. 필요한 경우 Oracle HSM 소프트웨어가 유휴 드라이브를 언로드하기 전까지 기다리는 시간을 지정합니다. *defaults.conf* 파일에서 *idle_unload* = *number-of-seconds* 형태의 지시어를 추가합니다. 여기서 *number-of-seconds*는 지정된 초 수를 나타내는 정수입니다.

   이 기능을 사용 안함으로 설정하려면 *0*을 지정합니다. 예제에서는 기본값(*600*초)을 *0*으로 변경하여 이 기능을 사용 안함으로 설정합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults.  To change the default behavior, uncomment the
# appropriate line (remove the '#' character and change the value.
...
li_delay = 90
li_unload = 35
idle_unload = 0
```

6. 필요한 경우 Oracle HSM 소프트웨어가 공유된 유휴 드라이브를 언로드하기 전까지 기다리는 시간을 지정합니다. *defaults.conf* 파일에서 *shared_unload* = *number-of-*

seconds 형태의 지시어를 추가합니다. 여기서 *number-of-seconds*는 지정된 초 수를 나타내는 정수입니다.

이동식 매체 드라이브를 공유하도록 Oracle HSM 서버를 구성할 수 있습니다. 이 지시어는 로드된 매체를 소유하는 서버가 실제로 드라이브를 사용 중이 아닌 경우 다른 서버가 사용할 수 있도록 드라이브를 확보합니다. 이 기능을 사용 안함으로 설정하려면 *0*을 지정합니다. 예제에서는 기본값(*600*초)을 *0*으로 변경하여 이 기능을 사용 안함으로 설정합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults.  To change the default behavior, uncomment the
# appropriate line (remove the '#' character and change the value.
...
idle_unload = 600
li_delay = 90
li_unload = 35
idle_unload = 0
shared_unload = 0
```

7. 파일을 저장하고 편집기를 닫습니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/defaults.conf
# These are the defaults.  To change the default behavior, uncomment the
# appropriate line (remove the '#' character and change the value.
...
idle_unload = 600
li_delay = 90
li_unload = 35
idle_unload = 0
shared_unload = 0
:wq
root@solaris:~#
```

8. 이제 아카이빙 파일 시스템 구성을 수행합니다.

아카이빙 파일 시스템 구성

아카이빙 파일 시스템을 만드는 절차는 데이터 파일의 추가 복사본을 저장하기 위한 장치를 추가한다는 점을 제외하고 비아카이빙 파일 시스템을 만드는 절차와 동일합니다.

1. QFS 파일 시스템을 구성하는 것부터 시작합니다. 범용 *ms* 또는 고성능 *ma* 파일 시스템을 구성할 수 있습니다.

   Oracle HSM Manager 그래픽 사용자 인터페이스를 사용하여 파일 시스템을 만들 수도 있지만 이 절의 예제에서는 *vi* 편집기를 사용합니다. 여기에서는 패밀리 세트 이름이 *samms*이고 장비 순서 번호가 *100*인 범용 *ms* 파일 시스템을 만듭니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Archiving file systems:
#
# Equipment              Equipment Equipment Family  Device Additional
# Identifier             Ordinal   Type      Set     State  Parameters
#---------------------- --------- --------- ------  ------ ------------------
samms                   100       ms        samms   on
 /dev/dsk/c1t3d0s3       101       md        samms   on
 /dev/dsk/c1t3d0s4       102       md        samms   on
```

2. 아카이브 테이프 스토리지를 추가하려면 라이브러리에 대한 항목을 추가하는 것부터 시작합니다. 장비 식별자 필드에 라이브러리에 대한 장치 ID를 입력하고 장비 순서 번호를 지정합니다.

   이 예제에서는 라이브러리 장비 식별자가 */dev/scsi/changer/c1t0d5*입니다. 디스크 아카이브에 대한 선택한 범위 다음에 장비 순서 번호를 *900*으로 설정합니다.

```
# Archival storage for copies:
#
# Equipment              Equipment Equipment Family    Device Additional
# Identifier             Ordinal   Type      Set       State  Parameters
#---------------------- --------- --------- --------- ------ -----------------
DISKVOL1                800       ms        DISKVOL1  on
 /dev/dsk/c6t0d1s7       801       md        DISKVOL1  on
 /dev/dsk/c4t0d2s7       802       md        DISKVOL1  on
/dev/scsi/changer/c1t0d5 900
```

3. 장비 유형을 일반 SCSI 연결 테이프 라이브러리인 *rb*로 설정하고 테이프 라이브러리 패밀리 세트의 이름을 제공한 다음 장치 상태를 *on*으로 설정합니다.

   이 예제에서는 *library1* 라이브러리를 사용합니다.

```
# Archival storage for copies:
#
# Equipment              Equipment Equipment Family    Device Additional
# Identifier             Ordinal   Type      Set       State  Parameters
#---------------------- --------- --------- --------- ------ -----------------
DISKVOL1                800       ms        DISKVOL1  on
 /dev/dsk/c6t0d1s7       801       md        DISKVOL1  on
 /dev/dsk/c4t0d2s7       802       md        DISKVOL1  on
/dev/scsi/changer/c1t0d5 900       rb        library1  on
```

4. 선택적으로 *Additional Parameters* 열에 라이브러리 카탈로그가 저장될 경로를 입력합니다.

카탈로그 경로를 제공하지 않을 경우 소프트웨어에서 기본 경로를 설정합니다.

문서 레이아웃 제한으로 인해 예제에서는 긴 경로가 *var/opt/SUNWsamfs/catalog/library1cat* 라이브러리 카탈로그까지 축약되어 있습니다.

```
# Archival storage for copies:
#
# Equipment              Equipment Equipment Family    Device Additional
# Identifier             Ordinal   Type      Set       State  Parameters
#----------------------- --------- --------- --------- ------ -----------------
DISKVOL1                 800       ms        DISKVOL1  on
 /dev/dsk/c6t0d1s7        801       md        DISKVOL1  on
 /dev/dsk/c4t0d2s7        802       md        DISKVOL1  on
/dev/scsi/changer/c1t0d5 900       rb        library1  on     ...catalog/library1cat
```

5. 다음으로 라이브러리 패밀리 세트의 일부인 각 테이프 드라이브에 대한 항목을 추가합니다. 각 드라이브를 라이브러리에 물리적으로 설치된 순서대로 추가합니다.

   "라이브러리에 드라이브가 설치되는 순서 결정"에서 만든 드라이브 매핑 파일에 나열된 드라이브 순서를 따릅니다. 예제에서는 */dev/rmt/1*, */dev/rmt/0*, */dev/rmt/2* 및 */dev/rmt/3*에 있는 Solaris에 연결된 드라이브가 라이브러리에서 각각 드라이브 *1*, *2*, *3* 및 *4*입니다. 따라서 */dev/rmt/1*이 *mcf* 파일에서 *901* 장치로 가장 먼저 나열됩니다. *tp* 장비 유형은 일반 SCSI 연결 테이프 드라이브를 지정합니다.

```
# Archival storage for copies:
#
# Equipment              Equipment Equipment Family    Device Additional
# Identifier             Ordinal   Type      Set       State  Parameters
#----------------------- --------- --------- --------  ------ -----------------
DISKVOL1                  800       ms        DISKVOL1 on
 /dev/dsk/c6t0d1s7         801       md        DISKVOL1 on
 /dev/dsk/c4t0d2s7         802       md        DISKVOL1 on
/dev/scsi/changer/c1t0d5  900       rb        library1  on     ...catalog/library1cat
 /dev/rmt/1cbn             901       tp        library1  on
 /dev/rmt/0cbn             902       tp        library1  on
 /dev/rmt/2cbn             903       tp        library1  on
 /dev/rmt/3cbn             904       tp        library1  on
```

6. 마지막으로 Oracle HSM 내역기를 직접 구성하려는 경우 장비 유형 *hy*를 사용하여 항목을 추가합니다. family-set 및 device-state 열에 하이픈을 입력하고 additional-parameters 열에 내역기 카탈로그 경로를 입력합니다.

내역기는 아카이브에서 내보낸 볼륨을 카탈로그화하는 가상 라이브러리입니다. 내역기를 구성하지 않을 경우 소프트웨어는 지정된 가장 높은 장비 순서 번호에 1을 더하여 내역기를 자동으로 만듭니다.

예제에서는 페이지 레이아웃 제한으로 인해 긴 경로가 내역기 카탈로그까지로 축약되었습니다. 전체 경로는 */var/opt/SUNWsamfs/catalog/historian_cat*입니다.

```
# Archival storage for copies:
#
# Equipment              Equipment Equipment Family    Device Additional
# Identifier             Ordinal   Type      Set       State  Parameters
#----------------------- --------- --------- --------- ------ -----------------
DISKVOL1                 800       ms        DISKVOL1  on
 /dev/dsk/c6t0d1s7        801       md        DISKVOL1  on
 /dev/dsk/c4t0d2s7        802       md        DISKVOL1  on
/dev/scsi/changer/c1t0d5 900       rb        library1  on     ...catalog/SL150cat
 /dev/rmt/0cbn            901       tp        library1  on
 /dev/rmt/1cbn            902       tp        library1  on
 /dev/rmt/2cbn            903       tp        library1  on
 /dev/rmt/3cbn            904       tp        library1  on
historian                999       hy        -         -      ...catalog/historian_cat
```

7. *mcf* 파일을 저장하고 편집기를 닫습니다.

```
...
 /dev/rmt/3cbn            904       tp        library1  on
historian                999       hy        -      -      ...catalog/historian_cat
:wq
root@solaris:~#
```

8. *sam-fsd* 명령을 실행하여 *mcf* 파일에서 오류를 확인합니다. 발견된 모든 오류를 수정합니다.

   sam-fsd 명령은 Oracle HSM 구성 파일을 읽고 파일 시스템을 초기화합니다. 오류가 발견되면 실행을 중지합니다.

```
root@solaris:~# sam-fsd
Trace file controls:
...
Would start sam-archiverd()
Would start sam-stagealld()
Would start sam-stagerd()
Would start sam-amld()
root@solaris:~#
```

9. 하나 이상의 파일 시스템을 아카이브 스토리지 볼륨으로 사용하는 중이면 텍스트 편집기에서 */etc/opt/SUNWsamfs/diskvols.conf* 파일을 만들고 각 파일 시스템에 VSN(볼륨 일련 번호)을 지정합니다. 각 파일 시스템에 대해 원하는 볼륨 일련 번호, 공백 및 파일 시스템 마운트 지점에 대한 경로로 구성된 새 라인을 시작합니다. 그런 다음 파일을 저장합니다.

   예제에서는 세 개의 디스크 기반 아카이브 볼륨이 있는데 *DISKVOL1*은 이 목적을 위해 로컬로 만든 QFS 파일 시스템입니다. *DISKVOL2* - *DISKVOL15*는 UFS 파일 시스템입니다. 이러한 파일 시스템은 모두 */diskvols/* 디렉토리에 마운트됩니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/diskvols.conf
# Volume
# Serial     Resource
# Number     Path
# ------     ----------------------
DISKVOL1     /diskvols/DISKVOL1
DISKVOL2     /diskvols/DISKVOL2
...
DISKVOL15     /diskvols/DISKVOL3
```

10. 새 파일 시스템의 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 */samms* 마운트 지점 디렉토리를 만들고 권한을 *755*로 설정합니다(*-rwxr-xr-x*).

```
root@solaris:~# mkdir /samms
root@solaris:~# chmod 755 /samms
```

11. Oracle HSM 소프트웨어에 *mcf* 파일을 다시 읽고 그에 따라 재구성하도록 지시합니다. 보고된 오류를 수정하고 필요한 경우 반복합니다

```
root@solaris:~# /opt/SUNWsamfs/sbin/samd config
Configuring SAM-FS
root@solaris:~#
```

12. 이제 아카이빙 파일 시스템 마운트를 수행합니다.

아카이빙 파일 시스템 마운트

1. 파일 시스템 호스트에 *root*로 로그인합니다. 호스트에 영역이 구성된 경우 전역 영역에 로그인합니다.
2. Solaris */etc/vfstab* 파일을 백업하고 텍스트 편집기에서 엽니다.

예제에서는 *vi* 편집기를 사용합니다.

```
root@solaris:~# cp /etc/vfstab /etc/vfstab.backup
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       -
...
samms      -        /samms    samfs   -     yes      -
```

3. Oracle HSM가 이전에 아카이브된 파일을 디스크에서 릴리스하게 만드는 백분율 단위의 디스크 캐시 사용률인 고수위를 설정합니다. Oracle HSM 파일 시스템 항목의 마지막 열에 마운트 옵션 *high=percentage*를 입력합니다. 여기서 *percentage*는 [*0-100*] 범위의 숫자입니다.

   디스크 스토리지 용량, 평균 파일 크기 및 특정 시점에 액세스할 것으로 예상되는 파일 수에 기초하여 이 값을 설정합니다. 사용자가 만드는 새 파일 및 사용자가 액세스해야 하는 아카이브된 파일 모두에 대한 항상 충분한 캐시 공간이 있어야 합니다. 그러나 또한 이동식 매체 볼륨 마운트와 연관된 오버헤드를 방지하기 위해 스테이징을 가능한 적게 수행해야 합니다.

   최신 고수위 디스크 또는 솔리드 상태 장치를 사용하여 주 캐시가 구현된 경우 고수위 값을 95%로 설정합니다. 그렇지 않은 경우 80-85%를 사용합니다. 예제에서는 고수위를 85%로 설정합니다.

```
root@solaris:~# cp /etc/vfstab /etc/vfstab.backup
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       -
...
samms      -        /samms    samfs   -     yes      high=85
```

4. Oracle HSM가 이전에 아카이브된 파일을 디스크에서 릴리스하는 것을 중지하게 만드는 백분율 단위의 디스크 캐시 사용률인 저수위를 설정합니다. Oracle HSM 파일 시스템 항목의 마지막 열에 마운트 옵션 *low=percentage*를 입력합니다. 여기서 *percentage*는 [*0-100*] 범위의 숫자입니다.

   디스크 스토리지 용량, 평균 파일 크기 및 특정 시점에 액세스할 것으로 예상되는 파일 수에 기초하여 이 값을 설정합니다. 특히 파일이 자주 요청 및 수정될 경우에는 성능상의

이유로 최근 활성 파일을 가능한 많이 캐시에 유지할 수 있습니다. 이렇게 하면 스테이징 관련 오버헤드가 최소화됩니다. 그러나 아카이브 복사본에서 디스크로 스테이지해야 하는 새 파일 및 새로 액세스한 파일에 필요한 공간을 이전에 캐시된 파일이 소비하는 것을 원하지는 않을 것입니다.

최신 고수위 디스크 또는 솔리드 상태 장치를 사용하여 주 캐시가 구현된 경우 저수위 값을 90%로 설정합니다. 그렇지 않은 경우 70-75%를 사용합니다. 예제에서는 로컬 요구 사항에 기초하여 고수위를 75%로 설정합니다.

```
root@solaris:~# cp /etc/vfstab /etc/vfstab.backup
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       -
...
samms      -        /samms    samfs   -     yes      high=85,low=75
```

5. 이전에 아카이브된 파일이 디스크에서 릴리스될 경우 사용자가 디스크 캐시에서 일부 파일 데이터를 보존해야 할 경우 Oracle HSM 파일 시스템 항목의 마지막 열에 부분 릴리스 마운트 옵션을 입력합니다.

   부분 릴리스를 지정하면 Oracle HSM는 아카이브된 파일을 릴리스하여 디스크 공간을 복구할 때 지정된 파일의 첫번째 부분을 디스크 캐시에 남겨 둡니다. 이 접근 방법을 사용하면 응용 프로그램이 파일의 시작 부분에서 데이터에 즉시 액세스할 수 있고 나머지 부분은 테이프와 같은 아카이브 매체에서 스테이지됩니다. 부분 릴리스를 제어하는 마운트 옵션은 다음과 같습니다.

   - *maxpartial=value*는 파일이 *value*까지 부분적으로 릴리스될 때 디스크 캐시에 남아 있을 수 있는 최대 파일 데이터 양을 설정합니다. 여기서 *value*는 *0-2097152* 범위의 킬로바이트 수이고 *0*은 부분 릴리스를 사용 안함으로 설정합니다. 기본값은 *16*입니다.
   - *partial=value*는 파일이 *value*까지 부분적으로 릴리스된 후 디스크 캐시에 남아 있는 기본 파일 데이터 양을 설정합니다. 여기서 *value*는 [*0-maxpartial*] 범위의 킬로바이트 수입니다. 기본값은 *16*입니다. 그러나 보존된 파일 부분은 항상 최소 하나의 DAU(디스크 할당 단위)에 해당하는 킬로바이트를 사용합니다.
   - *partial_stage=value*는 부분적으로 릴리스된 전체 파일이 *value*까지 스테이지되기 전에 읽어야 하는 최소 파일 데이터 양을 설정합니다. 여기서 *value*는 [*0-maxpartial*] 범위의 킬로바이트 수입니다. 기본값은 *-o partial*(설정된 경우)에 지정된 값 또는 *16*입니다.
   - *stage_n_window=value*는 자동 스테이징 없이 테이프 매체에서 직접 읽는 파일에서 특정 시점에 읽을 수 있는 최대 데이터 양을 설정합니다. 지정된 *value*는 [*64-2048000*] 범위의 킬로바이트 수입니다. 기본값은 *256*입니다.

테이프 매체에서 직접 읽는 파일에 대한 자세한 내용은 *stage* 매뉴얼 페이지의 *OPTIONS* 절에서 *-n* 아래를 참조하십시오.

예제에서는 응용 프로그램의 특성에 기초하여 *maxpartial*을 *128*로 설정하고 *partial*을 *64*로 설정하며 그 밖의 경우에는 기본값을 사용합니다.

```
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       -
...
samms      -        /samms    samfs   -     yes      ... maxpartial=128,partial=64
```

6. QFS 파일 시스템을 아카이빙에서 제외해야 할 경우 *nosam* 마운트 옵션을 각각에 대한 */etc/vfstab* 항목에 추가합니다.

   예제에서는 디스크 아카이브인 *DISKVOL1* 파일 시스템에 대해 *nosam* 옵션이 설정됩니다. 여기에서 *nosam* 마운트 옵션은 아카이브 복사본 자체가 아카이브되지 않게 합니다.

```
#File
#Device           Device   Mount               System  fsck  Mount    Mount
#to Mount         to fsck  Point               Type    Pass  at Boot  Options
#--------         -------  ------------------  ------  ----  -------  ---------
/devices          -        /devices            devfs   -     no       -
...
samms             -        /samms              samfs   -     yes      ... ,partial=64
DISKVOL1          -        /diskvols/DISKVOL1  samfs   -     yes      nosam
server:/DISKVOL2  -        /diskvols/DISKVOL2  nfs     -     yes
...
server:/DISKVOL15 -        /diskvols/DISKVOL15 nfs     -     yes
```

7. */etc/vfstab* 파일을 저장하고 편집기를 닫습니다.

```
...
server:/DISKVOL15 -        /diskvols/DISKVOL15 nfs     -     yes
:wq
root@solaris:~#
```

8. Oracle HSM 아카이빙 파일 시스템을 마운트합니다.

```
root@solaris:~# mount /samms
```

9. 이제 아카이빙 프로세스 구성을 수행합니다.

아카이빙 프로세스 구성

아카이빙 파일 시스템을 만들어 마운트하고 나면 일반적으로 모든 또는 대부분의 아카이빙 요구 사항을 별다른 추가 구성 없이 해결할 수 있습니다. 대부분의 경우 파일 시스템을 식별하고 각각의 아카이브 복사본 수를 지정하며 매체 볼륨을 각 복사본에 지정하는 *archiver.cmd* 텍스트 파일을 만드는 것 외에는 거의 할 일이 없습니다.

Oracle HSM 아카이빙 프로세스에 조정 가능한 많은 매개변수가 있지만 적절하게 정의된 특수한 요구 사항이 없는 경우 일반적으로 기본 설정을 사용해야 합니다. 기본값은 매체 마운트 수를 최소화하고 매체 사용률을 극대화하며 가장 광범위한 환경에서 종단간 아카이빙 성능을 최적화하도록 신중하게 선택된 값입니다. 따라서 조정이 필요한 경우 변경사항으로 인해 아카이버가 작업을 예약하고 매체를 선택하는 기능이 불필요하게 제한되지 않는지 특히 주의해야 합니다. 스토리지 작업을 세부적으로 관리하려는 경우 성능과 전체 효율성이 크게 저하될 수도 있습니다.

그러나 거의 모든 상황에서 아카이브 로깅을 사용으로 설정해야 합니다. 로그 파일을 적절하게 관리하지 않는 경우 과도한 크기가 될 수 있으므로 아카이브 로깅은 기본적으로 사용으로 설정되지 않습니다(관리에 대한 내용은 *Oracle Hierarchical Storage Manager and StorageTek QFS Software* 유지 관리 및 관리 설명서에서 설명됨). 그러나 파일 시스템이 손상 또는 손실된 경우 쉽게 복원할 수 없는 파일을 아카이브 로그 파일을 통해 복구할 수 있습니다. 파일 시스템에 대한 보호 구성을 수행할 경우 복구 지점 파일의 파일 시스템 메타데이터를 사용하여 아카이브 복사본에 저장된 데이터에서 파일 시스템을 신속하게 재작성할 수 있습니다. 그러나 불가피하게 마지막 복구 지점이 생성된 후 파일 시스템이 손상 또는 손실되기 전에 약간의 파일이 아카이브됩니다. 이 경우 아카이브 매체는 유효한 복사본을 보유하지만 파일 시스템 메타데이터가 없으면 복사본을 자동으로 찾을 수 없습니다. 파일 시스템의 아카이브 로그가 각 아카이브 복사본 및 각 볼륨 내의 해당 *tar* 파일 위치를 보유하는 매체의 볼륨 일련 번호를 기록하므로 *tar* 유틸리티를 사용하여 이러한 파일을 복구하고 파일 시스템을 완전히 복원할 수 있습니다.

archiver.cmd 파일을 만들고 아카이빙 프로세스를 구성하려면 다음과 같이 하십시오.

1. 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. 텍스트 편집기에서 새 */etc/opt/SUNWsamfs/archiver.cmd* 파일을 엽니다.

   *archiver.cmd*의 각 라인은 공백으로 구분된 하나 이상의 필드로 구성됩니다(선행 공백은 무시됨).

   예제에서는 *vi* 편집기를 사용하여 파일을 열고 주석을 입력합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/archiver.cmd
# Configuration file for archiving file systems
```

3. *archiver.cmd* 파일의 시작 부분에 필요한 모든 일반 아카이빙 지시어를 입력합니다.

   일반 지시어는 등호(=) 문자를 두번째 필드에 포함하거나 추가 필드가 없습니다. 대부분의 경우 일반 지시어를 설정하는 대신에 기본값을 사용할 수 있습니다(자세한 내용은 *archiver.cmd* 매뉴얼 페이지의 *GENERAL DIRECTIVES SECTION* 참조).

   이 섹션을 비워 둘 수 있지만 예제에서는 해당 형태를 보여주기 위해 두 개의 일반 지시어에 대한 기본값을 입력했습니다.

   - *archivemeta = off* 지시어는 메타데이터를 아카이브하지 않아야 한다는 것을 아카이빙 프로세스에 지시합니다.
   - *examine = noscan* 지시어는 파일이 변경되었다는 것을 파일 시스템이 보고할 때마다 아카이빙이 필요한 파일을 검사하도록 아카이빙 프로세스에 지시합니다(기본값).

     이전 버전의 Oracle HSM는 전체 파일 시스템을 주기적으로 스캔했습니다. 레거시 Oracle HSM 구성과의 호환성을 위해 필요한 경우가 아니라면 일반적으로 이 지시어를 변경하지 않아야 합니다.

```
# Configuration file for archiving file systems
#---------------------------------------------------------------------
# General Directives
archivemeta = off                                                 # default
examine = noscan                                                  # default
```

4. 필요한 모든 일반 아카이빙 지시어를 입력했으면 아카이브 세트에 파일을 지정하는 것을 시작합니다. 새 라인에 지정 지시어 *fs = filesystem-name*을 입력합니다. 여기서 *filesystem-name*은 */etc/opt/SUNWsamfs/mcf* 파일에 정의된 파일 시스템에 대한 패밀리 세트 이름입니다.

   이 지정 지시어는 지정된 파일 시스템의 파일 세트를 아카이브 매체의 복사본 세트에 매핑합니다. 파일 세트는 모든 파일 시스템만큼 크거나 약간의 파일만큼 작을 수 있습니다. 그러나 최상의 성능과 효율성을 위해 너무 많이 지정하면 안됩니다. 과도한 매체 마운트, 불필요한 매체 재배치 및 전반적인 매체 사용률 저하를 가져올 수 있으므로 아카이브 세트를 필요한 것 이상으로 만들지 마십시오. 대부분의 경우 파일 시스템당 하나의 아카이브 세트를 지정합니다.

   예제에서는 아카이빙 파일 시스템 *samms*에 대한 아카이브 세트 지정 지시어를 시작합니다.

```
# Configuration file for archiving file systems
#---------------------------------------------------------------------
# General Directives
archivemeta = off                                                 # default
examine = noscan                                                  # default
#---------------------------------------------------------------------
```

```
# Archive Set Assignments
fs = samms                                      # Archiving File System
```

5. 다음 라인에서 아카이브 로깅을 사용으로 설정합니다. *logfile = path/filename* 지시어를 입력합니다. 여기서 *path/filename*은 위치 및 파일 이름을 지정합니다.

   위에서 언급한 것처럼 파일 시스템 손실 후에 완벽한 복구를 위해 아카이브 로그 데이터가 필수적입니다. 따라서 아카이버 로그를 */var/adm/*과 같은 Oracle HSM 디렉토리가 아닌 디렉토리에 기록하도록 Oracle HSM를 구성하고 복사본을 정기적으로 저장합니다. 모든 파일 시스템에 대한 아카이버 작업을 함께 기록하는 전역 *archiver.log*를 만들 수도 있지만 각 파일 시스템에 대한 로그를 구성하면 파일 복구 도중 로그를 더 쉽게 검색할 수 있습니다. 따라서 예제에서는 파일 시스템 지정 지시어와 함께 */var/adm/samms.archiver.log*를 지정합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/archiver.cmd
...
#-----------------------------------------------------------------------
# Archive Set Assignments
fs = samms                                       # Archiving File System
logfile = /var/adm/samms.archiver.log
```

6. 다음 라인에서 이 파일 시스템의 파일을 아카이브 세트에 지정합니다. 만들어야 하는 각 아카이브 세트에 대해 *archiveset-name starting-directory expression* 지시어를 입력합니다. 여기서 각 항목은 다음과 같습니다.
   - *archiveset-name*은 새 아카이브 세트에 대해 선택한 이름입니다.
   - *starting-directory*는 Oracle HSM가 파일 검색을 시작할 디렉토리에 대한 경로입니다(파일 시스템 마운트 지점 기준).
   - *expression*은 Solaris *find* 명령에 의해 정의된 부울 표현식 중 하나입니다.

   대부분의 경우 아카이브 세트 정의를 가능한 포괄적이고 간단하게 유지해야 합니다. 그러나 상황에 따라서는 정규 표현식을 사용하여 사용자 또는 그룹 파일 소유권, 파일 크기, 파일 날짜/시간 기록, 파일 이름 등과 같은 더 제한적인 추가 식별자를 지정함으로써 아카이브 세트 멤버쉽을 제한할 수 있습니다. 자세한 내용은 *archiver.cmd* 매뉴얼 페이지를 참조하십시오.

   예제에서는 *samms* 파일 시스템에서 발견된 모든 파일을 *allsamms*라는 단일 아카이브 세트에 포함합니다. 마운트 지점 디렉토리 자체(*/samms*)에서 검색을 시작하도록 점(*.*)을 사용하여 경로를 지정합니다.

```
...
#-----------------------------------------------------------------------
# Archive Set Assignments
fs = samms                                       # Archiving File System
logfile = /var/adm/samms.archiver.log
allsamms .
```

7. 다음으로 *samms* 파일 시스템의 *allsamms* 아카이브 세트에 대한 복사본 지시어를 추가합니다. 각 복사본에 대해 하나 이상의 공백으로 라인을 시작하고 *copy-number -release -norelease archive-age unarchive-age* 지시어를 입력합니다. 여기서 각 항목은 다음과 같습니다.
   - *copy-number*는 정수입니다.
   - *-release* 및 *-norelease*는 복사본이 작성된 후 디스크 캐시 공간이 관리되는 방법을 제어하는 선택적 매개변수입니다. *-release*는 그 자체로 해당 복사본이 작성되자마자 디스크 공간이 자동으로 릴리스되도록 합니다. *-norelease*는 그 자체로 *-norelease*가 설정된 모든 복사본이 작성되고 릴리서 프로세스가 실행될 때까지 디스크 공간의 릴리스를 방지합니다. *-release* 및 *-norelease*는 함께 사용되어 *-norelease*가 설정된 모든 복사본이 작성되자마자 디스크 캐시 공간을 자동으로 해제합니다.
   - *archive-age*는 파일이 아카이브되기 전에 마지막으로 수정된 시간으로부터 경과해야 하는 시간입니다. 정수 및 식별자 *s*(초), *m*(분), *h*(시간), *d*(일), *w*(주) 및 *y*(년)를 임의로 조합하여 시간을 표시합니다. 기본값은 *4m*입니다.
   - *unarchive-age*는 파일의 아카이브를 취소할 수 있기 전에 마지막으로 수정된 시간으로부터 경과해야 하는 시간입니다. 기본값은 복사본의 아카이브를 취소하지 않는 것입니다.

   전체 중복성을 위해 항상 각 아카이브 세트의 복사본을 *2*개 이상 지정합니다(최대값은 4). 예제에서는 복사본이 도달하는 아카이브 수명을 15분으로 하여 복사본 3개를 각각 *-norelease*와 함께 지정합니다. 복사본 *1*은 디스크 아카이브 볼륨을 사용하여 작성되고 복사본 *2* 및 *3*은 테이프 매체에 작성됩니다.

```
...
#------------------------------------------------------------------------
# Archive Set Assignments
fs = samms                                        # Archiving File System
logfile = /var/adm/samms.archiver.log
allsamms .
    1 -norelease 15m
    2 -norelease 15m
    3 -norelease 15m
```

8. 나머지 모든 파일 시스템에 대한 아카이브 세트를 정의합니다.

   예제에서는 QFS 파일 시스템 *DISKVOL1*을 복사본 프로세스에 대한 아카이브 매체로 구성했습니다. 따라서 *fs = DISKVOL1*에 대한 항목을 시작합니다. 그러나 아카이브 복사본의 아카이브 복사본을 작성하지는 않을 것입니다. 따라서 로그 파일을 지정하지 않으며 이 파일 시스템의 파일에 대한 아카이빙을 방지하는 *no_archive*라는 특수한 아카이브 세트를 사용합니다.

```
...
```

```
#------------------------------------------------------------------------
# Archive Set Assignments
fs = samms                                        # Archiving File System
logfile = /var/adm/samms.archiver.log
allsamms .
    1 -norelease 15m
    2 -norelease 15m
    3 -norelease 15m
fs = DISKVOL1                             # QFS File System (Archival Media)
no_archive .
```

9. 다음으로 복사본이 만들어지는 방법을 제어하는 지시어를 입력합니다. 새 라인에서 *params* 키워드를 입력하여 복사 매개변수 섹션을 시작합니다.

```
...
fs = DISKVOL1                              # QFS File System (Archival Media)
no_archive .
#------------------------------------------------------------------------
# Copy Parameter Directives
params
```

10. 모든 아카이브 세트의 모든 복사본에 적용되는 공통 복사 매개변수를 설정해야 할 경우 *allsets -param value ...* 형태의 라인을 입력합니다. 여기서 *allsets*는 구성된 모든 아카이브 세트를 나타내는 특수한 아카이브 세트이고 *-param value ...*는 공백으로 구분된 하나 이상의 매개변수/값 쌍을 나타냅니다.

    매개변수 및 해당 값에 대한 자세한 내용은 *archiver.cmd* 매뉴얼 페이지의 *ARCHIVE SET COPY PARAMETERS SECTION* 절을 참조하십시오.

    예제의 지시어는 대부분의 파일 시스템에 적합합니다. 특수한 *allsets* 아카이브 세트는 최적의 성능 및 쉬운 관리를 위해 모든 아카이브 세트가 일관되게 처리되도록 합니다. *-sort path* 매개변수는 동일한 디렉토리의 파일이 아카이브 매체에 함께 남아 있도록 모든 아카이브 세트의 모든 복사본에 대한 테이프 아카이브(*tar*) 파일이 경로에 따라 정렬되도록 합니다. *-offline_copy stageahead* 매개변수는 오프라인 파일을 아카이브할 때 성능을 향상시킬 수 있습니다.

```
...
#------------------------------------------------------------------------
# Copy Parameter Directives
params
allsets -sort path -offline_copy stageahead
```

11. 모든 아카이브 세트의 특정 복사본에 대한 복사 매개변수를 설정해야 할 경우 *allfiles.copy-number -param value ...* 형태의 라인을 입력합니다. 여기서 *allsets*는 구성된 모든 아카이브 세트를 나타내는 특수한 아카이브 세트이고 *copy-*

*number*는 지시어가 적용되는 복사본 번호이며 *-param value ...*는 공백으로 구분된 하나 이상의 매개변수/값 쌍을 나타냅니다.

매개변수 및 해당 값에 대한 자세한 내용은 *archiver.cmd* 매뉴얼 페이지의 *ARCHIVE SET COPY PARAMETERS SECTION* 절을 참조하십시오.

예제에서 *allfiles.1 -startage 10m -startsize 500M -drives 10 -archmax 1G* 지시어는 디스크 볼륨에 맞게 복사본 *1*을 최적화합니다. 아카이빙을 위해 선택한 첫번째 파일이 10분 동안 대기했거나 대기 중인 모든 파일의 총 크기가 최소한 500MB 이상인 경우 아카이빙이 시작됩니다. 최대 10개의 드라이브를 사용하여 복사본을 만들 수 있고 복사본의 각 *tar* 파일은 1GB보다 클 수 없습니다.

나머지 지시어 *allfiles.2 -startage 24h -startsize 20G -drives 2 -archmax 24G -reserve set* 및 *allfiles.3 -startage 48h -startsize 20G -drives 2 -archmax 24G -reserve set*는 테이프 매체에 맞게 복사본 *2* 및 *3*을 최적화합니다. 아카이빙을 위해 선택한 첫번째 파일이 각각 24 또는 48시간 동안 대기했거나 대기 중인 모든 파일의 총 크기가 최소한 20GB 이상인 경우 아카이빙이 시작됩니다. 최대 2개의 드라이브를 사용하여 이러한 복사본을 만들 수 있고(기반구조에 맞게 개수 조정) 복사본의 각 *tar* 파일은 24GB보다 클 수 없습니다. *-reserve set*는 각 아카이브 세트의 복사본 *2* 및 *3*이 동일한 아카이브 세트의 복사본만 포함하는 테이프 매체를 사용하여 작성되도록 합니다.

```
...
#------------------------------------------------------------------------
# Copy Parameter Directives
params
allsets -sort path -offline_copy stageahead
allfiles.1 -startage 10m -startsize 500M -drives 10 -archmax 1G
allfiles.2 -startage 24h -startsize 20G -drives 2 -archmax 24G -reserve set
allfiles.3 -startage 48h -startsize 20G -drives 2 -archmax 24G -reserve set
```

이 절의 예제에서는 디스크 볼륨을 아카이빙에 사용한다고 가정합니다. 테이프 볼륨만 사용할 경우 복사본 2개를 지정하고 테이프에 더 자주 아카이브합니다. 기반구조에 맞게 지정된 드라이브 수를 조정하고 나면 다음 구성은 대부분의 파일 시스템에 적합합니다.

```
allsets -sort path -offline_copy stageahead -reserve set
allfiles.1 -startage 8h -startsize 8G -drives 2 -archmax 10G
allfiles.2 -startage 24h -startsize 20G -drives 2 -archmax 24G
```

12. 특정 아카이브 세트 및 복사본에 대한 지시어를 설정해야 할 경우 *archive-set-name.copy-number -param value ...* 형태의 라인을 입력합니다. 여기서 *archive-set-name*은 아카이브 세트에 사용한 이름이고 *copy-number*는 지시어가 적용되는 복사본의 번호이며 *-param value ...*는 공백으로 구분된 하나 이상의 매개변수/값 쌍을 나타냅니다.

매개변수 및 해당 값에 대한 자세한 내용은 *archiver.cmd* 매뉴얼 페이지의 *ARCHIVE SET COPY PARAMETERS SECTION* 절을 참조하십시오.

아래 예제에서는 *corpfs* 파일 시스템에 대한 2개의 아카이브 세트 *hq* 및 *branches*가 정의됩니다. *hq.1* 및 *hq.2*에 대한 복사본 지시어는 아카이브 세트 *hq*에만 적용됩니다. 아카이브 세트 *branches*는 영향을 받지 않습니다.

```
#-----------------------------------------------------------------------
# Archive Set Assignments
fs = corpfs
logfile = /var/adm/corporatefs.archive.log
hq /corpfs/hq/
    1 -norelease 15m
    2 -norelease 15m
branches /corpfs/branches/
    1 -norelease 15m
    2 -norelease 15m
#-----------------------------------------------------------------------
# Copy Parameter Directives
params
hq.1 -drives 4
hq.2 -drives 2
```

13. 필요한 모든 복사 매개변수를 설정하고 나면 새 라인에 *endparams* 키워드를 입력하여 복사 매개변수 목록을 닫습니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/archiver.cmd
...
#-----------------------------------------------------------------------
# Copy Parameter Directives
params
allsets -sort path -offline_copy stageahead
allfiles.1 -startage 10m -startsize 500M -drives 10 -archmax 1G
allfiles.2 -startage 24h -startsize 20G -drives 2 -archmax 24G -reserve set
allfiles.3 -startage 48h -startsize 20G -drives 2 -archmax 24G -reserve set
endparams
```

14. 선택적으로 *vsnpools* 키워드와 *pool-name media-type volumes* 형태인 하나 이상의 지시어를 입력하여 매체 풀을 정의할 수 있습니다. 여기서 *pool-name*은 풀에 지정한 이름이고 *media-type*은 부록 A. 장비 유형 용어집에 정의된 매체 유형 코드 중 하나이며 *volumes*는 하나 이상의 VSN(볼륨 일련 번호)과 일치하는 정규 표현식입니다. *endvsnpools* 키워드를 사용하여 지시어 목록을 닫습니다.

매체 풀은 선택적이며 일반적으로 아카이빙 프로세스에 사용할 수 있는 매체를 사용자가 제한하지는 않습니다. 따라서 예제에서는 매체 풀을 정의하지 않습니다. 자세한 내용은 *archiver.cmd* 매뉴얼 페이지의 *VSN POOL DEFINITIONS SECTION*을 참조하십시오.

15. 다음으로 아카이브 세트 복사본이 사용해야 하는 아카이브 매체를 식별하는 것을 시작합니다. 새 라인에 *vsns* 키워드를 입력합니다.

```
...
#------------------------------------------------------------------------
# Copy Parameter Directives
params
allsets -sort path -offline_copy stageahead
allfiles.1 -startage 10m -startsize 500M -drives 10 -archmax 1G
allfiles.2 -startage 24h -startsize 20G -drives 2 -archmax 24G -reserve set
allfiles.3 -startage 48h -startsize 20G -drives 2 -archmax 24G -reserve set
endparams
#------------------------------------------------------------------------
# VSN Directives
vsns
```

16. *archive-set-name.copy-number media-type volumes* 형태의 라인을 입력하여 각 아카이브 세트 복사본에 대한 매체를 지정합니다. 여기서 *archive-set-name.copy-number*는 지시어를 적용할 아카이브 세트 및 복사본을 지정하고 *media-type*은 부록 A. 장비 유형 용어집에 정의된 매체 유형 코드 중 하나이며 *volumes*는 하나 이상의 VSN(볼륨 일련 번호)과 일치하는 정규 표현식입니다.

    전체 중복성을 위해 항상 각 아카이브 세트 복사본을 매체의 다른 범위에 지정하여 두 복사본이 동일한 물리적 볼륨에 상주하지 않게 합니다. 가능한 경우 항상 하나 이상의 복사본을 이동식 매체(예: 테이프)에 지정합니다.

    예제에서는 모든 아카이브 세트의 첫번째 복사본을 *DISKVOL1* - *DISKVOL15* 범위의 볼륨 일련 번호를 가진 아카이브 디스크 매체(유형 *dk*)로 보냅니다. 또한 모든 아카이브 세트의 두번째 복사본을 *VOL000* - *VOL199* 범위의 볼륨 일련 번호를 가진 테이프 매체(유형 *tp*)로 보내고 세번째 복사본을 *VOL200* - *VOL399* 범위의 볼륨 일련 번호를 가진 테이프 매체(유형 *tp*)로 보냅니다.

```
...
#------------------------------------------------------------------------
# Copy Parameter Directives
params
allsets -sort path -offline_copy stageahead
allfiles.1 -startage 10m -startsize 500M -drives 10 -archmax 1G
allfiles.2 -startage 24h -startsize 20G -drives 2 -archmax 24G -reserve set
allfiles.3 -startage 48h -startsize 20G -drives 2 -archmax 24G -reserve set
endparams
```

```
#------------------------------------------------------------------------
# VSN Directives
vsns
allfiles.1 dk DISKVOL[1-15]
allfiles.2 tp VOL[0-1][0-9][0-9]
allfiles.2 tp VOL[2-3][0-9][0-9]
```

17. 모든 아카이브 세트 복사본에 대한 매체를 지정한 경우 *endvsns* 키워드를 새 라인에 입력하여 *vsns* 지시어 목록을 닫습니다. 파일을 저장하고 편집기를 닫습니다.

```
...
#------------------------------------------------------------------------
# Copy Parameter Directives
params
allsets -sort path -offline_copy stageahead
allfiles.1 -startage 10m -startsize 500M -drives 10 -archmax 1G
allfiles.2 -startage 24h -startsize 20G -drives 2 -archmax 24G -reserve set
allfiles.3 -startage 48h -startsize 20G -drives 2 -archmax 24G -reserve set
endparams
#------------------------------------------------------------------------
# VSN Directives
vsns
allfiles.1 dk DISKVOL[1-15]
allfiles.2 tp VOL[0-1][0-9][0-9]
allfiles.2 tp VOL[2-3][0-9][0-9]
endvsns
:wq
root@solaris:~#
```

18. *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. 예제에서는 오류가 있습니다.

```
root@solaris:~# archiver -lv
Reading '/etc/opt/SUNWsamfs/archiver.cmd'.
...
13: # File System Directives
14: #
15: fs = samms
16: logfile = /var/adm/samms.archiver.log
17: all .
18:     1 -norelease 15m
19:     2 -norelease 15m
```

```
20: fs=DISKVOL1                          # QFS File System (Archival Media)
21:
...
42: endvsns
DISKVOL1.1 has no volumes defined
1 archive set has no volumes defined
root@solaris:~#
```

19. *archiver.cmd* 파일에서 오류가 발견된 경우 오류를 수정하고 파일을 다시 검사합니다.

    위 예제에서는 디스크 아카이브로 구성한 QFS 파일 시스템인 *DISKVOL1* 파일 시스템의 지시어에 대한 *no_archive* 지시어를 잊어버리고 입력하지 않았습니다. 생략된 부분을 수정할 경우 *archiver -lv*는 오류 없이 실행됩니다.

```
root@solaris:~# archiver -lv
Reading '/etc/opt/SUNWsamfs/archiver.cmd'.
...
20: fs=DISKVOL1                           # QFS File System (Archival Media)
21: no_archive .
...
42: endvsns
Notify file: /etc/opt/SUNWsamfs/scripts/archiver.sh
...
allfiles.1
    startage: 10m startsize: 500M drives 10: archmax: 1G
 Volumes:
  DISKVOL1 (/diskvols/DISKVOL15)
  ...
  DISKVOL15 (/diskvols/DISKVOL3)
 Total space available:   150T
allfiles.2
    startage: 24h startsize: 20G drives: 2 archmax: 24G reserve: set
 Volumes:
   VOL000
...
   VOL199
 Total space available:  300T
allfiles.3
    startage: 48h startsize: 20G drives: 2 archmax: 24G reserve: set
 Volumes:
   VOL200
...
   VOL399
 Total space available:  300T
root@solaris:~#
```

20. Oracle HSM 소프트웨어에 *archiver.cmd* 파일을 다시 읽고 그에 따라 재구성하도록 지시합니다. *samd config* 명령을 사용합니다.

```
root@solaris:~# /opt/SUNWsamfs/sbin/samd config
Configuring SAM-FS
root@solaris:~#
```

21. 텍스트 편집기에서 */etc/opt/SUNWsamfs/releaser.cmd* 파일을 열고 *list_size = 300000* 라인을 추가한 다음 파일을 저장하고 편집기를 닫습니다.

    list_size 지시어는 파일 시스템에서 한 번에 릴리스할 수 있는 파일 수를 [*10-2147483648*] 범위의 정수로 설정합니다. inode 100만개를 위한 충분한 공간이 *.inodes* 파일에 있는 경우(inode 노드당 512- 바이트 허용) 기본값은 *100000*입니다. 그렇지 않은 경우 기본값은 *30000*입니다. 이 숫자를 *300000*으로 늘리면 많은 수의 작은 파일을 포함하는 일반적인 파일 시스템에 더 적합합니다.

    이 예에서는 *vi* 편집기를 사용합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/releaser.cmd
# releaser.cmd
logfile = /var/opt/SUNWsamfs/releaser.log
list_size = 300000
:wq
root@solaris:~#
```

22. 텍스트 편집기에서 */etc/opt/SUNWsamfs/stager.cmd* 파일을 열고 *maxactive = stage-requests* 라인을 추가합니다. 여기서 *stage-requests*는 8GB 이상의 RAM이 있는 호스트의 경우 *500000*이고 8GB 미만의 RAM이 있는 호스트의 경우 *100000*입니다. 파일을 저장하고 편집기를 닫습니다.

    maxactive 지시어는 한 번에 활성 상태일 수 있는 최대 스테이지 요청 수를 [*1-500000*] 범위의 정수로 설정합니다. 기본값은 호스트 메모리의 GB당 스테이지 요청 5000개를 허용하는 것입니다.

    이 예에서는 *vi* 편집기를 사용합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/stager.cmd
# stager.cmd
logfile = /var/opt/SUNWsamfs/stager.log
maxactive = 300000
:wq
root@solaris:~#
```

23. 재활용은 기본적으로 사용으로 설정되지 않습니다. 따라서 이동식 매체 볼륨의 재활용이 필요한 경우 "재활용 프로세스 구성"으로 이동합니다.
24. 아카이빙 Oracle HSM 파일 시스템에 대한 *mcf* 파일에서 아카이빙 장비 섹션에 네트워크 연결 테이프 라이브러리가 포함된 경우 "네트워크 연결 테이프 라이브러리에 저장된 아카이브 매체 카탈로그화"로 이동합니다.
25. 아카이브 테이프 볼륨의 데이터 무결성을 확인할 수 있어야 하는 경우 "아카이브 매체 검증 구성"으로 이동합니다.
26. 그렇지 않은 경우 "파일 시스템 보호 구성" 작업을 수행합니다.

재활용 프로세스 구성

사용자가 지정한 수보다 적은 유효 아카이브 세트가 이동식 매체 볼륨에 포함된 경우 리사이클러는 원래 볼륨을 장기적인 저장을 위해 내보내거나 재사용을 위해 레이블을 다시 지정할 수 있도록 다른 볼륨에서 유효 데이터를 통합합니다. 다음 두 가지 방법 중 하나로 재활용을 구성할 수 있습니다.

- 아카이브 세트별 재활용 구성

  아카이브 세트별로 매체를 재활용하는 경우 *archiver.cmd* 파일에 재활용 지시어를 추가합니다. 각 아카이브 세트 복사본의 매체가 재활용되는 방법을 정확하게 지정할 수 있습니다. 아카이브 세트의 멤버만 고려되므로 재활용 조건은 더 좁은 범위로 적용됩니다.

  가능한 경우 라이브러리가 아니라 아카이브 세트별로 매체를 재활용합니다. Oracle HSM 아카이빙 파일 시스템에서 재활용은 논리적으로 라이브러리 관리가 아니라 파일 시스템 작업의 일부입니다. 재활용은 아카이빙, 릴리스 및 스테이징을 보완합니다. 따라서 아카이빙 프로세스의 일부로 구성하는 것이 합리적입니다. 디스크 아카이브 볼륨 및/또는 SAM-Remote가 구성에 포함된 경우 아카이브 세트별로 재활용을 구성해야 합니다.
- 라이브러리별 재활용 구성

  라이브러리별로 매체를 재활용하는 경우 *recycler.cmd* 파일에 재활용 지시어를 추가합니다. 따라서 지정한 라이브러리에 포함된 모든 매체에 공통된 재활용 매개변수를 설정할 수 있습니다. 재활용 지시어는 라이브러리의 모든 볼륨에 적용되므로 본질적으로 아카이브 세트 특정 지시어보다 덜 세부적입니다. 지정된 VSN(볼륨 일련 번호)을 검사에서 명시적으로 제외할 수 있습니다. 그렇지 않을 경우 재활용 프로세스는 단순히 현재 유효한 아카이브 파일로 인식되지 않는 항목을 포함하는 볼륨을 찾습니다.

  결과적으로 라이브러리별 재활용은 재활용 중인 파일 시스템의 일부가 아닌 파일을 삭제할 수 있습니다. 재활용 지시어가 이러한 파일을 명시적으로 제외하지 않을 경우 아카이브 로그 및 라이브러리 카탈로그나 다른 파일 시스템의 아카이브 매체에 대한 백업 복사본과 같은 유용한 데이터가 위험해질 수 있습니다. 이러한 이유로 SAM-Remote를 사용하는 중이면 라이브러리별로 재활용할 수 없습니다. SAM-Remote 서버에 의해 제어되는 라이브러리의 볼륨에는 서버가 아니라 클라이언트가 소유하는 외래 아카이브 파일이 포함되어 있습니다.

아카이브 세트별 재활용 구성

1. Oracle HSM 파일 시스템 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. */etc/opt/SUNWsamfs/archiver.cmd* 파일을 텍스트 편집기에서 열고 아래로 스크롤하여 복사본 *params* 절을 찾습니다.

   예제에서는 *vi* 편집기를 사용합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/archiver.cmd
...
#------------------------------------------------------------------------
# Copy Parameter Directives
params
allsets -sort path -offline_copy stageahead
allfiles.1 -startage 6h  -startsize 6G  -startcount 500000
allfiles.2 -startage 24h -startsize 20G -startcount 500000 -drives 5
```

3. *archiver.cmd* 파일의 *params* 섹션에 아카이브 세트별 리사이클러 지시어를 *archive-set directive-list* 형태로 입력합니다. 여기서 archive-set는 아카이브 세트 중 하나이고 *directive-list*는 지시어 이름/값 쌍의 공백으로 구분된 목록입니다(재활용 지시어 목록은 *archiver.cmd* 매뉴얼 페이지 참조). 그런 다음 파일을 저장하고 편집기를 닫습니다.

   예제에서는 아카이브 세트 *allfiles.1* 및 *allfiles.2*에 대한 재활용 지시어를 추가합니다. *-recycle_mingain 30* 및 *-recycle_mingain 90* 지시어는 각각 볼륨 용량의 최소한 30% 및 90%를 복구할 수 없을 경우 볼륨을 재활용하지 않습니다. *-recycle_hwm 60* 지시어는 이동식 매체 용량의 60%가 사용된 경우 재활용을 시작합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/archiver.cmd
...
#------------------------------------------------------------------------
# Copy Parameters Directives
params
allsets -sort path -offline_copy stageahead
allfiles.1 -startage 6h  -startsize 6G -startcount 500000
allfiles.1 -recycle_mingain 30 -recycle_hwm 60
allfiles.2 -startage 6h -startsize 6G -startcount 500000
allfiles.2 -recycle_mingain 90 -recycle_hwm 60
endparams
#------------------------------------------------------------------------
# VSN Directives
vsns
```

```
allfiles.1 dk DISKVOL1
allfiles.2 tp VOL0[0-1][0-9]
endvsns
:wq
[root@solaris:~#
```

4. *archiver.cmd* 파일에 오류가 있는지 확인합니다. *archiver -lv* 명령을 사용합니다.

   archiver -lv 명령은 *archiver.cmd*를 읽고 오류가 발견되지 않은 경우 구성 보고서를 생성합니다. 그렇지 않고 오류가 발견되면 작업을 중지합니다. 예제에서는 파일에 오류가 없습니다.

```
root@solaris:~# archiver -lv
Reading '/etc/opt/SUNWsamfs/archiver.cmd'.
...
   VOL399
 Total space available:  300T
root@solaris:~#
```

5. *archiver.cmd* 파일에서 오류가 발견된 경우 오류를 수정하고 파일을 다시 검사합니다.
6. 텍스트 편집기에서 *recycler.cmd* 파일을 만듭니다. 리사이클러 로그의 경로와 파일 이름을 지정합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   */var/adm/*과 같은 Oracle HSM 디렉토리가 아닌 디렉토리에 로그를 기록하도록 Oracle HSM를 구성합니다. 예제에서는 *vi* 편집기를 사용하고 */var/adm/recycler.log*를 지정합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/recycler.cmd
logfile = /var/adm/recycler.log
:wq
root@solaris:~#
```

7. 텍스트 편집기에서 */etc/opt/SUNWsamfs/scripts/recycler.sh* 스크립트를 열고 재활용된 이동식 매체 볼륨을 처리하기 위한 셸 명령을 입력합니다.

   재활용 프로세스는 유효한 아카이브 복사본이 소모된 이동식 매체 볼륨을 식별한 경우 재활용된 매체의 처리를 수행하도록 설계된 C-셸 스크립트인 *recycler.sh* 파일을 호출합니다.

   볼륨을 재활용할 준비가 되었다는 것을 관리자에게 알리는 것부터 재사용을 위해 볼륨의 레이블을 다시 지정하거나 장기적인 내역 보존을 위해 라이브러리에서 볼륨을 내보내는 것에 이르기까지 필요한 작업을 수행하도록 파일을 편집합니다.

   기본적으로 이 스크립트는 스크립트를 설정하도록 *root* 사용자에게 미리 알립니다.

8. 아카이빙 Oracle HSM 파일 시스템에 대한 *mcf* 파일에서 아카이빙 장비 섹션에 네트워크 연결 테이프 라이브러리가 포함된 경우 "네트워크 연결 테이프 라이브러리에 저장된 아카이브 매체 카탈로그화"로 이동합니다.
9. 그렇지 않은 경우 "파일 시스템 보호 구성"으로 이동합니다.

라이브러리별 재활용 구성

1. Oracle HSM 파일 시스템 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. 텍스트 편집기에서 */etc/opt/SUNWsamfs/recycler.cmd* 파일을 만듭니다.

   예제에서는 *vi* 편집기를 사용합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/recycler.cmd
# Configuration file for archiving file systems
#-------------------------------------------------------------------------
```

3. *logfile* 지시어를 사용하여 리사이클러 로그의 경로와 파일 이름을 지정합니다.

   */var/adm/*과 같은 Oracle HSM 디렉토리가 아닌 디렉토리에 로그를 기록하도록 Oracle HSM를 구성합니다. 예제에서는 */var/adm/recycler.log*를 지정합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/recycler.cmd
# Configuration file for archiving file systems
#-------------------------------------------------------------------------
logfile = /var/adm/recycler.log
```

4. 재활용하지 않아야 하는 볼륨이 아카이브 매체 라이브러리에 있는 경우 *no_recycle media-type volumes* 지시어를 입력합니다. 여기서 *media-type*은 부록 A. 장비 유형 용어집에 정의된 매체 유형 코드 중 하나이고 *volumes*는 하나 이상의 VSN(볼륨 일련 번호)과 일치하는 정규 표현식입니다.

   예제에서는 [*VOL020-VOL999*] 범위의 볼륨에 대한 재활용을 사용 안함으로 설정합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/recycler.cmd
# Configuration file for archiving file systems
#-------------------------------------------------------------------------
logfile = /var/adm/recycler.log
no_recycle tp VOL[0-9][2-9][0-9]
```

5. 새 라인에 *library parameters* 지시어를 입력합니다. 여기서 *library*는 */etc/opt/SUNWsamfs/mcf* 파일이 이동식 매체 라이브러리에 지정한 패밀리 세트 이름이고 *parameters*는 다음 목록에 제공된 매개변수/값 쌍의 공백으로 구분된 목록입니다.

- *-dataquantity size*는 한 번에 다시 아카이브하도록 예약할 수 있는 최대 데이터 양을 *size*로 설정합니다. 여기서 *size*는 바이트 수입니다. 기본값은 1GB입니다.
- *-hwm percent*는 사용될 경우 재활용을 트리거하는 총 매체 용량의 백분율에 해당하는 라이브러리의 *high-water mark*를 설정합니다. 고수위는 [*0-100*] 범위의 숫자인 *percent*로 지정됩니다. 기본값은 *95*입니다.
- *-ignore*는 *recycler.cmd* 파일을 손상 없이 테스트할 수 있도록 이 라이브러리의 재활용을 방지합니다.
- *-mail address*는 재활용 메시지를 *address*로 보냅니다. 여기서 *address*는 유효한 전자 메일 주소입니다. 기본적으로 메시지는 전송되지 않습니다.
- *-mingain percent*는 총 용량의 백분율로 표시된 최소 크기만큼 사용 가능한 여유 공간을 늘릴 수 있는 볼륨만 재활용하도록 제한합니다. 이 최소 증가는 [*0-100*] 범위의 숫자인 *percent*로 지정됩니다. 총 용량이 200GB 미만인 볼륨의 경우 기본값은 *60*이고 총 용량이 200GB 이상인 볼륨의 경우 기본값은 *90*입니다.
- *-vsncount count*는 한 번에 다시 아카이브하도록 예약할 수 있는 최대 볼륨 수를 *count*로 설정합니다. 기본값은 *1*입니다.

예제에서는 *library1* 라이브러리에 대한 고수위를 95%로 설정하고 카트리지당 60%의 최소 용량 증가를 지정합니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/recycler.cmd
# Configuration file for archiving file systems
#---------------------------------------------------------------------
logfile = /var/adm/recycler.log
no_recycle tp VOL[0-9][2-9][0-9]
library1 -hwm 95 -mingain 60
```

6. Oracle HSM 구성의 일부인 다른 모든 라이브러리에 대해 앞의 단계를 반복합니다. 그런 다음 *recycler.cmd* 파일을 저장하고 편집기를 닫습니다.

```
root@solaris:~# vi /etc/opt/SUNWsamfs/recycler.cmd
# Configuration file for archiving file systems
#---------------------------------------------------------------------
logfile = /var/adm/recycler.log
no_recycle tp VOL[0-9][2-9][0-9]
library1 -hwm 95 -mingain 60
:wq
root@solaris:~#
```

7. 아카이빙 Oracle HSM 파일 시스템에 대한 *mcf* 파일에서 아카이빙 장비 섹션에 네트워크 연결 테이프 라이브러리가 포함된 경우 "네트워크 연결 테이프 라이브러리에 저장된 아카이브 매체 카탈로그화"로 이동합니다.
8. 그렇지 않은 경우 “파일 시스템 보호 구성”으로 이동합니다.

네트워크 연결 테이프 라이브러리에 저장된 아카이브 매체 카탈로그화

파일 시스템을 마운트한 후 Oracle HSM 소프트웨어는 *mcf* 파일에 구성된 각각의 자동화된 라이브러리에 대한 카탈로그를 만듭니다. 그러나 네트워크 연결 라이브러리가 있는 경우 해당 카탈로그를 채우기 위해 추가 단계를 수행해야 합니다.

다음과 같이 하십시오.

1. 파일 시스템 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. 아카이빙 파일 시스템에서 Oracle StorageTek ACSLS 연결 테이프 라이브러리를 사용할 경우 필요한 Oracle HSM 아카이브 매체를 라이브러리의 스크래치 풀에서 가져와서 카탈로그를 자동으로 생성합니다. *samimport -c volumes -s pool* 명령을 사용합니다. 여기서 *volumes*는 필요한 볼륨 수이고 *pool*은 라이브러리에 정의된 스크래치 매체 풀의 이름입니다. 여기서 중지합니다.

   예제에서는 *scratch*라는 풀에서 가져온 테이프 볼륨 *20*개를 요청합니다.

```
root@solaris:~# samimport -c 20 -s scratch
```

3. 단일 비공유 논리적 라이브러리로 구성된 IBM 3494 네트워크 연결 라이브러리가 아카이빙 파일 시스템에서 사용될 경우 필요한 테이프 볼륨을 라이브러리 메일 슬롯에 배치하여 라이브러리에서 자동으로 카탈로그화할 수 있게 합니다. 여기서 중지합니다.

   mcf 파일의 *Additional Parameters* 필드에 *access=private*가 지정된 경우 IBM 3494 라이브러리는 단일 논리적 라이브러리로 구성됩니다. *access=shared*인 경우 IBM 3494 라이브러리는 여러 논리적 라이브러리로 구분되며 사용자는 아래 지정된 방법을 사용해야 합니다.
4. 그렇지 않고 공유 IBM 3494 네트워크 연결 라이브러리나 다른 네트워크 연결 라이브러리가 아카이빙 파일 시스템에서 사용될 경우 텍스트 편집기를 사용하여 카탈로그 입력 파일을 만듭니다.

   예제에서는 *vi* 편집기를 사용하여 *input3494cat* 파일을 만듭니다.

```
root@solaris:~# vi input3494cat
~
"~/input3494cat" [New File]
```

5. 레코드 *index*를 입력하여 레코드를 시작합니다. 첫번째 레코드의 경우 항상 *0*을 입력하고 이후의 각 레코드에 대해 인덱스를 증분합니다. 필드의 끝을 나타내는 공백을 입력합니다.

행은 레코드를 정의하고 공백은 *build_cat* 입력 파일의 필드를 구분합니다. 첫번째 필드 *index*의 값은 단순히 Oracle HSM 카탈로그 내의 레코드를 식별하는 *0*부터 시작하는 연속 정수입니다. 예제에서는 첫번째 레코드에 해당하므로 *0*을 입력합니다.

```
0
~
"~/input3494cat" [New File]
```

6. 레코드의 두번째 필드에는 테이프 볼륨의 VSN(볼륨 일련 번호)을 입력하고 VSN이 없는 경우 단일 *?*(물음표)를 입력합니다. 그런 다음 필드의 끝을 나타내는 공백을 입력합니다.

   공백 문자를 포함하는 값(있는 경우)을 큰 따옴표로 묶습니다(*"VOL 01"*). 이 예제에서는 첫번째 볼륨의 VSN에 공백이 포함되어 있지 않습니다.

```
0 VOL001
~
"~/input3494" [New File]
```

7. 세번째 필드에 볼륨의 바코드(볼륨 일련 번호와 다른 경우), 볼륨 일련 번호 또는 볼륨 일련 번호가 없는 경우 문자열 *NO_BAR_CODE*를 입력합니다. 그런 다음 필드의 끝을 나타내는 공백을 입력합니다.

   예제에서는 첫번째 볼륨의 바코드 값이 VSN과 같습니다.

```
0 VOL001 VOL001
~
"~/input3494cat" [New File]
```

8. 마지막으로 네번째 필드에 볼륨의 매체 유형을 입력합니다. 그런 다음 필드의 끝을 나타내는 공백을 입력합니다.

   매체 유형은 LTO 매체의 경우 *li*와 같은 두 글자의 코드입니다. 매체 장비 유형의 포괄적인 목록은 부록 A. 장비 유형 용어집을 참조하십시오. 예제에서는 LTO 테이프 드라이브가 있는 IBM 3494 네트워크 연결 테이프 라이브러리를 사용하는 중이므로 *li*(종료 공백 포함)를 입력합니다.

```
0 VOL001 VOL001 li
~
"~/input3494cat" [New File]
```

9. 3-6단계를 반복하여 Oracle HSM와 함께 사용하려는 각 볼륨에 대한 추가 레코드를 만듭니다. 그런 다음 파일을 저장합니다.

```
0 VOL001 VOL001 li
1 VOL002 VOL002 li
```

```
...
13 VOL014 VOL014 li
:wq
root@solaris:~#
```

10. *build_cat input-file catalog-file* 명령을 사용하여 카탈로그를 만듭니다. 여기서 *input-file*은 입력 파일의 이름이고 *catalog-file*은 라이브러리 카탈로그에 대한 전체 경로입니다.

    mcf 파일의 *Additional Parameters* 필드에 카탈로그 이름을 지정한 경우 해당 이름을 사용합니다. 그렇지 않고 카탈로그를 만들지 않은 경우 Oracle HSM 소프트웨어는 파일 이름 *family-set-name*을 사용하여 */var/opt/SUNWsamfs/catalog/* 디렉토리에 기본 카탈로그를 만듭니다. 여기서 *family-set-name*은 *mcf* 파일에 있는 라이브러리에 사용되는 장비 이름입니다. 예제에서는 패밀리 세트 *i3494*를 사용합니다.

```
root@solaris:~# build_cat input_vsns /var/opt/SUNWsamfs/catalog/i3494
```

11. 아카이빙 파일 시스템이 공유될 경우 각각의 잠재적 메타데이터 서버에서 앞의 단계를 반복합니다.

    이제 아카이빙 파일 시스템이 완료되었으며 사용할 준비가 되었습니다.

12. 이제 파일 시스템 보호 구성을 수행합니다.

파일 시스템 보호 구성

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.

따라서 파일 시스템 보호에는 약간의 계획이 필요합니다. 한편으로는 손실 또는 손상된 파일 및 파일 시스템을 최대한 복구할 수 있도록 복구 지점 및 로그 파일 복사본을 충분히 자주 만들고 이러한 복사본을 오랫동안 보유해야 할 수 있습니다. 다른 한편으로는 데이터 파일이 활발하게 변경되는 동안 복구 지점 및 로그 파일 복사본을 만들지 않고 이러한 항목에 소비되는 디스크 공간을 알아야 할 수 있습니다(복구 지점 파일 및 로그가 클 수 있음). 따라서 이 절에서는 많은 파일 시스템 구성에서 수정 없이 사용할 수 있는 광범위하게 적용 가능한 구성을 권장합니다. 변경이 필요한 경우 권장되는 구성을 통해 문제를 확인하고 적절하게 작업을 시작할 수 있을 것입니다. 이 절의 나머지 부분에서는 복구 지점을 만들고 관리하기 위한 지침을 제공합니다. 다음 세부 절이 포함됩니다.

- 아카이버 로그의 복구 지점 파일 및 복사본을 저장할 위치 만들기
- 자동으로 복구 지점 만들기 및 아카이버 로그 저장

아카이버 로그의 복구 지점 파일 및 복사본을 저장할 위치 만들기

구성한 각 아카이빙 파일 시스템에 대해 다음과 같이 하십시오.

1. 파일 시스템 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. 복구 지점 파일을 위한 스토리지 위치를 선택합니다. 파일 시스템 호스트에 마운트할 수 있는 독립된 파일 시스템을 선택합니다.

3. 선택한 파일 시스템에 새 복구 지점 파일 및 특정 시점에 보유할 복구 지점 파일 수를 저장할 수 있는 충분한 공간이 있는지 확인합니다.

   복구 지점 파일이 클 수 있으며 복구 지점 파일을 만드는 빈도 및 보유할 기간에 따라 많은 수의 파일을 저장해야 합니다.
4. 선택한 파일 시스템이 물리적 장치를 아카이빙 파일 시스템과 공유하지 않는지 확인합니다.

   보호해야 하는 파일 시스템에 복구 지점 파일을 저장하지 마십시오. 또한 아카이빙 파일 시스템도 호스팅하는 물리적 장치에 상주하는 분할 영역 또는 LUN과 같은 논리적 장치에 복구 지점 파일을 저장하지 마십시오.
5. 선택한 파일 시스템에서 복구 지점 파일을 보유할 디렉토리를 만듭니다. *mkdir mount-point/path* 명령을 사용합니다. 여기서 *mount-point*는 선택한 독립된 파일 시스템의 마운트 지점이고 *path*는 선택한 디렉토리의 경로 및 이름입니다.

   여러 아카이빙 파일 시스템에 대한 복구 지점 파일을 단일 전체 수집 디렉토리에 저장하지 마십시오. 복구 지점 파일이 구성되어 필요 시 쉽게 찾을 수 있도록 각각에 대한 별개의 디렉토리를 만듭니다.

   예제에서는 아카이빙 파일 시스템 */samms*에 대한 복구 지점을 구성합니다. 따라서 독립된 파일 시스템 */zfs1*에 */zfs1/samms_recovery* 디렉토리를 만들었습니다.

```
root@solaris:~# mkdir /zfs1/samms_recovery
```

6. 파일 시스템에서 물리적 장치를 아카이빙 파일 시스템과 공유하지 않는 경우 파일 시스템에 대한 아카이버 로그의 적시 복사본을 저장하기 위한 하위 디렉토리를 만듭니다.

   예제에서는 호스트의 루트 파일 시스템의 */var* 디렉토리에 로그 복사본을 저장합니다. 여기서는 아카이빙 파일 시스템 */samms*에 대한 파일 시스템 보호를 구성하는 중입니다. 따라서 */var/samms_archlogs* 디렉토리를 만듭니다.

```
root@solaris:~# mkdir /var/samms_archlogs
```

7. 이제 복구 지점 만들기 및 아카이버 로그 저장 자동화를 수행합니다.

자동으로 복구 지점 만들기 및 아카이버 로그 저장

crontab 파일에 항목을 만들거나 Oracle HSM Manager 그래픽 사용자 인터페이스의 예약 기능을 사용하여 메타데이터 복구 지점 파일을 자동으로 만들 수 있지만 후자의 방법에서는 아카이버 로그 데이터가 자동으로 저장되지 않습니다. 따라서 이 절에서는 *crontab* 접근 방법에 초점을 맞춥니다. 그래픽 사용자 인터페이스를 사용하여 복구 지점을 예약하려는 경우 Manager 온라인 도움말을 참조하십시오.

아래 절차에서는 매일 실행되는 *crontab* 항목 2개를 만드는데 하나는 오래된 복구 지점 파일을 삭제한 다음 새 복구 지점을 만들고 다른 하나는 아카이버 로그를 저장합니다. 구성한 각 아카이빙 파일 시스템에 대해 다음과 같이 하십시오.

1. 파일 시스템 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. 편집을 위해 *root* 사용자의 *crontab* 파일을 엽니다. *crontab -e* 명령을 사용합니다.

   crontab 명령은 *root* 사용자의 *crontab* 파일의 편집 가능 복사본을 *EDITOR* 환경 변수에 지정된 텍스트 편집기에서 엽니다(자세한 내용은 Solaris *crontab* 매뉴얼 페이지 참조). 이 예에서는 *vi* 편집기를 사용합니다.

```
root@solaris:~# crontab -e
...
# The root crontab should be used to perform accounting data collection.
10 3 * * * /usr/sbin/logadm
15 3 * * 0 [ -x /usr/lib/fs/nfs/nfsfind ] && /usr/lib/fs/nfs/nfsfind
30 3 * * * [ -x /usr/lib/gss/gsscred_clean ] && /usr/lib/gss/gsscred_clean
30 0,9,12,18,21 * * * /usr/lib/update-manager/update-refresh.sh
```

3. 먼저 오래된 복구 지점 파일을 삭제하고 새 복구 지점을 만드는 항목을 만듭니다. 새 라인에서 작업이 완료되는 시간을 지정합니다. *minutes hour * * **을 입력합니다. 설명:
   - *minutes*는 작업이 시작되는 분을 지정하는 [*0-59*] 범위의 정수입니다.
   - *hour*는 작업이 시작되는 시간을 지정하는 [*0-23*] 범위의 정수입니다.
   - *(별표)는 사용되지 않는 값을 지정합니다.

     매일 실행되는 작업의 경우 월의 일 [*1-31*], 월 [*1-12*] 및 요일 [*0-6*] 값이 사용되지 않습니다.
   - 시간 지정에서 필드는 공백으로 구분됩니다.
   - *minutes hour*는 파일을 만들거나 수정하는 중이 아닌 시간을 지정합니다.

     파일 시스템 작업이 최소한으로 수행될 때 복구 지점 파일을 만들면 아카이브 상태가 가능한 정확하고 완벽하게 파일에 반영됩니다. 이상적인 경우라면 모든 새 파일과 변경된 파일이 지정한 시간 이전에 아카이브되어 있습니다.

   예제에서는 매일 오전 2시 10분에 시작하도록 작업을 예약합니다.

```
...
30 3 * * * [ -x /usr/lib/gss/gsscred_clean ] && /usr/lib/gss/gsscred_clean
30 0,9,12,18,21 * * * /usr/lib/update-manager/update-refresh.sh
10 2 * * *
```

4. 계속해서 동일한 라인에서 이전 복구 지점 파일을 지우는 셸 명령을 입력합니다. *(find directory -type f -mtime +retention -print | xargs -l1 rm -f;* 텍스트를 입력합니다. 여기서 각 항목은 다음과 같습니다.
   - *(*(여는 괄호)는 *crontab* 항목이 실행할 명령 시퀀스의 시작을 표시합니다.

- *directory*는 복구 지점 파일이 저장되고 따라서 Solaris *find* 명령이 검색을 시작할 지점에 해당하는 디렉토리의 경로 및 디렉토리 이름입니다.
- *-type f*는 블록 특수 파일, 문자 특수 파일, 디렉토리, 파이프 등이 아니라 일반 파일을 지정하는 *find* 명령 옵션입니다.
- *-mtime +retention*은 복구 지점 파일이 보존되는 시간을 나타내는 정수인 *retention*을 초과하여 수정되지 않은 파일을 지정하는 *find* 명령 옵션입니다.
- *-print*는 발견된 모든 파일을 표준 출력으로 나열하는 *find* 명령 옵션입니다.
- |*xargs -l1 rm -f*는 *-print*의 출력을 Solaris 명령 *xargs -l1*에 파이프하고 이 명령은 발견된 각 파일을 삭제하는 Solaris 명령 *rm -f*에 한 번에 하나씩 라인을 인수로 보냅니다.
- *;*(세미콜론)은 명령줄의 끝을 표시합니다.

예제에서 *crontab* 항목은 */zfs1/samms_recovery* 디렉토리에서 72시간(3일) 넘게 수정되지 않은 파일을 찾아 발견된 파일을 삭제합니다. *crontab* 항목은 여전히 한 라인이며 줄바꿈이 백슬래시로 이스케이프됩니다.

```
# The root crontab should be used to perform accounting data collection.
10 3 * * * /usr/sbin/logadm
15 3 * * 0 [ -x /usr/lib/fs/nfs/nfsfind ] && /usr/lib/fs/nfs/nfsfind
30 3 * * * [ -x /usr/lib/gss/gsscred_clean ] && /usr/lib/gss/gsscred_clean
30 0,9,12,18,21 * * * /usr/lib/update-manager/update-refresh.sh
10 2 * * * ( find /zfs1/samms_recovery -type f -mtime +72 -print | /
xargs -l1 rm -f;
```

5. 계속해서 동일한 라인에서 복구 지점이 만들어질 디렉토리로 변경하는 셸 명령을 입력합니다. *cd mount-point;* 텍스트를 입력합니다. 여기서 *mount-point*는 아카이빙 파일 시스템의 루트 디렉토리이고 세미콜론(;)은 명령줄의 끝을 표시합니다.

   복구 지점 파일을 만드는 *samfsdump* 명령은 현재 디렉토리 및 모든 하위 디렉토리의 모든 파일에 대한 메타데이터를 백업합니다. 예제에서는 보호 중인 파일 시스템의 마운트 지점인 */samms* 디렉토리로 변경합니다.

```
# The root crontab should be used to perform accounting data collection.
10 3 * * * /usr/sbin/logadm
15 3 * * 0 [ -x /usr/lib/fs/nfs/nfsfind ] && /usr/lib/fs/nfs/nfsfind
30 3 * * * [ -x /usr/lib/gss/gsscred_clean ] && /usr/lib/gss/gsscred_clean
30 0,9,12,18,21 * * * /usr/lib/update-manager/update-refresh.sh
10 2 * * * ( find /zfs1/samms_recovery -type f -mtime +72 -print | /
xargs -l1 rm -f; cd /samms;
```

6. 계속해서 동일한 라인에서 매일 새 복구 지점을 만드는 셸 명령을 입력합니다. */opt/SUNWsamfs/sbin/samfsdump -f directory/'date +/%y/%m/%d')* 텍스트를 입력합니다. 여기서 각 항목은 다음과 같습니다.

- */opt/SUNWsamfs/sbin/samfsdump*는 복구 지점을 만드는 명령입니다(자세한 내용은 매뉴얼 페이지 참조).
- *-f*는 복구 지점 파일이 저장되는 위치를 지정하는 *samfsdump* 명령 옵션입니다.
- *directory*는 이 파일 시스템의 복구 지점을 보관하기 위해 만든 디렉토리입니다.
- *'date +/%y/%m/%d'*는 Solaris *date* 명령 및 복구 지점 파일의 이름 *YYMMDD*를 만드는 형식 지정 템플리트입니다. 여기서 *YYMMDD*는 현재 연도의 마지막 2자리 숫자, 현재 월의 2자리 숫자, 월의 일에 해당하는 2자리 숫자입니다(예를 들어, 2015년 1월 22일의 경우 *150122*).
- ; (세미콜론)은 명령줄의 끝을 표시합니다.
- *)*(닫는 괄호)는 *crontab* 항목이 실행할 명령 시퀀스의 끝을 표시합니다.

예제에서는 위에서 만든 복구 지점 디렉토리 */zfs1/samms_recovery*를 지정합니다. *crontab* 항목은 여전히 한 라인이며 줄바꿈이 백슬래시로 이스케이프됩니다.

```
# The root crontab should be used to perform accounting data collection.
10 3 * * * /usr/sbin/logadm
15 3 * * 0 [ -x /usr/lib/fs/nfs/nfsfind ] && /usr/lib/fs/nfs/nfsfind
30 3 * * * [ -x /usr/lib/gss/gsscred_clean ] && /usr/lib/gss/gsscred_clean
30 0,9,12,18,21 * * * /usr/lib/update-manager/update-refresh.sh
10 2 * * * ( find /zfs1/samms_recovery -type f -mtime +72 -print | /
xargs -l1 rm -f; cd /samms ; /opt/SUNWsamfs/sbin/samfsdump /
-f /zfs1/samms_recovery/'date +/%y/%m/%d')
```

7. 이제 아카이버 로그를 저장하는 항목을 만듭니다. 새 라인에 *minutes hour * * **를 입력하여 작업을 수행할 시간을 지정합니다. 여기서 각 항목은 다음과 같습니다.
   - *minutes*는 작업이 시작되는 분을 지정하는 [*0-59*] 범위의 정수입니다.
   - *hour*는 작업이 시작되는 시간을 지정하는 [*0-23*] 범위의 정수입니다.
   - ***(별표)는 사용되지 않는 값을 지정합니다.

     매일 실행되는 작업의 경우 월의 일 [*1-31*], 월 [*1-12*] 및 요일 [*0-6*] 값이 사용되지 않습니다.
   - 시간 지정에서 필드는 공백으로 구분됩니다.
   - *minutes hour*는 파일을 만들거나 수정하는 중이 아닌 시간을 지정합니다.

   예제에서는 매주 일요일 오전 3시 15분에 시작하도록 작업을 예약합니다.

```
...
30 0,9,12,18,21 * * * /usr/lib/update-manager/update-refresh.sh
10 2 * * * ( find /zfs1/samms_recovery -type f -mtime +72 -print | /
xargs -l1 rm -f; cd /samms ; /opt/SUNWsamfs/sbin/samfsdump /
-f /zfs1/samms_recovery/'date +/%y/%m/%d')
15 3 * * 0
```

8. 계속해서 동일한 라인에서 현재 아카이버 로그를 백업 위치로 이동하고 고유한 이름을 지정하는 셸 명령을 입력합니다. *(mv /var/adm/samms.archive.log /var/samms_archlogs/"date +%y%m%d";* 텍스트를 입력합니다.

   이 단계에서는 활성 로그 파일에 남겨 둘 경우 덮어쓰여지는 로그 항목을 저장합니다. 예제에서는 *samms* 파일 시스템에 대한 아카이버 로그를 선택한 위치 */var/samms_archlogs/*로 이동하고 이름을 *YYMMDD*로 지정합니다. 여기서 *YYMMDD*는 현재 연도의 마지막 2자리 숫자, 현재 월의 2자리 숫자, 월의 일에 해당하는 2자리 숫자입니다(예를 들어, 2015년 1월 22일의 경우 *150122*).

```
...
30 0,9,12,18,21 * * * /usr/lib/update-manager/update-refresh.sh
10 2 * * * ( find /samms_recovery/dumps -type f -mtime +72 -print | xargs -l1 rm -f; / cd /samms ; /
opt/SUNWsamfs/sbin/samfsdump -f /zfs1/samms_recovery/'date +/%y/%m/%d')
15 3 * * 0 ( mv /var/adm/samms.archiver.log /var/samms_archlogs/"date +%y%m%d";
```

9. 계속해서 동일한 라인에서 아카이버 로그 파일을 다시 초기화하는 셸 명령을 입력합니다. *touch /var/adm/samms.archive.log)* 텍스트를 입력합니다.

   예제에서 *crontab* 항목은 여전히 한 라인이며 줄바꿈이 백슬래시로 이스케이프됩니다.

```
...
30 0,9,12,18,21 * * * /usr/lib/update-manager/update-refresh.sh
10 2 * * * ( find /samms_recovery/dumps -type f -mtime +72 -print | xargs -l1 rm -f; / cd /samms ; /
opt/SUNWsamfs/sbin/samfsdump -f /zfs1/samms_recovery/'date +/%y/%m/%d')
15 3 * * 0 ( mv /var/adm/samms.archive.log /var/samms_archlogs/"date +%y%m%d";/
touch /var/adm/samms.archiver.log )
```

10. 파일을 저장하고 편집기를 닫습니다.

```
# The root crontab should be used to perform accounting data collection.
10 3 * * * /usr/sbin/logadm
15 3 * * 0 [ -x /usr/lib/fs/nfs/nfsfind ] && /usr/lib/fs/nfs/nfsfind
30 3 * * * [ -x /usr/lib/gss/gsscred_clean ] && /usr/lib/gss/gsscred_clean
30 0,9,12,18,21 * * * /usr/lib/update-manager/update-refresh.sh
10 2 * * * ( find /samms_recovery/dumps -type f -mtime +72 -print | xargs -l1 rm -f; / cd /samms ; /
opt/SUNWsamfs/sbin/samfsdump -f /zfs1/samms_recovery/'date +/%y/%m/%d')
15 3 * * 0 ( mv /var/adm/samms.archive.log /var/samms_archlogs/"date +%y%m%d";/ touch /var/adm/samms
.archive.log )
:wq
root@solaris:~#
```

11. 파일 시스템에서 WORM(Write Once Read Many) 기능을 사용으로 설정해야 할 경우 "WORM(Write Once Read Many) 파일 지원을 사용으로 설정"을 참조하십시오.

12. LTFS를 사용하는 시스템과 상호 작용해야 하거나 원격 사이트 간에 많은 양의 데이터를 전송해야 할 경우 “LTFS(Linear Tape File System)에 대한 지원을 사용으로 설정”을 참조하십시오.
13. 아카이브 테이프 볼륨의 데이터 무결성을 확인할 수 있어야 하는 경우 “아카이브 매체 검증 구성”으로 이동합니다.
14. 다중 호스트 파일 시스템 액세스 또는 고가용성 구성과 같은 추가 요구 사항이 있는 경우 “기본 과정 이후”를 참조하십시오.
15. 그렇지 않은 경우 11장. 알림 및 로깅 구성으로 이동합니다.

아카이브 매체 검증 구성

매체 검증은 SCSI *verify* 명령을 사용하여 테이프 매체의 데이터 무결성을 평가하는 기술입니다. 호스트의 SCSI 드라이버는 드라이브에 기록하는 데이터의 논리적 블록에 대한 CRC 체크섬을 계산하고 *verify* 명령을 보냅니다. 드라이브는 데이터 블록을 읽고 고유한 체크섬을 계산한 다음 결과를 드라이브에 의해 제공된 값과 비교합니다. 불일치가 있는 경우 오류를 반환합니다. 드라이브는 체크섬이 완료되자마자 자신이 읽은 데이터를 삭제하므로 호스트에 추가 I/O 관련 오버헤드가 없습니다.

Oracle HSM는 다음과 같은 두 가지 방법으로 매체 검증을 지원합니다.

- DIV(데이터 무결성 검증)를 지원하도록 Oracle HSM 구성을 통해 StorageTek T10000 테이프 매체의 데이터에 대한 검증을 수동으로 수행하거나 Oracle HSM 주기적 매체 확인에서 자동으로 수행할 수 있습니다.
- 또한 Oracle HSM 주기적 매체 확인 구성을 통해 StorageTek T10000 테이프 매체 및 다른 형식(예: LTO Ultrium) 모두에서 데이터를 자동으로 검증할 수 있습니다.

DIV(데이터 무결성 검증)를 지원하도록 Oracle HSM 구성

DIV(데이터 무결성 검증)는 Oracle HSM 소프트웨어와 함께 작동하여 저장된 데이터의 무결성을 보장하는 Oracle StorageTek 테이프 드라이브의 기능입니다. 이 기능이 사용으로 설정된 경우(*div = on* 또는 *div = verify*) 서버 호스트 및 드라이브 모두는 I/O 도중 체크섬을 계산 및 비교합니다. 쓰기 작업 도중 서버는 각 데이터 블록에 대해 4바이트 체크섬을 계산하고 체크섬을 데이터와 함께 드라이브에 전달합니다. 그런 다음 테이프 드라이브는 체크섬을 재계산하고 결과를 서버에 의해 제공된 값과 비교합니다. 값이 일치할 경우 드라이브는 데이터 블록 및 체크섬 둘 다를 테이프에 기록합니다. 읽기 작업 도중 드라이브 및 호스트 둘 다는 데이터 블록 및 연관된 해당 체크섬을 테이프에서 읽습니다. 각각은 데이터 블록에서 체크섬을 재계산하고 결과를 저장된 체크섬과 비교합니다. 특정 시점에 체크섬이 일치하지 않을 경우 드라이브는 오류가 발생했다는 것을 응용 프로그램 소프트웨어에 알립니다.

div = verify 옵션은 데이터를 기록할 때 추가적 보호 계층을 제공합니다. 쓰기 작업이 완료된 경우 호스트는 데이터를 재확인할 것을 테이프 드라이브에 요청합니다. 그런 다음 드라이브는 데이터를 다시 스캔하고 체크섬을 재계산한 다음 결과를 테이프에 저장된 체크섬과 비교합니다. 드라이브는 추가 I/O 없이(데이터가 삭제됨) 모든 작업을 내부적으로 수행하므로 호스트 시스템에 추가 오버헤드가 없습니다. 또한 Oracle HSM *tpverify*(테이프 확인) 명령을 사용하여 이 단계를 요구 시 수행할 수 있습니다.

데이터 무결성 검증을 구성하려면 다음과 같이 하십시오.

1. Oracle HSM 서버에 *root*로 로그인합니다.

   예제에서는 메타데이터 서버의 이름이 *samfs-mds*로 지정됩니다.

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

2. 메타데이터 서버가 Oracle Solaris 11 이상을 실행 중인지 확인합니다.

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

3. Oracle HSM *mcf* 파일에 정의된 아카이브 스토리지 장비에 호환되는 테이프 드라이브인 StorageTek T10000C(최소 펌웨어 레벨 1.53.315) 또는 T10000D가 포함되어 있는지 확인합니다.
4. 모든 아카이빙 프로세스를 유휴 설정합니다(있는 경우). *samcmd aridle* 명령을 사용합니다.

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

```
[samfs-mds]root@solaris:~# samcmd aridle
[samfs-mds]root@solaris:~#
```

5. 모든 스테이징 프로세스를 유휴 설정합니다(있는 경우). *samcmd stidle* 명령을 사용합니다.

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

```
[samfs-mds]root@solaris:~# samcmd stidle
[samfs-mds]root@solaris:~#
```

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

   아카이빙 프로세스가 *Waiting for :arrun*이면 아카이빙 프로세스가 유휴 상태임을 나타냅니다.

```
[samfs-mds]root@solaris:~# samcmd a
Archiver status samcmd     6.0 14:20:34 Feb 22 2015
samcmd on samfs-mds
sam-archiverd:  Waiting for :arrun
sam-arfind: ...
```

```
Waiting for :arrun
```

7. 모든 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.

```
[samfs-mds]root@solaris:~# samcmd u
Staging queue samcmd      6.0 14:20:34 Feb 22 2015
samcmd on solaris.demo.lan
Staging queue by media type: all
sam-stagerd:  Waiting for :strun
root@solaris:~#
```

8. 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.

```
[samfs-mds]root@solaris:~# samcmd 801 idle
[samfs-mds]root@solaris:~# samcmd 802 idle
[samfs-mds]root@solaris:~# samcmd 803 idle
[samfs-mds]root@solaris:~# samcmd 804 idle
[samfs-mds]root@solaris:~#
```

9. 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.

```
[samfs-mds]root@solaris:~# samcmd r
Removable media samcmd     6.0 14:20:34 Feb 22 2015
samcmd on samqfs1host
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
[samfs-mds]root@solaris:~#
```

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

```
[samfs-mds]root@solaris:~# samd stop
[samfs-mds]root@solaris:~#
```

11. 텍스트 편집기에서 */etc/opt/SUNWsamfs/defaults.conf* 파일을 엽니다. 필요한 경우 *#div = off* 라인의 주석 처리를 해제하거나 없는 경우 이 라인을 추가합니다.

    기본적으로 *div*(데이터 무결성 검증)는 *off*(사용 안함)입니다.

    예제에서는 *vi* 편집기에서 파일을 열고 행 주석 처리를 해제합니다.

```
[samfs-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.
...
div = off
```

12. 데이터 무결성 검증 읽기, 쓰기 및 확인 작업을 사용으로 설정하려면 *#div = off* 라인을 *div = on*으로 변경하고 파일을 저장합니다.

    각 블록을 읽고 쓸 때 데이터가 확인되지만 Oracle HSM 아카이버 소프트웨어는 아카이브된 후의 완료된 파일 복사본을 확인하지 않습니다.

```
[samfs-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.
...
div = on
:wq
[samfs-mds]root@solaris:~#
```

13. 데이터 무결성 검증 기능의 쓰기 후 확인 옵션을 사용으로 설정하려면 *#div = off* 라인을 *div = verify*로 변경하고 파일을 저장합니다.

    각 블록을 읽거나 쓸 때 호스트와 드라이브는 데이터 무결성 검증을 수행합니다. 또한 완료된 아카이브 요청이 테이프에 기록될 때마다 드라이브는 새로 저장된 데이터 및 체크섬을 다시 읽고 재계산한 다음 저장된 결과와 계산된 결과를 비교합니다.

```
[samfs-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.
...
div = verify
:wq
[samfs-mds]root@solaris:~#
```

14. Oracle HSM 소프트웨어에 *defaults.conf* 파일을 다시 읽고 그에 따라 재구성하도록 지시합니다. *samd config* 명령을 사용합니다.

```
[samfs-mds]root@solaris:~# /opt/SUNWsamfs/sbin/samd config
```

15. 이전 단계에서 Oracle HSM 작업을 중지한 경우 이제 *samd start* 명령을 사용하여 다시 시작합니다.

```
[samfs-mds]root@solaris:~# samd start
[samfs-mds]root@solaris:~#
```

이제 데이터 무결성 검증이 구성되었습니다.

16. 데이터 무결성 검증을 자동화해야 할 경우 “Oracle HSM 주기적 매체 확인 구성”으로 이동합니다.
17. 파일 시스템에서 WORM(Write Once Read Many) 기능을 사용으로 설정해야 할 경우 “WORM(Write Once Read Many) 파일 지원을 사용으로 설정”을 참조하십시오.
18. LTFS를 사용하는 시스템과 상호 작용해야 하거나 원격 사이트 간에 많은 양의 데이터를 전송해야 할 경우 “LTFS(Linear Tape File System)에 대한 지원을 사용으로 설정”을 참조하십시오.
19. 다중 호스트 파일 시스템 액세스 또는 고가용성 구성과 같은 추가 요구 사항이 있는 경우 “기본 과정 이후”를 참조하십시오.

Oracle HSM 주기적 매체 확인 구성

Oracle HSM 아카이빙 파일 시스템을 위한 PMV(주기적 매체 확인)를 설정할 수 있습니다. 주기적 매체 확인은 파일 시스템에서 이동식 매체의 데이터 무결성을 자동으로 검사합니다. StorageTek T10000 매체는 StorageTek 데이터 무결성 검증을 사용하여 검사되고 다른 드라이브는 널리 지원되는 SCSI *verify(6)* 명령을 사용하여 검사됩니다.

주기적 매체 확인 기능은 *tpverify* 명령을 주기적으로 적용하여 감지된 모든 오류를 기록하고 관리자에게 알리며 지정된 복구 작업을 자동으로 수행하는 Oracle HSM 데몬인 *verifyd*를 추가합니다. 구성 파일 *verifyd.cmd*에서 정책 지시어를 설정하여 주기적 매체 확인을 구성합니다. 정책은 확인 스캔이 실행되는 시점, 수행되는 스캔의 유형, 사용할 수 있는 라이브러리 및 드라이브, 스캔해야 하는 테이프 볼륨, 오류가 감지된 경우 Oracle HSM에

서 수행하는 작업 등을 지정할 수 있습니다. 예를 들어, Oracle HSM은 오류가 포함된 파일을 자동으로 다시 아카이브하고 오류가 포함된 테이프 볼륨을 재활용할 수 있습니다.

1. Oracle HSM 서버에 *root*로 로그인합니다.

   예제에서는 메타데이터 서버의 이름이 *samfs-mds*로 지정됩니다.

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

2. 아직 수행하지 않은 경우 계속하기 전에 DIV(데이터 무결성 검증)를 지원하도록 Oracle HSM 구성을 수행합니다.
3. 메타데이터 서버가 Oracle Solaris 11 이상을 실행 중인지 확인합니다.

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

4. 텍스트 편집기에서 */etc/opt/SUNWsamfs/verifyd.cmd* 파일을 엽니다.

   예제에서는 *vi* 편집기에서 파일을 엽니다.

```
[samfs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/verifyd.cmd
# For additional information about the format of the verifyd.cmd file,
# type "man verifyd.cmd".
# Enable Oracle HSM Periodic Media Validation (PMV)
pmv = off
```

5. 주기적 매체 확인을 사용으로 설정하려면 *pmv = on* 라인을 입력합니다.

   기본적으로 주기적 매체 확인은 *off*입니다. 예제에서는 *on*으로 설정합니다.

```
[samfs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/verifyd.cmd
# For additional information about the format of the verifyd.cmd file,
# type "man verifyd.cmd".
# Enable Oracle HSM Periodic Media Validation (PMV)
pmv = on
```

6. 실행 시간을 설정합니다. *run_time = always* 라인을 입력하여 확인을 연속적으로 실행하거나 *run_time = HHMM hhmm DD dd*를 입력합니다. 여기서 *HHMM* 및 *hhmm*은 각각 시작 및 종료 시간이고 *DD dd*는 선택적 시작 및 종료 날짜입니다.

   HH 및 *hh*는 *00-24* 범위의 시간이고 *MM* 및 *mm*은 *00-60* 범위의 분 수이며 *DD* 및 *dd*는 *[0-6]* 범위의 요일입니다(*0*은 일요일이고 *6*은 토요일). 기본값은 *2200 0500 6 0*입니다.

그러나 확인 프로세스는 직접적으로 더 중요한 파일 시스템 작업과 경쟁하지 않습니다. 확인 프로세스는 아카이버 및 스테이저에 필요한 테이프 볼륨 및/또는 드라이브를 자동으로 양보합니다. 따라서 예제에서는 실행 시간을 *always*로 설정합니다.

```
[samfs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/verifyd.cmd
# For additional information about the format of the verifyd.cmd file,
# type "man verifyd.cmd".
# Enable Oracle HSM Periodic Media Validation (PMV)
pmv = on
# Run all of the time.  PMV will yield VSNs and drives when
# resources are wanted by the SAM-QFS archiver and stager.
run_time = always
```

7. 확인 방법을 지정합니다. *pmv_method = specified-method* 라인을 입력합니다. 여기서 *specified-method*는 다음 중 하나입니다.
   - *standard* 방법은 특히 Oracle StorageTek T10000C 이상 버전의 테이프 드라이브에 사용됩니다. *standard* 방법은 속도에 최적화된 방식으로 매체의 모서리, 시작, 끝과 처음 1,000개 블록을 확인합니다.
   - *complete* 방법도 Oracle StorageTek T10000C 이상 버전의 테이프 드라이브에 사용됩니다. 이 방법은 매체의 모든 블록에 대한 매체 ECC(오류 수정 코드)를 확인합니다.
   - *complete plus*도 Oracle StorageTek T10000C 이상 버전의 테이프 드라이브에 사용됩니다. 이 방법은 매체의 각 블록에 대한 매체 ECC(오류 수정 코드) 및 데이터 무결성 검증 체크섬을 둘 다 확인합니다(“DIV(데이터 무결성 검증)를 지원하도록 Oracle HSM 구성” 참조).
   - *legacy* 방법은 다른 모든 테이프 드라이브에 사용할 수 있으며 매체가 카탈로그에서 불량으로 표시되거나 드라이브가 *verifyd.cmd* 파일에 지정된 방법을 지원하지 않을 경우 자동으로 사용됩니다. 이 방법은 6바이트 고정 블록 모드 SCSI 확인 명령을 실행하고 이전에 기록된 결함을 건너뜁니다. 새 영구 매체 오류가 발견될 경우 *legacy* 방법은 다음 파일로 건너뛰고 새로 검색된 오류를 매체 결함 데이터베이스에 기록합니다.
   - *mir rebuild* 방법은 MIR(매체 정보 영역)이 누락되거나 손상된 경우 Oracle StorageTek 테이프 카트리지의 MIR을 재작성합니다. 이 방법은 매체 카탈로그에서 불량으로 표시된 매체와 함께 작동하며 MIR 손상이 감지된 경우 자동으로 지정됩니다.

   예제에서는 LTO 드라이브를 사용하므로 *legacy*를 지정합니다.

```
[samfs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/verifyd.cmd
...
# resources are wanted by the SAM-QFS archiver and stager.
run_time = always
pmv_method = legacy
```

8. 사용 가능한 모든 라이브러리와 드라이브를 확인에 사용하려면 *pmv_scan = all* 라인을 입력합니다.

```
[samfs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/verifyd.cmd
...
pmv_method = legacy
pmv_scan = all
```

9. 지정된 라이브러리의 사용 가능한 모든 드라이브를 확인에 사용하려면 *pmv _scan = library equipment-number* 라인을 입력합니다. 여기서 *equipment-number*는 파일 시스템의 *mcf* 파일에 있는 라이브러리에 지정된 장비 번호입니다.

   예제에서는 *800* 라이브러리의 모든 드라이브가 확인 프로세스에 사용되도록 합니다.

```
[samfs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/verifyd.cmd
...
pmv_method = legacy
pmv_scan = library 800
```

10. 지정된 라이브러리에서 확인 프로세스에 사용될 수 있는 드라이브 수를 제한하려면 *pmv _scan = library equipment-number max_drives number* 라인을 입력합니다. 여기서 *equipment-number*는 파일 시스템의 *mcf* 파일에 있는 라이브러리에 지정된 장비 번호이고 number는 사용할 수 있는 최대 드라이브 수입니다.

    예제에서는 *800* 라이브러리에 있는 최대 *2*개의 드라이브가 확인 프로세스에 사용되도록 합니다.

```
[samfs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/verifyd.cmd
...
pmv_method = legacy
pmv_scan = library 800 max_drives 2
```

11. 지정된 라이브러리에서 확인 프로세스에 사용될 수 있는 드라이브를 지정하려면 *pmv _scan = library equipment-number drive drive-numbers* 라인을 입력합니다. 여기서 *equipment-number*는 파일 시스템의 *mcf* 파일에 있는 라이브러리에 지정된 장비 번호이고 *drive-numbers*는 *mcf* 파일에서 지정된 드라이브에 지정된 공백으로 구분된 장비 번호 목록입니다.

    예제에서는 *900* 라이브러리에 있는 *903* 및 *904* 드라이브가 확인 프로세스에 사용되도록 합니다.

```
[samfs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/verifyd.cmd
...
pmv_method = legacy
```

```
pmv_scan = library 900 drive 903 904
```

12. 둘 이상의 라이브러리에서 확인 프로세스에 사용될 수 있는 드라이브를 지정하려면 *pmv_scan = library-specification library-specification...* 라인을 입력합니다. 여기서 *equipment-number*는 파일 시스템의 *mcf* 파일에 있는 라이브러리에 지정된 장비 번호이고 *drive-numbers*는 *mcf* 파일에 있는 라이브러리에 지정된 공백으로 구분된 장비 번호 목록입니다.

    예제에서는 *800* 라이브러리에 있는 최대 *2*개의 드라이브와 *900* 라이브러리에 있는 *903* 및 *904* 드라이브가 확인 프로세스에 사용되도록 합니다.

```
[samfs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/verifyd.cmd
...
pmv_method = legacy
pmv_scan = library 800 max_drives 2 library 900 drive 903 904
```

13. 주기적 매체 확인을 사용 안함으로 설정하고 어떠한 장비도 사용하지 않게 하려면 *pmv_scan = off* 라인을 입력합니다.

```
[samfs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/verifyd.cmd
...
pmv_method = legacy
pmv_scan = off
```

14. 주기적 매체 확인에서 지정된 수의 영구 오류가 감지되고 나서 매체가 재활용 대상이라는 플래그를 자동으로 지정하려면 *action = recycle perms number-errors* 라인을 입력합니다. 여기서 *number-errors*는 오류 수입니다.

    예제에서는 *10*개의 오류가 감지된 후 매체가 재활용 대상이라는 플래그를 지정하도록 Oracle HSM를 구성합니다.

```
[samfs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/verifyd.cmd
...
pmv_scan = all
action = recycle perms 10
```

15. 지정된 기간 동안 오류가 누적된 후 불량 블록을 포함하는 파일을 자동으로 다시 아카이브하려면 *action = rearch age time* 라인을 입력합니다. 여기서 *time*은 임의로 조합한 *SECONDSs*, *MINUTESm*, *HOURSh*, *DAYSd* 및/또는 *YEARSy*의 공백으로 구분된 목록이고 *SECONDS*, *MINUTES*, *HOURS*, *DAYS* 및 *YEARS*는 정수입니다.

    파일 시스템에서 아카이빙이 필요한 파일이 스캔되기 전에 가장 오래된 매체 결함이 지정된 기간을 경과했어야 합니다. 예제에서는 다시 아카이빙 수명을 *1*분으로 설정합니다.

```
[samfs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/verifyd.cmd
```

```
...
pmv_scan = all
action = rearch age 1m
```

16. 주기적 매체 확인에서 영구 매체 오류를 감지한 경우 매체를 불량으로 표시하고 그렇지 않은 경우 아무 조치도 수행하지 않으려면 *action = none* 라인을 입력합니다.

```
[samfs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/verifyd.cmd
...
pmv_scan = all
action = none
```

17. 주기적으로 확인해야 하는 테이프 볼륨을 지정합니다. *pmv_vsns = selection-criterion* 라인을 입력합니다. 여기서 *selection-criterion*은 *all* 또는 하나 이상의 VSN(볼륨 일련 번호)을 지정하는 공백으로 구분된 정규 표현식 목록입니다.

    기본값은 *all*입니다. 예제에서는 정규 표현식 3개가 제공되는데 *^VOL0[01][0-9]* 및 *^VOL23[0-9]*는 각각 *VOL000* - *VOL019* 및 *VOL230* - *VOL239* 범위의 볼륨 일련 번호를 가진 두 세트의 볼륨을 지정하고 *VOL400*은 해당 특정 볼륨 일련 번호를 가진 볼륨을 지정합니다.

```
[samfs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/verifyd.cmd
...
pmv_scan = all
action = none
pmv_vsns = ^VOL0[01][0-9] ^VOL23[0-9] VOL400
```

    Oracle HSM은 감사해야 하는지 여부, 재활용이 예약되어 있는지 여부, 사용할 수 없는지 여부, 외래(비Oracle HSM) 볼륨인지 여부 또는 데이터가 포함되지 않았는지 여부를 볼륨에서 확인하지 않습니다. 청소 카트리지, 레이블이 지정되지 않은 볼륨 및 중복된 볼륨 일련 번호를 가진 볼륨도 제외됩니다.

18. 원하는 확인 정책을 정의합니다. *pmv_policy = verified age vertime [modified age modtime] [mounted age mnttime]* 라인을 입력합니다. 설명:
    - *verified age*는 볼륨이 마지막으로 확인된 이후로 경과해야 하는 최소 시간을 지정합니다.
    - *modified age*(선택사항)는 볼륨이 마지막으로 수정된 이후로 경과해야 하는 최소 시간을 지정합니다.
    - *mounted age*(선택사항)는 볼륨이 마지막으로 마운트된 이후로 경과해야 하는 최소 시간을 지정합니다.
    - 매개변수 값 *vertime*, *modtime* 및 *mnttime*은 음수가 아닌 정수와 시간 단위 *y*(년), *m*(월), *d*(일), *H*(시), *M*(분), *S*(초)의 조합입니다.

Oracle HSM는 볼륨이 마지막으로 확인되고 선택적으로 수정 및/또는 마운트된 이후에 경과한 시간에 기초하여 확인 후보를 식별하고 순위를 지정합니다. 기본 정책은 단일 매개변수 *verified age 6m*(6개월)입니다. 예제에서는 verified age를 3개월로 설정하고 modified age를 15개월로 설정합니다.

```
[samfs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/verifyd.cmd
...
pmv_scan = all
action = none
pmv_vsns = ^VOL0[01][0-9] ^VOL23[0-9] VOL400
pmv_policy = verified age 3m modified age 15m
```

19. */etc/opt/SUNWsamfs/verifyd.cmd* 파일을 저장하고 편집기를 닫습니다.

```
[samfs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/verifyd.cmd
...
pmv_vsns = ^VOL0[01][0-9] ^VOL23[0-9] VOL400
pmv_policy = verified age 3m modified age 15m
:wq
root@solaris:~#
```

20. *tpverify -x* 명령을 입력하여 *verifyd.cmd* 파일에 오류가 있는지 확인합니다. 발견된 모든 오류를 수정합니다.

    tpverify -x 명령은 *verifyd.cmd*를 읽고 오류를 발견할 경우 중지됩니다.

```
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: legacy
STA Scan: off
Action: none
PMV VSNs: all
PMV Policy:
     Last Verified Age: 6m
root@solaris:~#
```

21. 새로운 *verifyd.cmd* 파일을 사용하여 확인 서비스를 다시 시작합니다. *tpverify -r* 명령을 입력합니다.

```
root@solaris:~# tpverify -r
root@solaris:~#
```

주기적 매체 확인의 구성이 끝났습니다.

22. 파일 시스템에서 WORM(Write Once Read Many) 기능을 사용으로 설정해야 할 경우 "WORM(Write Once Read Many) 파일 지원을 사용으로 설정"을 참조하십시오.
23. LTFS를 사용하는 시스템과 상호 작용해야 하거나 원격 사이트 간에 많은 양의 데이터를 전송해야 할 경우 "LTFS(Linear Tape File System)에 대한 지원을 사용으로 설정"을 참조하십시오.
24. 다중 호스트 파일 시스템 액세스 또는 고가용성 구성과 같은 추가 요구 사항이 있는 경우 "기본 과정 이후"를 참조하십시오.
25. 그렇지 않은 경우 11장. 알림 및 로깅 구성으로 이동합니다.

WORM(Write Once Read Many) 파일 지원을 사용으로 설정

WORM(Write Once Read Many) 파일은 법률상 아카이브 목적으로 많은 응용 프로그램에서 사용됩니다. WORM 가능 Oracle HSM 파일 시스템은 기본/사용자 정의 가능 파일 보존 기간, 데이터 및 경로 불변성, 하위 디렉토리의 WORM 설정 상속을 지원합니다. 두 가지 WORM 모드 중 하나를 사용할 수 있습니다.

- 표준 준수 모드(기본값)

  표준 WORM 모드는 사용자가 디렉토리나 비실행 파일에 UNIX *setuid* 권한을 설정할 때 WORM 보존 기간을 시작합니다(*chmod 4000 directory|file*). *setuid* (*set user ID upon execution*) 권한을 실행 파일에 설정하면 보안 위험에 노출되므로 이 모드를 사용하여 UNIX 실행 권한을 가진 파일을 보관할 수 없습니다.
- 에뮬레이션 모드

  WORM 에뮬레이션 모드는 사용자가 쓰기 가능 파일이나 디렉토리를 읽기 전용으로 만들 때 WORM 보존 기간을 시작하므로(*chmod 444 directory|file*) 실행 파일을 보관할 수 있습니다.

표준 및 에뮬레이션 모드는 엄격한 WORM 구현과 *root* 사용자의 제한을 완화하는 덜 제한적인 라이트 구현을 지원합니다. 엄격한 구현과 라이트 구현 모두 파일이나 디렉토리에 보존 기간이 트리거된 이후 데이터나 경로 변경을 허용하지 않습니다. 엄격한 구현에서는 누구도 지정된 보존 기간(기본값 43,200분/30일)을 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 법적 규정 준수 요구 사항을 충족하기에 매우 적합합니다. 라이트 구현에서는 *root* 사용자가 보존 기간을 단축하고, 파일 및 디렉토리를 삭제하고, 파일 시스템 만들기 명령 *sammkfs*를 사용하여 볼륨을 삭제할 수 있습니다. 따라서 데이터 무결성 및 유연한 관리가 주요 요구 사항인 경우 라이트 구현이 더 나은 선택일 수 있습니다.

WORM 구현을 선택하고 파일에서 보존을 사용으로 설정할 때 주의합니다. 일반적으로 요구 사항과 일치하는 가장 덜 제한적인 옵션을 사용합니다. 표준 모드에서 에뮬레이션 모드로 변경하거나 그 반대로 변경할 수 없으므로 신중하게 선택해야 합니다. 관리 유연성이 우선시되거나 보존 요구 사항이 이후에 변경될 수 있는 경우 라이트 구현을 선택합니다. 나중에 필요한 경우 라이트 버전의 WORM 모드에서 엄격한 버전으로 업그레이드할 수 있습니다. 그러나 엄격한 구현을 라이트 구현으로 변경할 수는 없습니다. 엄격한 WORM 구현이 적용되고 나면 지정된 전체 보존 기간 동안 파일을 보존해야 합니다. 따라서 요구 사항과 일치하는 가장 짧은 값으로 보존을 설정합니다.

Oracle HSM 파일 시스템에서 WORM 지원을 사용으로 설정

마운트 옵션을 사용하여 파일 시스템에서 WORM 지원을 사용으로 설정할 수 있습니다. 다음과 같이 하십시오.

1. *root*로 로그인합니다.

```
root@solaris:~#
```

2. 운영체제의 */etc/vfstab* 파일을 백업합니다.

```
root@solaris:~# cp /etc/vfstab /etc/vfstab.backup
```

3. 텍스트 편집기에서 */etc/vfstab* 파일을 열고 WORM 지원을 사용으로 설정하려는 Oracle HSM 파일 시스템에 대한 항목을 찾습니다.

   예제에서는 */etc/vfstab* 파일을 *vi* 편집기에서 열고 아카이빙 파일 시스템 *worm1*을 찾습니다.

```
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      -
...
worm1     -       /worm1   samfs  -    yes     -
```

4. 표준 WORM 준수 모드의 엄격한 구현을 사용으로 설정하려면 *worm_capable* 옵션을 *vfstab* 파일의 *Mount Options* 열에 입력합니다.

```
#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      -
...
worm1     -       /worm1   samfs  -    yes     worm_capable
```

5. 표준 WORM 준수 모드의 라이트 구현을 사용으로 설정하려면 *worm_lite* 옵션을 *vfstab* 파일의 *Mount Options* 열에 입력합니다.

```
#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      -
...
worm1     -       /worm1   samfs  -    yes     worm_lite
```

6. WORM 에뮬레이션 모드의 엄격한 구현을 사용으로 설정하려면 *worm_emul* 옵션을 *vfstab* 파일의 *Mount Options* 열에 입력합니다.

```
#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      -
...
worm1     -       /worm1   samfs  -    yes     worm_emul
```

7. WORM 에뮬레이션 모드의 라이트 구현을 사용으로 설정하려면 *emul_lite* 옵션을 *vfstab* 파일의 *Mount Options* 열에 입력합니다.

```
#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      -
...
worm1     -      /worm1    samfs  -    yes     emul_lite
```

8. 보존 기간이 명시적으로 지정되지 않은 파일에 대한 기본 보존 기간을 변경하려면 *def_retention=period* 옵션을 *vfstab* 파일의 *Mount Options* 열에 추가합니다. 여기서 *period*는 다음 단락에 설명된 형태 중 하나입니다.

   period 값은 다음 세 가지 형태 중 하나일 수 있습니다.

- *permanent* 또는 *0*은 영구 보존을 지정합니다.
- *YEARSyDAYSdHOURShMINUTESm*. 여기서 *YEARS*, *DAYS*, *HOURS* 및 *MINUTES*는 음수가 아닌 정수이고 지정자는 생략할 수 있습니다. 따라서 예를 들어, *5y3d1h4m*, *2y12h* 및 *365d*는 모두 유효합니다.
- *MINUTES*. 여기서 *MINUTES*는 *[1-2147483647]* 범위의 정수입니다.

2038년을 넘어가는 보존 기간을 설정해야 할 경우 기본 보존 기간을 설정합니다. *touch*와 같은 UNIX 유틸리티는 부호 있는 32비트 정수를 사용하여 1970년 1월 1일 이후로 경과된 시간을 초 단위로 나타냅니다. 32비트 정수가 표현할 수 있는 가장 큰 초 수는 2038년 1월 18일 오후 10시 14분으로 변환됩니다.

값이 제공되지 않은 경우 *def_retention*은 기본적으로 *43200*분(30일)으로 설정됩니다. 예제에서는 표준 WORM 가능 파일 시스템에 대한 보존 기간을 *777600*분(540일)으로 설정합니다.

```
#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      -
...
worm1     -       /worm1   samfs  -    no     worm_capable,def_retention=777600
```

9. *vfstab* 파일을 저장하고 편집기를 닫습니다.

   파일 시스템에 WORM이 사용으로 설정됩니다. 하나 이상의 WORM 파일이 파일 시스템에 상주하면 Oracle HSM 소프트웨어는 WORM 기능을 반영하기 위해 파일 시스템 수퍼 블록을 업데이트합니다. 엄격한 *worm_capable* 또는 *worm_emul* 마운트 옵션을 사용하여 파일 시스템이 마운트된 경우 *sammkfs*를 사용하여 파일 시스템을 재작성하려는 이후의 모든 시도가 실패합니다.
10. LTFS를 사용하는 시스템과 상호 작용해야 하거나 원격 사이트 간에 많은 양의 데이터를 전송해야 할 경우 "LTFS(Linear Tape File System)에 대한 지원을 사용으로 설정"을 참조하십시오.
11. 다중 호스트 파일 시스템 액세스 또는 고가용성 구성과 같은 추가 요구 사항이 있는 경우 "기본 과정 이후"를 참조하십시오.
12. 그렇지 않은 경우 11장. 알림 및 로깅 구성으로 이동합니다.

LTFS(Linear Tape File System)에 대한 지원을 사용으로 설정

Oracle HSM은 데이터를 LTFS(Linear Tape File System) 볼륨에서 가져오고 LTFS 볼륨으로 내보낼 수 있습니다. 이 기능으로 인해 LTFS를 표준 테이프 형식으로 사용하는 시스템과의 상호 작용이 용이해집니다. 또한 일반 WAN(Wide Area Network) 연결이 너무 느리거

나 비싸서 작업에 적합하지 않은 경우 이 기능을 사용하여 원격 Oracle HSM 사이트 간에 매우 많은 양의 데이터를 쉽게 전송할 수 있습니다.

Oracle HSM 소프트웨어는 LTFS 기능을 지원하지만 포함하고 있지는 않습니다. LTFS 파일 시스템을 사용하려면 호스트의 Solaris 운영체제에 *SUNWltfs* 패키지가 포함되어 있어야 합니다. 필요한 경우 계속하기 전에 *SUNWltfs* 패키지를 다운로드하고 설치하십시오.

LTFS 볼륨 사용 및 관리에 대한 자세한 내용은 *samltfs* 매뉴얼 페이지 및 *Oracle Hierarchical Storage Manager and StorageTek QFS Software* 유지 관리 및 관리 설명서를 참조하십시오.

Oracle HSM LTFS 지원을 사용으로 설정하려면 다음과 같이 하십시오.

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

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

2. 모든 아카이빙 프로세스를 유휴 설정합니다(있는 경우). *samcmd aridle* 명령을 사용합니다.

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

   ```
   [samfs-mds]root@solaris:~# samcmd aridle
   [samfs-mds]root@solaris:~#
   ```

3. 모든 스테이징 프로세스를 유휴 설정합니다(있는 경우). *samcmd stidle* 명령을 사용합니다.

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

   ```
   [samfs-mds]root@solaris:~# samcmd stidle
   [samfs-mds]root@solaris:~#
   ```

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

   아카이빙 프로세스가 *Waiting for :arrun*이면 아카이빙 프로세스가 유휴 상태임을 나타냅니다.

   ```
   [samfs-mds]root@solaris:~# samcmd a
   Archiver status samcmd     6.0 14:20:34 Feb 22 2015
   samcmd on samfs-mds
   sam-archiverd:  Waiting for :arrun
   sam-arfind: ...
   ```

```
Waiting for :arrun
```

5. 모든 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.

```
[samfs-mds]root@solaris:~# samcmd u
Staging queue samcmd      6.0 14:20:34 Feb 22 2015
samcmd on solaris.demo.lan
Staging queue by media type: all
sam-stagerd:  Waiting for :strun
root@solaris:~#
```

6. 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.

```
[samfs-mds]root@solaris:~# samcmd 801 idle
[samfs-mds]root@solaris:~# samcmd 802 idle
[samfs-mds]root@solaris:~# samcmd 803 idle
[samfs-mds]root@solaris:~# samcmd 804 idle
[samfs-mds]root@solaris:~#
```

7. 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.

```
[samfs-mds]root@solaris:~# samcmd r
Removable media samcmd     6.0 14:20:34 Feb 22 2015
samcmd on samqfs1host
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
[samfs-mds]root@solaris:~#
```

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

```
[samfs-mds]root@solaris:~# samd stop
[samfs-mds]root@solaris:~#
```

9. 텍스트 편집기에서 */etc/opt/SUNWsamfs/defaults.conf*를 엽니다.

   예제에서는 *vi* 편집기에서 파일을 엽니다.

```
[samfs-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.
...
```

10. *defaults.conf* 파일에서 *ltfs = mountpoint workers volumes* 라인을 추가합니다. 여기서 마운트 지점은 LTFS 파일 시스템을 마운트해야 하는 호스트 파일 시스템의 디렉토리이고 *workers*는 LTFS에 사용할 선택적인 최대 드라이브 수이며 *volumes*는 선택적인 드라이브당 최대 테이프 볼륨 수입니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

    예제에서는 LTFS 마운트 지점을 */mnt/ltfs*로 지정하고 다른 매개변수에는 기본값을 사용합니다.

```
[samfs-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.
...
ltfs = /mnt/ltfs
:wq
[samfs-mds]root@solaris:~#
```

11. Oracle HSM 소프트웨어에 *defaults.conf* 파일을 다시 읽고 그에 따라 재구성하도록 지시합니다. 보고된 모든 오류를 수정하고 필요에 따라 반복합니다.

```
[samfs-mds]root@solaris:~# /opt/SUNWsamfs/sbin/samd config
```

12. 이전 단계에서 Oracle HSM 작업을 중지한 경우 이제 *samd start* 명령을 사용하여 다시 시작합니다.

```
[samfs-mds]root@solaris:~# samd start
```

13. 이제 LTFS에 대한 Oracle HSM 지원이 사용으로 설정되었습니다. 다중 호스트 파일 시스템 액세스 또는 고가용성 구성과 같은 추가 요구 사항이 있는 경우 “기본 과정 이후”를 참조하십시오.
14. 그렇지 않은 경우 11장. 알림 및 로깅 구성으로 이동합니다.

기본 과정 이후

Oracle HSM 파일 시스템의 기본 설치 및 구성이 완료되었습니다. 이제 광범위한 목적을 위해 최적으로 구성된 완벽하게 작동하는 파일 시스템이 설정되었습니다.

이 설명서의 나머지 장에서는 더 특수한 요구 사항을 다룹니다. 따라서 아래에 설명된 추가 조정 및 기능 구현 작업을 시작하기 전에 요구 사항을 신중하게 평가해야 합니다. 그런 다음 고가용성 및 공유 파일 시스템 구성과 같은 추가 기능이 필요한 경우 기본 구성부터 시작하여 추가 기능을 적절하게 구현할 수 있습니다. 그러나 지금까지 수행한 작업으로 요구 사항을 충족할 수 있는 경우 추가 변경사항으로 개선되지는 않을 것입니다. 단지 이로 인해 유지 관리 및 관리가 복잡해질 수 있습니다.

- 응용 프로그램에서 매우 많거나 일정한 양의 데이터를 파일 시스템에 전송할 경우 추가 마운트 옵션을 설정하여 파일 시스템 성능을 향상시킬 수 있습니다. 자세한 내용은 12장. 특수한 요구 사항에 맞게 *I/O* 특성 조정을 참조하십시오.
- 파일 시스템에 대한 공유 액세스를 구성해야 할 경우 “Oracle HSM 소프트웨어를 사용하여 여러 호스트에서 파일 시스템 액세스” 및/또는 “NFS 및 SMB/CIFS를 사용하여 여러 호스트에서 파일 시스템 액세스”를 참조하십시오.
- 고가용성 QFS 파일 시스템 또는 Oracle HSM 아카이빙 파일 시스템을 구성해야 할 경우 9장. 고가용성 솔루션 준비 를 참조하십시오.
- 원격 위치에서 호스팅된 아카이브 스토리지를 공유하도록 Oracle HSM 아카이빙 파일 시스템을 구성해야 할 경우 8장. *SAM-Remote* 구성 을 참조하십시오.
- 사이드밴드 데이터베이스 기능을 사용할 계획인 경우 10장. 보고 데이터베이스 구성을 참조하십시오.
- 그렇지 않은 경우 11장. 알림 및 로깅 구성으로 이동합니다.

7

7장. 여러 호스트에서 파일 시스템 액세스

여러 호스트에서 Oracle HSM 파일 시스템을 다양한 방법으로 공유할 수 있습니다. 특정 상황에서 강점을 나타내는 방법이 다른 상황에서 심각한 단점을 나타낼 수도 있습니다. 따라서 요구 사항에 따라 다른 방법을 선택해야 합니다. 공유 방법은 다음과 같습니다.

- Oracle HSM 소프트웨어를 사용하여 여러 호스트에서 파일 시스템 액세스
- NFS 및 SMB/CIFS를 사용하여 여러 호스트에서 파일 시스템 액세스

Oracle HSM 소프트웨어를 사용하여 여러 호스트에서 파일 시스템 액세스

Oracle HSM를 사용하면 하나 이상의 클라이언트와 서버에서 파일 시스템을 동시에 마운트하도록 구성하여 여러 호스트에서 파일 시스템을 사용할 수 있습니다. 그러면 NFS 및 CIFS 공유와 관련한 네트워크 및 중간 서버 대기 시간 없이 고성능 로컬 경로 I/O를 통해 디스크 장치에서 호스트로 직접 파일 데이터를 전달합니다. 호스트는 한 번에 하나만 메타데이터 서버로 활성화될 수 있지만, 중복성을 위해 원하는 만큼의 클라이언트를 잠재적 메타데이터 서버로 구성할 수 있습니다. 파일 시스템 마운트 지점 수에는 제한이 없습니다.

Oracle HSM은 아카이빙에 상관없이 다중 읽기/단일 쓰기 구성과 공유 구성 모두에서 고성능(*ma*) 파일 시스템과 범용(*ms*) 파일 시스템 모두에 대한 다중 호스트 액세스를 지원합니다. 몇 가지 제한이 있습니다.

- 블록(*b*–) 특수 파일은 지원되지 않습니다.
- 문자(*c*–) 특수 파일은 지원되지 않습니다.
- FIFO 명명된 파이프(*p*–) 특수 파일은 지원되지 않습니다.
- 세그먼트된 파일은 지원되지 않습니다.

  세그먼트된 파일 환경에서는 Oracle HSM 공유 파일 시스템을 구현할 수 없습니다.
- 필수 잠금은 지원되지 않습니다.

  필수 잠금을 설정한 경우 *EACCES* 오류가 반환됩니다. 잠금 권고는 지원됩니다. 잠금 권고에 대한 자세한 내용은 `fcntl` 매뉴얼 페이지를 참조하십시오.

Oracle HSM 소프트웨어 호스트는 주어진 응용 프로그램에서 각각 고유한 장점과 제한 사항이 있는 두 구성 중 하나를 사용하여 파일 시스템 데이터에 액세스할 수 있습니다.

다중 읽기, 단일 쓰기 구성에서 단일 호스트는 읽기/쓰기 권한이 있는 파일 시스템을 호스트하고 모든 다른 호스트는 읽기 전용 파일 시스템을 호스트합니다. 구성은 단순히 마운트 옵션

을 설정하는 것입니다. 단일 호스트에서 파일에 대한 모든 변경을 수행하는 경우 추가적인 파일 잠금이나 일관성 검사 없이도 파일 일관성과 데이터 무결성이 보장됩니다. 최적의 성능을 위해 모든 호스트에서 디스크의 데이터를 직접 읽고 메타데이터도 읽습니다. 하지만 모든 호스트에서 파일 시스템 메타데이터에 액세스할 수 있어야 하므로, *ma* 파일 시스템의 모든 호스트가 데이터와 메타데이터 장치 모두에 액세스할 수 있어야 합니다.

공유 구성에서는 모든 호스트가 임대를 사용하여 파일 데이터를 읽고, 쓰고, 추가할 수 있습니다. 임대는 단일 호스트에서 지정된 기간 동안 지정된 방법으로 파일에 액세스할 수 있도록 허용하는 기능입니다. 메타데이터 서버는 읽기, 쓰기 및 추가 임대를 발급하고 충돌하는 임대 요청과 갱신을 관리합니다. 공유 파일 시스템은 융통성을 확대하지만, 구성이 복잡하고 많은 파일 시스템 오버헤드를 발생합니다. 모든 호스트는 디스크에서 직접 파일 데이터를 읽지만, 클라이언트는 네트워크를 통해 메타데이터에 액세스합니다. 따라서 메타데이터 장치에 액세스할 수 없는 클라이언트는 *ma* 파일 시스템을 공유할 수 있습니다.

여러 호스트에서 데이터 액세스를 구성하려면 다음 두 가지 접근법 중 하나를 선택하십시오.

- Oracle HSM 단일 쓰기, 다중 읽기 파일 시스템 구성
- Oracle HSM 공유 파일 시스템 구성.

Oracle HSM 단일 쓰기, 다중 읽기 파일 시스템 구성

단일 쓰기, 다중 읽기 파일 시스템을 구성하려면 다음 작업을 수행합니다.

- 쓰기 장치에서 파일 시스템 만들기
- 읽기 장치 구성

쓰기 장치에서 파일 시스템 만들기

다음과 같이 하십시오.

1. *root* 계정을 사용하여 *writer* 역할을 하는 호스트에 로그인합니다.

   예제에서 *writer* 호스트 이름은 *swriterfs-mds-writer*로 지정됩니다.

```
[swriterfs1-mds-writer]root@solaris:~#
```

2. *writer* 역할을 하는 호스트에서 */etc/opt/SUNWsamfs/mcf* 파일을 텍스트 편집기에서 열고 QFS 파일 시스템을 추가합니다. 범용 *ms* 또는 고성능 *ma* 파일 시스템을 구성할 수 있습니다.

   별도의 메타데이터 장치를 포함하는 *ma* 파일 시스템에서, 파일 시스템에 대한 메타데이터 서버를 쓰기 장치로 구성합니다. 아래 예제에서는 *vi* 텍스트 편집기를 사용하여 *swriterfs1-mds-writer* 호스트의 *mcf* 파일을 편집합니다. 예제에서는 장비 식별자 및 패밀리 세트 이름 *swriterfs1* 및 장비 순서 번호 *300*을 사용하여 *ma* 파일 시스템을 지정합니다.

```
[swriterfs1-mds-writer]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
```

```
# Equipment           Equipment  Equipment  Family      Device   Additional
# Identifier          Ordinal    Type       Set         State    Parameters
#------------------  ---------  ---------  ---------   ------   ---------------
swriterfs1            300         ma          swriterfs1   on
 /dev/dsk/c0t0d0s0    301        mm         swriterfs1   on
 /dev/dsk/c0t3d0s0    302        mr         swriterfs1   on
 /dev/dsk/c0t3d0s1    303        mr         swriterfs1   on
```

3. */etc/opt/SUNWsamfs/mcf* 파일을 저장하고 편집기를 종료합니다.

   예제에서는 변경사항을 저장하고 *vi* 편집기를 종료합니다.

```
# Equipment           Equipment  Equipment  Family      Device   Additional
# Identifier          Ordinal    Type       Set         State    Parameters
#------------------  ---------  ---------  ---------   ------   ---------------
swriterfs1            300        ma         swriterfs1  on
 /dev/dsk/c0t0d0s0     301        mm          swriterfs1  on
 /dev/dsk/c0t3d0s0     302        mr          swriterfs1  on
 /dev/dsk/c0t3d0s1     303        mr          swriterfs1  on
:wq
[swriterfs1-mds-writer]root@solaris:~#
```

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

   sam-fsd 명령은 Oracle HSM 구성 파일을 읽고 파일 시스템을 초기화합니다. 오류가 발견되면 실행을 중지합니다.

```
[swriterfs1-mds-writer]root@solaris:~# sam-fsd
...
Would start sam-stagerd()
Would start sam-amld()
[swriterfs1-mds-writer]root@solaris:~#
```

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

```
[swriterfs1-mds-writer]root@solaris:~# samd config
Configuring SAM-FS
[swriterfs1-mds-writer]root@solaris:~#
```

6. "고성능 **ma** 파일 시스템 구성"에 설명한 대로 *sammkfs* 명령과 파일 시스템의 패밀리 세트 이름을 사용하여 파일 시스템을 만듭니다.

   예제에 사용된 명령은 단일 쓰기/다중 읽기 파일 시스템 *swriterfs1*을 만듭니다.

```
[swriterfs1-mds-writer]root@solaris:~# sammkfs swriterfs1
Building 'swriterfs1' will destroy the contents of devices:
  /dev/dsk/c0t0d0s0
  /dev/dsk/c0t3d0s0
  /dev/dsk/c0t3d0s1
Do you wish to continue? [y/N]yes ...
```

7. 운영체제의 */etc/vfstab* 파일을 백업합니다.

```
[swriterfs1-mds-writer]root@solaris:~# cp /etc/vfstab /etc/vfstab.backup
[swriterfs1-mds-writer]root@solaris:~#
```

8. "고성능 ma 파일 시스템 구성"에 설명한 대로 운영체제의 */etc/vfstab* 파일에 새 파일 시스템을 추가합니다.

   예제에서는 */etc/vfstab* 파일을 *vi* 텍스트 편집기에서 열고 *swriterfs1* 패밀리 세트 장치에 대한 행을 추가합니다.

```
[swriterfs1-mds-writer]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       -
...
swriterfs1 -        /swriterfs1 samfs   -     no
```

9. */etc/vfstab* 파일의 *Mount Options* 열에 *writer* 마운트 옵션을 입력합니다.

   주의:

   한 번에 하나의 호스트만 *writer* 역할을 하도록 합니다. *writer* 옵션을 사용하여 여러 호스트에서 다중 읽기, 단일 쓰기 파일 시스템을 마운트하도록 허용하면 파일 시스템이 손상될 수 있습니다.

```
#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       -
...
swriterfs1 -        /swriterfs1 samfs   -     no       writer
```

10. */etc/vfstab* 파일에 다른 원하는 변경을 수행합니다. 콤마를 구분자로 사용하여 마운트 옵션을 추가합니다.

    예를 들어, 첫번째 시도가 실패할 경우 백그라운드에서 파일 시스템을 마운트하려면 *Mount Options* 필드에 *bg* 마운트 옵션을 추가합니다. 사용 가능한 전체 마운트 옵션 목록은 *mount_samfs* 매뉴얼 페이지를 참조하십시오.

```
#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       -
...
swriterfs1 -        /swriterfs1 samfs   -     no       writer,bg
```

11. */etc/vfstab* 파일을 저장하고 편집기를 종료합니다.

```
#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       -
...
swriterfs1 -        /swriterfs1 samfs   -     no       writer,bg
:wq
[swriterfs1-mds-writer]root@solaris:~#
```

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

    마운트 지점 권한이 모든 호스트에서 동일해야 합니다. 사용자가 마운트 지점 디렉토리를 변경하고 마운트된 파일 시스템에서 파일에 액세스하려면 실행(*x*) 권한이 있어야 합니다. 예제에서는 */swriterfs1* 마운트 지점 디렉토리를 만들고 *755*(*-rwxr-xr-x*)로 권한을 설정합니다.

```
[swriterfs1-mds-writer]root@solaris:~# mkdir /swriterfs1
[swriterfs1-mds-writer]root@solaris:~# chmod 755 /swriterfs1
[swriterfs1-mds-writer]root@solaris:~#
```

13. 새 파일 시스템을 마운트합니다.

```
[swriterfs1-mds-writer]root@solaris:~# mount /swriterfs1
[swriterfs1-mds-writer]root@solaris:~#
```

14. 공유 파일 시스템이 만들어졌으면 읽기 장치 구성을 수행합니다.

읽기 장치 구성

읽기 장치는 파일 시스템을 읽기 전용으로 마운트하는 호스트입니다. 읽기 장치로 구성하는 각 호스트에 대해 다음과 같이 하십시오.

1. 호스트에 *root*로 로그인합니다.

   예제에서 *reader* 호스트 이름은 *swriterfs-reader1]*로 지정됩니다.

```
[swriterfs-reader1]root@solaris:~#
```

2. 단말기 창에서 *samfsconfig device-path* 명령을 사용하여 다중 읽기, 단일 쓰기 파일 시스템에 대한 구성 정보를 검색합니다. 여기서 *device-path*는 이 명령이 파일 시스템 디스크 장치 검색을 시작해야 하는 위치입니다(예: */dev/dsk/**).

   samfsconfig 유틸리티는 *sammkfs*가 Oracle HSM 파일 시스템에 포함된 각 장치에 쓰는 식별 수퍼 블록을 읽고 파일 시스템 구성 정보를 검색합니다. 이 명령은 현재 호스트에서 시작하여 구성의 각 장치에 대한 올바른 경로를 반환하고 연결할 수 없는 장치를 플래그합니다. 명령 구문과 매개변수에 대한 자세한 내용은 *samfsconfig* 매뉴얼 페이지를 참조하십시오.

   장치 경로가 *swriterfs1-reader1* 호스트부터 지정된다는 점만 제외하고, 예제의 *samfsconfig* 출력에는 *swriterfs1-mds-writer*의 *mcf* 파일에 나열된 것과 동일한 장비가 표시됩니다.

```
[swriterfs1-reader1]root@solaris:~# samfsconfig /dev/dsk/*
# Family Set 'swriterfs1' Created Thu Nov 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   -
```

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

   또는 *samfsconfig*의 출력을 *mcf* 파일로 재지정할 수 있습니다. 또는 *samd buildmcf* 명령을 사용하여 *samfsconfig*를 실행하고 클라이언트 *mcf* 파일을 자동으로 만들 수 있습니다.

예제에서 주석 처리된 열 머리글을 추가한 후 *swriterfs1-reader1* 호스트의 *mcf* 파일은 다음과 비슷합니다.

```
[swriterfs1-reader1]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    -
```

4. 모든 장치에 대해 *Device State* 필드가 *on*으로 설정되어 있는지 확인합니다. 그런 다음 *mcf* 파일을 저장합니다.

```
[swriterfs1-reader1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment           Equipment  Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#-----------------   ---------  ---------  ---------  ------   ---------------
sharefs               300        ma         sharefs    on
 /dev/dsk/c1t0d0s0     301        mm         sharefs    on
 /dev/dsk/c1t3d0s0     302        mr         sharefs    on
 /dev/dsk/c1t3d0s1     303        mr         sharefs    on
:wq
[swriterfs1-reader1]root@solaris:~#
```

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

   sam-fsd 명령은 Oracle HSM 구성 파일을 읽고 파일 시스템을 초기화합니다. 오류가 발견되면 실행을 중지합니다.

```
[swriterfs1-reader1]root@solaris:~# sam-fsd
...
Would start sam-stagerd()
Would start sam-amld()
[swriterfs1-reader1]root@solaris:~#
```

6. 운영체제의 */etc/vfstab* 파일을 백업합니다.

```
[swriterfs1-reader1]root@solaris:~# cp /etc/vfstab /etc/vfstab.backup
[swriterfs1-reader1]root@solaris:~#
```

7. 호스트 운영체제의 */etc/vfstab* 파일에 단일 쓰기, 다중 읽기 파일 시스템을 추가합니다.

   예제에서는 */etc/vfstab* 파일을 *vi* 텍스트 편집기에서 열고 *swriterfs1* 패밀리 세트 장치에 대한 행을 추가합니다.

```
[swriterfs1-reader1]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       -
...
swriterfs1 -        /swriterfs1 samfs   -     no
```

8. */etc/vfstab* 파일의 *Mount Options* 열에 *reader* 옵션을 입력합니다.

   주의:

   호스트에서 *reader* 옵션을 사용하여 파일 시스템을 마운트하는지 확인합니다. 실수로 여러 호스트에서 *writer* 마운트 옵션을 사용할 경우 파일 시스템이 손상될 수 있습니다.

```
#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       -
...
swriterfs1 -        /swriterfs1 samfs   -     no       reader
```

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

```
#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       -
...
swriterfs1 -        /swriterfs1 samfs   -     no       writer,bg
:wq
[swriterfs1-reader1]root@solaris:~#
```

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

    마운트 지점 권한이 모든 호스트에서 동일해야 합니다. 사용자가 마운트 지점 디렉토리를 변경하고 마운트된 파일 시스템에서 파일에 액세스하려면 실행(*x*) 권한이 있어야 합니다. 예제에서는 방금 쓰기 호스트에서 수행한 것처럼 */swriterfs1* 마운트 지점 디렉토리를 만들고 *755*(*-rwxr-xr-x*)로 권한을 설정합니다.

```
[swriterfs1-reader1]root@solaris:~# mkdir /swriterfs1
[swriterfs1-reader1]root@solaris:~# chmod 755 /swriterfs1
[swriterfs1-reader1]root@solaris:~#
```

11. 새 파일 시스템을 마운트합니다.

```
[swriterfs1-reader1]root@solaris:~# mount /swriterfs1
[swriterfs1-reader1]root@solaris:~#
```

12. 모든 읽기 호스트가 파일 시스템을 읽기 전용으로 마운트하도록 구성될 때까지 이 절차를 반복합니다.
13. 사이드밴드 데이터베이스 기능을 사용할 계획인 경우 10장. 보고 데이터베이스 구성을 참조하십시오.
14. 그렇지 않은 경우 11장. 알림 및 로깅 구성으로 이동합니다.

Oracle HSM 공유 파일 시스템 구성

Oracle HSM 공유 파일 시스템은 여러 Oracle HSM 호스트에 파일에 대한 읽기, 쓰기 및 추가 권한을 제공합니다. 모든 호스트에서 파일 시스템을 마운트하고 스토리지 장치에 직접 연결합니다. 또한 호스트 중 하나인 메타데이터 서버(MDS)에서 파일 시스템 메타데이터를 배타적으로 제어하고, 동일한 파일에 액세스하려는 호스트 간을 중재합니다. 이 서버는 이더넷 로컬 네트워크를 통해 클라이언트 호스트에 메타데이터 업데이트를 제공하고, 읽기, 쓰기 및 추가 임대를 실행, 갱신, 취소하여 파일 액세스를 제어합니다. 고성능 *ma* 또는 범용 *ms* 유형의 비아카이빙 파일 시스템과 아카이빙 파일 시스템을 모두 공유할 수 있습니다.

공유 파일 시스템을 구성하려면 다음 작업을 수행합니다.

- 공유를 위한 파일 시스템 메타데이터 서버 구성
- 공유를 위한 파일 시스템 클라이언트 구성
- 공유 파일 시스템에 대한 아카이브 스토리지 구성

공유를 위한 파일 시스템 메타데이터 서버 구성

공유 파일 시스템을 지원하도록 메타데이터 서버를 구성하려면 아래 나열된 작업을 수행합니다.

- 활성/잠재적 메타데이터 서버에서 hosts 파일 만들기

- 활성 서버에서 공유 파일 시스템 만들기
- 활성 서버에서 공유 파일 시스템 마운트

활성/잠재적 메타데이터 서버에서 hosts 파일 만들기

활성/잠재적 메타데이터 서버에서 공유 파일 시스템의 서버와 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. 서버에 *root*로 로그인합니다.

   예제에서 서버 이름은 *sharefs-mds*로 지정됩니다.

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

2. 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.sharefs* 파일을 만듭니다. 몇 가지 선택적 머리글을 추가하고, 각 라인은 주석을 나타내는 해시 기호(#)로 시작합니다.

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

3. 메타데이터 서버의 호스트 이름 및 IP 주소/도메인 이름을 공백 문자로 구분하여 2개 열에 추가합니다.

```
# /etc/opt/SUNWsamfs/hosts.sharefs
#                                        Server   On/  Additional
#Host Name            Network Interface  Ordinal  Off  Parameters
#------------------   ------------------ -------  ---  ----------
sharefs-mds           10.79.213.117
```

4. 네트워크 주소와 공백 문자로 구분된 세번째 열을 추가합니다. 이 열에 활성 메타데이터 서버의 순서 번호인 *1*을 입력합니다.

   이 예제에서는 메타데이터 서버가 하나뿐이므로 *1*을 입력합니다.

```
# /etc/opt/SUNWsamfs/hosts.sharefs
```

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

5. 네트워크 주소와 공백 문자로 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 1)은 호스트가 파일 시스템에 액세스할 수 있도록 구성되지 않은 *off* 상태라는 것을 나타냅니다(공유 파일 시스템을 관리할 때 이러한 값을 사용하는 방법에 대한 자세한 내용은 *samsharefs* 매뉴얼 페이지 참조).

```
# /etc/opt/SUNWsamfs/hosts.sharefs
#                                        Server   On/  Additional
#Host Name            Network Interface  Ordinal  Off  Parameters
#------------------   ------------------ -------  ---  ----------
sharefs-mds          10.79.213.117      1        0
```

6. 네트워크 주소와 공백 문자로 구분된 다섯번째 열을 추가합니다. 이 열에서 현재 활성 메타데이터 서버를 나타내는 *server* 키워드를 입력합니다.

```
# /etc/opt/SUNWsamfs/hosts.sharefs
#                                        Server   On/  Additional
#Host Name            Network Interface  Ordinal  Off  Parameters
#------------------   ------------------ -------  ---  ----------
sharefs-mds          10.79.213.117      1        0    server
```

7. 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).

   예제에서 호스트 *sharefs-mds_alt*는 서버 순서가 *2*인 잠재적 메타데이터 서버입니다.

```
# /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
```

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

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

```
# /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       10.79.213.147         0        0
```

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

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

```
# /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       10.79.213.147         0        0
:wq
[sharefs-mds]root@solaris:~#
```

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

활성 서버에서 공유 파일 시스템 만들기

다음과 같이 하십시오.

1. 서버에 *root*로 로그인합니다.

   예제에서 서버 이름은 *sharefs-mds*로 지정됩니다.

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

2. 메타데이터 서버(MDS)에서 */etc/opt/SUNWsamfs/mcf* 파일을 텍스트 편집기에서 열고 QFS 파일 시스템을 추가합니다. 범용 *ms* 또는 고성능 *ma* 파일 시스템을 구성할 수 있습니다.

아래 예제에서는 *vi* 텍스트 편집기를 사용하여 *sharefs-mds* 호스트에서 *mcf* 파일을 편집합니다. 이 예제에서는 장비 식별자와 패밀리 세트 이름 *sharefs* 및 장비 순서 번호 *300*을 사용하여 *ma* 파일 시스템을 지정합니다.

```
[sharefs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment           Equipment  Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#------------------   ---------  ---------  ---------  ------   ---------------
sharefs               300        ma         sharefs    on
 /dev/dsk/c0t0d0s0    301        mm         sharefs    on
 /dev/dsk/c0t3d0s0    302        mr         sharefs    on
 /dev/dsk/c0t3d0s1    303        mr         sharefs    on
```

3. *ma* 파일 시스템 장비에 대한 행의 *Additional Parameters* 필드에 *shared* 매개변수를 입력합니다.

```
# Equipment           Equipment   Equipment  Family     Device   Additional
# Identifier          Ordinal    Type       Set        State    Parameters
#------------------   ---------  ---------  ---------  ------   ---------------
sharefs               300         ma        sharefs    on       shared
 /dev/dsk/c0t0d0s0     301         mm        sharefs    on
 /dev/dsk/c0t3d0s0     302         mr        sharefs    on
 /dev/dsk/c0t3d0s1     303         mr        sharefs    on
```

4. */etc/opt/SUNWsamfs/mcf* 파일을 저장하고 편집기를 종료합니다.

   예제에서는 변경사항을 저장하고 *vi* 편집기를 종료합니다.

```
sharefs               300         ma         sharefs    on      shared
 /dev/dsk/c0t0d0s0     301         mm         sharefs    on
 /dev/dsk/c0t3d0s0     302         mr         sharefs    on
 /dev/dsk/c0t3d0s1     303         mr         sharefs    on
:wq
[sharefs-mds]root@solaris:~#
```

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

   sam-fsd 명령은 Oracle HSM 구성 파일을 읽고 파일 시스템을 초기화합니다. 오류가 발견되면 실행을 중지합니다.

```
[sharefs-mds]root@solaris:~# sam-fsd
...
Would start sam-stagerd()
```

```
Would start sam-amld()
[sharefs-mds]root@solaris:~#
```

6. Oracle HSM 서비스에 *mcf* 파일을 다시 읽고 그에 따라 재구성하도록 지시합니다. 보고된 모든 오류를 수정하고 필요에 따라 반복합니다.

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

7. "고성능 ma 파일 시스템 구성"에서 설명한 대로 *sammkfs -S* 명령과 파일 시스템의 패밀리 세트 이름을 사용하여 파일 시스템을 만듭니다.

   sammkfs 명령은 *hosts.family-set-name* 및 *mcf* 파일을 읽고 지정된 등록 정보를 가진 공유 파일 시스템을 만듭니다. 예제의 명령은 *hosts.sharefs* 파일에서 공유 매개변수를 읽고 공유 파일 시스템 *sharefs*를 만듭니다.

```
[sharefs-mds]root@solaris:~# sammkfs -S sharefs
Building 'sharefs' will destroy the contents of devices:
  /dev/dsk/c0t0d0s0
  /dev/dsk/c0t3d0s0
  /dev/dsk/c0t3d0s1
Do you wish to continue? [y/N]yes ...
[sharefs-mds]root@solaris:~#
```

8. 이제 활성 메타데이터 서버에서 공유 파일 시스템 마운트를 수행합니다.

활성 서버에서 공유 파일 시스템 마운트

1. 서버에 *root*로 로그인합니다.

   예제에서 서버 이름은 *sharefs-mds*로 지정됩니다.

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

2. 운영체제의 */etc/vfstab* 파일을 백업합니다.

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

3. "고성능 ma 파일 시스템 구성"에 설명한 대로 운영체제의 */etc/vfstab* 파일에 새 파일 시스템을 추가합니다.

   예제에서는 */etc/vfstab* 파일을 *vi* 텍스트 편집기에서 열고 *sharefs* 패밀리 세트 장치에 대한 행을 추가합니다.

```
[sharefs-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       -
...
sharefs    -        /sharefs   samfs   -     no
```

4. *Mount Options* 열에 *shared* 옵션을 입력합니다.

```
#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. */etc/vfstab* 파일에 다른 원하는 변경을 수행합니다.

   예를 들어 초기 시도가 실패할 경우 백그라운드에서 파일 시스템 마운트를 재시도하려면 *bg* 마운트 옵션을 *Mount Options* 필드에 추가합니다. 사용 가능한 마운트 옵션에 대한 자세한 내용은 *mount_samfs* 매뉴얼 페이지를 참조하십시오.

```
#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,bg
```

6. */etc/vfstab* 파일을 저장하고 편집기를 종료합니다.

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

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

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

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

8. 새 파일 시스템을 마운트합니다.

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

9. 호스트를 다중 네트워크 인터페이스로 구성한 경우 로컬 hosts 파일을 사용하여 네트워크 통신 경로 지정 작업을 수행할 수 있습니다.
10. 그렇지 않은 경우 공유 파일 시스템이 메타데이터 서버에서 만들어졌으면 공유를 위한 파일 시스템 클라이언트 구성을 수행합니다.

공유를 위한 파일 시스템 클라이언트 구성

클라이언트는 순수하게 클라이언트로 구성된 호스트와 잠재적 메타데이터 서버로 구성된 호스트를 모두 포함합니다. 대부분의 경우 클라이언트 구성은 서버를 구성하는 방법과 거의 동일합니다. 각 클라이언트는 서버와 동일한 장치를 포함합니다. 마운트 옵션과 정확한 장치 경로만 변경됩니다. 제어기 번호는 클라이언트 호스트별로 지정되므로 다를 수 있습니다.

하나 이상의 클라이언트에서 공유 파일 시스템을 지원하도록 구성하려면 아래 나열된 작업을 수행합니다.

- Solaris 클라이언트에서 공유 파일 시스템 만들기
- Solaris 클라이언트에서 공유 파일 시스템 마운트
- Linux 클라이언트에서 공유 파일 시스템 만들기(있는 경우)
- Linux 클라이언트에서 공유 파일 시스템 마운트(있는 경우)

Solaris 클라이언트에서 공유 파일 시스템 만들기

각 클라이언트에 대해 다음과 같이 하십시오.

1. 클라이언트에서 *root*로 로그인합니다.

   예제에서 서버 이름은 *sharefs-client1*로 지정됩니다.

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

2. 단말기 창에서 *samfsconfig device-path* 명령을 입력합니다. 여기서 *device-path*는 명령이 파일 시스템 디스크 장치(예: */dev/dsk/** 또는 */dev/zvol/dsk/rpool/**) 검색을 시작해야 하는 위치입니다.

   samfsconfig 명령은 공유 파일 시스템에 대한 구성 정보를 검색합니다.

```
[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 붙여 넣습니다.

   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.

   예제에서는 *sharefs-client1* 호스트에서 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 저장합니다.

    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 명령은 Oracle HSM 구성 파일을 읽고 파일 시스템을 초기화합니다. 만일 오류가 발생하면 중지됩니다. 예제에서는 *sharefs-client1*에서 *mcf* 파일을 확인합니다.

```
[sharefs-client1]root@solaris:~# sam-fsd
...
Would start sam-stagerd()
Would start sam-amld()
[sharefs-client1]root@solaris:~#
```

12. 이때 호스트를 다중 네트워크 인터페이스로 구성한 경우 로컬 hosts 파일을 사용하여 네트워크 통신 경로 지정 작업을 수행할 수 있습니다.
13. 이제 Solaris 클라이언트에서 공유 파일 시스템 마운트를 수행합니다.

Solaris 클라이언트에서 공유 파일 시스템 마운트

각 클라이언트에 대해 다음과 같이 하십시오.

1. Solaris 클라이언트에서 *root*로 로그인합니다.

   예제에서 서버 이름은 *sharefs-client1*로 지정됩니다.

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

2. 운영체제의 */etc/vfstab* 파일을 백업합니다.

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

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

5. */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:~#
```

6. 공유 파일 시스템을 마운트합니다.

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

7. 공유 파일 시스템에 Linux 클라이언트가 포함되어 있는 경우 Linux 클라이언트에서 공유 파일 시스템 만들기를 수행합니다.
8. Oracle HSM 공유 아카이빙 파일 시스템을 구성할 경우 "공유 파일 시스템에 대한 아카이브 스토리지 구성" 작업으로 이동합니다.
9. 그렇지 않은 경우 여기서 중지합니다. Oracle HSM 공유 파일 시스템을 구성했습니다.

Linux 클라이언트에서 공유 파일 시스템 만들기

각 클라이언트에 대해 다음과 같이 하십시오.

1. Linux 클라이언트에서 *root*로 로그인합니다.

   예제에서 Linux 클라이언트 호스트 이름은 *sharefs-clientL*로 지정됩니다.

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

2. 단말기 창에서 *samfsconfig device-path* 명령을 입력합니다. 여기서 *device-path*는 명령이 파일 시스템 디스크 장치(예: */dev/**) 검색을 시작해야 하는 위치입니다.

   samfsconfig 명령은 공유 파일 시스템에 대한 구성 정보를 검색합니다. Linux 호스트는 파일 시스템의 메타데이터 장치에 액세스할 수 없으므로 *samfsconfig*가 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.

   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 붙여넣습니다.

   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 데이터 장치 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 명령은 Oracle HSM 구성 파일을 읽고 파일 시스템을 초기화합니다. 만일 오류가 발생하면 중지됩니다. 예제에서는 Linux 클라이언트 *sharefs-clientL*에서 *mcf* 파일을 확인합니다.

```
[sharefs-clientL][root@linux ~]# sam-fsd
...
Would start sam-stagerd()
Would start sam-amld()
[sharefs-clientL][root@linux ~]#
```

9. 이제 Linux 클라이언트에서 공유 파일 시스템 마운트를 수행합니다.

Linux 클라이언트에서 공유 파일 시스템 마운트

각 클라이언트에 대해 다음과 같이 하십시오.

1. Linux 클라이언트에서 *root*로 로그인합니다.

   예제에서 Linux 클라이언트 호스트 이름은 *sharefs-clientL*로 지정됩니다.

```
[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 *sharefs-clientL*에서 */etc/fstab* 파일을 백업한 후에 *vi* 텍스트 편집기에서 파일을 열고 *sharefs* 패밀리 세트 장치 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.

```
#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 지점입니다.

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

9. Oracle HSM 공유 아카이빙 파일 시스템을 구성할 경우 "공유 파일 시스템에 대한 아카이브 스토리지 구성" 작업으로 이동합니다.
10. 사이드밴드 데이터베이스 기능을 사용할 계획인 경우 10장. 보고 데이터베이스 구성을 참조하십시오.
11. 그렇지 않은 경우 11장. 알림 및 로깅 구성으로 이동합니다.

로컬 hosts 파일을 사용하여 네트워크 통신 경로 지정

개별 호스트에는 로컬 hosts 파일이 필요 없습니다. 파일 시스템은 모든 파일 시스템 호스트에 대한 활성 메타데이터 서버와 활성 및 잠재적 메타데이터 서버의 네트워크 인터페이스를 식별합니다("활성/잠재적 메타데이터 서버에서 hosts 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 네트워크 인터페이스에서 메타데이터 서버의 다른 호스트를 조회합니다. 호스트 이름과 IP 주소는 파일 시스템에 대한 전역 hosts 파일 */etc/opt/SUNWsamfs/hosts.family-set-name*에 나열됩니다. 여기서 *family-set-name*은 공유 파일 시스템의 패밀리 세트 번호입니다. 그런 다음 호스트는 로컬 hosts 파일 */etc/opt/SUNWsamfs/hosts.family-set-name.local*을 찾습니다.

로컬 hosts 파일이 없는 경우 호스트는 전역 hosts 파일에 지정된 인터페이스 주소를 사용합니다. 전역 파일에서 지정된 순서대로 호스트가 사용됩니다.

로컬 hosts 파일이 있는 경우 호스트는 로컬 파일을 전역 파일과 비교하여 양쪽 파일에 나열된 인터페이스만 사용합니다. 로컬 파일에서 지정된 순서대로 호스트가 사용됩니다.

따라서 각 파일에 다른 주소를 사용하여 다양한 호스트에서 사용되는 인터페이스를 제어할 수 있습니다. 로컬 hosts 파일을 구성하려면 아래 설명된 절차를 사용하십시오.

1. 각 활성/잠재적 메타데이터 서버 호스트에서 서버 및 호스트 통신을 필요한 방식으로 라우팅하도록 공유 파일 시스템에 대한 전역 hosts 파일을 편집합니다.

   이 절의 예제에서는 공유 파일 시스템 *sharefs2nic*에 활성 메타데이터 서버*sharefs2-mds*와 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 hosts 파일인 */etc/opt/SUNWsamfs/hosts.sharefs2*를 편집합니다. 활성/잠재적 서버에 대한 개인 네트워크 인터페이스 주소를 지정합니다. 그러나 클라이언트의 경우 주소 대신 호스트 이름을 지정합니다.

```
[sharefs2-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sharefs2
# /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
:wq
[sharefs2-mds]root@solaris:~#
```

2. 각 활성/잠재적 메타데이터 서버에서 경로 및 파일 이름 */etc/opt/SUNWsamfs/hosts.family-set-name.local*을 사용하여 로컬 hosts 파일을 만듭니다. 여기서 *family-set-name*은 공유 파일 시스템의 장비 식별자입니다. 활성/잠재적 서버에서 사용할 네트워크 인터페이스만 포함하십시오.

   예제에서는 활성/잠재적 메타데이터 서버에서 개인 네트워크를 통해 서로 통신하므로 각 서버의 로컬 hosts 파일인 *hosts.sharefs2.local*에 두 호스트인 활성/잠재적 메타데이터 서버의 개인 주소만 나열됩니다.

```
[sharefs2-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sharefs2.local
# /etc/opt/SUNWsamfs/hosts.sharefs2 on sharefs2-mds
#                                    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.local on sharefs2-mds_alt
#                                    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. 각 클라이언트에서 경로 및 파일 이름 */etc/opt/SUNWsamfs/hosts.family-set-name.local*을 사용하여 로컬 호스트 파일을 만듭니다. 여기서 *family-set-name*은 공유 파일 시스템의 장비 식별자입니다. 클라이언트에서 사용할 네트워크 인터페이스만 포함하십시오.

   예제에서는 클라이언트가 공용 네트워크를 통해서만 서버와 통신합니다. 따라서 파일에는 두 호스트인 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.local on sharefs2-client1
#                                     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.local on sharefs2-client2
#                                     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 “활성 서버에서 공유 파일 시스템 마운트”로 이동합니다.
5. 클라이언트를 구성하는 동안 이 절차를 시작한 경우 지금 “Solaris 클라이언트에서 공유 파일 시스템 마운트”해야 합니다.

공유 파일 시스템에 대한 아카이브 스토리지 구성

아카이빙 Oracle HSM 공유 파일 시스템에 대한 아카이브 스토리지를 설정하려면 다음 작업을 수행합니다.

- 지속 바인딩을 사용하여 서버 및 Datamover 호스트에 테이프 드라이브 연결
- 아카이브 스토리지를 사용하도록 아카이빙 파일 시스템의 호스트 구성
- 공유 아카이브 파일 시스템의 호스트 전체로 테이프 I/O 분산(필요한 경우)

지속 바인딩을 사용하여 서버 및 Datamover 호스트에 테이프 드라이브 연결

공유 아카이빙 파일 시스템에서 모든 잠재적 메타데이터 서버는 라이브러리 및 테이프 드라이브에 대한 액세스 권한이 있어야 합니다. 공유 아카이브 파일 시스템의 호스트 전체로 테이

프 I/O 분산을 결정할 경우 하나 이상의 클라이언트도 드라이브에 액세스해야 합니다. 따라서 각 드라이브를 일관된 방법으로 처리하도록 각 호스트를 구성해야 합니다.

Solaris 운영체제는 시작 시 장치를 발견한 순서대로 시스템 장치 트리에 드라이브를 연결합니다. 이 순서는 다른 파일 시스템 호스트에서 장치를 발견한 순서나 이동식 매체 라이브러리에 장치가 물리적으로 설치된 순서를 반영할 수도 있고 그렇지 않을 수도 있습니다. 따라서 다른 호스트에 바인드된 것과 동일한 방법과 이동식 매체 라이브러리에 설치된 것과 동일한 순서로 각 호스트에 장치를 지속적으로 바인드해야 합니다.

아래 절차는 필요한 단계를 간략히 설명합니다. 지속 바인딩을 만드는 방법에 대한 자세한 내용은 Solaris *devfsadm* 및 *devlinks* 매뉴얼 페이지와 사용하는 Solaris 운영체제 버전의 관리 설명서를 참조하십시오.

1. 활성 메타데이터 서버에 *root*로 로그인합니다.

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

2. 현재 라이브러리에 있는 드라이브의 물리적 순서를 모를 경우 "라이브러리에 드라이브가 설치되는 순서 결정"에 설명된 대로 매핑 파일을 만듭니다.

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

```
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.

   이 예에서는 *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 */etc/devlink.tab* 파일에 행을 추가하여 Solaris 테이프 장치 트리의 시작 노드 *rmt/node-number*를 라이브러리의 첫 번째 드라이브에 재매핑합니다. *type=ddi_byte:tape; addr=device_address,0; rmt/node-number/M0* 형식으로 라인을 입력합니다. 여기서 *device_address*는 장치

의 물리적 주소이고 *node-number*는 Solaris 장치 트리에서 Solaris가 자동으로 구성하는 모든 장치와 충돌을 피하기에 충분히 높은 위치입니다(Solaris는 *0* 노드부터 시작됨).

예제에서 라이브러리의 첫번째 장치 *1*에 대한 장치 주소는 *w500104f0008120fe*입니다. 장치가 현재 *rmt/1*의 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 충돌하지 않는 노드 *rmt/60*을 라이브러리의 번호 *1* 드라이브 *w500104f0008120fe*에 재매핑하는 라인을 */etc/devlink.tab*에 만듭니다.

```
[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
```

그런 후에 라이브러리와 동일한 순서를 유지하여 다음 세 개의 Solaris 장치 노드(**rmt/61**, **rmt/62** 및 **rmt/63**)에 장치 주소를 매핑합니다.

```
[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 구성의 각 잠재적 메타데이터 서버 및 datamover에 동일한 지속 테이프 장치 링크를 만듭니다. */etc/devlink.tab* 파일에 동일한 라인을 추가하고 */dev/rmt*에서 링크를 삭제한 다음 *devfsadm -c tape*를 실행합니다.

   예제에는 잠재적 메타데이터 서버 *sharefs-mds_alt*와 datamover 클라이언트 *sharefs-client1*이 있습니다. 각 위치의 */etc/devlink.tab* 파일을 활성 서버 *sharefs-mds*의 해당 파일과 일치하도록 편집합니다. 그런 다음 *sharefs-mds_alt* 및 *sharefs-client1*의 */dev/rmt*에서 기존 링크를 삭제하고 각각에 대해 *devfsadm -c tape*를 실행합니다.

```
[sharefs-mds]root@solaris:~# ssh 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 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. 이제 아카이브 스토리지를 사용하도록 아카이빙 파일 시스템의 호스트 구성을 수행합니다.

아카이브 스토리지를 사용하도록 아카이빙 파일 시스템의 호스트 구성

활성 메타데이터 서버와 각 잠재적 메타데이터 서버 및 datamover 클라이언트에 대해 다음과 같이 하십시오.

1. 호스트에 *root*로 로그인합니다.

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

2. */etc/opt/SUNWsamfs/mcf* 파일을 텍스트 편집기에서 엽니다.

   예제에서는 *vi* 편집기를 사용합니다.

```
[sharefs-host]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
sharefs                  100       ms        sharefs on
 /dev/dsk/c1t3d0s3        101       md        sharefs on
 /dev/dsk/c1t3d0s4        102       md        sharefs on
...
```

3. */etc/opt/SUNWsamfs/mcf* 파일의 파일 시스템 정의에 따라 아카이브 스토리지 장비 섹션을 시작합니다.

   예제에서는 명확성을 위해 일부 머리글을 추가합니다.

```
[sharefs-host]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
...
# Archival storage for copies:
#
# Equipment              Equipment Equipment Family    Device Additional
# Identifier             Ordinal   Type      Set       State  Parameters
#----------------------- --------- --------- --------- ------ ----------------
```

4. 아카이브 테이프 스토리지를 추가하려면 라이브러리에 대한 항목을 추가하는 것부터 시작합니다. 장비 식별자 필드에 라이브러리에 대한 장치 ID를 입력하고 장비 순서 번호를 지정합니다.

   이 예제에서는 라이브러리 장비 식별자가 */dev/scsi/changer/c1t0d5*입니다. 디스크 아카이브에 대한 선택한 범위 다음에 장비 순서 번호를 *900*으로 설정합니다.

```
# Archival storage for copies:
#
# Equipment              Equipment Equipment Family    Device Additional
# Identifier             Ordinal   Type      Set       State  Parameters
#----------------------- --------- --------- --------- ------ ----------------
/dev/scsi/changer/c1t0d5 900
```

5. 장비 유형을 일반 SCSI 연결 테이프 라이브러리인 *rb*로 설정하고 테이프 라이브러리 패밀리 세트의 이름을 제공한 다음 장치 상태를 *on*으로 설정합니다.

   이 예제에서는 *library1* 라이브러리를 사용합니다.

```
# Archival storage for copies:
#
# Equipment              Equipment Equipment Family    Device Additional
# Identifier             Ordinal   Type      Set       State  Parameters
#----------------------- --------- --------- --------- ------ ----------------
/dev/scsi/changer/c1t0d5 900       rb        library1  on
```

6. *Additional Parameters* 열에서 라이브러리 카탈로그에 대한 선택적 사용자 정의 경로와 이름을 입력할 수 있습니다.

   기본 경로가 아닌 선택적 경로는 127자를 초과할 수 없습니다. 예제에서는 기본 경로인 */var/opt/SUNWsamfs/catalog/*를 사용자 정의 카탈로그 파일 이름인 *library1cat*와 함께 사용합니다. 문서 레이아웃 제한 때문에 예제에서는 다음과 같이 경로를 축약합니다.

```
# Archival storage for copies:
#
# Equipment              Equipment Equipment Family    Device Additional
# Identifier             Ordinal   Type      Set       State  Parameters
#----------------------- --------- --------- --------- ------ ----------------
/dev/scsi/changer/c1t0d5 900       rb        library1  on     .../library1cat
```

7. 다음에는 각 테이프 드라이브에 대한 항목을 추가합니다. "지속 바인딩을 사용하여 서버 및 Datamover 호스트에 테이프 드라이브 연결 " 절차에서 설정한 지속 장비 식별자를 사용합니다.

```
# Archival storage for copies:
#
# Equipment              Equipment Equipment Family    Device Additional
# Identifier             Ordinal   Type      Set       State  Parameters
#----------------------- --------- --------- --------  ------ -----------------
DISKVOL1                  800       ms       DISKVOL1  on
 /dev/dsk/c6t0d1s7         801       md       DISKVOL1  on
 /dev/dsk/c4t0d2s7         802       md       DISKVOL1  on
/dev/scsi/changer/c1t0d5  900       rb       library1  on     .../library1cat
 /dev/rmt/60cbn            901       tp       library1 on
 /dev/rmt/61cbn            902       tp       library1  on
 /dev/rmt/62cbn            903       tp       library1  on
 /dev/rmt/63cbn            904       tp       library1  on
```

8. 끝으로, Oracle HSM 내역기를 직접 구성하려는 경우 장비 유형 *hy*를 사용하여 항목을 추가합니다. family-set 및 device-state 열에 하이픈을 입력하고 additional-parameters 열에 내역기 카탈로그 경로를 입력합니다.

   내역기는 아카이브에서 내보낸 볼륨을 카탈로그화하는 가상 라이브러리입니다. 내역기를 구성하지 않을 경우 소프트웨어는 지정된 가장 높은 장비 순서 번호에 1을 더하여 내역기를 자동으로 만듭니다.

   예제에서는 page-layout 이유에 대한 내역기 카탈로그 경로를 약어로 표시합니다. 전체 경로는 */var/opt/SUNWsamfs/catalog/historian_cat*입니다.

```
# Archival storage for copies:
#
# Equipment              Equipment Equipment Family    Device Additional
# Identifier             Ordinal   Type      Set       State  Parameters
#----------------------- --------- --------- --------- ------ ----------------
/dev/scsi/changer/c1t0d5 900       rb        library1  on     ...catalog/library1cat
 /dev/rmt/60cbn            901       tp        library1  on
 /dev/rmt/61cbn            902       tp        library1  on
 /dev/rmt/62cbn            903       tp        library1  on
 /dev/rmt/63cbn            904       tp        library1  on
historian                 999       hy        -         -      .../historian_cat
```

9. *mcf* 파일을 저장하고 편집기를 닫습니다.

```
...
 /dev/rmt/3cbn             904       tp        library1  on
historian                 999       hy        -       -      .../historian_cat
:wq
[sharefs-host]root@solaris:~#
```

10. *sam-fsd* 명령을 실행하여 *mcf* 파일에서 오류를 확인합니다. 발견된 모든 오류를 수정합니다.

    sam-fsd 명령은 Oracle HSM 구성 파일을 읽고 파일 시스템을 초기화합니다. 오류가 발견되면 실행을 중지합니다.

```
[sharefs-host]root@solaris:~# sam-fsd
...
Would start sam-stagealld()
Would start sam-stagerd()
Would start sam-amld()
[sharefs-host]root@solaris:~#
```

11. Oracle HSM 서비스에 *mcf* 파일을 다시 읽고 그에 따라 재구성하도록 지시합니다. 보고된 모든 오류를 수정하고 필요에 따라 반복합니다.

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

12. 모든 활성 및 잠재적 메타데이터 서버와 모든 datamover 클라이언트가 아카이브 스토리지를 사용하도록 구성될 때까지 이 절차를 반복합니다.
13. 필요한 경우 공유 아카이브 파일 시스템의 호스트 전체로 테이프 I/O 분산을 수행합니다.
14. 사이드밴드 데이터베이스 기능을 사용할 계획인 경우 10장. 보고 데이터베이스 구성을 참조하십시오.
15. 그렇지 않은 경우 11장. 알림 및 로깅 구성으로 이동합니다.

공유 아카이브 파일 시스템의 호스트 전체로 테이프 I/O 분산

Oracle HSM 릴리스 6.1부터 Oracle Solaris 11 이상에서 실행되는 공유 아카이빙 파일 시스템의 클라이언트에 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).

공유 파일 시스템 호스트 전체로 테이프 I/O를 분산시키려면 다음과 같이 하십시오.

1. 분산된 I/O에 사용될 모든 장치를 파일 시스템 메타데이터 서버와 테이프 I/O를 처리할 모든 파일 시스템 클라이언트에 연결합니다.
2. 아직 수행하지 않은 경우 지속 바인딩을 사용하여 Datamover로 동작할 각 클라이언트에 테이프 드라이브 연결을 수행합니다. 그런 다음 여기로 돌아옵니다.

3. 공유 아카이빙 파일 시스템의 메타데이터 서버에 *root*로 로그인합니다.

   예제에서 서버의 호스트 이름은 *samsharefs-mds*입니다.

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

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

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

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

   예제에서는 *ssh*를 사용하여 원격으로 클라이언트 호스트 *samsharefs-client1* 및 *samsharefs-client2*에 로그인하고 로그인 배너에서 Solaris 버전을 얻습니다.

```
[samsharefs-mds]root@solaris:~# ssh root@samsharefs-client1
Password:
Oracle Corporation      SunOS 5.11      11.1    September 2013
[samsharefs-client1]root@solaris:~# exit
[samsharefs-mds]root@solaris:~# ssh root@samsharefs-client2
Password:
Oracle Corporation      SunOS 5.11      11.1    September 2013
[samsharefs-client2]root@solaris:~# exit
[samsharefs-mds]root@solaris:~#
```

6. 분산된 I/O 구성에서 각 테이프 드라이브에 대한 버퍼 공간으로 할당할 수 있는 시스템 메모리의 용량을 계산합니다. 총 사용 가능한 메모리를 드라이브 수로 나누고 안심할 수 있는 여유분을 뺍니다.

   ```
   (total-memory bytes)/(drive-count drives) = memory bytes/drive
   (memory bytes/drive) - (safe-margin bytes/drive) = buffsize bytes/drive
   ```

   Oracle HSM에서는 사용되는 각 드라이브에 대해 버퍼를 할당합니다. 따라서 시스템 메모리가 제공할 수 있는 것보다 많은 버퍼 공간을 실수로 구성하지 않도록 하십시오. 예제에서는 드라이브당 224KB 이하를 할당할 수 있음을 알 수 있습니다. 따라서 안심할 수 있는 여유분을 확보하기 위해 128로 줄입니다.

   ```
   ((3584 kilobytes)/(16 drives)) = 224 kilobytes/drive
   buffsize = 128 kilobytes/drive
   ```

7. 각 드라이브에 할당할 수 있는 버퍼의 크기를 계산했으면 지정된 크기의 버퍼에 맞는 Oracle HSM 장치 블록 크기 및 블록 수를 계산합니다.

   ```
   (number blocks/buffer)*block-size bytes/block/drive = buffersize bytes/drive
   ```

   블록 수와 블록 크기의 곱이 계산된 버퍼 크기보다 작거나 같을 때까지 조정합니다. 블록 수는 *[2-8192]* 범위에 있어야 합니다. 예제에서는 버퍼당 각각 64KB의 블록 2개로 결정합니다.

   ```
   (2 blocks/buffer)*(64 kilobytes/block/drive) = 128 kilobytes/drive
   ```

8. 메타데이터 서버에서 */etc/opt/SUNWsamfs/archiver.cmd* 파일을 텍스트 편집기로 엽니다. 파일의 위쪽에 있는 일반 지시어 섹션의 새 라인에서 *bufsize = media-type media-blocks*를 입력합니다. 여기서,
   - *media-type*은 *mcf* 파일이 분산된 I/O에 사용되는 드라이브 및 매체에 지정하는 유형 코드입니다.
   - *media-blocks*는 위에서 계산한 버퍼당 블록 수입니다.

   파일을 저장하고 편집기를 닫습니다.

   예제에서는 *samsharefs-mds* 서버에 로그인하고 *vi* 편집기를 사용하여 *bufsize = ti 2* 라인을 추가합니다. 여기서 *ti*는 사용 중인 Oracle StorageTek T10000 드라이브에 대한 매체 유형이고 *2*는 계산된 드라이브 버퍼당 블록 수입니다.

```
[samsharefs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/archiver.cmd
# archiver.cmd
#----------------------------------------------------------------------
# General Directives
archivemeta = off
examine = noscan
bufsize = ti 2
:wq
[samsharefs-mds]root@solaris:~#
```

9. 메타데이터 서버에서 */etc/opt/SUNWsamfs/defaults.conf* 파일을 텍스트 편집기로 엽니다. 분산된 I/O에 참여할 각 매체 유형에 대해 *media-type_blksize =size* 형식의 라인을 입력합니다. 여기서,
   - *media-type*은 *mcf* 파일이 분산된 I/O에 사용되는 드라이브 및 매체에 지정하는 유형 코드입니다.
   - *size*는 이 절차의 앞에서 계산한 블록 크기입니다.

   기본적으로 StorageTek T10000 드라이브에 대한 장치 블록 크기는 2MB 또는 2048KB(*ti_blksize = 2048*)입니다. 따라서 예제에서는 기본값을 계산된 블록 크기인 64KB로 대체합니다.

```
[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.
...
#li_blksize = 256
ti_blksize = 64
[samsharefs-mds]root@solaris:~#
```

10. */etc/opt/SUNWsamfs/defaults.conf* 파일에서는 필요한 경우 *#distio = off* 라인의 주석 처리를 해제하거나 없는 경우 이 라인을 추가합니다.

    기본적으로 *distio*는 *off*(사용 안함)입니다. 예제에서는 *distio = on* 라인을 추가합니다.

```
...
distio = on
```

11. */etc/opt/SUNWsamfs/defaults.conf* 파일에서는 분산된 I/O에 참여해야 하는 각 장치 유형을 사용으로 설정합니다. 새 라인에서 *media-type _distio = on*을 입력합니다. 여기서 *media-type*은 *mcf* 파일이 드라이브 및 매체에 지정하는 유형 코드입니다.

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

```
...
distio = on
ti_distio = on
```

12. */etc/opt/SUNWsamfs/defaults.conf* 파일에서는 분산된 I/O에 참여하지 않아야 하는 각 장치 유형을 사용 안함으로 설정합니다. 새 라인에서 *media-type _distio = off*를 입력합니다. 여기서 *media-type*은 *mcf* 파일이 드라이브 및 매체에 지정하는 유형 코드입니다.

    예제에서는 LTO 드라이브를 제외합니다.

```
...
distio = on
ti_distio = on
li_distio = off
```

13. */etc/opt/SUNWsamfs/defaults.conf* 파일 편집을 마쳤으면 파일 내용을 저장하고 편집기를 닫습니다.

```
...
distio = on
ti_distio = on
li_distio = off
:wq
[samsharefs-mds]root@solaris:~#
```

14. datamover로 작동할 각 클라이언트에서 *defaults.conf* 파일을 편집하여 서버의 파일과 일치하도록 합니다.
15. datamover 역할을 할 각 클라이언트에서 */etc/opt/SUNWsamfs/mcf* 파일을 텍스트 편집기에서 열고 메타데이터 서버에서 분산된 테이프 I/O에 사용 중인 모든 테이프 장치를 포함하도록 파일을 업데이트합니다. 장치 순서 및 장비 번호가 메타데이터 서버의 *mcf* 파일과 동일한지 확인합니다.

    예제에서는 *vi* 편집기를 사용하여 *samsharefs-client1* 호스트에서 *mcf* 파일을 구성합니다.

```
[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
```

16. 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 *samsharefs-client1* 호스트에서 구성되었으므로, 테이프 장치에 대해 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     .../library1cat
 /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:~#
```

17. 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. 그런 다음 파일을 저장하고 편집기를 닫습니다.

    예제에서는 라이브러리가 *samsharefs-client2* 호스트에 구성되어 있지 않으므로, 테이프 장치에 대한 패밀리 세트 이름으로 하이픈을 사용합니다.

```
[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:~#
```

18. 특정 아카이브 세트 복사본에 대해 분산된 테이프 I/O를 사용 또는 사용 안함으로 설정해야 하는 경우 텍스트 편집기에서 */etc/opt/SUNWsamfs/archiver.cmd* 파일을 열고 copy 지시어에 *-distio* 매개변수를 추가합니다. 분산된 I/O를 사용으로 설정하려면 *-distio on*을 설정하고, 사용 안함으로 설정하려면 *-distio off*로 설정합니다. 파일을 저장합니다.

    예제에서는 *samsharefs-mds* 서버에 로그인하고 *vi* 편집기를 사용하여 복사본 *1*에 대해 분산된 I/O *off*를 설정합니다.

```
[samsharefs-mds]root@solaris:~# vi /etc/opt/SUNWsamfs/archiver.cmd
# archiver.cmd
...
params
allsets -sort path -offline_copy stageahead
```

```
allfiles.1 -startage 10m -startsize 500M -startcount 500000 -distio off
allfiles.2 -startage 24h -startsize 20G  -startcount 500000 -reserve set
:wq
[samsharefs-mds]root@solaris:~#
```

19. *sam-fsd* 명령을 실행하여 구성 파일에 오류가 있는지 확인합니다. 발견된 모든 오류를 수정합니다.

    sam-fsd 명령은 Oracle HSM 구성 파일을 읽고 파일 시스템을 초기화합니다. 만일 오류가 발생하면 중지됩니다. 예제에서는 *sharefs-mds* 서버에서 명령을 실행합니다.

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

20. Oracle HSM 서비스에 수정된 구성 파일을 읽고 그에 따라 재구성하도록 지시합니다. 보고된 모든 오류를 수정하고 필요에 따라 반복합니다.

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

21. 분산된 I/O가 성공적으로 활성화되었는지 확인하려면 *samcmd g* 명령을 사용합니다. *DATAMOVER* 플래그가 클라이언트에 대한 출력에 표시되면 분산된 I/O가 성공적으로 활성화된 것입니다.

    예제에서는 플래그가 표시됩니다.

```
[samsharefs-mds]root@solaris:~# samcmd g
Shared clients samcmd 6.0.dist_tapeio 11:09:13 Feb 20 2014
samcmd on samsharefs-mds
samsharefs is shared, server is samsharefs-mds, 2 clients 3 max
ord hostname             seqno nomsgs status   config  conf1  flags
  1 samsharefs-mds          14      0   8091  808540d   4051      0 MNT SVR

    config   :   CDEVID      ARCHIVE_SCAN    GFSID   OLD_ARCHIVE_FMT
    "        :   SYNC_META   TRACE   SAM_ENABLED     SHARED_MO
    config1  :   NFSV4_ACL   MD_DEVICES      SMALL_DAUS      SHARED_FS
    flags    :
    status   :   MOUNTED     SERVER  SAM     DATAMOVER
    last_msg :  Wed Jul  2 10:13:50 2014

  2 samsharefs-client1     127      0   a0a1  808540d   4041      0 MNT CLI

    config   :   CDEVID      ARCHIVE_SCAN    GFSID   OLD_ARCHIVE_FMT
    "        :   SYNC_META   TRACE   SAM_ENABLED     SHARED_MO
    config1  :   NFSV4_ACL   MD_DEVICES      SHARED_FS
    flags    :
    status   :   MOUNTED     CLIENT  SAM     SRVR_BYTEREV
```

```
"          :    DATAMOVER
...
```

22. 사이드밴드 데이터베이스 기능을 사용할 계획인 경우 10장. 보고 데이터베이스 구성을 참조하십시오.
23. 그렇지 않은 경우 11장. 알림 및 로깅 구성으로 이동합니다.

NFS 및 SMB/CIFS를 사용하여 여러 호스트에서 파일 시스템 액세스

네트워크 파일 시스템(NFS) 또는 SMB(Server Message Block)/CIFS(Common Internet File System) 공유를 multiple-host file-system 액세스에 대한 Oracle HSM 소프트웨어의 고유 지원 대신 또는 고유 지원과 함께 사용하여 여러 호스트에서 Oracle HSM 파일 시스템에 액세스할 수 있습니다("Oracle HSM 소프트웨어를 사용하여 여러 호스트에서 파일 시스템 액세스" 참조). 다음 절에서는 기본 구성 단계를 간략히 설명합니다.

- NFS를 사용하여 Oracle HSM 파일 시스템 공유
- SMB/CIFS를 사용하여 Oracle HSM 파일 시스템 공유

NFS를 사용하여 Oracle HSM 파일 시스템 공유

다음 작업을 수행합니다.

- NFS 4를 사용하여 Oracle HSM 공유 파일 시스템을 공유하기 전에 위임을 사용 안함으로 설정
- WORM 파일과 디렉토리를 공유하도록 NFS 서버 및 클라이언트 구성(필요한 경우)
- Oracle HSM 호스트에서 NFS 서버 구성
- NFS 공유로 Oracle HSM 파일 시스템 공유
- NFS 클라이언트에서 NFS 공유 Oracle HSM 파일 시스템 마운트

NFS 4를 사용하여 Oracle HSM 공유 파일 시스템을 공유하기 전에 위임을 사용 안함으로 설정

NFS를 사용하여 Oracle HSM 공유 파일 시스템을 공유할 경우 Oracle HSM 소프트웨어에서 NFS와 간섭하지 않고 파일에 대한 액세스를 제어하는지 확인해야 합니다. NFS 서버에서 클라이언트를 대신하여 파일에 액세스하는 경우에는 Oracle HSM 공유 파일 시스템의 클라이언트로 액세스하기 때문에 이는 일반적으로 문제가 되지 않습니다. 하지만 클라이언트에 대한 읽기 및 쓰기 권한에 대한 제어를 위임하도록 NFS 버전 4 서버를 구성한 경우에는 문제가 발생할 수 있습니다. 서버는 잠재적 충돌을 차단하기 위해서만 작업에 영향을 주면 되므로 위임은 매력적인 기능입니다. 서버의 작업 로드가 NFS 클라이언트 전체에 부분적으로 분산되므로 네트워크 트래픽이 감소합니다. 하지만 위임은 Oracle HSM 서버와 별개로 액세스 권한(특히, 쓰기 권한)을 부여하며, 자체 공유 파일 시스템 클라이언트에서의 액세스도 제어합니다. 충돌과 잠재적 파일 손상을 방지하려면 위임을 사용 안함으로 설정해야 합니다. 다음과 같이 하십시오.

1. NFS 공유로 구성하려는 Oracle HSM 파일 시스템의 호스트에 로그인합니다. *root*로 로그인합니다.

파일 시스템이 Oracle HSM 공유 파일 시스템인 경우 파일 시스템에 대한 메타데이터 서버에 로그인합니다. 아래 예제에서는 서버 이름이 *qfsnfs*입니다.

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

2. NFS 버전 4를 사용 중이고 NFS 서버에서 Solaris 11.1 이상을 실행하는 경우 SMF(서비스 관리 기능)의 *sharectl set -p* 명령을 사용하여 NFS *server_delegation* 등록 정보를 *off*로 설정합니다.

```
[qfsnfs]root@solaris:~# sharectl set -p server_delegation=off
```

3. NFS 버전 4를 사용하고 있고 NFS 서버에서 Solaris 11.0 이전 버전을 실행하는 경우 텍스트 편집기에서 */etc/default/nfs* 파일을 열고 *NFS_SERVER_DELEGATION* 매개변수를 *off*로 설정하여 위임을 사용 안함으로 설정합니다. 파일을 저장하고 편집기를 닫습니다.

   이 예에서는 *vi* 편집기를 사용합니다.

```
[qfsnfs]root@solaris:~# vi /etc/default/nfs
# ident "@(#)nfs        1.10    04/09/01 SMI"
# Copyright 2004 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
...
NFS_SERVER_DELEGATION=off
:wq
[qfsnfs]root@solaris:~#
```

4. 공유하려는 Oracle HSM 파일 시스템에서 WORM(Write-Once Read-Many) 기능을 지원하는 경우 지금 WORM 파일과 디렉토리를 공유하도록 NFS 서버 및 클라이언트 구성을 수행합니다.
5. 그렇지 않은 경우 Oracle HSM 호스트에서 NFS 서버 구성을 수행합니다.

WORM 파일과 디렉토리를 공유하도록 NFS 서버 및 클라이언트 구성

1. NFS를 사용하여 공유할 Oracle HSM 파일 시스템의 호스트에 로그인합니다. *root*로 로그인합니다.

   파일 시스템이 Oracle HSM 공유 파일 시스템인 경우 파일 시스템에 대한 메타데이터 서버에 로그인합니다. 아래 예제에서 서버 이름은 *qfsnfs*이고 클라이언트 이름은 *nfsclient1*입니다.

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

2. 공유하려는 Oracle HSM 파일 시스템이 WORM 기능을 사용하고 Oracle Solaris 10 이상에서 실행 중인 서버에 호스팅된 경우 NFS 서버와 모든 클라이언트에서 NFS 버전 4가 사용으로 설정되어 있는지 확인합니다.

   예제에서는 *qfsnfs* 서버와 *nfsclient1* 클라이언트를 확인합니다. 각 경우에 먼저 *uname -r* 명령을 사용하여 Solaris 버전 레벨을 확인합니다. 그런 다음 *modinfo* 명령의 출력을 NFS 버전 정보를 찾는 정규 표현식 및 *grep*에 연결합니다.

```
[qfsnfs]root@solaris:~# uname -r
5.11
[qfsnfs]root@solaris:~# modinfo | grep -i "nfs.* version 4"
258 7a600000  86cd0  28   1  nfs (network filesystem version 4)
[qfsnfs]root@solaris:~# ssh root@nfsclient1
Pasword: ...
[nfsclient1]root@solaris:~# uname -r
5.11
[nfsclient1]root@solaris:~# modinfo | grep -i "nfs.* version 4"
278 fffffffff8cba000  9df68  27   1  nfs (network filesystem version 4)
[nfsclient1]root@solaris:~# exit
[qfsnfs]root@solaris:~#
```

3. Oracle Solaris 10 이상에서 실행 중인 서버에서 NFS 버전 4를 사용으로 설정하지 않은 경우 서버와 각 클라이언트에 *root*로 로그인합니다. 그런 다음 *sharectl set* 명령을 사용하여 NFS 4를 사용으로 설정합니다.

```
[qfsnfs]root@solaris:~# sharectl set -p server_versmax=4 nfs
[qfsnfs]root@solaris:~# ssh root@nfsclient1
Password ...
[nfsclient1]root@solaris:~# sharectl set -p server_versmax=4 nfs
[nfsclient1]root@solaris:~# exit
[qfsnfs]root@solaris:~#
```

4. 이제 Oracle HSM 호스트에서 NFS 서버 구성을 수행합니다.

Oracle HSM 호스트에서 NFS 서버 구성

클라이언트에서 NFS(네트워크 파일 시스템)를 사용하여 Oracle HSM 파일 시스템을 성공적으로 마운트하려면 파일 시스템이 호스트에 성공적으로 마운트되기 전에 Oracle HSM 파일 시스템을 공유하려고 시도하지 않도록 NFS 서버를 구성해야 합니다. Oracle Solaris 10 이상 버전의 운영체제에서는 SMF(서비스 관리 기능)에서 부트 시 파일 시스템 마운트를 관리합니다. 아래 절차를 사용하여 NFS를 구성하지 않은 경우 QFS 마운트 또는 NFS 공유 중 하나는 성공하고 하나는 실패합니다.

1. NFS 공유로 구성하려는 Oracle HSM 파일 시스템의 호스트에 로그인합니다. *root*로 로그인합니다.

파일 시스템이 Oracle HSM 공유 파일 시스템인 경우 파일 시스템에 대한 메타데이터 서버에 로그인합니다. 아래 예제에서는 서버 이름이 *qfsnfs*입니다.

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

2. *svccfg export /network/nfs/server* 명령의 출력을 재지정하여 기존 NFS 구성을 XML 매니페스트 파일로 내보냅니다.

   예제에서는 내보낸 구성을 매니페스트 파일 */var/tmp/server.xml*로 보냅니다.

```
[qfsnfs]root@solaris:~# svccfg export /network/nfs/server > /var/tmp/server.xml
[qfsnfs]root@solaris:~#
```

3. 텍스트 편집기에서 매니페스트 파일을 열고 *filesystem-local* 종속성을 찾습니다.

   예제에서는 *vi* 편집기에서 파일을 엽니다. *filesystem-local* 종속성에 대한 항목은 종속 *nfs-server_multi-user-server*에 종속 항목 바로 앞에 나열됩니다.

```
[qfsnfs]root@solaris:~# vi /var/tmp/server.xml
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM '/usr/share/lib/xml/dtd/service_bundle.dtd.1'>
<service_bundle type='manifest' name='export'>
  <service name='network/nfs/server' type='service' version='0'>
    ...
    <dependency name='filesystem-local' grouping='require_all' restart_on='error' type='service'>
      <service_fmri value='svc:/system/filesystem/local'/>
    </dependency>
    <dependent name='nfs-server_multi-user-server' restart_on='none'
        grouping='optional_all'>
      <service_fmri value='svc:/milestone/multi-user-server'/>
    </dependent>
    ...
```

4. *filesystem-local* 종속성 바로 뒤에 QFS 공유 파일 시스템을 마운트하는 *qfs* 종속성을 추가합니다. 그런 다음 파일을 저장하고 편집기를 종료합니다.

   그러면 서버에서 NFS를 통해 공유하려고 시도하기 전에 Oracle HSM 공유 파일 시스템이 마운트됩니다.

```
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM '/usr/share/lib/xml/dtd/service_bundle.dtd.1'>
<service_bundle type='manifest' name='export'>
  <service name='network/nfs/server' type='service' version='0'>
    ...
    <dependency name='filesystem-local' grouping='require_all' restart_on='error' type='service'>
```

```
      <service_fmri value='svc:/system/filesystem/local'/>
    </dependency>
    <dependency name='qfs' grouping='require_all' restart_on='error' type='service'>
      <service_fmri value='svc:/network/qfs/shared-mount:default'/>
    </dependency>
    <dependent name='nfs-server_multi-user-server' restart_on='none'
        grouping='optional_all'>
      <service_fmri value='svc:/milestone/multi-user-server'/>
    </dependent>
:wq
[qfsnfs]root@solaris:~#
```

5. *svccfg validate* 명령을 사용하여 매니페스트 파일을 검증합니다.

```
[qfsnfs]root@solaris:~# svccfg validate /var/tmp/server.xml
```

6. *svccfg validate* 명령이 오류를 보고하면 오류를 수정하고 파일을 다시 검증합니다.

   예제에서 *svccfg validate* 명령은 XML 구문 분석 오류를 반환합니다. 파일을 저장할 때 실수로 종료 태그 *</dependency>*를 누락했습니다. 따라서 *vi* 편집기에서 파일을 다시 열고 문제를 해결합니다.

```
[qfsnfs]root@solaris:~# svccfg validate /var/tmp/server.xml
/var/tmp/server.xml:75: parser error : Opening and ending tag mismatch: dependency line 29 and
 service
  </service>
            ^
/var/tmp/server.xml:76: parser error : expected '>'
</service_bundle>
          ^
/var/tmp/server.xml:77: parser error : Premature end of data in tag service_bundle line 3
^
svccfg: couldn't parse document
[qfsnfs]root@solaris:~# vi /var/tmp/server.xml
...
:wq
[qfsnfs]root@solaris:~#
```

7. *svccfg validate* 명령이 오류 없이 완료되는 경우 *svcadm disable nfs/server* 명령을 사용하여 NFS를 사용 안함으로 설정합니다.

   예제에서는 *svccfg validate* 명령이 출력을 반환하지 않았으므로, 파일이 유효하고 NFS를 사용 안함으로 설정할 수 있습니다.

```
[qfsnfs]root@solaris:~# svccfg validate /var/tmp/server.xml
```

```
[qfsnfs]root@solaris:~# svcadm disable nfs/server
```

8. *svccfg delete nfs/server* 명령을 사용하여 기존 NFS 서버 구성을 삭제합니다.

```
[qfsnfs]root@solaris:~# svccfg delete nfs/server
```

9. *svccfg import* 명령을 사용하여 매니페스트 파일을 SMF(서비스 관리 기능)로 가져옵니다.

```
[qfsnfs]root@solaris:~# svccfg import /var/tmp/server.xml
```

10. *svcadm enable nfs/server* 명령을 사용하여 NFS를 다시 사용으로 설정합니다.

    NFS가 업데이트된 구성을 사용하도록 구성됩니다.

```
[qfsnfs]root@solaris:~# svcadm enable nfs/server
```

11. *qfs* 종속성이 적용되었는지 확인합니다. *svcs -d svc:/network/nfs/server:default* 명령이 */network/qfs/shared-mount:default* 서비스를 표시하는지 확인합니다.

```
[qfsnfs]root@solaris:~# svcs -d svc:/network/nfs/server:default
STATE          STIME    FMRI
...
online         Nov_01   svc:/network/qfs/shared-mount:default
...
```

12. 이제 NFS 공유로 Oracle HSM 파일 시스템 공유를 수행합니다.

NFS 공유로 Oracle HSM 파일 시스템 공유

Oracle Solaris 운영체제 버전의 관리 설명서에 설명된 절차를 사용하여 Oracle HSM 파일 시스템을 공유합니다. 아래 단계에서는 Solaris 11.1에 대한 절차를 요약합니다.

1. NFS를 사용하여 공유할 Oracle HSM 파일 시스템의 호스트에 로그인합니다. *root*로 로그인합니다.

   파일 시스템이 Oracle HSM 공유 파일 시스템인 경우 파일 시스템에 대한 메타데이터 서버에 로그인합니다. 아래 예제에서는 서버 이름이 *qfsnfs*입니다.

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

2. *share -F nfs -o sharing-options sharepath* 명령줄을 입력합니다. 여기서 *-F* 스위치는 *nfs* 공유 프로토콜을 지정하고 *sharepath*는 공유 리소스의 경로입니다. 선택적 *-o* 매개변수를 사용할 경우 *sharing-options*는 다음 중 하나를 포함할 수 있습니다.

- *rw*는 모든 클라이언트에서 읽기 및 쓰기 권한으로 *sharepath*를 사용할 수 있도록 지정합니다.
- *ro*는 모든 클라이언트에서 읽기 전용 권한으로 *sharepath*를 사용할 수 있도록 지정합니다.
- *rw=clients*는 공유에 액세스할 수 있는 하나 이상의 클라이언트를 콜론으로 구분한 목록인 *clients*에서 읽기 및 쓰기 권한으로 *sharepath*를 사용할 수 있도록 지정합니다.
- *ro=clients*는 공유에 액세스할 수 있는 하나 이상의 클라이언트를 콜론으로 구분한 목록인 *clients*에서 읽기 전용 권한으로 *sharepath*를 사용할 수 있도록 지정합니다.

예제에서는 */qfsms* 파일 시스템 읽기/쓰기를 *nfsclient1* 및 *nfsclient2* 클라이언트와 공유하고 읽기 전용을 *nfsclient3*과 공유합니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시로 이스케이프됨).

```
[qfsnfs]root@solaris:~# share -F nfs -o rw=nfsclient1:nfsclient2 /
 ro=nfsclient3 /qfsms
```

명령을 입력하면 시스템에서 NFS 서버 데몬 *nfsd*를 자동으로 다시 시작합니다. 추가 옵션과 자세한 내용은 *share_nfs* 매뉴얼 페이지를 참조하십시오.

3. *share -F nfs* 명령줄을 사용하여 공유 매개변수를 확인합니다.

   예제에서 명령 출력은 공유를 올바르게 구성했음을 보여 줍니다.

```
[qfsnfs]root@solaris:~# share -F nfs
/qfsms   sec=sys,rw=nfsclient1:nfsclient2,ro=nfsclient3
[qfsnfs]root@solaris:~#
```

4. 이제 NFS 클라이언트에서 NFS 공유 Oracle HSM 파일 시스템 마운트를 수행합니다.

NFS 클라이언트에서 NFS 공유 Oracle HSM 파일 시스템 마운트

클라이언트 시스템의 편리한 마운트 지점에 NFS 서버의 파일 시스템을 마운트합니다. 각 클라이언트에 대해 다음과 같이 하십시오.

1. 클라이언트에 *root*로 로그인합니다.

   예제에서 NFS 클라이언트의 이름은 *nfsclient1*로 지정됩니다.

```
[nfsclient1]root@solaris:~#
```

2. 운영체제의 */etc/vfstab* 파일을 백업합니다.

```
[nfsclient1]root@solaris:~# cp /etc/vfstab /etc/vfstab.backup
```

```
[nfsclient1]root@solaris:~#
```

3. 텍스트 편집기에서 */etc/vfstab* 파일을 엽니다.

   예제에서는 *vi* 편집기를 사용합니다.

```
[nfsclient1]root@solaris:~# vi /etc/vfstab
#File         Device                         Mount
#Device       to      Mount     System fsck  at     Mount
#to Mount     fsck    Point     Type   Pass  Boot   Options
#------------ ------  --------- ------ ----  -----  ----------------
/devices      -       /devices  devfs  -     no     -
...
```

4. */etc/vfstab* 파일의 첫번째 열에서 NFS 서버의 이름과 공유할 파일 시스템의 마운트 지점을 콜론으로 구분하여 지정하여 파일 장치의 이름을 지정합니다.

   예제에서 NFS 서버의 이름은 *qfsnfs*로 지정되고, 공유 파일 시스템의 이름은 *qfsms*로 지정되고, 서버의 마운트 지점은 */qfsms*입니다.

```
#File         Device                         Mount
#Device       to      Mount     System fsck  at     Mount
#to Mount     fsck    Point     Type   Pass  Boot   Options
#------------ ------  --------- ------ ----  -----  ----------------
/devices      -       /devices  devfs  -     no     -
...
qfsnfs:/qfsms
```

5. 로컬 시스템에서 원격 파일 시스템의 일관성 검사를 시도하지 않도록 */etc/vfstab* 파일의 두번째 열에 하이픈(-)을 입력합니다.

```
#File         Device                         Mount
#Device       to      Mount     System fsck  at     Mount
#to Mount     fsck    Point     Type   Pass  Boot   Options
#------------ ------  --------- ------ ----  -----  ----------------
/devices      -       /devices  devfs  -     no     -
...
qfsnfs:/qfsms -
```

6. */etc/vfstab* 파일의 세번째 열에 원격 파일 시스템을 마운트할 로컬 마운트 지점을 입력합니다.

   예제에서 마운트 지점은 */qfsnfs* 디렉토리입니다.

```
#File         Device                         Mount
```

```
#Device       to      Mount     System fsck  at     Mount
#to Mount     fsck    Point     Type   Pass  Boot   Options
#------------ ------  --------- ------ ----  -----  ---------------
/devices      -       /devices  devfs  -     no     -
...
qfsnfs:/qfsms -       /qfsnfs
```

7. */etc/vfstab* 파일의 네번째 열에 파일 시스템 유형 *nfs*를 입력합니다.

```
#File         Device                          Mount
#Device       to      Mount     System fsck  at     Mount
#to Mount     fsck    Point     Type   Pass  Boot   Options
#------------ ------  --------- ------ ----  -----  ---------------
/devices      -       /devices  devfs  -     no     -
...
qfsnfs:/qfsms -       /qfsnfs   nfs
```

   클라이언트에서는 원격 QFS 파일 시스템을 NFS 파일 시스템으로 마운트하므로 *nfs* 파일 시스템 유형을 사용합니다.

8. 로컬 시스템에서는 원격 파일 시스템의 일관성을 검사하지 않으므로, */etc/vfstab* 파일의 다섯번째 열에 하이픈(-)을 입력합니다.

```
#File         Device                          Mount
#Device       to      Mount     System fsck  at     Mount
#to Mount     fsck    Point     Type   Pass  Boot   Options
#------------ ------  --------- ------ ----  -----  ---------------
/devices      -       /devices  devfs  -     no     -
...
qfsnfs:/qfsms -       /qfsnfs   nfs    -
```

9. */etc/vfstab* 파일의 여섯번째 열에는 부트 시 원격 파일 시스템을 마운트하려면 *yes*를 입력하고, 요구 시 수동으로 마운트하려면 *no*를 입력합니다.

   예제에서는 *yes*를 입력합니다.

```
#File         Device                          Mount
#Device       to      Mount     System fsck  at     Mount
#to Mount     fsck    Point     Type   Pass  Boot   Options
#------------ ------  --------- ------ ----  -----  ---------------
/devices      -       /devices  devfs  -     no     -
...
qfsnfs:/qfsms -       /qfsnfs   nfs    -     yes
```

10. */etc/vfstab* 파일의 마지막 열에서 *hard* 및 *intr* NFS 마운트 옵션을 입력하여 중단할 수 없는 무제한 재시도를 적용하거나, *soft*, *retrans* 및 *timeo* 마운트 옵션을 입력하고 *retrans*를 *120* 이상으로 설정하고 *timeo*를 1초의 *3000*/10으로 설정하여 지정된 재시도 횟수를 설정합니다.

    hard 재시도 옵션을 설정하거나, 긴 시간 초과 값과 충분한 재시도 횟수로 *soft* 옵션을 지정하면 요청된 파일이 즉시 마운트할 수 없는 이동식 볼륨에 있는 경우에도 NFS 요청이 실패하지 않습니다. 이러한 마운트 옵션에 대한 자세한 내용은 Solaris *mount_nfs* 매뉴얼 페이지를 참조하십시오.

    예제에서는 *soft* 마운트 옵션을 입력합니다.

```
#File         Device                         Mount
#Device       to      Mount     System fsck  at     Mount
#to Mount     fsck    Point     Type   Pass  Boot   Options
#------------ ------  --------- ------ ----  -----  ----------------
/devices      -       /devices  devfs  -     no     -
...
qfsnfs:/qfsms -       /qfsnfs   nfs    -     yes    soft,retrans=120,timeo=3000
```

11. NFS 2를 사용 중인 경우 *rsize* 마운트 매개변수를 *32768*로 설정합니다.

    다른 NFS 버전인 경우 기본값을 사용합니다.

    rsize 마운트 매개변수는 읽기 버퍼 크기를 *32768*바이트(기본값: *8192*바이트)로 설정합니다. 예제에서는 NFS 2 구성을 보여줍니다.

```
#File         Device                         Mount
#Device       to      Mount     System fsck  at     Mount
#to Mount     fsck    Point     Type   Pass  Boot   Options
#------------ ------  --------- ------ ----  -----  ----------------
/devices      -       /devices  devfs  -     no     -
...
qfsnfs2:/qfs2 -       /qfsnfs2  nfs    -     yes    ...,rsize=32768
```

12. NFS 2를 사용 중인 경우 *wsize* 마운트 매개변수를 *32768*로 설정합니다.

    다른 NFS 버전인 경우 기본값을 사용합니다.

    wsize 마운트 매개변수는 쓰기 버퍼 크기를 지정된 바이트 수(기본값: *8192*바이트)로 설정합니다. 예제에서는 NFS 2 구성을 보여줍니다.

```
#File         Device                         Mount
#Device       to      Mount     System fsck  at     Mount
#to Mount     fsck    Point     Type   Pass  Boot   Options
#------------ ------  --------- ------ ----  -----  ----------------
```

```
/devices      -       /devices  devfs  -     no     -
...
qfsnfs2:/qfs2 -       /qfsnfs2  nfs    -     yes    ...,wsize=32768
```

13. */etc/vfstab* 파일을 저장하고 편집기를 종료합니다.

```
#File         Device                         Mount
#Device       to      Mount     System fsck  at     Mount
#to Mount     fsck    Point     Type   Pass  Boot   Options
#------------ ------  --------- ------ ----  -----  ----------------
/devices      -       /devices  devfs  -     no     -
...
qfsnfs:/qfsms -       /qfsnfs   nfs    -     yes    soft,retrans=120,timeo=3000
:wq
[nfsclient1]root@solaris:~#
```

14. 공유 파일 시스템에 대한 마운트 지점 디렉토리를 만듭니다.

    예제에서는 */qfsnfs* 디렉토리에 공유 파일 시스템을 마운트합니다.

```
[nfsclient1]root@solaris:~# mkdir /qfsnfs
[nfsclient1]root@solaris:~#
```

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

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

```
[nfsclient1]root@solaris:~# mkdir /qfsnfs
[nfsclient1]root@solaris:~# chmod 755 /qfsnfs
[nfsclient1]root@solaris:~#
```

16. 공유 파일 시스템을 마운트합니다.

```
[nfsclient1]root@solaris:~# mount /qfsnfs
[nfsclient1]root@solaris:~#
```

17. 사이드밴드 데이터베이스 기능을 사용할 계획인 경우 10장. 보고 데이터베이스 구성을 참조하십시오.
18. 그렇지 않은 경우 11장. 알림 및 로깅 구성으로 이동합니다.

SMB/CIFS를 사용하여 Oracle HSM 파일 시스템 공유

SMB는 Oracle HSM을 Microsoft Windows 호스트에서 액세스할 수 있도록 만들고 대소문자 구분 사용 안함, DOS 속성 지원, NFSv4 ACL(액세스 제어 목록) 지원 등과 같은 상호운용성 기능을 제공합니다. Oracle Solaris OS는 NT LM 0.12 및 CIFS(Common Internet File System)를 비롯하여 여러 SMB 언어에 대한 지원을 포함한 SMB(Server Message Block) 프로토콜 서버 및 클라이언트 구현을 제공합니다.

Oracle HSM은 Windows SID(보안 식별자)를 지원합니다. Windows ID는 더 이상 *idmap* 서비스를 사용하여 명시적으로 정의하거나 Active Directory 서비스에서 제공할 필요가 없습니다.

Oracle HSM 파일 시스템을 사용하여 SMB 서비스를 구성하려면 다음 작업을 수행합니다.

- Oracle Solaris SMB 구성 및 관리 설명서 검토.
- SMB 서버에 대한 Windows ID 명시적 매핑(선택사항).
- SMB/CIFS와 공유하도록 Oracle HSM 파일 시스템 구성.
- Windows Active Directory 도메인 또는 작업 그룹에 대한 SMB 서버 구성.
- SMB/CIFS 공유로 Oracle HSM 파일 시스템 공유.

Oracle Solaris SMB 구성 및 관리 설명서 검토

아래 절에서는 Oracle HSM 파일 시스템에 적용되는 SMB 구성 프로세스 부분에 대해 간략히 설명합니다. 자세한 설명이나 일부 시나리오에 대해서는 다루지 않습니다. 따라서 Oracle Solaris SMB 서버 구성, 기존 Windows 환경에 서버 통합, Solaris 시스템에 SMB 공유 마운트 등에 대한 전체 지침을 검토하십시오. 전체 지침은 *Oracle Solaris Information Library*의 *Managing SMB and Windows Interoperability in Oracle Solaris* 볼륨에서 확인할 수 있습니다.

SMB 서버에 대한 Windows ID 명시적 매핑(선택사항)

이제 Oracle HSM는 Windows SID(보안 식별자)를 완벽하게 지원하지만, UNIX ID와 SID 사이의 관계를 명시적으로 정의하면 유용한 경우도 있습니다. 예를 들어 사용자가 UNIX ID와 Windows ID를 모두 사용하는 유형이 다른 환경에서 *idmap* 서비스 또는 Active Directory 서비스를 사용하여 명시적 매핑을 만들 수도 있습니다. SMB 및 Windows 상호운용성에 대한 자세한 내용은 사용하는 Oracle Solaris 버전의 제품 설명서를 참조하십시오.

SMB/CIFS와 공유하도록 Oracle HSM 파일 시스템 구성

SMB/CIFS를 사용하여 공유되는 Oracle HSM 파일 시스템은 NFS(네트워크 파일 시스템) 버전 4에서 채택되고 Oracle Solaris 11에서 소개된 새로운 ACL(액세스 제어 목록) 구현을 사용해야 합니다. 이전 버전의 Solaris 및 NFS에서는 Windows ACL 구현과 호환되지 않는 POSIX 드래프트 사양을 기반으로 하는 ACL을 사용합니다.

Oracle HSM으로 만드는 새로운 파일 시스템에서는 기본적으로 Solaris 11의 NFS 버전 4 ACL을 사용합니다. 하지만 기존 Oracle HSM 파일 시스템을 SMB/CIFS 클라이언트와 공유해야 하는 경우 적절한 절차를 사용하여 기존 POSIX 스타일 ACL을 변환해야 합니다.

POSIX 스타일 ACL을 사용하는 Oracle HSM 비공유 파일 시스템 변환

다음과 같이 하십시오.

1. 호스트에 *root*로 로그인합니다.

   예제에서는 *qfs-host* 호스트에 로그인합니다.

```
[qfs-host]root@solaris:~#
```

2. 호스트에서 Oracle Solaris 11.1 이상을 실행하는지 확인합니다. *uname -r* 명령을 사용합니다.

```
[qfs-host]root@solaris:~# uname -r
5.11
[qfs-host]root@solaris:~#
```

3. *umount mount-point* 명령을 사용하여 파일 시스템을 마운트 해제합니다. 여기서 *mount-point*는 Oracle HSM 파일 시스템의 마운트 지점입니다.

   자세한 내용은 *umount_samfs* 매뉴얼 페이지를 참조하십시오. 아래 예제에서 서버 이름은 *qfs-host*이고 파일 시스템은 */qfsms*입니다.

```
[qfs-host]root@solaris:~# umount /qfsms
```

4. *samfsck -F -A file-system* 명령을 사용하여 파일 시스템을 변환합니다. 여기서 *-F* 옵션은 파일 시스템의 검사와 오류 수정을 지정하고, *-A* 옵션은 ACL 변환을 지정하고, *file-system*은 변환해야 하는 파일 시스템의 이름입니다.

   -A 옵션을 지정한 경우 *-F* 옵션이 필요합니다. *samfsck -F -A* 명령이 오류를 반환하는 경우 프로세스가 중지되고 ACL이 변환되지 않습니다. 이러한 옵션에 대한 자세한 내용은 *samfsck* 매뉴얼 페이지를 참조하십시오.

```
[qfs-host]root@solaris:~# samfsck -F -A /qfsms
```

5. 오류가 반환되고 ACL이 변환되지 않는 경우 *samfsck -F -a file-system* 명령을 사용하여 ACL을 강제로 변환합니다.

   -a 옵션은 강제 변환을 지정합니다. *-a* 옵션을 지정한 경우 *-F* 옵션이 필요합니다. 이러한 옵션에 대한 자세한 내용은 *samfsck* 매뉴얼 페이지를 참조하십시오.

```
[qfs-host]root@solaris:~# samfsck -F -a /qfsms
```

6. 이제 Windows Active Directory 도메인 또는 작업 그룹에 대한 SMB 서버 구성을 수행합니다.

POSIX 스타일 ACL을 사용하는 Oracle HSM 공유 파일 시스템 변환

1. 파일 시스템 메타데이터 서버에 *root*로 로그인합니다.

   예제에서는 메타데이터 서버 *sharedqfs-mds*에 로그인합니다.

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

2. 메타데이터 서버에서 Oracle Solaris 11.1 이상을 실행하는지 확인합니다. *uname -r* 명령을 사용합니다.

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

3. 각 Oracle HSM 클라이언트에 *root*로 로그인하고, 각 클라이언트에서 Oracle Solaris 11.1 이상을 실행하는지 확인합니다.

   예제에서는 단말기 창을 열고 *ssh*를 사용하여 클라이언트 호스트 *sharedqfs-client1* 및 *sharedqfs-client2*에 원격으로 로그인하여 로그인 배너에서 Solaris 버전을 가져옵니다.

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

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

4. *umount mount-point* 명령을 사용하여 각 Oracle HSM 클라이언트에서 Oracle HSM 공유 파일 시스템을 마운트 해제합니다. 여기서 *mount-point*는 Oracle HSM 파일 시스템의 마운트 지점입니다.

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

```
Oracle Corporation      SunOS 5.11      11.1    September 2013
[sharedqfs-client1]root@solaris:~# umount /sharedqfs
```

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

Oracle Corporation      SunOS 5.11      11.1    September 2013
[sharedqfs-client2]root@solaris:~# umount /sharedqfs
[sharedqfs-client1]root@solaris:~#
```

5. *umount -o await_clients=interval mount-point* 명령을 사용하여 메타데이터 서버에서 Oracle HSM 공유 파일 시스템을 마운트 해제합니다. 여기서 *mount-point*는 Oracle HSM 파일 시스템의 마운트 지점이고, interval은 *-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 */sharedqfs* 파일 시스템을 *sharedqfs-mds* 메타데이터 서버에서 마운트 해제하고 클라이언트에서 *60*초 동안 마운트 해제하도록 허용합니다.

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

6. 파일 시스템을 POSIX 스타일 ACL에서 NFS 버전 4 ACL로 변환합니다. 메타데이터 서버에서 *samfsck -F -A file-system* 명령을 사용하여 파일 시스템을 변환합니다. 여기서 *-F* 옵션은 파일 시스템의 검사와 오류 수정을 지정하고, *-A* 옵션은 ACL 변환을 지정하고, *file-system*은 변환해야 하는 파일 시스템의 이름입니다.

   -A 옵션을 지정한 경우 *-F* 옵션이 필요합니다. *samfsck -F -A file-system* 명령이 오류를 반환하는 경우 프로세스가 중지되고 ACL이 변환되지 않습니다. 이러한 옵션에 대한 자세한 내용은 *samfsck* 매뉴얼 페이지를 참조하십시오. 예제에서는 */sharedqfs*라는 Oracle HSM 파일 시스템을 변환합니다.

```
[sharedqfs-mds]root@solaris:~# samfsck -F -A /sharedqfs
```

7. 오류가 반환되고 ACL이 변환되지 않는 경우 ACL을 강제로 변환합니다. 메타데이터 서버에서는 *samfsck -F -a file-system* 명령을 사용합니다.

   -a 옵션은 강제 변환을 지정합니다. *-a* 옵션을 지정한 경우 *-F* 옵션이 필요합니다. 이러한 옵션에 대한 자세한 내용은 *samfsck* 매뉴얼 페이지를 참조하십시오. 예제에서는 */qfsma*라는 Oracle HSM 파일 시스템을 강제로 변환합니다.

```
[sharedqfs-mds]root@solaris:~# samfsck -F -a /sharedqfs
```

8. 이제 Windows Active Directory 도메인 또는 작업 그룹에 대한 SMB 서버 구성을 수행합니다.

Windows Active Directory 도메인 또는 작업 그룹에 대한 SMB 서버 구성

Oracle Solaris SMB 서비스는 도메인 또는 작업 그룹의 두 가지 상호 배타적인 모드로 작동할 수 있습니다. 사용 환경과 인증 요구 사항을 기반으로 두 모드 중 하나를 선택합니다.

- Active Directory 도메인 사용자에게 Solaris SMB 서비스에 대한 액세스 권한을 부여해야 하는 경우 도메인 모드로 SMB 서버 구성을 수행합니다.
- 로컬 Solaris 사용자에게 SMB 서비스에 대한 액세스 권한을 부여해야 하고 Active Directory 도메인이 없거나 Active Directory 도메인 사용자에게 서비스에 대한 액세스 권한을 부여해야 하는 경우 작업 그룹 모드로 SMB 서버 구성을 수행합니다.

도메인 모드로 SMB 서버 구성

1. Windows Active Directory 관리자에게 문의하여 다음 정보를 확인하십시오.
   - Active Directory 도메인에 조인할 때 사용해야 하는 인증된 Active Directory 사용자 계정의 이름
   - 계정에 대한 기본 *Computers* 컨테이너 대신 사용해야 하는 조직 단위(있는 경우)
   - Oracle HSM 파일 시스템을 공유할 도메인에 대한 정규화된 LDAP/DNS 도메인 이름
2. SMB/CIFS 공유로 구성하려는 Oracle HSM 파일 시스템의 호스트에 로그인합니다. *root*로 로그인합니다.

   파일 시스템이 Oracle HSM 공유 파일 시스템인 경우 파일 시스템에 대한 메타데이터 서버에 로그인합니다. 아래 예제에서 서버 이름은 *qfssmb*입니다.

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

3. 오픈 소스 Samba 서버와 SMB 서버를 단일 Oracle Solaris 시스템에서 함께 사용할 수 없습니다. 따라서 Samba 서비스가 실행 중인지 확인하십시오. 서비스 상태 명령 *svcs*의 출력을 *grep* 및 정규식 *samba*에 연결합니다.

   예제에서는 *svcs* 명령의 출력에 정규식과 일치하는 항목이 포함되어 있으므로 SMB 서비스가 실행 중인 것입니다.

```
[qfssmb]root@solaris:~# svcs | grep samba
legacy_run     Nov_03   lrc:/etc/rc3_d/S90samba
```

4. Samba 서비스(*svc:/network/samba*)가 실행 중인 경우 Windows Internet Naming Service/WINS(*svc:/network/wins*)(실행 중인 경우)와 함께 서비스를 사용 안함으로 설정합니다. *svcadm disable* 명령을 사용합니다.

```
[qfssmb]root@solaris:~# svcadm disable svc:/network/samba
[qfssmb]root@solaris:~# svcadm disable svc:/network/wins
```

5. 이제 *svcadm enable -r smb/server* 명령을 사용하여 SMB 서버와 이 서버에 종속되는 모든 서비스를 시작합니다.

```
[qfssmb]root@solaris:~# svcadm enable -r smb/server
```

6. Oracle HSM 호스트의 시스템 시계와 Microsoft Windows 도메인 컨트롤러의 시스템 시계의 시차가 5분 이내여야 합니다.
   - Windows 도메인 컨트롤러에서 NTP(Network Time Protocol) 서버를 사용하는 경우 Oracle HSM 호스트에서 동일한 서버를 사용하도록 구성합니다. Oracle HSM 호스트에서 */etc/inet/ntpclient.conf* 파일을 만들고 *svcadm enable ntp* 명령을 사용하여 *ntpd* 데몬을 시작합니다. 자세한 내용은 *ntpd* 매뉴얼 페이지와 Oracle Solaris 관리 설명서를 참조하십시오.
   - 그렇지 않은 경우 *ntpdate domain-controller-name* 명령을 실행하여 Oracle HSM 호스트를 도메인 컨트롤러와 동기화하거나(자세한 내용은 *ntpdate* 매뉴얼 페이지 참조) Oracle HSM 호스트의 시스템 시계를 도메인 컨트롤러의 시스템 시계에 표시된 시간으로 수동으로 설정합니다.
7. *smbadm join -u username -o organizational-unit domain-name* 명령을 사용하여 Windows 도메인에 조인합니다. 여기서 *username*은 Active Directory 관리자가 지정한 사용자 계정의 이름이고, 선택적 *organizational-unit*은 지정된 계정 컨테이너이고(있는 경우), *domain-name*은 지정된 정규화된 LDAP 또는 DNS 도메인 이름입니다.

   예제에서는 사용자 계정을 사용하여 Windows 도메인 *this.example.com*에 조인합니다.

```
[qfssmb]root@solaris:~# smbadm join -u admin -o smbsharing this.example.com
```

8. 이제 SMB/CIFS 공유로 Oracle HSM 파일 시스템 공유를 수행합니다.

작업 그룹 모드로 SMB 서버 구성

1. Windows 네트워크 관리자에게 문의하여 Oracle HSM 파일 시스템의 호스트를 조인할 Windows 작업 그룹의 이름을 확인합니다.

   기본 작업 그룹의 이름은 *WORKGROUP*입니다.
2. Oracle HSM 파일 시스템의 호스트에 로그인합니다. *root*로 로그인합니다.

   파일 시스템이 Oracle HSM 공유 파일 시스템인 경우 파일 시스템에 대한 메타데이터 서버에 로그인합니다. 아래 예제에서 서버 이름은 *qfssmb*입니다.

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

3. 오픈 소스 Samba 서버와 SMB 서버를 단일 Oracle Solaris 시스템에서 함께 사용할 수 없습니다. 따라서 Samba 서비스가 실행 중인지 확인하십시오. *svcs* 서비스 상태 명령의 출력을 *grep* 및 정규식 *samba*에 연결합니다.

   예제에서는 *svcs* 명령의 출력에 정규식과 일치하는 항목이 포함되어 있으므로 SMB 서비스가 실행 중인 것입니다.

```
[qfssmb]root@solaris:~# svcs | grep samba
legacy_run     Nov_03   lrc:/etc/rc3_d/S90samba
```

4. Samba 서비스(*svc:/network/samba*)가 실행 중인 경우 Windows Internet Naming Service/WINS(*svc:/network/wins*) 서비스(실행 중인 경우)와 함께 이 서비스를 사용 안함으로 설정합니다. *svcadm disable* 명령을 사용합니다.

   Samba 서버와 SMB 서버를 단일 Oracle Solaris 시스템에서 함께 사용할 수 없습니다.

```
[qfssmb]root@solaris:~# svcadm disable svc:/network/samba
[qfssmb]root@solaris:~# svcadm disable svc:/network/wins
```

5. 이제 *svcadm enable -r smb/server* 명령을 사용하여 SMB 서버와 이 서버에 종속되는 모든 서비스를 시작합니다.

```
[qfssmb]root@solaris:~# svcadm enable -r smb/server
```

6. 작업 그룹을 조인합니다. *smbadm join* 명령을 *-w*(작업 그룹) 스위치 및 Windows 네트워크 관리자가 지정한 작업 그룹의 이름과 함께 사용합니다.

   예제에서 지정된 작업 그룹의 이름은 *crossplatform*입니다.

```
[qfssmb]root@solaris:~# smbadm join -w crossplatform
```

7. SMB 암호를 암호화하도록 Oracle HSM 호스트를 구성합니다. 텍스트 편집기에서 */etc/pam.d/other* 파일을 열고 *password required pam_smb_passwd.so.1 nowarn* 명령줄을 추가한 다음 파일을 저장합니다.

   이 예에서는 *vi* 편집기를 사용합니다.

```
[qfssmb]root@solaris:~# vi /etc/pam.d/other
# Copyright (c) 2012, Oracle and/or its affiliates. All rights reserved.
#
# PAM configuration
#
# Default definitions for Authentication management
# Used when service name is not explicitly mentioned for authentication
#
auth definitive  pam_user_policy.so.1
```

```
...
password required pam_authtok_store.so.1
password required pam_smb_passwd.so.1 nowarn
:wq
[qfssmb]root@solaris:~#
```

자세한 내용은 *pam_smb_passwd* 매뉴얼 페이지를 참조하십시오.

8. *pam_smb_passwd* 모듈이 설치된 경우 SMB 서버에서 Windows 작업 그룹에 로그인할 수 있도록 *passwd local-username* 명령을 사용하여 *local-username* 사용자에 대한 암호의 암호화된 버전을 생성합니다.

   SMB 서버에서는 Solaris 운영체제에 사용된 암호의 동일한 암호화된 버전을 사용하여 사용자를 인증할 수 없습니다. 예제에서는 *smbsamqfs* 사용자에 대한 암호화된 SMB 암호를 생성합니다.

```
[qfssmb]root@solaris:~# passwd smbsamqfs
```

9. 이제 SMB/CIFS 공유로 Oracle HSM 파일 시스템 공유를 수행합니다.

SMB/CIFS 공유로 Oracle HSM 파일 시스템 공유

Oracle Solaris 운영체제 버전의 관리 설명서에 설명된 절차를 사용하여 Oracle HSM 파일 시스템을 공유합니다. 아래 단계에서는 Solaris 11.1에 대한 절차를 요약합니다.

1. SMB/CIFS 공유로 구성하려는 Oracle HSM 파일 시스템의 호스트에 로그인합니다. *root*로 로그인합니다.

   파일 시스템이 Oracle HSM 공유 파일 시스템인 경우 파일 시스템에 대한 메타데이터 서버에 로그인합니다. 아래 예제에서 서버 이름은 *qfssmb*입니다.

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

2. 공유를 구성합니다. *share -F smb -o specific-options sharepath sharename* 명령을 사용합니다. 여기서 *-F* 스위치는 *smb* 공유 프로토콜을 지정하고, *sharepath*는 공유 리소스의 경로이고, *sharename*은 공유에 사용할 이름입니다. 선택적 *-o* 매개변수의 값 *sharing-options*는 다음을 포함할 수 있습니다.
   - *abe=[true|false]*

     공유에 대한 ABE(액세스 기반 열거) 정책이 *true*인 경우 요청하는 사용자가 액세스할 수 없는 디렉토리 항목은 클라이언트로 반환되는 디렉토리 목록에서 생략됩니다.
   - *ad-container=cn=user,ou=organization,dc=domain-dns*

     Active Directory 컨테이너에서는 LDAP(Lightweight Directory Access Protocol) RDN(상대적인 식별 이름) 속성 값인 *cn*(사용자 객체 클래스), *ou*(조직 단위 객체 클

래스), *dc*(도메인 DNS 객체 클래스)로 지정된 도메인 객체로 공유 액세스를 제한합니다.

Active Directory 컨테이너를 SMB/CIFS와 함께 사용하는 방법에 대한 자세한 내용은 *Internet Engineering Task Force Request For Comment (RFC) 2253* 및 Microsoft Windows 디렉토리 서비스 설명서를 참조하십시오.

- *catia=[true|false]*

  CATIA 문자 대체가 *true*인 경우 Windows에서 허용되지 않는 CATIA 버전 4 파일 시스템의 모든 문자가 올바른 문자로 대체됩니다. 대체 목록은 *share_smb* 매뉴얼 페이지를 참조하십시오.

- *csc=[manual|auto|vdo|disabled]*

  클라이언트측 캐싱(*csc*) 정책은 오프라인 사용을 위해 파일의 클라이언트측 캐싱을 제어합니다. *manual* 정책을 사용하면 사용자가 요청한 경우 클라이언트에서 파일을 캐시할 수 있지만, 자동 파일별 재통합(기본값)이 사용 안함으로 설정됩니다. *auto* 정책을 사용하면 클라이언트에서 파일을 자동으로 캐시할 수 있고 자동 파일별 재통합이 사용으로 설정됩니다. *vdo* 정책을 사용하면 오프라인에서 사용하기 위해 클라이언트에서 파일을 자동으로 캐시하고, 파일별 재통합을 사용으로 설정하고, 오프라인 상태에서도 로컬 캐시에서 클라이언트 작업을 수행할 수 있습니다. *disabled* 정책은 클라이언트측 캐싱을 허용하지 않습니다.

- *dfsroot=[true|false]*

  Microsoft DFS(분산 파일 시스템)에서 루트 공유(*dfsroot=true*)는 널리 분산된 공유 폴더 그룹을 보다 쉽게 관리할 수 있는 단일 DFS 파일 시스템으로 구성하는 공유입니다. 자세한 내용은 Microsoft Windows Server 설명서를 참조하십시오.

- *guestok=[true|false]*

  guestok 정책이 *true*이면 로컬로 정의된 *guest* 계정으로 공유에 액세스할 수 있습니다. 이 정책이 *false*이거나 정의되지 않은 경우(기본값) *guest* 계정으로 공유에 액세스할 수 없습니다. 이 정책을 사용하면 Windows *Guest* 사용자를 로컬로 정의된 UNIX 사용자 이름(예: *guest* 또는 *nobody*)에 매핑할 수 있습니다.

```
# idmap add winname:Guest unixuser:guest
```

그런 다음 원하는 경우 */var/smb/smbpasswd*에 저장된 암호를 기준으로 로컬로 정의된 계정을 인증할 수 있습니다. 자세한 내용은 *idmap* 매뉴얼 페이지를 참조하십시오.

- *rw=[*|[[-]criterion][:[-]criterion*]...

  rw 정책은 제공된 액세스 목록과 일치하는 모든 클라이언트에 권한을 허용하거나 거부합니다.

액세스 목록에는 모두를 의미하는 단일 별표(*) 또는 클라이언트 액세스 기준의 콜론으로 구분된 목록이 포함되어 있습니다. 여기서 각 *criterion*은 거부를 의미하는 선택적 빼기 기호(-), 호스트 이름, 네트워크 그룹, 전체 LDAP 또는 DNS 도메인 이름 및/또는 기호 @과 IP 주소 또는 도메인 이름의 전체 또는 일부로 구성됩니다. 클라이언트가 기준 중 하나를 충족할 때까지 왼쪽에서 오른쪽으로 액세스 목록을 평가합니다. 자세한 내용은 *share_smb* 매뉴얼 페이지를 참조하십시오.

- *ro=[*|[[-]criterion][:[-]criterion*]...

  ro 정책은 액세스 목록과 일치하는 클라이언트에 읽기 전용 권한을 부여하거나 거부합니다.

- *none=[*|[[-]criterion][:[-]criterion*]...

  none 정책은 액세스 목록과 일치하는 클라이언트에 대한 액세스를 거부합니다. 액세스 목록이 별표(*)인 경우 *ro* 및 *rw* 정책이 *none* 정책을 대체합니다.

예제에서는 */qfsms* 파일 시스템 읽기/쓰기를 *smbclient1* 및 *smbclient2* 클라이언트와 공유하고 읽기 전용을 *smbclient3*과 공유합니다.

```
[qfssmb]root@solaris:~# share -F smb -o rw=smbclient1:smbclient2
ro=smbclient3 /qfsms
```

명령을 입력하면 시스템에서 SMB 서버 데몬 *smbd*를 자동으로 다시 시작합니다.

3. 공유 매개변수를 확인합니다. *share -F nfs* 명령을 사용합니다.

   예제에서 명령 출력은 공유를 올바르게 구성했음을 보여 줍니다.

```
[qfssmb]root@solaris:~# share -F smb /qfsms
sec=sys,rw=smbclient1:smbclient2,ro=smbclient3
[qfssmb]root@solaris:~#
```

4. 사이드밴드 데이터베이스 기능을 사용할 계획인 경우 10장. 보고 데이터베이스 구성을 참조하십시오.
5. 그렇지 않은 경우 11장. 알림 및 로깅 구성으로 이동합니다.

8

8장. SAM-Remote 구성

Oracle Hierarchical Storage Manager 소프트웨어의 SAM-Remote 기능을 통해 Oracle HSM 파일 시스템 호스트는 원격 Oracle HSM 파일 시스템 호스트에 호스트된 테이프 매체 및 드라이브에 액세스할 수 있습니다. 로컬 호스트는 SAM-Remote 서버 역할을 하는 원격 호스트의 SAM-Remote 클라이언트로 테이프 리소스에 액세스합니다. 클라이언트의 아카이빙 정책은 일반적으로 로컬 자기 또는 SSD(Solid State Disk) 디스크 아카이브에 한두 개의 복사본을 유지 관리하고 서버가 제공하는 원격 테이프에 한두 개의 복사본을 유지 관리합니다. 각 호스트의 마스터 구성 파일 */etc/opt/SUNWsamfs/mcf*는 특수 SAM-Remote 장비 유형을 사용하여 공유 리소스 및 클라이언트/서버 관계를 정의합니다.

SAM-Remote 클라이언트와 서버에 대한 다양한 아카이빙 및 데이터 보호 요구 사항을 처리할 수 있습니다.

- 테이프 아카이빙의 이점을 라이브러리와 드라이브가 부족한 Oracle HSM 호스트까지 확장할 수 있습니다.
- 지역 사무실 및 위성 캠퍼스에서 호스트된 Oracle HSM 파일 시스템에 대한 테이프 리소스를 중앙에서 유지 및 관리할 수 있습니다.

  중앙의 주 사무실 데이터 센터에서는 Oracle HSM 파일 시스템 호스트에 연결된 테이프 라이브러리가 있으며 SAM-Remote 서버로 작동합니다. 소규모 분산 사무실에서는 Oracle HSM 파일 시스템 호스트에 디스크 아카이브만 있으며 SAM-Remote 클라이언트로 작동합니다. 모든 호스트가 아카이브된 데이터의 로컬 및 테이프 복사본을 모두 유지 관리합니다. 그러나 하드웨어 및 매체 인벤토리는 중앙 데이터 센터에 집중되며, 여기서 가장 효율적 및 최소 비용으로 유지 관리할 수 있습니다.
- 백업 및 재해 복구를 위해 오프사이트 테이프 복사본을 자동으로 만들고 유지 관리할 수 있습니다.

  모든 Oracle HSM 파일 시스템 호스트에 연결된 테이프 라이브러리가 있습니다. 각 호스트는 오프사이트 위치의 대응 쌍과 관련해서 SAM-Remote 클라이언트 및 서버 둘 다로 작동합니다. 각 Oracle HSM 호스트는 로컬 리소스를 사용하여 로컬 디스크 및 테이프 복사본을 만듭니다. 각 호스트는 대응 쌍이 제공하는 리소스를 사용하여 원격 테이프 복사본을 만들고, 각각 대응 쌍에 테이프 리소스를 제공합니다. 따라서 두 파일 시스템의 오프사이트 복사본이 정상적인 아카이빙 프로세스의 일부로 자동으로 생성됩니다.
- 로컬 리소스를 사용할 수 없는 경우 원격 아카이브 스토리지에 액세스하도록 Oracle HSM 파일 시스템 호스트를 구성할 수 있습니다.

  모든 Oracle HSM 파일 시스템에 연결된 테이프 라이브러리가 있으며 각각 다른 위치의 대응 쌍과 관련해서 SAM-Remote 클라이언트 및 서버 둘 다로 작동합니다. 각 Oracle

HSM 호스트는 로컬 리소스를 사용하여 로컬 디스크 및 테이프 복사본을 만듭니다. 그러나 로컬 라이브러리에 액세스할 수 없는 경우에도 원격 대응 쌍이 제공하는 매체와 리소스를 사용하여 파일을 아카이브 및 검색할 수 있습니다.

이 장에서는 SAM-Remote 클라이언트/서버 네트워크 구성 프로세스를 설명합니다. 다음 작업을 다룹니다.

모든 SAM-Remote 호스트에서 동일한 소프트웨어를 사용하는지 확인

SAM-Remote 클라이언트와 서버에 동일한 Oracle HSM 소프트웨어 개정판을 설치해야 합니다. 아래 절차를 사용하여 개정 레벨을 확인합니다.

1. SAM-Remote 서버 호스트에 *root*로 로그인합니다.

   예제에서 서버 호스트는 *server1*입니다.

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

2. SAM-Remote 클라이언트 호스트에 *root*로 로그인합니다.

   예제에서는 단말기 창을 열고 *ssh*를 사용하여 호스트 *client1*에 로그인합니다.

```
[server1]root@solaris:~# ssh root@client1
Password: ...
[client1]root@solaris:~#
```

3. 모든 SAM-Remote 서버와 클라이언트에서 Oracle HSM 패키지 개정 레벨이 동일한지 확인합니다. 각 SAM-Remote 호스트에서 *samcmd l* 명령을 사용하여 구성 세부정보를 나열합니다. 결과를 비교합니다.

   예제에서는 *server1*의 결과를 *client1*의 결과와 비교합니다. 둘 다 동일한 릴리스의 Oracle HSM 소프트웨어를 사용합니다.

```
[server1]root@solaris:~# samcmd l
Usage information samcmd     6.0  10:20:34 Feb 20 2015
samcmd on server1
...
[server1]root@solaris:~#
```

```
[client1]root@solaris:~# samcmd l
Usage information samcmd     6.0  10:20:37 Feb 20 2015
samcmd on client1
...
[server1]root@solaris:~#
```

4. 필요한 경우 모든 SAM-Remote 서버와 클라이언트의 개정 레벨이 같아질 때까지 4장. *Oracle HSM* 및 *QFS* 소프트웨어 설치 의 절차를 사용하여 호스트 소프트웨어를 업데이트합니다.
5. 이제 Oracle HSM 프로세스 중지를 수행합니다.

Oracle HSM 프로세스 중지

1. SAM-Remote 서버 호스트에 *root*로 로그인합니다.

   예제에서는 서버 이름이 *server1*로 지정됩니다.

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

2. 구성된 장치의 장비 순서 번호를 가져옵니다. *samcmd c* 명령을 사용합니다.

   예제에서 장치 번호가 *801*, *802*, *803* 및 *804*로 지정됩니다.

```
[server1]root@solaris:~# samcmd c
Device configuration samcmd     6.0  10:20:34 Feb 20 2015
samcmd on server1
Device configuration:
ty  eq  state   device_name                       fs   family_set
rb  800 on      /dev/scsi/changer/c1t0d5          800  rb800
tp  801 on      /dev/rmt/0cbn                     801  rb800
tp  802 on      /dev/rmt/1cbn                     802  rb800
tp  803 on      /dev/rmt/2cbn                     803  rb800
tp  804 on      /dev/rmt/3cbn                     804  rb800
```

3.
4. 모든 아카이빙 프로세스를 유휴 설정합니다(있는 경우). *samcmd aridle* 명령을 사용합니다.

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

```
[samfs-mds]root@solaris:~# samcmd aridle
[samfs-mds]root@solaris:~#
```

5. 모든 스테이징 프로세스를 유휴 설정합니다(있는 경우). *samcmd stidle* 명령을 사용합니다.

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

   ```
   [samfs-mds]root@solaris:~# samcmd stidle
   [samfs-mds]root@solaris:~#
   ```

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

   아카이빙 프로세스가 *Waiting for :arrun*이면 아카이빙 프로세스가 유휴 상태임을 나타냅니다.

   ```
   [samfs-mds]root@solaris:~# samcmd a
   Archiver status samcmd     6.0 14:20:34 Feb 22 2015
   samcmd on samfs-mds
   sam-archiverd:  Waiting for :arrun
   sam-arfind: ...
   Waiting for :arrun
   ```

7. 모든 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.

   ```
   [samfs-mds]root@solaris:~# samcmd u
   Staging queue samcmd      6.0 14:20:34 Feb 22 2015
   samcmd on solaris.demo.lan
   Staging queue by media type: all
   sam-stagerd:  Waiting for :strun
   root@solaris:~#
   ```

8. 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.

   ```
   [samfs-mds]root@solaris:~# samcmd 801 idle
   [samfs-mds]root@solaris:~# samcmd 802 idle
   [samfs-mds]root@solaris:~# samcmd 803 idle
   ```

```
[samfs-mds]root@solaris:~# samcmd 804 idle
[samfs-mds]root@solaris:~#
```

9. 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.

```
[samfs-mds]root@solaris:~# samcmd r
Removable media samcmd     6.0 14:20:34 Feb 22 2015
samcmd on samqfs1host
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
[samfs-mds]root@solaris:~#
```

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

```
[samfs-mds]root@solaris:~# samd stop
[samfs-mds]root@solaris:~#
```

11. 이제 SAM-Remote 서버 구성을 수행합니다.

SAM-Remote 서버 구성

SAM-Remote 서버는 연결된 로봇 테이프 라이브러리 및 테이프 드라이브를 원격 클라이언트에서 사용할 수 있게 해주는 Oracle HSM 파일 시스템 호스트입니다. 여기서 원격 클라이언트는 그 자체로 Oracle HSM 파일 시스템 호스트입니다. SAM-Remote 서버에서 Oracle HSM 프로세스를 시작하려면 하나 이상의 QFS 파일 시스템을 마운트해야 합니다.

SAM-Remote 서버를 구성하려면 다음 작업을 수행합니다.

- SAM-Remote 서버의 **mcf** 파일에서 원격 공유 아카이빙 장비 정의
- **samremote** 서버 구성 파일 만들기

SAM-Remote 서버의 mcf 파일에서 원격 공유 아카이빙 장비 정의

1. SAM-Remote 서버 호스트에 *root*로 로그인합니다.

예제에서는 서버 이름이 *server1*로 지정됩니다.

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

2. 서버에서 */etc/opt/SUNWsamfs/mcf* 파일을 텍스트 편집기에서 열고 아래로 스크롤하여 아카이빙 장비 정의로 이동합니다.

   예제에서는 *vi* 편집기를 사용합니다. 파일에서는 Oracle HSM 아카이빙 파일 시스템 *fs600*과 4개 드라이브를 보관하는 테이프 라이브러리 *rb800*을 정의합니다. 예제에는 실제 파일에 표시되지 않고 긴 장치 경로를 약어로 표시하는 분류 머리글이 포함되어 있습니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
#==============================================================================
# Oracle HSM archiving file system fs600
# Equipment               Equipment Equipment Family  Device Additional
# Identifier              Ordinal   Type      Set     State  Parameters
#------------------------  --------- --------- ------  ------ ----------
fs600                     600       ms        fs600   on
 /dev/dsk/c9t60...F4d0s7    610       md        fs600   on
 /dev/dsk/c9t60...81d0s7    611       md        fs600   on
#==============================================================================
# Local tape archive rb800
# Equipment               Equipment Equipment Family  Device Additional
# Identifier              Ordinal   Type      Set     State  Parameters
#------------------------  --------- --------- ------  ------ ----------
/dev/scsi/changer/c1t0d5  800       rb        rb800   on
 /dev/rmt/0cbn              801       tp        rb800   on
 /dev/rmt/1cbn              802       tp        rb800   on
 /dev/rmt/2cbn              803       tp        rb800   on
 /dev/rmt/3cbn              804       tp        rb800   on
```

3. 아카이빙 장비 정의의 맨 끝에 테이프 리소스를 클라이언트에서 사용할 수 있게 해주는 서버에 대한 항목을 시작합니다. *Equipment Identifier* 필드에 SAM-Remote 서버 구성 파일의 경로 */etc/opt/SUNWsamfs/samremote*를 입력하고 장비 순서 번호를 지정합니다.

   예제에서는 일부 머리글을 설명으로 추가하고 *samremote* 서버에 장비 순서 번호 *500*을 지정합니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
...
#==============================================================================
# Server samremote shares tape hardware and media with clients
```

```
# Equipment                   Equipment Equipment Family  Device Additional
# Identifier                  Ordinal   Type      Set     State  Parameters
#---------------------------  --------- --------- ------  ------ ----------
/etc/opt/SUNWsamfs/samremote  500
```

4. 새 항목의 *Equipment Type* 필드에 SAM-Remote 서버 장비를 나타내는 *ss*를 입력합니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
...
#===============================================================================
# Server samremote shares tape hardware and media with clients
# Equipment                   Equipment Equipment Family  Device Additional
# Identifier                  Ordinal   Type      Set     State  Parameters
#---------------------------  --------- --------- ------  ------ ----------
/etc/opt/SUNWsamfs/samremote  500       ss
```

5. 모든 호스트와 서버에서 고유한 *Family Set* 이름을 지정하고 장치를 *on*으로 설정합니다.

   예제에서는 새 장비에 패밀리 세트 이름 *ss500*을 지정합니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
...
#===============================================================================
# Server samremote shares tape hardware and media with clients
# Equipment                   Equipment Equipment Family  Device Additional
# Identifier                  Ordinal   Type      Set     State  Parameters
#---------------------------  --------- --------- ------  ------ ----------
/etc/opt/SUNWsamfs/samremote  500       ss        ss500   on
```

6. SAM-Remote 클라이언트를 11개 이상 구성하려면 연속하는 1-10개의 클라이언트로 구성된 그룹마다 추가 서버 장비(*ss* 유형) 항목을 하나씩 추가합니다.
7. 파일을 저장하고 편집기를 닫습니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
...
/etc/opt/SUNWsamfs/samremote  500       ss        ss500   on
:wq
[server1]root@solaris:~#
```

8. 이제 *samremote* 서버 구성 파일 만들기를 수행합니다.

samremote 서버 구성 파일 만들기

SAM-Remote 서버 구성 파일에는 디스크 버퍼 특성과 각 클라이언트에 사용할 매체를 정의합니다. 구성할 각 서버에 대해 다음과 같이 하십시오.

1. SAM-Remote 서버 호스트에 *root*로 로그인합니다.

   예제에서는 서버 이름이 *server1*로 지정됩니다.

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

2. 서버에서 텍스트 편집기로 */etc/opt/SUNWsamfs/samremote* 파일을 만듭니다.

   예제에서는 *vi* 편집기를 사용하여 파일을 만듭니다. 먼저 해시(#) 기호로 표시된 일부 설명을 사용하여 파일을 문서화합니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/samremote
# Server Configuration File:
# Defines the disk buffer and media that is available to each client.
```

3. 새 라인을 시작하고 첫번째 열에 클라이언트의 호스트 이름, IP 주소 또는 정규화된 도메인 이름을 입력하여 첫번째 클라이언트 항목을 시작합니다.

   클라이언트 식별자 행은 비공백 문자로 시작해야 합니다. 예제에서는 호스트 이름 *client1*을 사용하여 클라이언트를 식별합니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/samremote
# Server Configuration File:
# Defines the disk buffer and media that is available to each client.
client1
```

4. 클라이언트와 공유할 매체 식별을 시작합니다. *indent media* 형식으로 새 행을 시작합니다. 여기서 *indent*는 하나 이상의 공백이고 *media*는 SAM-remote 키워드입니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/samremote
# Server Configuration File:
# Defines the disk buffer and media that is available to each client.
client1
        media
```

5. *indent equipment-number media-type VSN* 형식의 새 행을 사용하여 각 매체 유형과 소스를 식별합니다. 설명:
   - *indent*는 하나 이상의 공백입니다.
   - *equipment-number*는 *mcf* 파일에서 아카이브 스토리지 장비를 식별하는 장비 순서 번호입니다.

- *media-type*은 이 장비에 사용되는 테이프 매체의 매체 식별자입니다. 전체 Oracle HSM 매체 유형 목록은 부록 A. 장비 유형 용어집을 참조하십시오.
- *VSN*는 하나 이상의 볼륨 일련 번호를 공백으로 구분한 목록이며 최대 31자의 영숫자 문자입니다.

예제에서는 테이프 라이브러리에 있는 테이프 볼륨 범위(유형 *tp*)인 공유 매체의 소스를 장비 순서 번호 *800*으로 식별합니다. 괄호로 묶은 정규 표현식으로 사용 가능한 볼륨을 지정합니다. 표현식 *VOL0[0-1][0-9]*는 *client1*을 볼륨 *VOL000-VOL019*로 제한합니다.

```
client1
       media
         800 tp (VOL0[0-1][0-9])
```

각 행에 매체 유형을 하나씩만 지정할 수 있습니다. 따라서 라이브러리에서 여러 매체 유형을 지원할 경우 각 유형을 새 항목으로 지정합니다.

```
      media
         800 ti VOL500 VOL501
         800 li (VOL0[0-1][0-9])
```

6. 클라이언트와 공유할 매체 식별을 완료한 후 SAM-Remote 키워드 *endmedia*를 입력하여 목록을 닫습니다.

   이제 예제에서 *client1*이 완전히 구성되었습니다.

```
client1
       media
         800 tp (VOL0[0-1][0-9])
       endmedia
```

7. 추가 클라이언트를 구성해야 하는 경우 지금 구성합니다. 최대 10개의 클라이언트에 대해 각각 새로운 클라이언트 구성 레코드를 추가합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   볼륨 경합 및 데이터 손실을 방지하려면 클라이언트가 동일한 이동식 매체 볼륨을 공유하지 않는지 확인합니다.

   예제에서는 *client2*라는 추가 클라이언트 하나를 구성합니다. 두번째 클라이언트는 *client1*(장비 순서 번호 *800*)과 동일한 테이프 라이브러리에 있는 테이프 볼륨 범위에 액세스할 수 있습니다. 하지만 구성의 정규 표현식은 다른 볼륨 세트(*VOL020-VOL039*)를 지정합니다.

```
# Server Configuration File:
# Defines the disk buffer and media that is available to each client.
```

```
client1
        media
          800 tp (VOL0[0-1][0-9])
        endmedia
client2
        media
          800 tp (VOL02-3][0-9])
        endmedia
:wq
[server1]root@solaris:~#
```

8. 이제 SAM-Remote 클라이언트 구성을 수행합니다.

SAM-Remote 클라이언트 구성

각 SAM-Remote 클라이언트에 대해 다음 작업을 수행합니다.

- SAM-Remote 클라이언트의 MCF 파일에서 원격 아카이브 장비 정의
- SAM-Remote 클라이언트 구성 파일 만들기
- SAM-Remote 클라이언트의 MCF 파일에서 원격 아카이브 장비 정의
- SAM-Remote 클라이언트 구성 파일 만들기
- SAM-Remote 클라이언트에서 **`archiver.cmd`** 파일 구성

SAM-Remote 클라이언트의 MCF 파일에서 원격 아카이브 장비 정의

1. SAM-Remote 클라이언트 호스트에 *`root`*로 로그인합니다.

   예제에서는 SAM-Remote 클라이언트의 이름이 *`client1`*로 지정됩니다.

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

2. 클라이언트에서 */etc/opt/SUNWsamfs/mcf* 파일을 텍스트 편집기에서 열고 아래로 스크롤하여 아카이빙 장비 정의로 이동합니다.

   예제에서는 *`vi`* 편집기를 사용합니다. 파일에서는 Oracle HSM 아카이빙 파일 시스템 *`fs100`*을 정의합니다. 로컬 복사본은 로컬 ZFS 파일 시스템인 디스크 아카이브 *`DISKVOL1`*에 저장됩니다. 예제에는 실제 파일에 표시되지 않고 긴 장치 경로를 약어로 표시하는 분류 머리글이 포함되어 있습니다.

```
[client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Client's /etc/opt/SUNWsamfs/mcf file
#================================================================
# Oracle HSM archiving file system "fs100"
# Equipment               Equipment Equipment Family  Device Additional
# Identifier              Ordinal   Type      Set     State  Parameters
```

```
#------------------------  --------- --------- ------  ------ ----------
fs100                      100       ms        fs100   on
 /dev/dsk/c10t60...7Bd0s7   110       md        fs100   on
 /dev/dsk/c10t60...48d0s7   111       md        fs100   on
#================================================================
# Disk archive "/diskvols/DISKVOL1" stores local archive copies
```

3. 아카이빙 장비 정의의 맨 끝에 서버를 클라이언트에서 사용할 수 있도록 해주는 장비 입력을 시작합니다. *Equipment Identifier* 필드에 SAM-Remote 서버 구성 파일의 경로를 입력하고 장비 순서 번호를 지정합니다.

   예제에서는 클라이언트 구성 이름을 */etc/opt/SUNWsamfs/sc400*으로 지정하고 클라이언트에 장비 순서 번호 *400*을 지정합니다. 또한 일부 머리글을 설명으로 추가합니다.

```
[client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
...
# Disk archive "/diskvols/DISKVOL1" stores local archive copies
#
#================================================================
# Client "sc400" accesses tape resources on server "samremote" (ss500)
# Equipment               Equipment Equipment Family  Device Additional
# Identifier              Ordinal   Type      Set     State  Parameters
#------------------------  --------- --------- ------  ------ ----------
/etc/opt/SUNWsamfs/sc400   400
```

4. 새 항목의 *Equipment Type* 필드에 SAM-Remote 클라이언트 장비를 나타내는 *sc*를 입력합니다.

```
[client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
...
#================================================================
# Client "sc400" accesses tape resources on server "samremote" (ss500)
# Equipment               Equipment Equipment Family  Device Additional
# Identifier              Ordinal   Type      Set     State  Parameters
#------------------------  --------- --------- ------  ------ ----------
/etc/opt/SUNWsamfs/ss500   400       sc
```

5. 모든 호스트와 서버에서 고유한 *Family Set* 이름을 지정하고 장치를 *on*으로 설정합니다.

   예제에서는 새 장비에 패밀리 세트 이름 *ss500*을 지정합니다.

```
[client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
...
#================================================================
```

```
# Client "sc400" accesses tape resources on server "samremote" (ss500)
# Equipment               Equipment Equipment Family  Device Additional
# Identifier              Ordinal   Type      Set     State  Parameters
#-------------------------  --------- --------- ------  ------ ----------
/etc/opt/SUNWsamfs/ss500   400       sc        ss500   on
```

6. SAM-Remote 서버에서 사용할 각 테이프 드라이브에 대해 SAM-Remote 클라이언트 *sc* 장비에 SAM-Remote 의사 장치 하나를 추가합니다. *Equipment Identifier* 필드에서 */dev/samrd/rddevice-number* 형식의 항목을 추가합니다. 여기서 *device-number*는 정수입니다.

   예제에서는 두 의사 장치 */dev/samrd/rd 0* 및 */dev/samrd/rd 1*에 대한 항목을 시작합니다.

```
[client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
...
#=========================================================================
# Client "sc400" accesses tape resources on server "samremote" (ss500)
# Equipment               Equipment Equipment Family  Device Additional
# Identifier              Ordinal   Type      Set     State  Parameters
#-------------------------  --------- --------- ------  ------ ----------
/etc/opt/SUNWsamfs/sc400   400       sc        sc400   on
 /dev/samrd/rd0
 /dev/samrd/rd1
```

7. 각 의사 장치의 *Equipment Ordinal* 필드에 *sc* 장비에 지정된 범위에 속하는 번호를 입력합니다.

   예제에서는 장비 순서 *410*을 */dev/samrd/rd0*에 지정하고 장비 순서 *420*을 */dev/samrd/rd1*에 지정합니다.

```
[client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
...
#=========================================================================
# Client "sc400" accesses tape resources on server "samremote" (ss500)
# Equipment               Equipment Equipment Family  Device Additional
# Identifier              Ordinal   Type      Set     State  Parameters
#-------------------------  --------- --------- ------  ------ ----------
/etc/opt/SUNWsamfs/ss500   400       sc        ss500   on
 /dev/samrd/rd0             410
 /dev/samrd/rd1             420
```

8. 각 SAM-Remote 의사 장치의 *Equipment Type* 필드에 SAM-Remote 의사 장치의 장비 유형을 나타내는 *rd*를 입력합니다.

```
[client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
...
#========================================================================
# Client "sc400" accesses tape resources on server "samremote" (ss500)
# Equipment               Equipment Equipment Family  Device Additional
# Identifier              Ordinal   Type      Set     State  Parameters
#-----------------------  --------- --------- ------  ------ ----------
/etc/opt/SUNWsamfs/ss500   400       sc        ss500   on
 /dev/samrd/rd0             410       rd
 /dev/samrd/rd1             420       rd
```

9. 각 의사 장치의 *Family Set* 필드에 *sc* 장비에 대한 패밀리 세트 이름을 입력합니다.

   예제에서는 패밀리 세트 이름 *ss500*을 사용합니다.

```
[client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
...
#========================================================================
# Client "sc400" accesses tape resources on server "samremote" (ss500)
# Equipment               Equipment Equipment Family  Device Additional
# Identifier              Ordinal   Type      Set     State  Parameters
#-----------------------  --------- --------- ------  ------ ----------
/etc/opt/SUNWsamfs/ss500   400       sc        ss500   on
 /dev/samrd/rd0             410       rd        ss500
 /dev/samrd/rd1             420       rd        ss500
```

10. 각 의사 장치의 *Device State* 필드에 *on*을 입력합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

    예제에서는 장비 순서 *410*을 */dev/samrd/rd0*에 지정하고 장비 순서 *420*을 */dev/samrd/rd1*에 지정합니다.

```
[client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
...
#========================================================================
# Client "sc400" accesses tape resources on server "samremote" (ss500)
# Equipment               Equipment Equipment Family  Device Additional
# Identifier              Ordinal   Type      Set     State  Parameters
#-----------------------  --------- --------- ------  ------ ----------
/etc/opt/SUNWsamfs/ss500   400       sc        ss500   on
 /dev/samrd/rd0             410       rd        ss500   on
 /dev/samrd/rd1             420       rd        ss500   on
:wq
[client1]root@solaris:~#
```

11. 이제 SAM-Remote 클라이언트 구성 파일 만들기를 수행합니다.

SAM-Remote 클라이언트 구성 파일 만들기

각 SAM-Remote 클라이언트에 대해 다음과 같이 하십시오.

1. SAM-Remote 클라이언트 호스트에 *root*로 로그인합니다.

   예제에서는 SAM-Remote 클라이언트의 이름이 *client1*로 지정됩니다.

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

2. 클라이언트에서 텍스트 편집기를 사용하여 */etc/opt/SUNWsamfs/family-set-name* 파일을 만듭니다. 여기서 *family-set-name*은 *mcf* 파일에 사용된 원격 장비의 패밀리 세트 이름입니다.

   예제에서는 *vi* 편집기를 사용하여 파일을 만들고 패밀리 세트 *ss500*에 대해 파일 이름을 지정합니다. 또한 앞에 해시(#) 기호가 표시된 일부 설명을 사용하여 파일을 문서화합니다.

```
[client1]root@solaris:~# vi /etc/opt/SUNWsamfs/sc400
# Client's SAM-Remote client configuration file: /opt/SUNWsamfs/sc400
# This file identifies the host of the SAM-Remote server.
```

3. 새 라인을 시작하고 첫번째 열에 서버의 호스트 이름, IP 주소 또는 정규화된 도메인 이름을 입력하여 서버에 대한 단일 항목을 추가합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   행은 공백이 아닌 문자로 시작해야 합니다. 예제에서는 호스트 이름 *server1*을 사용하여 서버를 식별합니다.

```
[client1]root@solaris:~# vi /etc/opt/SUNWsamfs/samremote
# Client's SAM-Remote server configuration file: /opt/SUNWsamfs/sc400
# This file identifies the host of the SAM-Remote server.
server1
:wq
[client1]root@solaris:~#
```

4. 이제 SAM-Remote 클라이언트에서 *archiver.cmd* 파일 구성을 수행합니다.

SAM-Remote 클라이언트에서 archiver.cmd 파일 구성

1. SAM-Remote 클라이언트 호스트에 *root*로 로그인합니다.

   예제에서는 SAM-Remote 클라이언트의 이름이 *client1*로 지정됩니다.

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

2. */etc/opt/SUNWsamfs/archiver.cmd* 파일을 텍스트 편집기에서 열고 아래로 스크롤하여 *params* 키워드로 시작하고 *endparams* 키워드로 끝나는 복사 매개변수 지시어를 찾습니다.

   예제에서는 *vi* 편집기에서 파일을 엽니다.

```
[client1]root@solaris:~# vi /etc/opt/SUNWsamfs/archiver.cmd
...
#------------------------------------------------------------------------
# Copy Parameter Directives
params
allsets -sort path -offline_copy direct
allfiles.1 -startage 10m -startsize 500M -drives 10
allfiles.2 -startage 24h -startsize 20G -drives 2 -reserve set
endparams
```

3. 원격 매체에 아카이브될 모든 아카이브 세트에 대한 복사 매개변수를 확인합니다. 매개변수에 *-tapenonstop* 및/또는 *-offline_copy direct* 지시어가 포함되어 있는 경우 지금 지시어를 제거합니다.

   예제에서 *all* 매개변수는 모든 복사본에 대한 *-offline_copy direct* 지시어를 지정합니다. 따라서 원격 매체 *allfiles.3*으로 보낼 복사본에 대해 *-offline_copy none*을 지정하여 이 지시어를 대체합니다.

```
#------------------------------------------------------------------------
# Copy Parameter Directives
# Copy Parameter Directives
params
allsets -sort path -offline_copy direct
allfiles.1 -startage 10m -startsize 500M -drives 10
allfiles.2 -startage 24h -startsize 20G -drives 2 -reserve set offline_copy none
endparams
```

4. 아래로 스크롤하여 SAM-Remote 키워드 *vsns*로 시작하고 키워드 *endvsns*로 끝나는 VSN 지시어를 찾습니다.

   예제에서는 *vi* 편집기를 사용합니다. 현재 지정된 매체가 있는 복사본 *allfiles.1*만 로컬 디스크 아카이브 볼륨 *qfs200*을 사용하여 생성됩니다.

```
...
endparams
#------------------------------------------------------------------------
```

```
# VSN Directives
vsns
allfiles.1 dk qfs200
endvsns
```

5. 서버의 */etc/opt/SUNWsamfs/samremote* 파일에서 이 클라이언트에 지정된 아카이브 복사본을 원격 매체에 지정합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   예제에서는 *client1*을 구성합니다. 복사본 *allfiles.2*가 *samremote* 서버 구성 파일에 지정된 *VOL000-VOL019* 범위에 속하는 원격 테이프 볼륨을 사용하여 생성됩니다.

```
...
endparams
#------------------------------------------------------------------------
# VSN Directives
vsns
allfiles.1 dk qfs200
allfiles.2 tp VOL0[0-1][0-9]
endvsns
:wq
[client1]root@solaris:~#
```

6. 이제 SAM-Remote 서버에서 아카이빙 구성 검증을 수행합니다.

SAM-Remote 서버에서 아카이빙 구성 검증

1. SAM-Remote 서버 호스트에 *root*로 로그인합니다.

   예제에서는 SAM-Remote 서버의 이름이 *server1*로 지정됩니다.

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

2. 서버에서 Oracle HSM 프로세스를 시작합니다. *samd start* 명령을 사용합니다.

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

3. 서버 호스트에서 공유 장치 서버의 상태를 확인합니다. *samcmd s* 명령을 사용합니다.

   예제에서는 장비 순서 번호가 *500*인 SAM-Remote 서버 장비(유형 *ss*)가 *on* 상태이고 정상적으로 작동하고 있습니다.

```
[server1]root@solaris:~# samcmd s
Device status samcmd     6.0  11:20:34 Feb 20 2015
samcmd on server1
ty    eq  state   device_name                    fs     status
```

```
rb    800 on      /dev/scsi/changer/c1t0d5         800    m--------r
tp    801 on      /dev/rmt/0cbn                    800    ---------p
  empty
tp    802 on      /dev/rmt/1cbn                    800    ---------p
  empty
tp    803 on      /dev/rmt/2cbn                    800    ---------p
  empty
tp    804 on      /dev/rmt/3cbn                    800    ---------p
  empty
ss    500 on      /etc/opt/SUNWsamfs/samremote     ss500  -------o-r
[server1]root@solaris:~#
```

4. 공유 장치 서버가 *on* 상태가 아닌 경우 해당 서버가 서버 호스트의 */etc/opt/SUNWsamfs/mcf* 파일에 올바르게 정의되어 있는지 확인합니다. */etc/opt/SUNWsamfs/samremote* 파일이 올바르고 올바른 위치에 있는지 확인하십시오.

   "SAM-Remote 서버의 **mcf** 파일에서 원격 공유 아카이빙 장비 정의" 및 "**samremote** 서버 구성 파일 만들기" 절차를 참조하십시오.

5. 서버에서 SAM-Remote 클라이언트의 연결 상태를 확인합니다. *samcmd R* 명령을 사용합니다.

   예제에서는 *client1*과 *client2* 모두 *0005* 즉, *connected* 상태입니다. *0004*는 연결되지 않은 상태를 나타냅니다.

```
[server1]root@solaris:~# samcmd R
Remote server eq: 500 addr: 00003858 samcmd 6.0  11:20:44 Feb 20 2015
samcmd on server1
message:
Client IPv4: client1 192.10.10.3 port - 5000
 client index - 0 port - 31842 flags - 0005 connected
Client IPv4: client2 10.1.229.97 port - 5000
 client index - 1 port - 32848 flags - 0005 connected
[server1]root@solaris:~#
```

6. 공유 장치 클라이언트가 연결되어 있지 않은 경우(상태 *0004*) 네트워크 연결을 확인합니다. 서버와 클라이언트에서 서로 상대측의 호스트 이름과 주소를 확인할 수 있는지 확인합니다. 서버와 클라이언트가 서로 연결되는지 확인합니다.

   예제에서는 *ssh*를 *getent* 및 *ping* 명령과 함께 사용하여 SAM-Remote 구성의 각 호스트 간 연결을 확인합니다.

```
[server1]root@solaris:~# getent hosts client1
192.10.10.3  client1
[server1]root@solaris:~# getent hosts 192.10.10.3
192.10.10.3  client1
```

```
[server1]root@solaris:~# ping 192.10.10.3
192.10.10.31 is alive
[server1]root@solaris:~# getent hosts client2
10.1.229.97 client2
[server1]root@solaris:~# getent hosts 10.1.229.97
10.1.229.97 client2
[server1]root@solaris:~# ping 10.1.229.97
192.10.10.31 is alive
[server1]root@solaris:~# ssh root@client1
Password: ...
[client1]root@solaris:~# getent hosts server1
192.10.201.12  server1
...
[client1]root@solaris:~# exit
[server1]root@solaris:~# ssh root@client2
Password: ...
[client2]root@solaris:~# getent hosts server1
192.10.201.12  server1
...
[client2]root@solaris:~# exit
[server1]root@solaris:~#
```

7. 공유 장치 클라이언트가 연결되지 않는 경우(상태 *0004*) 해당 클라이언트가 클라이언트 호스트의 */etc/opt/SUNWsamfs/mcf* 파일에 올바르게 정의되어 있는지 확인합니다. 서버 호스트가 */etc/opt/SUNWsamfs/family-set-name* 파일에서 올바르게 식별되고 파일이 클라이언트 호스트의 올바른 위치에 있는지 확인합니다. 그런 다음 클라이언트 호스트가 서버 호스트의 */etc/opt/SUNWsamfs/samremote* 파일에서 올바르게 식별되는지 확인합니다.

   "SAM-Remote 클라이언트의 MCF 파일에서 원격 아카이브 장비 정의" 및 "SAM-Remote 클라이언트 구성 파일 만들기" 절차를 참조하십시오.
8. 클라이언트에서 서버 호스트가 */etc/opt/SUNWsamfs/family-set-name* 파일에서 올바르게 식별되고 파일이 클라이언트 호스트의 올바른 위치에 있는지 확인합니다.

   "SAM-Remote 클라이언트 구성 파일 만들기" 절차를 참조하십시오.
9. 공유 장치 클라이언트가 연결되지 않고(상태 *0004*) 클라이언트측 구성 파일에 문제가 없는 경우 서버를 확인합니다. 클라이언트 호스트가 서버 호스트의 */etc/opt/SUNWsamfs/samremote* 파일에서 올바르게 식별되는지 확인합니다.

   "**samremote** 서버 구성 파일 만들기" 절차를 참조하십시오.
10. 서버에서 각 클라이언트가 공유 테이프 라이브러리에 대한 카탈로그에 액세스할 수 있고 사용 가능한 볼륨이 표시되는지 확인합니다. *samcmd v equipment-number* 명령을 사용합니다. 여기서 *equipment-number*는 클라이언트의 *mcf* 파일에서 SAM-Remote 클라이언트 장비에 지정한 장비 순서입니다.

예제에서는 *client1*을 확인하므로, *400*은 SAM-Remote 클라이언트 장비 */etc/opt/SUNWsamfs/sc400*에 대한 장비 번호입니다. 출력에 *client1*에서 액세스할 수 있는 볼륨 *VOL000* - *VOL019*가 올바르게 나열됩니다.

```
[server1]root@solaris:~# samcmd v 400
Robot catalog samcmd     6.0  12:20:40 Feb 20 2015
samcmd on server1
Robot VSN catalog by slot       : eq 400
slot     access time  count use flags         ty vsn
   3     none         0     0%  -il-o-b-----  li VOL000
   7     none         0     0%  -il-o-b-----  li VOL001
...
  24     none         0     0%  -il-o-b-----  li VOL019
[server1]root@solaris:~#
```

11. 공유 장비 클라이언트에 올바른 볼륨이 표시되지 않는 경우 호스트 파일을 확인합니다. 서버 호스트에서 지정된 볼륨이 */etc/opt/SUNWsamfs/samremote* 파일에서 올바르게 식별되는지 확인합니다. 클라이언트 호스트에서 */etc/opt/SUNWsamfs/family-set-name* 파일이 서버 호스트를 올바르게 식별하는지 확인합니다.

    "**samremote** 서버 구성 파일 만들기" 및 "SAM-Remote 클라이언트 구성 파일 만들기" 절차를 참조하십시오.
12. 이제 각 SAM-Remote 클라이언트에서 아카이빙 구성 검증을 수행합니다.

각 SAM-Remote 클라이언트에서 아카이빙 구성 검증

각 SAM-Remote 클라이언트에 대해 다음과 같이 하십시오.

1. SAM-Remote 클라이언트 호스트에 *root*로 로그인합니다.

   예제에서는 SAM-Remote 클라이언트의 이름이 *client1*로 지정됩니다.

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

2. 클라이언트 호스트에서 Oracle HSM 프로세스를 시작합니다. *samd start* 명령을 사용합니다.

```
[client1]root@solaris:~# samd start
[client1]root@solaris:~#
```

3. 클라이언트 호스트에서 공유 장치 클라이언트의 상태를 확인합니다. *samcmd s* 명령을 사용합니다.

   예제에서는 장비 순서 번호가 *400*인 SAM-Remote 클라이언트 장비(유형 *sc*)가 *on* 상태이고 정상적으로 작동하고 있습니다.

```
[client1]root@solaris:~# samcmd s
Device status samcmd     6.0  12:20:49 Feb 20 2015
samcmd on client1
ty    eq  state   device_name                    fs     status
sc    400 on      /etc/opt/SUNWsamfs/sc400       sc400  -------o-r
```

4. 공유 장치 클라이언트가 *on* 상태가 아닌 경우 *sc* 장치가 올바르게 정의되어 있는지 확인합니다. 클라이언트 호스트에서 */etc/opt/SUNWsamfs/mcf* 파일을 확인하고 */etc/opt/SUNWsamfs/family-set-name* 파일이 올바르며 올바른 위치에 있는지 확인합니다.

   "SAM-Remote 클라이언트의 MCF 파일에서 원격 아카이브 장비 정의" 및 "SAM-Remote 클라이언트 구성 파일 만들기" 절차를 참조하십시오.
5. 클라이언트 호스트에서 */etc/opt/SUNWsamfs/archiver.cmd* 파일에 원격 매체에 대한 올바른 볼륨 일련 번호가 지정되어 있는지 확인합니다. *archiver -A* 명령을 사용하여 파일을 나열합니다.

   예제에서는 *client1*을 구성합니다. 복사본 *allfiles.2*가 *samremote* 서버 구성 파일에 지정된 *VOL000-VOL019* 범위에 속하는 원격 테이프 볼륨 중 하나를 사용하여 생성됩니다.

```
[client1]root@solaris:~# archiver -A
Reading '/etc/opt/SUNWsamfs/archiver.cmd'.
1: # archiver.cmd
2: #----------------------------------------------------------------------
3: # Global Directives
4: archivemeta = off
5: examine = noscan
...
30: #----------------------------------------------------------------------
31: # VSN Directives
32: vsns
33: allfiles.1 dk qfs200
34: allfiles.2 tp VOL0[0-1][0-9]
36: endvsns
[client1]root@solaris:~#
```

6. *archiver.cmd* 파일에 불일치가 있는 경우 계속하기 전에 해결합니다.
7. 재활용을 구성하려면 SAM-Remote에 대한 재활용 구성을 참조합니다.

SAM-Remote에 대한 재활용 구성

SAM-Remote를 구성할 때 한 호스트에서 재활용해도 다른 호스트의 유효한 데이터가 삭제되지 않도록 해야 합니다. SAM-Remote 서버에서 구성하는 모든 재활용 지시어는 서버에서

자체 아카이브 세트에 대해 사용하는 매체만 재활용해야 합니다. 서버에서는 SAM-Remote 클라이언트에서 사용할 수 있는 매체 볼륨을 재활용하지 않아야 합니다. 마찬가지로 SAM-Remote 클라이언트에서 구성하는 재활용 지시어는 서버에서 사용할 수 있도록 지정된 볼륨 또는 로컬에 아카이브된 데이터를 보관하는 매체만 재활용해야 합니다.

SAM-Remote 환경에서 리사이클러를 사용하기 전에 재활용 프로세스를 완벽하게 이해해야 합니다. 따라서 "재활용" 및 *sam-recycler*, *archiver.cmd*, *recycler.cmd* 및 *recycler.sh* 매뉴얼 페이지를 읽어 보시기 바랍니다.

그런 다음 재활용 방법에 익숙해지면 아래 작업을 수행하십시오.

- SAM-Remote 서버에서 재활용 구성
- SAM-Remote 클라이언트에서 재활용 구성.

SAM-Remote 서버에서 재활용 구성

SAM-Remote 서버에 호스트되는 파일 시스템에 대한 재활용을 구성해야 하는 경우 다음과 같이 하십시오.

1. SAM-Remote 서버에 *root*로 로그인합니다.

   예제에서는 SAM-Remote 서버의 이름이 *server1*로 지정됩니다.

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

2. 텍스트 편집기에서 */etc/opt/SUNWsamfs/archiver.cmd* 파일을 엽니다. 아래로 스크롤하여 *params* 섹션을 찾습니다.

   예제에서는 *vi* 편집기에서 파일을 엽니다.

```
[client1]root@solaris:~# vi /etc/opt/SUNWsamfs/archiver.cmd
...
#------------------------------------------------------------------------
# Copy Parameter Directives
params
allsets -sort path -offline_copy direct
allfiles.1 -startage 10m -startsize 500M -drives 10
allfiles.2 -startage 24h -startsize 20G -drives 2 -reserve set
endparams
```

3. 아카이브 세트별 리사이클러 지시어를 *archive-set directive-list* 형식으로 입력합니다. 여기서 archive-set는 아카이브 세트 중 하나이고 *directive-list*는 지시어 이름/값 쌍의 공백으로 구분된 목록입니다. 전체 재활용 지시어 목록은 *archiver.cmd* 매뉴얼 페이지를 참조하십시오.

   SAM-Remote를 사용하는 경우 *archiver.cmd* 파일의 *params* 절에서 아카이브 세트별 재활용을 구성해야 합니다. 라이브러리별로 재활용을 지정할 수 없습니다.

예제에서는 아카이브 세트 *allfiles.1* 및 *allfiles.2*에 대한 재활용 지시어를 추가합니다. *-recycle_mingain 90* 지시어는 볼륨 용량의 최소 90% 이상을 복구할 수 있는 경우에만 볼륨을 재활용합니다. *-recycle_hwm 60* 지시어는 이동식 매체 용량의 60%가 사용된 경우 재활용을 시작합니다. *-recycle_vsncount 1*은 재활용에 대해 이동식 매체 볼륨을 한 번에 하나만 예약합니다.

```
#------------------------------------------------------------------
#  Copy Parameters Directives
params
allsetsallfiles. -sort path -offline_copy direct
allfiles.1 -startage 10m -startsize 500M -drives 10
allfiles.1 -recycle_mingain 90
allfiles.2 -startage 24h -startsize 20G -drives 2 -reserve set offline_copy none
allfiles.2 -recycle_hwm 60 -recycle_mingain 90 -recycle_vsncount 1
endparams
```

SAM-Remote 서버에서 정의된 재활용 지시어는 서버가 고유한 아카이브 세트에 사용하는 아카이브 볼륨에만 적용됩니다. 클라이언트에서 액세스할 수 있는 볼륨에는 서버의 재활용 지시어가 적용되지 않습니다.

예제에서 *allfiles.2* 복사본에 대한 서버의 재활용 지시어는 *VSN Directives* 절에서 서버 사용으로 나열된 테이프 볼륨 *VOL100-VOL199*에 적용됩니다. *client1*에 대해 예약된 *VOL000-VOL019* 볼륨이나 *client2*에 대해 예약된 *VOL020-VOL039* 볼륨에는 서버의 재활용 지시어가 적용되지 않습니다.

```
...
endparams
#------------------------------------------------------------------------
# VSN Directives
vsns
allfiles.1 dk DISKVOL1
allfiles.2 tp VOL1[0-9][0-9]
endvsns
```

4. *archiver.cmd* 파일을 저장하고 편집기를 닫습니다.

```
...
endvsns
:wq
[server1]root@solaris:~#
```

5. 서버에서 텍스트 편집기를 사용하여 *recycler.cmd* 파일을 만듭니다. 리사이클러 로그의 경로와 파일 이름을 지정합니다.

예제에서는 *vi* 편집기를 사용합니다. 로그 파일의 기본 위치를 지정합니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/recycler.cmd
logfile = /var/adm/recycler.log
```

6. 서버의 *recycler.cmd* 파일에서 *no-recycle media-type volumes* 형식의 지시어를 추가합니다. 여기서 *media-type*은 부록 A. 장비 유형 용어집에 지정된 매체 유형 중 하나이고 *volumes*는 SAM-Remote 클라이언트에 지정한 모든 아카이브 스토리지 볼륨에 대한 볼륨 일련 번호를 지정하며 공백으로 구분된 목록 또는 정규 표현식입니다. 파일을 저장하고 편집기를 닫습니다.

   no-recycle 지시어는 클라이언트 전용 스토리지 리소스를 추가적으로 보호합니다. 또한 호스트 재활용 프로세스에 지정된 볼륨을 건너뛰도록 명시적으로 지시합니다.

   예제에서는 *VOL000-VOL019* 및 *VOL020-VOL039* 범위의 매체 유형 *tp*(테이프) 볼륨에 대해 *no-recycle* 지시어를 추가합니다.

```
[server1]root@solaris:~# vi /etc/opt/SUNWsamfs/recycler.cmd
logfile = /var/opt/SUNWsamfs/recycler/recycler.log
no_recycle tp VOL0[0-1][0-9] VOL0[2-3][0-9]
:wq
[server1]root@solaris:~#
```

7. 이제 SAM-Remote 클라이언트에서 재활용 구성을 수행합니다.

SAM-Remote 클라이언트에서 재활용 구성

각 클라이언트에 대해 다음과 같이 하십시오.

1. SAM-Remote 클라이언트에 *root*로 로그인합니다.

   예제에서는 SAM-Remote 클라이언트의 이름이 *client1*로 지정됩니다.

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

2. 클라이언트에서 */etc/opt/SUNWsamfs/archiver.cmd* 파일을 텍스트 편집기에서 열고 아래로 스크롤하여 복사본 *params* 섹션을 찾습니다.

   예제에서는 *vi* 편집기에서 파일을 엽니다.

```
[client1]root@solaris:~# vi /etc/opt/SUNWsamfs/archiver.cmd
...
#------------------------------------------------------------------------
# Copy Parameters Directives
params
```

```
allsets -sort path -offline_copy stageahead
allfiles.1 -startage 6h  -startsize 6G  -startcount 500000
allfiles.2 -startage 24h -startsize 20G -startcount 500000 -archmax 24G
endparams
#------------------------------------------------------------------------
# VSN Directives
vsns
allfiles.1 dk qfs200
allfiles.2 tp VOL0[0-1][0-9]
endvsns
```

3. *archiver.cmd* 파일의 *params* 섹션에 아카이브 세트별 리사이클러 지시어를 *archive-set directive-list* 형태로 입력합니다. 여기서 archive-set는 아카이브 세트 중 하나이고 *directive-list*는 지시어 이름/값 쌍의 공백으로 구분된 목록입니다(재활용 지시어 목록은 *archiver.cmd* 매뉴얼 페이지 참조). 그런 다음 파일을 저장하고 편집기를 닫습니다.

   SAM-Remote를 사용하는 경우 *archiver.cmd* 파일의 *params* 절에서 아카이브 세트별 재활용을 구성해야 합니다. 라이브러리별로 재활용을 지정할 수 없습니다.

   예제에서는 아카이브 세트 *allfiles.1* 및 *allfiles.2*에 대한 재활용 지시어를 추가합니다. *-recycle_mingain 90* 지시어는 볼륨 용량의 최소 90% 이상을 복구할 수 있는 경우에만 볼륨을 재활용합니다. *-recycle_hwm 60* 지시어는 이동식 매체 용량의 60%가 사용된 경우 재활용을 시작합니다. *-recycle_vsncount 1* 지시어는 재활용에 대해 이동식 매체 볼륨을 한 번에 하나만 예약합니다.

```
#------------------------------------------------------------------------
# Copy Parameters Directives
params
allsets -sort path -offline_copy stageahead
allfiles.1 -startage 6h  -startsize 6G  -startcount 500000
allfiles.1 -recycle_mingain 90
allfiles.2 -startage 24h -startsize 20G -startcount 500000 -archmax 24G
allsets.2 -recycle_hwm 60 -recycle_mingain 90 -recycle_vsncount 1
endparams
```

   클라이언트에 정의된 재활용 지시어는 클라이언트에서 자체 아카이브 세트에 사용하는 매체에만 적용됩니다. 예제에서 복사본 *allfiles.2*에 대한 클라이언트의 재활용 지시어는 *VOL000-VOL019* 범위의 서버 제공 원격 테이프 볼륨에 적용됩니다. client2에 대해 예약된 *VOL020-VOL039* 범위의 볼륨이나 서버에 대해 예약된 *VOL100-VOL119* 범위의 볼륨에는 적용되지 않습니다.

```
...
endparams
```

```
#-------------------------------------------------------------------------
# VSN Directives
vsns
allfiles.1 dk qfs200
allfiles.2 tp VOL0[0-1][0-9]
endvsns
:wq
[client1]root@solaris:~#
```

4. *archiver.cmd* 파일을 저장하고 편집기를 닫습니다.

```
...
endvsns
:wq
[client]root@solaris:~#
```

5. 클라이언트에서 텍스트 편집기를 사용하여 *recycler.cmd* 파일을 만듭니다. 리사이클러 로그의 경로와 파일 이름을 지정합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   클라이언트에서는 서버 또는 *client2*에서 사용되는 아카이브 매체에 액세스하지 못하도록 서버 및 클라이언트를 구성했습니다. 따라서 *no-recyle* 지시어를 추가할 필요가 없습니다.

   예제에서는 *vi* 편집기를 사용합니다. 로그 파일의 기본 위치를 지정합니다.

```
[client1]root@solaris:~# vi /etc/opt/SUNWsamfs/recycler.cmd
logfile = /var/adm/recycler.log
:wq
[client1]root@solaris:~#
```

6. 모든 SAM-Remote 클라이언트가 구성될 때까지 이 절차를 반복합니다.
7. *sam-recycler -dvxn* 명령을 입력합니다. 여기서 매개변수는 다음과 같은 결과를 제공합니다.
   - *-d*는 재활용을 위해 각 볼륨을 선택하거나 선택하지 않은 이유를 나타내는 볼륨 선택 메시지를 표시합니다.
   - *-v*는 재활용하도록 표시되고 이동해야 하는 각 볼륨에 있는 파일을 나열합니다.
   - *-x*는 볼륨 레이블에 표시된 시간보다 더 오래되어 복구할 수 없는 아카이브 복사본이 나열될 경우 오류를 반환하고 중지합니다.
   - *-n*은 실제 재활용을 금지합니다. 재활용 프로세스는 *archiver.cmd* 파일의 모든 아카이브 세트 정의에 *-recycle_ignore*가 포함되어 있는 경우처럼 작동하므로 재활용 구성을 비파괴적으로 테스트할 수 있습니다.
8. 모든 SAM-Remote 클라이언트와 서버를 구성한 후 사이드밴드 데이터베이스 기능을 사용하려면 10장. 보고 데이터베이스 구성으로 이동합니다.

9. 그렇지 않은 경우 11장. 알림 및 로깅 구성으로 이동합니다.

9

9장. 고가용성 솔루션 준비

Oracle Hierarchical Storage Manager and StorageTek QFS Software 고가용성 구성은 무중단 파일 시스템 및 아카이빙 서비스를 유지 관리하도록 설계되었습니다. 고가용성 솔루션에서 Oracle Hierarchical Storage Manager 또는 QFS 소프트웨어는 Oracle Solaris Cluster 소프트웨어, 중복 하드웨어 및 중복 통신과 통합되어 있습니다. 따라서 호스트 시스템 또는 구성 요소가 실패하거나 관리자에 의해 서비스가 중단된 경우 Oracle HSM 서비스는 사용자 및 응용 프로그램이 액세스할 수 있는 대체 호스트로 자동으로 페일오버됩니다. 따라서 고가용성 구성은 장비 및 시스템 오류로 인한 작동 중지 시간을 최소화합니다.

고가용성 구성은 복잡하지만 예측할 수 없는 상호 작용과 잠재적인 데이터 손상을 방지하도록 신중하게 설계 및 배치해야 합니다. 따라서 이 장에서는 지원되는 구성 설명부터 시작합니다. 이 절을 학습하고 나서 가용성 요구 사항에 가장 적합한 구성을 선택합니다. 그런 다음 이후의 절에서는 선택한 구성을 설정하는 방법에 대해 설명합니다.

공유 Oracle Solaris Cluster 구성에서 하드웨어 아키텍처를 혼합할 수 없습니다. 모든 노드는 SPARC 아키텍처, x86-64 아키텍처(Solaris 11.1 전용) 또는 32비트 x86 아키텍처(Solaris 10 이전)를 사용해야 합니다.

지원되는 고가용성 구성 이해

클러스터화된 다중 호스트 솔루션에서는 저장된 데이터의 무결성을 보장하기 위해 파일 시스템, 응용 프로그램, 운영체제, 클러스터링 소프트웨어 및 스토리지 간의 상호 작용을 신중하게 제어해야 합니다. 따라서 복잡성 및 잠재적 위험을 최소화하기 위해 4가지 특정 배치 요구 사항 세트에 맞게 지원되는 고가용성 Oracle HSM 구성이 제공됩니다.

- HA-QFS, 고가용성 QFS 비공유 독립형 파일 시스템 구성
- HA-COTC, 고가용성 메타데이터 서버가 있는 QFS 공유 파일 시스템
- HA-SAM, 고가용성, 아카이빙, QFS 공유 파일 시스템 구성
- SC-RAC, Oracle RAC를 위한 고가용성 QFS 공유 파일 시스템 구성.

HA-QFS, 고가용성 QFS 비공유 독립형 파일 시스템 구성

HA-QFS(고가용성 QFS) 구성은 호스트 오류가 발생한 경우에도 QFS 비공유 독립형 파일 시스템을 계속 액세스할 수 있게 합니다. 파일 시스템은 Solaris Cluster 소프트웨어가 *SUNW.HAStoragePlus* 유형의 리소스로 관리하는 2노드 클러스터에서 두 노드 모두에 구성됩니다. 그러나 특정 시점에 하나의 노드만 QFS 파일 시스템을 마운트합니다. 파일 시스템을 마운트하는 노드가 실패할 경우 클러스터링 소프트웨어는 페일오버를 자동으로 시작하고 나머지 노드에 파일 시스템을 다시 마운트합니다.

클라이언트는 활성 클러스터 노드를 파일 서버로 사용하여 NFS(네트워크 파일 시스템), HA-NFS(고가용성 NFS) 또는 SMB/CIFS 공유를 통해 데이터에 액세스합니다.

구현 지침은 “고가용성 QFS 비공유 파일 시스템”을 참조하십시오.

HA-COTC, 고가용성 메타데이터 서버가 있는 QFS 공유 파일 시스템

HA-COTC(High Availability-Clients Outside the Cluster) 구성은 서버가 실패한 경우에도 QFS 파일 시스템 클라이언트가 해당 데이터에 계속 액세스할 수 있도록 QFS 메타데이터 서버의 가용성을 유지 관리합니다. 파일 시스템은 공유됩니다. QFS 활성/잠재적 메타데이터 서버는 Solaris Cluster 소프트웨어가 관리하는 2노드 클러스터에서 호스트됩니다. *SUNW.qfs* 유형의 Oracle HSM 고가용성 리소스는 클러스터 내의 공유 파일 시스템 서버에 대한 페일오버를 관리합니다. 모든 클라이언트는 클러스터 외부에서 호스팅됩니다. 클러스터화된 서버는 메타데이터의 가용성을 보장하고 I/O 라이센스를 발행하며 파일 시스템의 일관성을 유지 관리합니다.

활성 메타데이터 서버를 호스팅하는 노드가 실패할 경우 Solaris Cluster 소프트웨어는 정상적인 노드에서 잠재적 MDS를 자동으로 활성화하고 페일오버를 시작합니다. QFS 파일 시스템은 공유되므로 새로 활성화된 메타데이터 서버에 이미 마운트되어 있고 클라이언트에 마운트된 상태로 남아 있습니다. 클라이언트는 계속해서 메타데이터 업데이트 및 I/O 임대를 수신하므로 파일 시스템이 중단 없이 계속될 수 있습니다.

HA-COTC 구성에서는 물리적으로 별개인 *mm* 메타데이터 장치 및 *mr* 데이터 장치가 있는 고성능 *ma* 파일 시스템을 사용해야 합니다. 범용 *ms* 파일 시스템 및 *md* 장치는 지원되지 않습니다.

표준 NFS(네트워크 파일 시스템) 또는 SMB/CIFS를 사용하여 Oracle HSM을 실행하지 않는 클라이언트와 HA-COTC 파일 시스템을 공유할 수 있습니다. 그러나 HA-NFS는 지원되지 않습니다.

구현 지침은 “고가용성 QFS 공유 파일 시스템, 클러스터 외부의 클라이언트”를 참조하십시오.

HA-SAM, 고가용성, 아카이빙, QFS 공유 파일 시스템 구성

HA-SAM(High-Availability Oracle Hierarchical Storage Manager) 구성은 서버 호스트가 실패한 경우에도 QFS 메타데이터 서버 및 Oracle Hierarchical Storage Manager 응용 프로그램이 계속 작동하도록 보장하여 아카이빙 파일 시스템의 가용성을 유지 관리합니다. 파일 시스템은 Solaris Cluster 소프트웨어가 관리하는 2노드 클러스터에서 호스트되는 활성/잠재적 QFS 메타데이터 서버 간에 공유됩니다. *SUNW.qfs* 유형의 Oracle HSM 고가용성 리소스는 서버에 대한 페일오버를 관리합니다.

활성 Oracle HSM 메타데이터 서버 노드가 실패할 경우 클러스터링 소프트웨어는 잠재적 메타데이터 서버 노드를 자동으로 활성화하고 페일오버를 시작합니다. QFS 파일 시스템이 공유되고 이미 모든 노드에 마운트되어 있으므로 데이터 및 메타데이터에 중단 없이 계속 액세스할 수 있습니다.

클라이언트는 활성 클러스터 노드를 파일 서버로 사용하여 HA-NFS(고가용성 네트워크 파일 시스템), NFS 또는 SMB/CIFS 공유를 통해 데이터에 액세스합니다.

구현 지침은 "고가용성 Oracle HSM 공유 아카이빙 파일 시스템"을 참조하십시오.

SC-RAC, Oracle RAC를 위한 고가용성 QFS 공유 파일 시스템 구성

SC-RAC(Solaris Cluster-Oracle Real Application Cluster) 구성은 QFS 파일 시스템을 사용하는 고가용성 데이터베이스 솔루션을 지원합니다. RAC 소프트웨어는 I/O 요청을 조정하고 작업 로드를 분배하며 클러스터의 노드에서 실행되는 여러 Oracle 데이터베이스 인스턴스에 대한 일관된 단일 데이터베이스 파일 세트를 유지 관리합니다. SC-RAC 구성에서 Oracle 데이터베이스, Oracle RAC(Real Application Cluster) 및 QFS 소프트웨어는 클러스터에 있는 둘 이상의 노드에서 실행됩니다. Solaris Cluster 소프트웨어는 클러스터를 *SUNW.qfs* 유형의 리소스로 관리합니다. 하나의 노드가 QFS 공유 파일 시스템의 MDS(메타데이터 서버)로 구성됩니다. 나머지 노드는 파일 시스템을 클라이언트로 공유하는 잠재적 메타데이터 서버로 구성됩니다. 활성 메타데이터 서버 노드가 실패할 경우 Solaris Cluster 소프트웨어는 정상적인 노드에서 잠재적 메타데이터 서버를 자동으로 활성화하고 페일오버를 시작합니다. QFS 파일 시스템이 공유되고 이미 모든 노드에 마운트되어 있으므로 데이터에 중단 없이 계속 액세스할 수 있습니다.

고가용성 QFS 비공유 파일 시스템

HA-QFS(고가용성 QFS) 파일 시스템을 구성하려면 *SUNW.HAStoragePlus* 유형의 리소스로 관리되는 2노드 Solaris Cluster에서 동일한 호스트 2개를 설정합니다. 그런 다음 두 노드 모두에서 QFS 비공유 파일 시스템을 구성합니다. 특정 시점에 하나의 노드만 파일 시스템을 마운트합니다. 그러나 하나의 노드가 실패할 경우 클러스터링 소프트웨어는 페일오버를 자동으로 시작하고 작동 중인 노드에 파일 시스템을 다시 마운트합니다.

HA-QFS(고가용성 QFS) 파일 시스템을 설정하려면 다음과 같이 하십시오.

- 두 클러스터 노드 모두에서 비공유 QFS 파일 시스템 만들기
- 고가용성 QFS 파일 시스템 구성
- 필요한 경우 HA-NFS(고가용성 네트워크 파일 시스템) 공유 구성

  HA-NFS 설정에 대한 자세한 지침은 *Oracle Solaris Cluster* 온라인 설명서 라이브러리에 포함된 *Oracle Solaris Cluster Data Service for Network File System (NFS) Guide*에 나와 있습니다.

두 클러스터 노드 모두에서 비공유 QFS 파일 시스템 만들기

1. 클러스터 노드 중 하나에 *root*로 로그인합니다.

   예제에서는 호스트가 *qfs1mds-node1* 및 *qfs1mds-node2*입니다. *qfs1mds-node1* 호스트에 로그인합니다.

```
[qfs1mds-node1]root@solaris:~#
```

2. 원하는 QFS 파일 시스템을 호스트에서 구성합니다(마운트하지는 않음).

   "범용 **ms** 파일 시스템 구성" 또는 "고성능 **ma** 파일 시스템 구성"의 지침에 따라 파일 시스템을 구성합니다. HA-QFS 구성은 QFS 공유 파일 시스템을 지원하지 않습니다.
3. 나머지 클러스터 노드에 *root*로 로그인합니다.

   예제에서는 *ssh*를 사용하여 *qfs1mds-node2* 호스트에 로그인합니다.

```
[qfs1mds-node1]root@solaris:~# ssh root@qfs1mds-node2
Password:
[qfs1mds-node2]root@solaris:~#
```

4. 두번째 노드에서 동일한 QFS 파일 시스템을 구성합니다.
5. 이제 고가용성 QFS 파일 시스템 구성을 수행합니다.

고가용성 QFS 파일 시스템 구성

다음과 같이 하십시오.

1. 클러스터 노드 중 하나에 *root*로 로그인합니다.

   예제에서는 호스트가 *qfs1mds-node1* 및 *qfs1mds-node2*입니다. *qfs1mds-node1* 호스트에 로그인합니다.

```
[qfs1mds-node1]root@solaris:~#
```

2. 아직 정의하지 않은 경우 Solaris Cluster 소프트웨어에 대한 *SUNW.HAStoragePlus* 리소스 유형을 정의합니다. *clresourcetype register SUNW.HAStoragePlus* 명령을 사용합니다.

   *HAStoragePlus*는 디스크 장치 그룹, 클러스터 파일 시스템 및 로컬 파일 시스템 간의 종속성을 정의 및 관리하는 Solaris Cluster 리소스 유형입니다. 이 리소스 유형은 페일오버 이후 데이터 서비스의 시작을 조정하여 서비스를 다시 시작하려고 할 때 필요한 모든 구성 요소가 준비되도록 합니다. 자세한 내용은 *SUNW.HAStoragePlus* 매뉴얼 페이지를 참조하십시오.

```
[qfs1mds-node1]root@solaris:~# clresourcetype register SUNW.HAStoragePlus
[qfs1mds-node1]root@solaris:~#
```

3. *SUNW.HAStoragePlus* 유형의 새 Solaris Cluster 리소스와 이를 포함할 새 리소스 그룹을 만듭니다. */usr/global/bin/clresource create -g resource-group -t SUNW.HAStoragePlus -x FilesystemMountPoints=/global/mount-point -x FilesystemCheckCommand=/bin/true QFS-resource* 명령을 사용합니다. 설명:
   - *resource-group*은 파일 시스템 리소스 그룹에 사용하도록 선택한 이름입니다.
   - *mount-point*는 QFS 파일 시스템이 마운트되는 디렉토리입니다.

- *QFS-resource*는 *SUNW.HAStoragePlus* 리소스에 사용하도록 선택한 이름입니다.

예제에서는 마운트 지점 디렉토리 */global/qfs1* 및 *SUNW.HAStoragePlus* 리소스 *haqfs*가 있는 리소스 그룹 *qfsrg*를 만듭니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[qfs1mds-node1]root@solaris:~# clresource create -g qfsrg -t SUNW.HAStoragePlus /
-x FilesystemMountPoints=/global/hsmqfs1/qfs1 /
-x FilesystemCheckCommand=/bin/true haqfs
[qfs1mds-node1]root@solaris:~#
```

4. 클러스터의 노드를 표시합니다. *clresourcegroup status* 명령을 사용합니다.

   예제에서는 QFS 파일 시스템 호스트 노드가 *qfs1mds-1* 및 *qfs1mds-2*입니다. *qfs1mds-1* 노드는 *Online* 상태이므로 파일 시스템을 마운트하고 *qfsrg* 리소스 그룹을 호스팅하는 주 노드입니다.

```
[qfs1mds-node1]root@solaris:~# clresourcegroup status
=== Cluster Resource Groups ===
Group Name     Node Name     Suspended    Status
----------     ---------     ---------    ------
qfsrg          qfs1mds-1     No           Online
               qfs1mds-2     No           Offline
[qfs1mds-node1]root@solaris:~#
```

5. 리소스 그룹을 보조 노드로 이동하여 리소스 그룹이 올바르게 페일오버되는지 확인합니다. Solaris Cluster 명령 *clresourcegroup switch -n node2 group-name*을 사용합니다. 여기서 *node2*는 보조 노드의 이름이고 *group-name*은 HA-QFS 리소스 그룹에 사용하도록 선택한 이름입니다. 그런 다음 *clresourcegroup status*를 사용하여 결과를 확인합니다.

   예제에서는 *haqfs* 리소스 그룹을 *qfs1mds-node2*로 이동하고 이 리소스 그룹이 지정된 노드에서 온라인 상태가 되는지 확인합니다.

```
[qfs1mds-node1]root@solaris:~# clresourcegroup switch -n qfs1mds-node2 qfsrg
[qfs1mds-node1]root@solaris:~# clresourcegroup status
=== Cluster Resource Groups ===
Group Name     Node Name     Suspended    Status
----------     ---------     ---------    ------
qfsrg          qfs1mds-1     No           Offline
               qfs1mds-2     No           Online
[qfs1mds-node1]root@solaris:~#
```

6. 리소스 그룹을 다시 주 노드로 이동합니다. Solaris Cluster 명령 *clresourcegroup switch -n node1 group-name*을 사용합니다. 여기서 *node1*은 주 노드의 이름이고

*group-name*은 HA-QFS 리소스 그룹에 사용하도록 선택한 이름입니다. 그런 다음 *clresourcegroup status*를 사용하여 결과를 확인합니다.

예제에서는 *qfsrg* 리소스 그룹을 *qfs1mds-node1*로 성공적으로 다시 이동합니다.

```
[qfs1mds-node1]root@solaris:~# clresourcegroup switch -n qfs1mds-node1 qfsrg
[qfs1mds-node1]root@solaris:~# clresourcegroup status
=== Cluster Resource Groups ===
Group Name  Node Name       Suspended   Status
----------  -------------   ---------   ------
qfsrg       qfs1mds-node1   No          Online
            qfs1mds-node2   No            Offline
[qfs1mds-node1]root@solaris:~#
```

7. HA-NFS(고가용성 네트워크 파일 시스템) 공유를 구성해야 할 경우 지금 구성합니다. 지침은 Oracle Solaris Cluster 온라인 설명서 라이브러리에 포함된 Oracle Solaris Cluster Data Service for Network File System (NFS) Guide를 참조하십시오.
8. 사이드밴드 데이터베이스 기능을 사용할 계획인 경우 10장. 보고 데이터베이스 구성을 참조하십시오.
9. 그렇지 않은 경우 11장. 알림 및 로깅 구성으로 이동합니다.

고가용성 QFS 공유 파일 시스템, 클러스터 외부의 클라이언트

HA-COTC(High Availability-Clients Outside the Cluster) 구성은 Solaris Cluster 소프트웨어에 의해 관리되는 고가용성 클러스터의 노드에서 MDS(중요 메타데이터 서버)를 호스팅하는 비아카이빙 QFS 공유 파일 시스템입니다. 이 배열은 서버가 실패한 경우에도 파일 시스템 클라이언트가 해당 데이터에 계속 액세스할 수 있도록 QFS 메타데이터 및 파일 액세스 임대에 대한 페일오버 보호를 제공합니다. 그러나 파일 시스템 클라이언트와 데이터 장치가 클러스터 외부에 남아 있으므로 Solaris Cluster가 QFS 공유 데이터의 제어를 QFS 소프트웨어와 경합하지 않습니다.

HA-COTC 파일 시스템을 구성하려면 아래 작업을 수행합니다.

- 두 HA-COTC 클러스터 노드 모두에서 QFS 공유 파일 시스템 hosts 파일 만들기
- HA-COTC 클러스터 외부의 QFS 서버 및 클라이언트에서 로컬 hosts 파일 만들기
- 주 HA-COTC 클러스터 노드에서 활성 QFS 메타데이터 서버 구성
- 보조 HA-COTC 클러스터 노드에서 잠재적 QFS 메타데이터 서버 구성
- HA-COTC 메타데이터 서버의 페일오버 구성
- HA-COTC 클러스터 외부의 호스트를 QFS 공유 파일 시스템 클라이언트로 구성
- 필요한 경우 "NFS 및 SMB/CIFS를 사용하여 여러 호스트에서 파일 시스템 액세스"에 설명된 대로 NFS(네트워크 파일 시스템) 공유를 구성합니다. HA-NFS(고가용성 NFS)는 지원되지 않습니다.

두 HA-COTC 클러스터 노드 모두에서 QFS 공유 파일 시스템 hosts 파일 만들기

QFS 공유 파일 시스템에서는 모든 호스트가 파일 시스템의 메타데이터에 액세스할 수 있도록 메타데이터 서버에서 hosts 파일을 구성해야 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. HA-COTC 클러스터의 주 노드에 *root*로 로그인합니다.

   예제에서는 호스트가 *qfs1mds-node1* 및 *qfs1mds-node2*입니다. *qfs1mds-node1* 호스트에 로그인합니다.

```
[qfs1mds-node1]root@solaris:~#
```

2. 클러스터 구성을 표시합니다. */usr/global/bin/cluster show* 명령을 사용합니다. 각 *Node Name*에 대한 레코드를 찾은 다음 각 네트워크 어댑터의 *privatehostname*, *Transport Adapter* 이름 및 *ip_address* 등록 정보를 메모합니다.

   명령의 출력이 상당히 길 수 있으므로 아래 예제에서는 긴 표시가 줄임표(...)로 축약되어 있습니다.

   예제에서는 각 노드에 2개의 네트워크 인터페이스인 *qfe3* 및 *hme0*이 있습니다.

   - *hme0* 어댑터는 클러스터가 노드 간의 내부 통신에 사용하는 개인 네트워크의 IP 주소를 가집니다. Solaris Cluster 소프트웨어는 각 개인 주소에 해당하는 개인 호스트 이름을 지정합니다.

     기본적으로 주 노드의 개인 호스트 이름은 *clusternode1-priv*이고 보조 노드의 개인 호스트 이름은 *clusternode2-priv*입니다.
   - *qfe3* 어댑터는 클러스터가 데이터 전송에 사용하는 공용 IP 주소 및 호스트 이름인 *qfs1mds-node1* 및 *qfs1mds-node2*를 가집니다.

```
[qfs1mds-node1]root@solaris:~# cluster show
...
  === Cluster Nodes ===
  Node Name:                                    qfs1mds-node1...
    privatehostname:                               clusternode1-priv...
    Transport Adapter List:                        qfe3, hme0...
    Transport Adapter:                          qfe3...
      Adapter Property(ip_address):                172.16.0.12...
    Transport Adapter:                          hme0...
      Adapter Property(ip_address):                10.0.0.129...
  Node Name:                                    qfs1mds-node2...
    privatehostname:                               clusternode2-priv...
```

```
    Transport Adapter List:                        qfe3, hme0...
      Adapter Property(ip_address):                172.16.0.13...
    Transport Adapter:                           hme0
      Adapter Property(ip_address):                10.0.0.122
```

3. 텍스트 편집기를 사용하여 메타데이터 서버에서 */etc/opt/SUNWsamfs/hosts.family-set-name* 파일을 만듭니다. 여기서 *family-set-name*은 파일 시스템의 패밀리 세트 이름에 대한 이름입니다.

   예제에서는 *vi* 텍스트 편집기를 사용하여 *hosts.qfs1* 파일을 만듭니다. 일부 선택적 머리글을 추가하여 호스트 테이블의 열을 표시하고 각 라인을 해시 기호(#)로 시작하여 주석이라는 것을 나타냅니다.

```
[qfs1mds-node1]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.qfs1
# /etc/opt/SUNWsamfs/hosts.qfs1
#                                          Server   On/  Additional
#Host Name     Network Interface            Ordinal  Off  Parameters
#------------  ---------------------------  -------  ---  ----------
```

4. 테이블의 첫번째 열에 주 및 보조 메타데이터 서버 노드의 호스트 이름과 그 뒤에 약간의 공백을 입력합니다. 각 항목을 별개의 라인에 배치합니다.

   hosts 파일에서 라인은 행(레코드)이고 공백은 열(필드) 구분자입니다. 예제에서 처음 두 행의 *Host Name* 열에는 파일 시스템에 대한 메타데이터 서버를 호스트하는 클러스터 노드의 호스트 이름인 *qfs1mds-node1* 및 *qfs1mds-node2* 값이 포함됩니다.

```
#                                          Server   On/  Additional
#Host Name     Network Interface            Ordinal  Off  Parameters
#------------  ---------------------------  -------  ---  ----------
qfs1mds-node1
qfs1mds-node2
```

5. 각 라인의 두번째 열에서 *Host Name* 호스트에 대한 *Network Interface* 정보 제공을 시작합니다. 각 HA-COTC 클러스터 노드의 Solaris Cluster 개인 호스트 이름 또는 개인 네트워크 주소와 그 뒤에 콤마를 입력합니다.

   HA-COTC 서버 노드는 고가용성 클러스터 내의 서버-서버 통신에 개인 호스트 이름을 사용합니다. 예제에서는 Solaris Cluster 소프트웨어에 의해 지정된 기본 이름인 개인 호스트 이름 *clusternode1-priv* 및 *clusternode2-priv*를 사용합니다.

```
#                                          Server   On/  Additional
#Host Name     Network Interface            Ordinal  Off  Parameters
#------------  ---------------------------  -------  ---  ----------
qfs1mds-node1  clusternode1-priv,
qfs1mds-node2  clusternode2-priv,
```

6. 각 라인의 두번째 열에서 콤마 뒤에 활성 메타데이터 서버에 대한 가상 공용 호스트 이름과 그 뒤에 공백을 입력합니다.

   HA-COTC 서버 노드는 모두 클러스터 외부에 상주하는 클라이언트와 통신하기 위해 공용 데이터 네트워크를 사용합니다. 페일오버 도중 활성 메타데이터 서버의 IP 주소 및 호스트 이름이 변경되므로(*qfs1mds-node1*에서 *qfs1mds-node2*로 변경되거나 그 반대로 변경) 두 경우 모두에 가상 호스트 이름 *qfs1mds*를 사용합니다. 나중에 *qfs1mds*에 대한 요청을 항상 활성 메타데이터 서버로 경로를 지정하도록 Solaris Cluster 소프트웨어를 구성할 것입니다.

```
#                                       Server   On/  Additional
#Host Name     Network Interface         Ordinal  Off  Parameters
#------------  ------------------------  -------  ---  ----------
qfs1mds-node1  clusternode1-priv,qfs1mds
qfs1mds-node2  clusternode2-priv,qfs1mds
```

7. 각 라인의 세번째 열에 서버의 순서 번호(활성 메타데이터 서버의 경우 *1*이고 잠재적 메타데이터 서버의 경우 *2*)와 그 뒤에 공백을 입력합니다.

   이 예제에서는 메타데이터 서버가 하나만 있고 주 노드 *qfs1mds-node1*이 활성 메타데이터 서버이므로 해당 순서가 *1*이고 보조 노드 *qfs1mds-node2*는 해당 순서가 **2**입니다.

```
#                                       Server   On/  Additional
#Host Name     Network Interface         Ordinal  Off  Parameters
#------------  ------------------------  -------  ---  ----------
qfs1mds-node1  clusternode1-priv,qfs1mds  1
qfs1mds-node2  clusternode2-priv,qfs1mds  2
```

8. 각 라인의 네번째 열에 *0*(영)과 그 뒤에 공백을 입력합니다.

   네번째 열의 *0*(영), *-*(하이픈) 또는 빈 값은 호스트가 공유 파일 시스템에 액세스할 수 있도록 구성된 켜짐 상태라는 것을 나타냅니다. *1*(숫자 1)은 호스트가 파일 시스템에 액세스할 수 있도록 구성되지 않은 *off* 상태라는 것을 나타냅니다(공유 파일 시스템을 관리할 때 이러한 값을 사용하는 방법에 대한 자세한 내용은 *samsharefs* 매뉴얼 페이지 참조).

```
#                                       Server   On/  Additional
#Host Name     Network Interface         Ordinal  Off  Parameters
#------------  ------------------------  -------  ---  ----------
qfs1mds-node1  clusternode1-priv,qfs1mds  1        0
qfs1mds-node2  clusternode2-priv,qfs1mds  2        0
```

9. 주 노드의 라인에서 다섯번째 열에 *server* 키워드를 입력합니다.

   server 키워드는 기본 활성 메타데이터 서버를 식별합니다.

```
#                                          Server   On/  Additional
#Host Name     Network Interface            Ordinal  Off  Parameters
#------------  ---------------------------  -------  ---  ----------
qfs1mds-node1  clusternode1-priv,qfs1mds    1        0    server
qfs1mds-node2  clusternode2-priv,qfs1mds    2        0
```

10. 각 클라이언트 호스트에 대해 *Server Ordinal* 값을 *0*으로 설정하는 라인을 추가합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

    서버 순서 *0*은 호스트가 서버가 아니라 클라이언트라는 것을 식별합니다. HA-COTC 클라이언트는 클러스터의 멤버가 아니므로 클러스터의 공용 데이터 네트워크를 통해서만 통신합니다. 이 클라이언트는 공용 IP 주소만 가집니다. 예제에서는 호스트 이름 대신 공용 IP 주소 *172.16.0.133* 및 *172.16.0.147*을 사용하여 두 클라이언트 *qfs1client1* 및 *qfs1client2*를 추가합니다.

```
#                                          Server   On/  Additional
#Host Name     Network Interface            Ordinal  Off  Parameters
#------------  ---------------------------  -------  ---  ----------
qfs1mds-node1  clusternode1-priv,qfs1mds    1        0    server
qfs1mds-node2  clusternode2-priv,qfs1mds    2        0
qfs1client1    172.16.0.133                 0        0
qfs1client2    172.16.0.147                 0        0
:wq
[qfs1mds-node1]root@solaris:~#
```

11. 전역 */etc/opt/SUNWsamfs/hosts.family-set-name* 파일의 복사본을 QFS 잠재적 메타데이터 서버(두번째 HA-COTC 클러스터 노드)에 배치합니다.
12. 이제 HA-COTC 클러스터 외부의 QFS 서버 및 클라이언트에서 로컬 hosts 파일 만들기를 수행합니다.

HA-COTC 클러스터 외부의 QFS 서버 및 클라이언트에서 로컬 hosts 파일 만들기

클러스터 외부의 클라이언트와 파일 시스템을 공유하는 고가용성 구성에서는 클라이언트가 Solaris Cluster 소프트웨어에 의해 정의된 공용 데이터 네트워크만 사용하여 파일 시스템 서버와 통신하도록 해야 합니다. 이렇게 하려면 특별히 구성된 QFS 로컬 hosts 파일을 사용하여 클라이언트와 서버의 여러 네트워크 인터페이스 간에 네트워크 트래픽을 선택적으로 경로를 지정합니다.

각 파일 시스템 호스트는 먼저 메타데이터 서버에서 */etc/opt/SUNWsamfs/hosts.family-set-name* 파일을 검사하여 다른 호스트에 대한 네트워크 인터페이스를 식별합니다. 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 파

일이 있는 경우 로컬 파일을 전역 파일과 비교하여 양쪽 파일에 나열된 인터페이스만 로컬 파일에 지정된 순서대로 사용합니다. 따라서 각 파일의 다른 배열에서 다른 주소를 사용하여 다른 호스트에 사용되는 인터페이스를 제어할 수 있습니다.

로컬 hosts 파일을 구성하려면 아래 설명된 절차를 사용하십시오.

1. HA-COTC 클러스터의 주 노드에 *root*로 로그인합니다.

   예제에서는 호스트가 *qfs1mds-node1* 및 *qfs1mds-node2*입니다. *qfs1mds-node1* 호스트에 로그인합니다.

```
[qfs1mds-node1]root@solaris:~#
```

2. 각 활성/잠재적 메타데이터 서버에 로컬 hosts 파일을 만듭니다. 경로 및 파일 이름 */etc/opt/SUNWsamfs/hosts.family-set-name.local*을 사용합니다. 여기서 *family-set-name*은 공유 파일 시스템의 장비 식별자입니다. 활성/잠재적 서버에서 사용할 네트워크 인터페이스만 포함하십시오.

   예제에서는 활성 및 잠재적 메타데이터 서버가 개인 네트워크를 통해 서로 통신하고 공용 네트워크를 통해 클라이언트와 통신하게 하려고 합니다. 따라서 활성 및 잠재적 서버의 로컬 hosts 파일 *hosts.qfs1.local*에는 활성 및 잠재적 서버에 대한 클러스터 개인 주소만 나열됩니다.

```
[qfs1mds-node1]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.qfs1.local
# /etc/opt/SUNWsamfs/hosts.qfs1.local
#                                         Server   On/  Additional
#Host Name     Network Interface           Ordinal  Off  Parameters
#------------  --------------------------  -------  ---  ----------
qfs1mds-node1  clusternode1-priv           1        0    server
qfs1mds-node2  clusternode2-priv           2        0
qfs1client1    172.16.0.133                  0        0
qfs1client2    172.16.0.147                  0        0
:wq
[qfs1mds-node1]root@solaris:~# ssh root@qfs1mds-node2
Password:


[qfs1mds-node2]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.qfs1.local
# /etc/opt/SUNWsamfs/hosts.qfs1.local
#                                         Server   On/  Additional
#Host Name     Network Interface           Ordinal  Off  Parameters
#------------  --------------------------  -------  ---  ----------
qfs1mds-node1  clusternode1-priv           1        0    server
qfs1mds-node2  clusternode2-priv           2        0
qfs1client1    172.16.0.133                  0        0
qfs1client2    172.16.0.147                  0        0
:wq
```

```
[qfs1mds-node2]root@solaris:~# exit
[qfs1mds-node1]root@solaris:~#
```

3. 텍스트 편집기를 사용하여 각 클라이언트에 로컬 hosts 파일을 만듭니다. 경로 및 파일 이름 */etc/opt/SUNWsamfs/hosts.family-set-name.local*을 사용합니다. 여기서 *family-set-name*은 공유 파일 시스템의 장비 식별자입니다. 클라이언트에서 사용할 네트워크 인터페이스만 포함하십시오. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   예제에서는 *vi* 편집기를 사용합니다. 클라이언트가 공용 데이터 네트워크를 통해 서버와만 통신하게 하려고 합니다. 따라서 파일에는 활성 메타데이터 서버에 대한 가상 호스트 이름 *qfs1mds*만 포함됩니다. Solaris Cluster 소프트웨어는 *qfs1mds*에 대한 요청을 활성 상태인 서버 노드에 경로를 지정합니다.

```
[qfs1mds-node1]root@solaris:~# ssh root@qfsclient1
Password:
[qfs1client1]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.qfs1.local
# /etc/opt/SUNWsamfs/hosts.qfs1.local
#                                          Server   On/  Additional
#Host Name     Network Interface            Ordinal  Off  Parameters
#------------  ---------------------------  -------  ---  ----------
qfs1mds        qfs1mds                      1        0    server
:wq
qfs1client1]root@solaris:~# exit
[qfs1mds-node1]root@solaris:~# ssh root@qfs1client2
Password:
[qfs1client2]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.qfs1.local
# /etc/opt/SUNWsamfs/hosts.qfs1.local
#                                          Server   On/  Additional
#Host Name     Network Interface            Ordinal  Off  Parameters
#------------  ---------------------------  -------  ---  ----------
qfs1mds        qfs1mds                      1        0    server
:wq
[qfs1client2]root@solaris:~# exit
[qfs1mds-node1]root@solaris:~#
```

4. 이제 주 HA-COTC 클러스터 노드에서 활성 QFS 메타데이터 서버 구성을 수행합니다.

주 HA-COTC 클러스터 노드에서 활성 QFS 메타데이터 서버 구성

활성 메타데이터 서버를 구성하려면 다음 작업을 수행합니다.

- 주 HA-COTC 노드에서 고성능 QFS 파일 시스템 만들기
- 클러스터 제어에서 데이터 장치 제외
- 주 HA-COTC 노드에 QFS 파일 시스템 마운트.

주 HA-COTC 노드에서 고성능 QFS 파일 시스템 만들기

1. HA-COTC 클러스터에 대한 주 노드 및 QFS 공유 파일 시스템에 대한 활성 메타데이터 서버 둘 다로 사용될 클러스터 노드를 선택합니다. *root*로 로그인합니다.

   예제에서는 *qfs1mds-node1*이 주 노드 및 활성 메타데이터 서버입니다.

```
[qfs1mds-node1]root@solaris:~#
```

2. QFS 파일 시스템에 사용할 전역 스토리지 장치를 선택합니다. Solaris Cluster 명령 */usr/global/bin/cldevice list -v*를 사용합니다.

   Solaris Cluster 소프트웨어는 클러스터 노드에 연결되는 모든 장치에 고유 DID(장치 식별자)를 지정합니다. 전역 장치는 클러스터의 모든 노드에서 액세스할 수 있고 로컬 장치는 해당 장치를 마운트하는 호스트에서만 액세스할 수 있습니다. 전역 장치는 페일오버 후에 계속 액세스할 수 있습니다. 로컬 장치는 그렇지 않습니다.

   예제에서는 *d1*, *d2*, *d7* 및 *d8* 장치를 두 노드 모두에서 액세스할 수 없습니다. 따라서 고가용성 QFS 공유 파일 시스템을 구성할 때 *d3*, *d4* 및 *d5* 장치 중에서 선택합니다.

```
[qfs1mds-node1]root@solaris:~# cldevice list -v
DID Device          Full Device Path
----------          ----------------
d1                  qfs1mds-node1:/dev/rdsk/c0t0d0
d2                  qfs1mds-node1:/dev/rdsk/c0t6d0
d3                  qfs1mds-node1:/dev/rdsk/c1t1d0
d3                  qfs1mds-node2:/dev/rdsk/c1t1d0
d4                  qfs1mds-node1:/dev/rdsk/c1t2d0
d4                  qfs1mds-node2:/dev/rdsk/c1t2d0
d5                  qfs1mds-node1:/dev/rdsk/c1t3d0
d5                  qfs1mds-node2:/dev/rdsk/c1t3d0
d6                  qfs1mds-node2:/dev/rdsk/c0t0d0
d7                  qfs1mds-node2:/dev/rdsk/c0t1d0
```

3. 선택한 주 노드에서 *md* 또는 *mr* 데이터 서비스를 사용하는 고성능 *ma* 파일 시스템을 만듭니다. 텍스트 편집기에서 */etc/opt/SUNWsamfs/mcf* 파일을 엽니다.

   예제에서는 파일 시스템 *qfs1*을 구성합니다. *d3* 장치를 메타데이터 장치로 구성하고(장비 유형 *mm*) *d4* 및 *d5* 장치를 데이터 장치로 사용합니다(장비 유형 *mr*).

```
[qfs1mds-node1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment           Equipment  Equipment  Family   Device   Additional
# Identifier          Ordinal    Type       Set      State    Parameters
#------------------   ---------  ---------  -------  ------   ----------------
qfs1                  100        ma         qfs1     -
 /dev/did/dsk/d3s0    101        mm         qfs1     -
```

```
 /dev/did/dsk/d4s0     102        mr         qfs1     -
 /dev/did/dsk/d5s1     103        mr         qfs1     -
```

4. */etc/opt/SUNWsamfs/mcf* 파일에서 파일 시스템 항목의 *Additional Parameters* 열에 *shared* 매개변수를 입력합니다. 파일을 저장합니다.

```
[qfs1mds-node1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment           Equipment  Equipment  Family   Device   Additional
# Identifier          Ordinal    Type       Set      State    Parameters
#------------------   ---------  ---------  -------  ------   ----------------
qfs1                  100        ma         qfs1     -        shared
 /dev/did/dsk/d3s0     101        mm         qfs1     -
 /dev/did/dsk/d4s0     102        mr         qfs1     -
 /dev/did/dsk/d5s1     103        mr         qfs1     -
:wq
[qfs1mds-node1]root@solaris:~#
```

5. *mcf* 파일에서 오류를 검사합니다. */opt/SUNWsamfs/sbin/sam-fsd* 명령을 사용하고 발견된 모든 오류를 수정합니다.

   sam-fsd 명령은 Oracle HSM 구성 파일을 읽고 파일 시스템을 초기화합니다. 만일 오류가 발생하면 중지됩니다. 예제에서는 *qfs1mds-node1* 호스트에서 *mcf* 파일을 검사합니다.

```
[qfs1mds-node1]root@solaris:~# sam-fsd
...
Would start sam-archiverd()
Would start sam-stagealld()
Would start sam-stagerd()
Would start sam-amld()
[qfs1mds-node1]root@solaris:~#
```

6. 파일 시스템을 만듭니다. */opt/SUNWsamfs/sbin/sammkfs -S family-set-name* 명령을 사용합니다. 여기서 *family-set-name*은 파일 시스템의 장비 식별자입니다.

   sammkfs 명령은 주 노드 *qfs1mds-node1*에서 *hosts.family-set-name* 및 *mcf* 파일을 읽고 지정된 등록 정보를 가진 공유 파일 시스템을 만듭니다.

```
[qfs1mds-node1]root@solaris:~# sammkfs -S qfs1
Building 'qfs1' will destroy the contents of devices:
  ...
Do you wish to continue? [y/N]yes ...
[qfs1mds-node1]root@solaris:~#
```

7. 이제 클러스터 제어에서 데이터 장치 제외를 수행합니다.

클러스터 제어에서 데이터 장치 제외

기본적으로 Solaris Cluster 소프트웨어는 클러스터의 배타적인 사용을 위해 디스크 장치를 보호합니다. 그러나 HA-COTC 구성에서는 메타데이터(*mm*) 장치만 클러스터의 일부입니다. 데이터(*mr*) 장치는 클러스터 외부의 파일 시스템 클라이언트와 공유되고 클라이언트 호스트에 직접 연결됩니다. 따라서 데이터(*mr*) 장치를 클러스터 소프트웨어의 제어 밖에 두어야 합니다. 이를 위해 다음 두 가지 방법 중 하나를 사용할 수 있습니다.

- HA-COTC 클러스터에서 QFS 데이터 장치에 대한 보호를 사용 안함으로 설정 또는
- HA-COTC 클러스터의 로컬 전용 장치 그룹에 공유 데이터 장치 배치.

HA-COTC 클러스터에서 QFS 데이터 장치에 대한 보호를 사용 안함으로 설정

1. HA-COTC 클러스터의 주 노드 및 QFS 공유 파일 시스템에 대한 활성 메타데이터 서버에 로그인합니다. *root*로 로그인합니다.

   예제에서는 *qfs1mds-node1*이 주 노드 및 활성 메타데이터 서버입니다.

```
[qfs1mds-node1]root@solaris:~#
```

2. */etc/opt/SUNWsamfs/mcf* 파일에 정의된 각 데이터(*mr*) 장치에 대해 보호를 사용 안함으로 설정합니다. *cldevice set -p default_fencing=nofencing-noscrub device-identifier* 명령을 사용합니다. 여기서 *device-identifier*는 *mcf* 파일의 첫번째 열에 있는 장치에 대해 나열된 장치 식별자입니다.

   메타데이터(*mm*) 장치에 대한 보호를 사용 안함으로 설정하지 마십시오. HA-COTC 구성에서 QFS 메타데이터(*mm*) 장치는 클러스터의 일부이지만 QFS 공유 데이터(*mr*) 장치는 그렇지 않습니다. 데이터 장치는 클러스터 외부의 클라이언트에 직접 연결됩니다. 이러한 이유로 HA-COTC 데이터(*mr*) 장치는 Solaris Cluster 소프트웨어에서 관리하지 않는 로컬 장치로 관리되어야 합니다. 그렇지 않은 경우 Solaris Cluster 소프트웨어 및 QFS가 서로 작업을 방해하고 데이터를 손상시킬 수 있습니다.

   위 예제에서는 *d4* 및 *d5* 장치를 파일 시스템 *qfs1*에 대한 데이터 장치로 구성했습니다. 따라서 이러한 장치에 대한 보호를 전역적으로 사용 안함으로 설정합니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[qfs1mds-node1]root@solaris:~# cldevice set -p /
default_fencing=nofencing-noscrub d4
[qfs1mds-node1]root@solaris:~# cldevice set -p /
default_fencing=nofencing-noscrub d5
```

3. 이제 주 HA-COTC 클러스터 노드에 QFS 파일 시스템 마운트를 수행합니다.

HA-COTC 클러스터의 로컬 전용 장치 그룹에 공유 데이터 장치 배치

1. HA-COTC 클러스터의 주 노드 및 QFS 공유 파일 시스템에 대한 활성 메타데이터 서버에 로그인합니다. *root*로 로그인합니다.

   예제에서는 *qfs1mds-node1*이 주 노드 및 활성 메타데이터 서버입니다.

```
[qfs1mds-node1]root@solaris:~#
```

2. 파일 시스템의 일부인 모든 데이터(*mr*) 장치를 *localonly* 장치 그룹에 배치합니다. *cldevicegroup set -d device-identifier-list -p localonly=true -n active-mds-node device-group* 명령을 사용합니다. 여기서 *device-list*는 콤마로 구분된 장치 식별자 목록이고 *active-mds-node*는 활성 메타데이터 서버가 일반적으로 상주하는 주 노드이며 *device-group*은 장치 그룹에 사용하도록 선택한 이름입니다.

   다음 예제에서는 데이터 장치 *d4* 및 *d5*(*mcf* 장비 번호 *102* 및 *103*)를 주 노드의 로컬 장치 그룹 *mdsdevgrp*에 배치합니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[qfs1mds-node1]root@solaris:~# cldevicegroup set -d d4,d5 -p localonly=true /
-n node1mds mdsdevgrp
[qfs1mds-node1]root@solaris:~#
```

3. 이제 주 HA-COTC 클러스터 노드에 QFS 파일 시스템 마운트를 수행합니다.

주 HA-COTC 노드에 QFS 파일 시스템 마운트

1. HA-COTC 클러스터의 주 노드 및 QFS 공유 파일 시스템에 대한 활성 메타데이터 서버에 로그인합니다. *root*로 로그인합니다.

   예제에서는 *qfs1mds-node1*이 주 노드 및 활성 메타데이터 서버입니다.

```
[qfs1mds-node1]root@solaris:~#
```

2. 운영체제의 */etc/vfstab* 파일을 백업합니다.

```
[qfs1mds-node1]root@solaris:~# cp /etc/vfstab /etc/vfstab.backup
```

3. 텍스트 편집기에서 운영체제의 */etc/vfstab* 파일을 열고 새 파일 시스템에 대한 라인을 시작합니다. 첫번째 열(*Device to Mount*)에 파일 시스템 이름과 하나 이상의 공백을 차례로 입력합니다.

   예제에서는 *vi* 텍스트 편집기를 사용합니다. *qfs1* 파일 시스템에 대한 라인을 시작합니다.

```
[qfs1mds-node1]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       -
...
qfs1       -
```

4. */etc/vfstab* 파일의 두번째 열(*Device to fsck*)에 하이픈(-)과 하나 이상의 공백을 차례로 입력합니다.

   하이픈은 운영체제에 파일 시스템 무결성 검사를 건너뛰도록 지시합니다. 이러한 검사는 SAMFS 파일 시스템이 아니라 UFS를 위한 것입니다.

```
[qfs1mds-node1]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       -
...
qfs1       -
```

5. */etc/vfstab* 파일의 세번째 열에 클러스터를 기준으로 파일 시스템의 마운트 지점을 입력합니다. 시스템 루트 디렉토리의 바로 아래에 있지 않은 하위 디렉토리를 선택합니다.

   공유 QFS 파일 시스템을 루트 바로 아래에 마운트하면 *SUNW.qfs* 리소스 유형 사용 시에 페일오버 문제가 발생할 수 있습니다. 예제에서는 클러스터의 마운트 지점을 */global/ha-cotc/qfs1*로 설정합니다.

```
#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       -
...
qfs1       -        /global/ha-cotc/qfs1
```

6. 공유 QFS 파일 시스템의 경우와 마찬가지로 */etc/vfstab* 파일 레코드의 나머지 필드를 채웁니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

```
#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       -
...
qfs1       -        /global/ha-cotc/qfs1  samfs   -     no       shared
:wq
[qfs1mds-node1]root@solaris:~#
```

7. 고가용성 공유 파일 시스템에 대한 마운트 지점을 만듭니다.

   mkdir 명령을 *-p*(*parents*) 옵션과 함께 사용하면 */global* 디렉토리가 만들어집니다(아직 없는 경우).

```
[qfs1mds-node1]root@solaris:~# mkdir -p /global/ha-cotc/qfs1
```

8. 주 노드에 고가용성 공유 파일 시스템을 마운트합니다.

```
[qfs1mds-node1]root@solaris:~# mount /global/ha-cotc/qfs1
```

9. 이제 보조 HA-COTC 클러스터 노드에서 잠재적 QFS 메타데이터 서버 구성을 수행합니다.

보조 HA-COTC 클러스터 노드에서 잠재적 QFS 메타데이터 서버 구성

2노드 클러스터의 보조 노드는 잠재적 메타데이터 서버로 사용됩니다. 잠재적 메타데이터 서버는 메타데이터 장치에 액세스할 수 있으므로 메타데이터 서버의 역할을 담당할 수 있는 호스트입니다. 따라서 주 노드의 활성 메타데이터 서버가 실패할 경우 Solaris Cluster 소프트웨어는 보조 노드로 페일오버하여 잠재적 메타데이터 서버를 활성화할 수 있습니다. 잠재적 메타데이터 서버를 구성하려면 다음 작업을 수행합니다.

- 보조 HA-COTC 노드에서 고성능 QFS 파일 시스템 만들기
- 보조 HA-COTC 노드에 QFS 파일 시스템 마운트.

보조 HA-COTC 노드에서 고성능 QFS 파일 시스템 만들기

1. HA-COTC 클러스터의 보조 노드에 *root*로 로그인합니다.

   예제에서는 *qfs1mds-node2*가 보조 노드 및 잠재적 메타데이터 서버입니다.

```
[qfs1mds-node2]root@solaris:~#
```

2. */etc/opt/SUNWsamfs/mcf* 파일을 주 노드에서 보조 노드로 복사합니다.
3. *mcf* 파일에서 오류를 검사합니다. */opt/SUNWsamfs/sbin/sam-fsd* 명령을 사용하고 발견된 모든 오류를 수정합니다.

   sam-fsd 명령은 Oracle HSM 구성 파일을 읽고 파일 시스템을 초기화합니다. 만일 오류가 발생하면 중지됩니다. 예제에서는 *qfs1mds-node1* 호스트에서 *mcf* 파일을 검사합니다.

```
[qfs1mds-node2]root@solaris:~# sam-fsd
...
Would start sam-archiverd()
Would start sam-stagealld()
Would start sam-stagerd()
Would start sam-amld()
[qfs1mds-node2]root@solaris:~#
```

4. 이제 보조 HA-COTC 클러스터 노드에 QFS 파일 시스템 마운트를 수행합니다.

보조 HA-COTC 노드에 QFS 파일 시스템 마운트

1. HA-COTC 클러스터의 보조 노드에 *root*로 로그인합니다.

   예제에서는 *qfs1mds-node2*가 보조 노드입니다.

```
[qfs1mds-node2]root@solaris:~#
```

2. 운영체제의 */etc/vfstab* 파일을 백업합니다.

```
[qfs1mds-node2]root@solaris:~# cp /etc/vfstab /etc/vfstab.backup
```

3. 텍스트 편집기에서 운영체제의 */etc/vfstab* 파일을 열고 새 파일 시스템에 대한 라인을 추가합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   이 예에서는 *vi* 편집기를 사용합니다.

```
[qfs1mds-node2]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       -
...
qfs1       -        /global/ha-cotc/qfs1  samfs   -     no       shared
:wq
```

```
[qfs1mds-node2]root@solaris:~#
```

4. 보조 노드에서 고가용성 공유 파일 시스템에 대한 마운트 지점을 만듭니다.

```
[qfs1mds-node2]root@solaris:~# mkdir -p /global/ha-cotc/qfs1
[qfs1mds-node2]root@solaris:~#
```

5. 보조 노드에 고가용성 공유 파일 시스템을 마운트합니다.

```
[qfs1mds-node2]root@solaris:~# mount /global/ha-cotc/qfs1
[qfs1mds-node2]root@solaris:~#
```

6. 이제 HA-COTC 메타데이터 서버의 페일오버 구성을 수행합니다.

HA-COTC 메타데이터 서버의 페일오버 구성

Solaris Cluster 소프트웨어가 관리하는 클러스터에서 Oracle HSM 공유 파일 시스템을 호스팅할 경우 Oracle HSM 소프트웨어에 의해 정의된 리소스 유형인 *SUNW.qfs* 클러스터 리소스를 만들어 메타데이터 서버의 페일오버를 구성합니다(자세한 내용은 *SUNW.qfs* 매뉴얼 페이지 참조). HA-COTC 구성에 대한 리소스를 만들고 구성하려면 다음과 같이 하십시오.

1. HA-COTC 클러스터의 주 노드에 *root*로 로그인합니다.

   예제에서는 *qfs1mds-node1*이 주 노드입니다.

```
[qfs1mds-node1]root@solaris:~#
```

2. Solaris Cluster 소프트웨어에 대한 QFS 리소스 유형 *SUNW.qfs*를 정의합니다. *clresourcetype registerSUNW.qfs* 명령을 사용합니다.

```
[qfs1mds-node1]root@solaris:~# clresourcetype register SUNW.qfs
[qfs1mds-node1]root@solaris:~#
```

3. 등록 파일을 찾을 수 없어 등록에 실패한 경우 */opt/SUNWsamfs/sc/etc/* 디렉토리에 대한 심볼릭 링크를 Solaris Cluster가 리소스 유형 등록 파일을 유지하는 디렉토리인 */opt/cluster/lib/rgm/rtreg/*에 배치합니다.

   Oracle HSM 소프트웨어를 설치하기 전에 Oracle Solaris Cluster 소프트웨어를 설치하지 않았습니다. 일반적으로 Oracle HSM는 설치 도중 Solaris Cluster를 감지한 경우 *SUNW.qfs* 등록 파일의 위치를 자동으로 제공합니다. 따라서 링크를 수동으로 만들 필요가 없습니다.

```
[qfs1mds-node1]root@solaris:~# cd /opt/cluster/lib/rgm/rtreg/
[qfs1mds-node1]root@solaris:~# ln -s /opt/SUNWsamfs/sc/etc/SUNW.qfs SUNW.qfs
[qfs1mds-node1]root@solaris:~#
```

4. QFS 메타데이터 서버에 대한 리소스 그룹을 만듭니다. Solaris Cluster 명령 *clresourcegroup create -n node-list group-name*을 사용합니다. 여기서 *node-list*는 콤마로 구분된 클러스터 노드 이름 2개의 목록이고 *group-name*은 리소스 그룹에 사용할 이름입니다.

   예제에서는 HA-COTC 서버 노드가 멤버로 포함된 리소스 그룹 *qfsrg*를 만듭니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[qfs1mds-node1]root@solaris:~# clresourcegroup create -n / qfs1mds-node1,qfs1mds-node2 qfsrg
[qfs1mds-node1]root@solaris:~#
```

5. 새 리소스 그룹에서 활성 메타데이터 서버에 대한 가상 호스트 이름을 설정합니다. Solaris Cluster 명령 *clreslogicalhostname create -g group-name virtualMDS*를 사용합니다. 여기서 *group-name*은 QFS 리소스 그룹의 이름이고 *virtualMDS*는 가상 호스트 이름입니다.

   공유 파일 시스템에 대한 hosts 파일에 사용한 것과 같은 가상 호스트 이름을 사용합니다. 예제에서는 가상 호스트 *qfs1mds*를 *qfsr* 리소스 그룹에서 만듭니다.

```
[qfs1mds-node1]root@solaris:~# clreslogicalhostname create -g qfsrg qfs1mds
```

6. QFS 파일 시스템 리소스를 리소스 그룹에 추가합니다. *clresource create -g group-name -t SUNW.qfs -x QFSFileSystem=mount-point -y Resource_dependencies=virtualMDS resource-name* 명령을 사용합니다. 설명:
   - *group-name*은 QFS 리소스 그룹의 이름입니다.
   - *mount-point*는 시스템 루트 디렉토리 바로 아래에 있지 않은 하위 디렉토리인 클러스터의 파일 시스템에 대한 마운트 지점입니다.

     공유 QFS 파일 시스템을 루트 바로 아래에 마운트하면 *SUNW.qfs* 리소스 유형 사용 시에 페일오버 문제가 발생할 수 있습니다.
   - *virtualMDS*는 활성 메타데이터 서버의 가상 호스트 이름입니다.
   - *resource-name*은 리소스에 제공할 이름입니다.

   예제에서는 *SUNW.qfs* 유형의 *hasqfs*라는 리소스를 리소스 그룹 *qfsrg*에서 만듭니다. *SUNW.qfs* 확장 등록 정보 *QFSFileSystem*을 */global/ha-cotc/qfs1* 마운트 지점으로 설정하고 표준 등록 정보 *Resource_dependencies*를 활성 메타데이터 서버 *qfs1mds*에 대한 논리 호스트로 설정합니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[qfs1mds-node1]root@solaris:~# clresource create -g qfsrg -t SUNW.qfs /
-x QFSFileSystem=/global/ha-cotc/qfs1 -y Resource_dependencies=qfs1mds hasqfs
```

7. 자원 그룹을 온라인으로 전환합니다. *clresourcegroup online -emM group-name* 명령을 사용합니다. 여기서 *group-name*은 QFS 리소스 그룹의 이름입니다.

예제에서는 *qfsr* 리소스 그룹을 온라인 상태로 만듭니다.

```
[qfs1mds-node1]root@solaris:~# clresourcegroup manage qfsrg
[qfs1mds-node1]root@solaris:~# clresourcegroup online -emM qfsrg
```

8. QFS 리소스 그룹이 온라인 상태인지 확인합니다. Solaris Cluster *clresourcegroup status* 명령을 사용합니다.

   예제에서는 *qfsrg* 리소스 그룹이 주 노드 *sam1mds-node1*에서 *online* 상태입니다.

```
[qfs1mds-node1]root@solaris:~# clresourcegroup status
=== Cluster Resource Groups ===
Group Name  Node Name       Suspended   Status
----------  -------------   ---------   ------
qfsrg       qfs1mds-node1   No          Online
            qfs1mds-node2   No          Offline
```

9. 리소스 그룹을 보조 노드로 이동하여 리소스 그룹이 올바르게 페일오버되는지 확인합니다. Solaris Cluster 명령 *clresourcegroup switch -n node2 group-name*을 사용합니다. 여기서 *node2*는 보조 노드의 이름이고 *group-name*은 HA-QFS 리소스 그룹에 사용하도록 선택한 이름입니다. 그런 다음 *clresourcegroup status*를 사용하여 결과를 확인합니다.

   예제에서는 *qfsrg* 리소스 그룹을 *qfs1mds-node2*로 이동하고 이 리소스 그룹이 지정된 노드에서 온라인 상태가 되는지 확인합니다.

```
[qfs1mds-node1]root@solaris:~# clresourcegroup switch -n qfs1mds-node2 qfsrg
[qfs1mds-node1]root@solaris:~# clresourcegroup status
=== Cluster Resource Groups ===
Group Name  Node Name       Suspended   Status
----------  -------------   ---------   ------
qfsrg       qfs1mds-node1   No          Offline
            qfs1mds-node2   No          Online
```

10. 리소스 그룹을 다시 주 노드로 이동합니다. Solaris Cluster 명령 *clresourcegroup switch -n node1 group-name*을 사용합니다. 여기서 *node1*은 주 노드의 이름이고 *group-name*은 HA-QFS 리소스 그룹에 사용하도록 선택한 이름입니다. 그런 다음 *clresourcegroup status*를 사용하여 결과를 확인합니다.

    예제에서는 *qfsrg* 리소스 그룹을 *qfs1mds-node1*로 성공적으로 다시 이동합니다.

```
[qfs1mds-node1]root@solaris:~# clresourcegroup switch -n qfs1mds-node1 qfsrg
[qfs1mds-node1]root@solaris:~# clresourcegroup status
=== Cluster Resource Groups ===
Group Name  Node Name       Suspended   Status
```

```
----------  -------------   ---------   ------
qfsrg       qfs1mds-node1   No          Online
            qfs1mds-node2   No          Offline
```

11. 이제 HA-COTC 클러스터 외부의 호스트를 QFS 공유 파일 시스템 클라이언트로 구성을 수행합니다.

HA-COTC 클러스터 외부의 호스트를 QFS 공유 파일 시스템 클라이언트로 구성

클라이언트가 클러스터 내에 있는 메타데이터 서버의 고가용성 구성에 방해가 되지 않도록 각 호스트를 파일 시스템의 메타데이터 장치에 액세스할 수 없는 QFS 클라이언트로 구성합니다.

HA-COTC 공유 파일 시스템의 각 클라이언트에 대해 다음과 같이 하십시오.

1. HA-COTC 클러스터의 주 노드에 로그인합니다. *root*로 로그인합니다.

   ```
   [qfs1mds-node1]root@solaris:~#
   ```

2. 클러스터에 대한 장치 구성을 표시합니다. Solaris Cluster 명령 */usr/global/bin/cldevice list -v*를 사용합니다.

```
[qfs1mds-node1]root@solaris:~# cldevice list -v
DID Device          Full Device Path
----------          ----------------
d1                  qfs1mds-node1:/dev/rdsk/c0t0d0
d2                  qfs1mds-node1:/dev/rdsk/c0t6d0
...
d7                  qfs1mds-node2:/dev/rdsk/c0t1d0
[qfs1mds-node1]root@solaris:~#
```

3. *cldevice list -v* 명령의 출력을 검사합니다. 각 QFS 데이터(*mr*) 장치의 장치 식별자에 해당하는 */dev/rdsk/* 경로를 메모합니다.

   예제에서 QFS 데이터 장치는 *d4* 및 *d5*입니다.

```
[qfs1mds-node1]root@solaris:~# cldevice list -v
DID Device          Full Device Path
----------          ----------------
d1                  qfs1mds-node1:/dev/rdsk/c0t0d0
d2                  qfs1mds-node1:/dev/rdsk/c0t6d0
d3                  qfs1mds-node1:/dev/rdsk/c1t1d0
d3                  qfs1mds-node2:/dev/rdsk/c1t1d0
d4                  qfs1mds-node1:/dev/rdsk/c1t2d0
d4                  qfs1mds-node2:/dev/rdsk/c1t2d0
```

```
d5                    qfs1mds-node1:/dev/rdsk/c1t3d0
d5                    qfs1mds-node2:/dev/rdsk/c1t3d0
d6                    qfs1mds-node2:/dev/rdsk/c0t0d0
d7                    qfs1mds-node2:/dev/rdsk/c0t1d0
[qfs1mds-node1]root@solaris:~#
```

4. HA-COTC 클러스터의 클라이언트 호스트에 *root*로 로그인합니다.

   예제에서는 *qfs1client1*이 클라이언트 호스트입니다.

```
[qfs1mds-node1]root@solaris:~# ssh root@qfs1client1
[qfs1client1]root@solaris:~#
```

5. 클라이언트 호스트에서 공유 파일 시스템에 대한 구성 정보를 검색합니다. *samfsconfig /dev/rdsk/** 명령을 사용합니다.

   *samfsconfig /dev/rdsk/** 명령은 QFS 파일 시스템에 속하는 연결된 장치를 지정된 경로에서 검색합니다. 예제에서 이 명령은 *qfs1* 데이터(*mr*) 장치에 대한 경로를 찾습니다. 예상대로 이 명령은 메타데이터(*mm*) 장치를 찾지 않으므로 공유 데이터 장치를 나열하기 전에 *Missing slices* 및 *Ordinal 0* 메시지를 반환합니다.

```
[qfs1client1]root@solaris:~# samfsconfig /dev/rdsk/*
# Family Set 'qfs1' Created Thu Dec 21 07:17:00 2013
# Missing slices
# Ordinal 0
# /dev/rdsk/c1t2d0s0   102         mr       qfs1   -
# /dev/rdsk/c1t3d0s1   103         mr       qfs1   -
```

6. *samfsconfig* 명령 출력을 서버의 Solaris Cluster *cldevice list* 명령 출력과 비교합니다. 둘 다 *mr* 데이터 장치에 대해 동일한 장치 경로를 보고하는지 확인합니다.

   컨트롤러 번호(*cN*)가 다를 수 있지만 *samfsconfig* 및 *cldevice list* 명령은 동일한 장치를 나타내야 합니다. 예제에서는 *samfsconfig* 및 *cldevice list* 명령이 동일한 장치를 가리킵니다.

   메타데이터 서버 노드에서 */etc/opt/SUNWsamfs/mcf* 파일은 클러스터 장치 식별자 *d4* 및 *d5*를 사용하여 공유 *mr* 데이터 장치 *102* 및 *103*을 식별합니다.

```
/dev/did/dsk/d4s0     102         mr         qfs1   -
/dev/did/dsk/d5s1     103         mr         qfs1   -
```

   메타데이터 서버 노드의 *cldevice list* 명령은 클러스터 장치 식별자 *d4* 및 *d5*를 경로 */dev/rdisk/c1t2d0* 및 */dev/rdisk/c1t3d0*에 매핑합니다.

```
d4                    qfs1mds-node1:/dev/rdsk/c1t2d0
```

```
d5                  qfs1mds-node1:/dev/rdsk/c1t3d0
```

또한 클라이언트 노드에서 *samfsconfig* 명령은 경로 */dev/rdisk/c1t2d0* 및 */dev/rdisk/c1t3d0*을 사용하여 공유 *mr* 데이터 장치 *102* 및 *103*을 식별합니다.

```
/dev/rdsk/c1t2d0s0    102        mr       qfs1  -
/dev/rdsk/c1t3d0s1    103        mr       qfs1  -
```

7. 클라이언트의 */etc/opt/SUNWsamfs/mcf* 파일을 텍스트 편집기에서 엽니다. HA-COTC 공유 파일 시스템에 대한 항목을 추가합니다. 이 항목은 메타데이터 서버 *mcf* 파일의 해당 항목과 정확히 일치해야 합니다.

   예제에서는 *vi* 편집기를 사용하여 QFS 공유 파일 시스템 *qfs1*(장비 순서 번호 *100*)에 대한 항목을 만듭니다.

```
[qfs1client1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment           Equipment  Equipment  Family   Device   Additional
# Identifier          Ordinal    Type       Set      State    Parameters
#------------------   ---------  ---------  -------  ------   ----------------
qfs1                  100        ma         qfs1     -        shared
```

8. 새 라인에서 HA-COTC 공유 파일 시스템의 메타데이터(*mm*) 장치에 대한 항목을 시작합니다. 첫번째 열(*Equipment Identifier*)에 *nodev* 키워드를 입력합니다.

```
# Equipment           Equipment  Equipment  Family   Device   Additional
# Identifier          Ordinal    Type       Set      State    Parameters
#------------------   ---------  ---------  -------  ------   ----------------
qfs1                  100        ma         qfs1      -       shared
 nodev
```

9. HA-COTC 파일 시스템의 메타데이터(*mm*) 장치에 대한 나머지 필드에 메타데이터 서버 *mcf* 파일에서 사용된 것과 동일한 장비 순서 번호, 패밀리 세트 및 장치 상태 매개변수를 채웁니다.

```
# Equipment           Equipment  Equipment  Family   Device   Additional
# Identifier          Ordinal    Type       Set      State    Parameters
#------------------   ---------  ---------  -------  ------   ----------------
qfs1                  100        ma         qfs1      -       shared
 nodev                 101        mm         qfs1     -
```

10. *samfsconfig* 출력에서 데이터(*mr*) 장치에 대한 전체 항목을 복사합니다. 클라이언트의 */etc/opt/SUNWsamfs/mcf* 파일에 항목을 붙여넣습니다. *samfsconfig*에 의해 삽입된 선행 주석(#) 표시를 제거합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

```
# Equipment           Equipment  Equipment  Family   Device   Additional
# Identifier          Ordinal    Type       Set      State    Parameters
#-----------------   ---------  ---------  -------  ------   ----------------
qfs1                  100        ma         qfs1      -       shared
 nodev                 101        mm         qfs1     -
 /dev/rdsk/c1t2d0s0    102        mr         qfs1     -
 /dev/rdsk/c1t3d0s1    103        mr         qfs1     -
:wq
[qfs1client1]root@solaris:~#
```

11. *mcf* 파일에서 오류를 검사합니다. */opt/SUNWsamfs/sbin/sam-fsd* 명령을 사용하고 발견된 모든 오류를 수정합니다.

    sam-fsd 명령은 Oracle HSM 구성 파일을 읽고 파일 시스템을 초기화합니다. 만일 오류가 발생하면 중지됩니다. 예제에서는 *qfs1client1* 호스트에서 *mcf* 파일을 검사합니다.

```
[qfs1client1]root@solaris:~# sam-fsd
...
Would start sam-archiverd()
Would start sam-stagealld()
Would start sam-stagerd()
Would start sam-amld()
[qfs1client1]root@solaris:~#
```

12. 클라이언트 운영체제의 */etc/vfstab* 파일을 텍스트 편집기에서 열고 서버에서 사용된 것과 동일한 매개변수를 사용하여 새 파일 시스템에 대한 항목을 추가합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

    이 예에서는 *vi* 편집기를 사용합니다.

```
[qfs1client1]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       -
...
qfs1       -        /global/ha-cotc/qfs1  samfs   -     no       shared
:wq
[qfs1client1]root@solaris:~#
```

13. 클라이언트에서 고가용성 공유 파일 시스템에 대한 마운트 지점을 만듭니다.

```
[qfs1client1]root@solaris:~# mkdir -p /global/qfs1
[qfs1client1]root@solaris:~#
```

14. 고가용성 공유 파일 시스템을 클라이언트에 마운트합니다.

    예제에서

```
[qfs1client1]root@solaris:~# mount /global/qfs1
[qfs1client1]root@solaris:~#
```

15. 모든 HA-COTC 클라이언트가 구성될 때까지 이 절차를 반복합니다.
16. 사이드밴드 데이터베이스 기능을 사용할 계획인 경우 10장. 보고 데이터베이스 구성을 참조하십시오.
17. 그렇지 않은 경우 11장. 알림 및 로깅 구성으로 이동합니다.

고가용성 Oracle HSM 공유 아카이빙 파일 시스템

HA-SAM(High-Availability Oracle Hierarchical Storage Manager) 구성은 서버 호스트가 실패한 경우에도 QFS 메타데이터 서버 및 Oracle Hierarchical Storage Manager 응용 프로그램이 계속 작동하도록 보장하여 아카이빙 파일 시스템의 가용성을 유지 관리합니다. 파일 시스템은 Solaris Cluster 소프트웨어가 관리하는 2노드 클러스터에서 호스트되는 활성/잠재적 QFS 메타데이터 서버 간에 공유됩니다. 활성 클러스터 노드가 실패한 경우 클러스터링 소프트웨어는 작동 중인 노드에서 잠재적 Oracle HSM 서버를 자동으로 활성화하고 실행 중인 작업에 대한 제어 권한을 이전합니다. QFS 파일 시스템 및 Oracle HSM 응용 프로그램의 로컬 스토리지 디렉토리가 공유되고 이미 마운트되었으므로 데이터 및 메타데이터에 중단 없이 계속 액세스할 수 있습니다.

HA-SAM 구성은 활성 메타데이터 서버를 통해 모든 I/O를 전송하여 클러스터화된 환경에서 파일 시스템 일관성을 유지합니다. HA-SAM 파일 시스템은 순전히 접근성을 위해서만 공유됩니다. 다른 SAM-QFS 공유 파일 시스템 구성처럼 잠재적 메타데이터 서버 호스트를 파일 시스템 클라이언트로 사용할 수 없습니다. 잠재적 메타데이터 서버는 노드 페일오버 중 활성화되지 않은 경우 I/O를 수행하지 않습니다. NFS를 사용하는 클라이언트와 HA-SAM 파일 시스템을 공유할 수 있습니다. 그러나 활성 메타데이터 서버 노드에서 배타적으로 공유를 내보냈는지 확인해야 합니다.

고가용성 아카이빙 파일 시스템은 3개의 Solaris Cluster 리소스 유형에 따라 다릅니다.

- *SUNW.hasam*

  주 호스트가 실패할 경우 *SUNW.hasam* 리소스는 Oracle Hierarchical Storage Manager 응용 프로그램의 페일오버를 관리합니다. *SUNW.hasam* 소프트웨어는 Oracle HSM 소프트웨어 배포에 포함됩니다.

- *SUNW.qfs*

주 호스트가 실패할 경우 *SUNW.qfs* 리소스는 QFS 메타데이터 서버에 대한 페일오버를 관리합니다. *SUNW.qfs* 소프트웨어는 Oracle HSM 소프트웨어 배포에 포함되어 있습니다(자세한 내용은 *SUNW.qfs* 매뉴얼 페이지 참조).

- *SUNW.HAStoragePlus*

  주 호스트가 실패할 경우 *SUNW.HAStoragePlus* 리소스는 Oracle Hierarchical Storage Manager의 로컬 스토리지에 대한 페일오버를 관리합니다. Oracle HSM 응용 프로그램은 서버 호스트의 로컬 파일 시스템에서 휘발성 아카이빙 정보(작업 대기열 및 이동식 매체 카탈로그)를 유지 관리합니다. *SUNW.HAStoragePlus*는 Solaris Cluster 소프트웨어에 표준 리소스 유형으로 포함되어 있습니다(리소스 유형에 대한 자세한 내용은 *Oracle Solaris Cluster Documentation Library*에서 *Data Services Planning and Administration* 설명서 참조).

필요한 구성 요소의 인스턴스를 구성하고 작동 중인 HA-SAM 아카이빙 구성에 통합하려면 다음 작업을 수행합니다.

- 두 HA-SAM 클러스터 노드 모두에서 전역 hosts 파일 만들기
- 두 HA-SAM 클러스터 노드 모두에서 로컬 hosts 파일 만들기
- 주 HA-SAM 클러스터 노드에서 활성 QFS 메타데이터 서버 구성
- 보조 HA-SAM 클러스터 노드에서 잠재적 QFS 메타데이터 서버 구성
- Oracle HSM 구성 파일에 대한 고가용성 로컬 파일 시스템 구성
- Oracle HSM 구성 파일을 고가용성 로컬 파일 시스템으로 재배치
- 고가용성 로컬 파일 시스템을 사용하도록 HA-SAM 클러스터 구성
- QFS 파일 시스템 메타데이터 서버의 페일오버 구성
- Oracle Hierarchical Storage Manager 응용 프로그램의 페일오버 구성
- HA-SAM 솔루션에 대한 클러스터 리소스 종속성 정의
- HA-SAM 리소스 그룹을 온라인 상태로 전환 및 구성 테스트
- 필요한 경우 HA-NFS(고가용성 네트워크 파일 시스템) 공유 구성

  HA-NFS 설정에 대한 자세한 지침은 *Oracle Solaris Cluster* 온라인 설명서 라이브러리에 포함된 *Oracle Solaris Cluster Data Service for Network File System (NFS) Guide*에 나와 있습니다.

두 HA-SAM 클러스터 노드 모두에서 전역 hosts 파일 만들기

아카이빙 Oracle HSM 공유 파일 시스템에서 메타데이터 서버의 hosts 파일을 구성하여 두 노드 모두의 호스트가 파일 시스템의 메타데이터에 액세스할 수 있게 해야 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. HA-SAM 클러스터의 주 노드에 *root*로 로그인합니다.

예제에서는 *sam1mds-node1*이 주 노드입니다.

```
[sam1mds-node1]root@solaris:~#
```

2. 클러스터 구성을 표시합니다. */usr/global/bin/cluster show* 명령을 사용합니다. 출력에서 각 *Node Name*에 대한 레코드를 찾아서 *privatehostname*과 각 네트워크 어댑터의 *Transport Adapter* 이름 및 *ip_address* 등록 정보를 메모합니다.

   예제에서는 각 노드에 2개의 네트워크 인터페이스인 *hme0* 및 *qfe3*이 있습니다.

   - *hme0* 어댑터는 클러스터가 노드 간의 내부 통신에 사용하는 개인 네트워크의 IP 주소를 가집니다. Solaris Cluster 소프트웨어는 각 개인 주소에 해당하는 *privatehostname*을 지정합니다.

     기본적으로 주 노드의 개인 호스트 이름은 *clusternode1-priv*이고 보조 노드의 개인 호스트 이름은 *clusternode2-priv*입니다.
   - *qfe3* 어댑터는 클러스터가 데이터 전송에 사용하는 공용 IP 주소와 공용 호스트 이름인 *sam1mds-node1* 및 *sam1mds-node2*를 가집니다.

   표시는 줄임표(*...*)를 사용하여 축약되어 있습니다.

```
[sam1mds-node1]root@solaris:~# cluster show
...
  === Cluster Nodes ===
  Node Name:                                    sam1mds-node1...
    privatehostname:                               clusternode1-priv...
    Transport Adapter List:                        qfe3, hme0...
    Transport Adapter:                          qfe3...
      Adapter Property(ip_address):                172.16.0.12...
    Transport Adapter:                          hme0...
      Adapter Property(ip_address):                10.0.0.129...
  Node Name:                                    sam1mds-node2...
    privatehostname:                               clusternode2-priv...
    Transport Adapter List:                        qfe3, hme0...
      Adapter Property(ip_address):                172.16.0.13...
    Transport Adapter:                          hme0...
      Adapter Property(ip_address):                10.0.0.122
```

3. 텍스트 편집기를 사용하여 */etc/opt/SUNWsamfs/hosts.family-set-name* 파일을 만듭니다. 여기서 *family-set-name*은 */etc/opt/SUNWsamfs/mcf* 파일에서 파일 시스템 장비에 지정되는 패밀리 세트 이름입니다.

   예제에서는 *vi* 텍스트 편집기를 사용하여 *hosts.sam1* 파일을 만듭니다. 일부 선택적 머리글을 추가하여 호스트 테이블의 열을 표시하고 각 라인을 해시 기호(*#*)로 시작하여 주석이라는 것을 나타냅니다.

```
[sam1mds-node1]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sam1
# /etc/opt/SUNWsamfs/hosts.sam1
#                                         Server   On/  Additional
#Host Name     Network Interface           Ordinal  Off  Parameters
#------------  --------------------------  -------  ---  ----------
```

4. 테이블의 첫번째 열에 주 및 보조 메타데이터 서버 노드의 호스트 이름과 그 뒤에 약간의 공백을 입력합니다(각 항목을 별개의 라인에 입력).

   hosts 파일에서 라인은 행(레코드)이고 공백은 열(필드) 구분자입니다. 예제에서 처음 두 행의 *Host Name* 열에는 파일 시스템에 대한 메타데이터 서버를 호스트하는 클러스터 노드의 호스트 이름인 *sam1mds-node1* 및 *sam1mds-node2* 값이 포함됩니다.

```
#                                         Server   On/  Additional
#Host Name     Network Interface           Ordinal  Off  Parameters
#------------  --------------------------  -------  ---  ----------
sam1mds-node1
sam1mds-node2
```

5. 각 라인의 두번째 열에서 *Host Name* 열에 나열된 호스트에 대한 *Network Interface* 정보 제공을 시작합니다. 각 HA-SAM 클러스터 노드의 Solaris Cluster 개인 호스트 이름 또는 개인 네트워크 주소와 그 뒤에 콤마를 입력합니다.

   HA-SAM 서버 노드는 고가용성 클러스터 내의 서버-서버 통신에 개인 호스트 이름을 사용합니다. 예제에서는 Solaris Cluster 소프트웨어에 의해 지정된 기본 이름인 개인 호스트 이름 *clusternode1-priv* 및 *clusternode2-priv*를 사용합니다.

```
#                                         Server   On/  Additional
#Host Name     Network Interface           Ordinal  Off  Parameters
#------------  --------------------------  -------  ---  ----------
sam1mds-node1  clusternode1-priv,
sam1mds-node2  clusternode2-priv,
```

6. 각 라인의 두번째 열에서 콤마 뒤에 활성 메타데이터 서버에 대한 공용 호스트 이름과 그 뒤에 공백을 입력합니다.

   HA-SAM 서버 노드는 클러스터 외부의 호스트와 통신하기 위해 공용 데이터 네트워크를 사용합니다. 페일오버 도중 활성 메타데이터 서버의 IP 주소 및 호스트 이름이 변경되므로(*sam1mds-node1*에서 *sam1mds-node2*로 변경되거나 그 반대로 변경) 두 경우 모두에 가상 호스트 이름 *sam1mds*를 사용합니다. 나중에 *sam1mds*에 대한 요청을 항상 활성 메타데이터 서버로 경로를 지정하도록 Solaris Cluster 소프트웨어를 구성할 것입니다.

```
#                                         Server   On/  Additional
```

```
#Host Name     Network Interface              Ordinal  Off  Parameters
#------------  -----------------------------  -------  ---  ----------
sam1mds-node1  clusternode1-priv,sam1mds
sam1mds-node2  clusternode2-priv,sam1mds
```

7. 각 라인의 세번째 열에 서버의 순서 번호(활성 메타데이터 서버의 경우 *1*이고 잠재적 메타데이터 서버의 경우 *2*)와 그 뒤에 공백을 입력합니다.

   이 예제에서는 메타데이터 서버가 하나만 있고 주 노드 *sam1mds-node1*이 활성 메타데이터 서버이므로 해당 순서가 *1*이고 보조 노드 *sam1mds-node2*는 해당 순서가 *2*입니다.

```
#                                     Server   On/  Additional
#Host Name     Network Interface       Ordinal  Off  Parameters
#------------  ----------------------  -------  ---  ----------
sam1mds-node1  clusternode1-priv,sam1mds  1
sam1mds-node2  clusternode2-priv,sam1mds  2
```

8. 각 라인의 네번째 열에 *0*(영)과 그 뒤에 공백을 입력합니다.

   네번째 열의 *0*, *-*(하이픈) 또는 공백 값은 호스트가 *on*으로 구성되었고 공유 파일 시스템에 액세스할 수 있음을 나타냅니다. *1*(숫자 1)은 호스트가 파일 시스템에 액세스할 수 있도록 구성되지 않은 *off* 상태라는 것을 나타냅니다(공유 파일 시스템을 관리할 때 이러한 값을 사용하는 방법에 대한 자세한 내용은 *samsharefs* 매뉴얼 페이지 참조).

```
#                                     Server   On/  Additional
#Host Name     Network Interface       Ordinal  Off  Parameters
#------------  ----------------------  -------  ---  ----------
sam1mds-node1  clusternode1-priv,sam1mds  1        0
sam1mds-node2  clusternode2-priv,sam1mds  2        0
```

9. 주 노드의 라인에서 다섯번째 열에 *server* 키워드를 입력합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   server 키워드는 기본 활성 메타데이터 서버를 식별합니다.

```
#                                     Server   On/  Additional
#Host Name     Network Interface       Ordinal  Off  Parameters
#------------  ----------------------  -------  ---  ----------
sam1mds-node1  clusternode1-priv,sam1mds  1        0    server
sam1mds-node2  clusternode2-priv,sam1mds  2        0
:wq
[sam1mds-node1]root@solaris:~#
```

10. 전역 */etc/opt/SUNWsamfs/hosts.family-set-name* 파일의 복사본을 잠재적 메타데이터 서버에 배치합니다.

11. 이제 두 HA-SAM 클러스터 노드 모두에서 로컬 hosts 파일 만들기를 수행합니다.

두 HA-SAM 클러스터 노드 모두에서 로컬 hosts 파일 만들기

고가용성 아카이빙 공유 파일 시스템에서는 서버가 Solaris Cluster 소프트웨어에 의해 정의된 개인 네트워크를 사용하여 서로 통신하도록 해야 합니다. 이렇게 하려면 특별히 구성된 로컬 hosts 파일을 사용하여 서버의 네트워크 인터페이스 간에 네트워크를 트래픽을 선택적으로 경로를 지정합니다.

각 파일 시스템 호스트는 먼저 메타데이터 서버에서 */etc/opt/SUNWsamfs/hosts.family-set-name* 파일을 검사하여 다른 호스트에 대한 네트워크 인터페이스를 식별합니다. 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 파일의 다른 배열에서 다른 주소를 사용하여 다른 호스트에 사용되는 인터페이스를 제어할 수 있습니다.

로컬 hosts 파일을 구성하려면 아래 설명된 절차를 사용하십시오.

1. HA-SAM 클러스터의 주 노드에 *root*로 로그인합니다.

   예제에서는 *sam1mds-node1*이 주 노드입니다.

```
[sam1mds-node1]root@solaris:~#
```

2. 텍스트 편집기에서 경로 및 파일 이름 */etc/opt/SUNWsamfs/hosts.family-set-name.local*을 사용하여 활성 메타데이터 서버에 로컬 hosts 파일을 만듭니다. 여기서 *family-set-name*은 */etc/opt/SUNWsamfs/mcf* 파일에서 파일 시스템 장비에 지정되는 패밀리 세트 이름입니다. 활성 서버가 잠재적 서버와 통신할 때 사용할 네트워크 인터페이스만 포함합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   예제에서는 활성 및 잠재적 메타데이터 서버가 개인 네트워크를 통해 서로 통신하게 하려고 합니다. 따라서 활성 메타데이터 서버의 로컬 hosts 파일 *hosts.sam1.local*에는 활성 및 잠재적 서버에 대한 클러스터 개인 주소만 나열됩니다.

```
[sam1mds-node1]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sam1.local
#                                          Server   On/  Additional
#Host Name     Network Interface             Ordinal  Off  Parameters
#------------  ----------------------------  -------  ---  ----------
sam1mds-node1  clusternode1-priv             1        0    server
sam1mds-node2  clusternode2-priv             2        0
:wq
[sam1mds-node1]root@solaris:~#
```

3. 보조 클러스터 노드에 *root*로 로그인합니다.

예제에서는 *sam1mds-node2*가 보조 노드입니다.

```
[sam1mds-node1]root@solaris:~# ssh root@sam1mds-node2
Password:
[sam1mds-node2]root@solaris:~#
```

4. 텍스트 편집기를 사용하여 잠재적 메타데이터 서버에 로컬 hosts 파일을 만듭니다. 경로 및 파일 이름 */etc/opt/SUNWsamfs/hosts.family-set-name.local*을 사용합니다. 여기서 *family-set-name*은 */etc/opt/SUNWsamfs/mcf* 파일에서 파일 시스템 장비에 지정되는 패밀리 세트 이름입니다. 잠재적 서버가 활성 서버와 통신할 때 사용할 네트워크 인터페이스만 포함합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   예제에서는 활성 및 잠재적 메타데이터 서버가 개인 네트워크를 통해 서로 통신하게 하려고 합니다. 따라서 잠재적 메타데이터 서버의 로컬 hosts 파일 *hosts.sam1.local*에는 활성 및 잠재적 서버에 대한 클러스터 개인 주소만 나열됩니다.

```
[sam1mds-node2]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.sam1.local
#                                          Server   On/  Additional
#Host Name     Network Interface            Ordinal  Off  Parameters
#------------  ---------------------------  -------  ---  ----------
sam1mds-node1  clusternode1-priv            1        0    server
sam1mds-node2  clusternode2-priv            2        0
:wq
[sam1mds-node2]root@solaris:~# exit
[sam1mds-node1]root@solaris:~#
```

5. 이제 주 HA-SAM 클러스터 노드에서 활성 QFS 메타데이터 서버 구성을 수행합니다.

주 HA-SAM 클러스터 노드에서 활성 QFS 메타데이터 서버 구성

1. HA-SAM 클러스터에 대한 주 노드 및 QFS 공유 파일 시스템에 대한 활성 메타데이터 서버 둘 다로 사용될 클러스터 노드를 선택합니다. *root*로 로그인합니다.

   예제에서는 *sam1mds-node1*이 주 노드입니다.

```
[sam1mds-node1]root@solaris:~#
```

2. QFS 파일 시스템에 사용할 전역 스토리지 장치를 선택합니다. */usr/global/bin/cldevice list -v* 명령을 사용합니다.

   Solaris Cluster 소프트웨어는 클러스터 노드에 연결되는 모든 장치에 고유 DID(장치 식별자)를 지정합니다. 전역 장치는 클러스터의 모든 노드에서 액세스할 수 있고 로컬 장치는 해당 장치를 마운트하는 호스트에서만 액세스할 수 있습니다. 전역 장치는 페일오버 후에 계속 액세스할 수 있습니다. 로컬 장치는 그렇지 않습니다.

예제에서는 *d1*, *d2*, *d7* 및 *d8* 장치를 두 노드 모두에서 액세스할 수 없습니다. 따라서 고가용성 QFS 공유 파일 시스템을 구성할 때 *d3*, *d4* 및 *d5* 장치 중에서 선택합니다.

```
[sam1mds-node1]root@solaris:~# cldevice list -v
DID Device          Full Device Path
----------          ----------------
d1                  sam1mds-node1:/dev/rdsk/c0t0d0
d2                  sam1mds-node1:/dev/rdsk/c0t6d0
d3                  sam1mds-node1:/dev/rdsk/c1t1d0
d3                  sam1mds-node2:/dev/rdsk/c1t1d0
d4                  sam1mds-node1:/dev/rdsk/c1t2d0
d4                  sam1mds-node2:/dev/rdsk/c1t2d0
d5                  sam1mds-node1:/dev/rdsk/c1t3d0
d5                  sam1mds-node2:/dev/rdsk/c1t3d0
d6                  sam1mds-node2:/dev/rdsk/c0t0d0
d7                  sam1mds-node2:/dev/rdsk/c0t1d0
```

3. 선택한 주 노드에서 *mr* 데이터 장치를 사용하는 고성능 *ma* 파일 시스템을 만듭니다. 텍스트 편집기에서 */etc/opt/SUNWsamfs/mcf* 파일을 엽니다.

   예제에서는 파일 시스템 *sam1*을 구성합니다. *d3* 장치를 메타데이터 장치로 구성하고(장비 유형 *mm*) *d4* 및 *d5* 장치를 데이터 장치로 사용합니다(장비 유형 *mr*).

```
[sam1mds-node1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment           Equipment  Equipment  Family   Device   Additional
# Identifier          Ordinal    Type       Set      State    Parameters
#-----------------   ---------  ---------  -------  ------   -----------------
sam1                  100        ma         sam1     -
 /dev/did/dsk/d3s0    101        mm         sam1     -
 /dev/did/dsk/d4s0    102        mr         sam1     -
 /dev/did/dsk/d5s1    103        mr         sam1     -
```

4. */etc/opt/SUNWsamfs/mcf* 파일에서 파일 시스템 항목의 *Additional Parameters* 열에 *shared* 매개변수를 입력합니다. 파일을 저장합니다.

```
[sam1mds-node1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment           Equipment  Equipment  Family   Device   Additional
# Identifier          Ordinal    Type       Set      State    Parameters
#-----------------   ---------  ---------  -------  ------   -----------------
sam1                  100        ma         sam1     -        shared
 /dev/did/dsk/d3s0     101        mm         sam1     -
 /dev/did/dsk/d4s0     102        mr         sam1     -
 /dev/did/dsk/d5s1     103        mr         sam1     -
:wq
```

```
[sam1mds-node1]root@solaris:~#
```

5. *mcf* 파일에서 오류를 검사합니다. */opt/SUNWsamfs/sbin/sam-fsd* 명령을 사용하고 발견된 모든 오류를 수정합니다.

   sam-fsd 명령은 Oracle HSM 구성 파일을 읽고 파일 시스템을 초기화합니다. 만일 오류가 발생하면 중지됩니다. 예제에서는 *sam1mds-node1* 호스트에서 *mcf* 파일을 검사합니다.

```
[sam1mds-node1]root@solaris:~# sam-fsd
...
Would start sam-archiverd()
Would start sam-stagealld()
Would start sam-stagerd()
Would start sam-amld()
[sam1mds-node1]root@solaris:~#
```

6. 파일 시스템을 만듭니다. */opt/SUNWsamfs/sbin/sammkfs -S family-set-name* 명령을 사용합니다. 여기서 *family-set-name*은 */etc/opt/SUNWsamfs/mcf* 파일에서 파일 시스템 장비에 지정된 패밀리 세트 이름입니다.

   sammkfs 명령은 *hosts.family-set-name* 및 *mcf* 파일을 읽고 지정된 등록 정보를 가진 Oracle HSM 파일 시스템을 만듭니다.

```
[sam1mds-node1]root@solaris:~# sammkfs -S sam1
Building 'sam1' will destroy the contents of devices:
  ...
Do you wish to continue? [y/N]yes ...
[sam1mds-node1]root@solaris:~#
```

7. 텍스트 편집기에서 운영체제의 */etc/vfstab* 파일을 열고 새 파일 시스템에 대한 라인을 시작합니다. 첫번째 열에 파일 시스템 이름과 공백을 입력하고 두번째 열에 하이픈과 추가 공백을 입력합니다.

   예제에서는 *vi* 텍스트 편집기를 사용합니다. *sam1* 파일 시스템에 대한 라인을 시작합니다. 하이픈은 운영체제에서 UFS 도구를 사용하여 파일 시스템 무결성을 검사하려는 시도를 방지합니다.

```
[sam1mds-node1]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       -
```

```
...
sam1       -
```

8. */etc/vfstab* 파일의 세번째 열에 클러스터를 기준으로 파일 시스템의 마운트 지점을 입력합니다. 시스템 루트 디렉토리의 바로 아래에 있지 않은 하위 디렉토리를 선택합니다.

   공유 QFS 파일 시스템을 루트 바로 아래에 마운트하면 *SUNW.qfs* 리소스 유형 사용 시에 페일오버 문제가 발생할 수 있습니다. 예제에서는 클러스터의 마운트 지점을 */global/ha-sam/sam1*로 설정합니다.

```
#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       -
...
sam1       -        /global/ha-sam/sam1
```

9. 모든 Oracle HSM 공유 파일 시스템의 경우와 마찬가지로 */etc/vfstab* 파일 레코드의 나머지 필드를 채웁니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

```
#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       -
...
sam1       -        /global/ha-sam/sam1 samfs   -     no       shared
:wq
[sam1mds-node1]root@solaris:~#
```

10. 고가용성 파일 시스템에 대한 마운트 지점을 만듭니다.

    mkdir 명령을 *-p*(*parents*) 옵션과 함께 사용하면 */global* 디렉토리가 만들어집니다(아직 없는 경우).

```
[sam1mds-node1]root@solaris:~# mkdir -p /global/ha-sam/sam1
```

11. 주 노드에 고가용성 공유 파일 시스템을 마운트합니다.

```
[sam1mds-node1]root@solaris:~# mount /global/ha-sam/sam1
```

12. 이제 보조 HA-SAM 클러스터 노드에서 잠재적 QFS 메타데이터 서버 구성을 수행합니다.

보조 HA-SAM 클러스터 노드에서 잠재적 QFS 메타데이터 서버 구성

2노드 클러스터의 보조 노드는 잠재적 메타데이터 서버로 사용됩니다. 잠재적 메타데이터 서버는 메타데이터 장치에 액세스할 수 있으므로 메타데이터 서버의 역할을 담당할 수 있는 호스트입니다. 따라서 주 노드의 활성 메타데이터 서버가 실패할 경우 Solaris Cluster 소프트웨어는 보조 노드로 페일오버하여 잠재적 메타데이터 서버를 활성화할 수 있습니다.

1. HA-SAM 클러스터의 보조 노드에 *root*로 로그인합니다.

   예제에서는 *sam1mds-node2*가 보조 노드입니다.

```
[sam1mds-node2]root@solaris:~#
```

2. */etc/opt/SUNWsamfs/mcf* 파일을 주 노드에서 보조 노드로 복사합니다.
3. *mcf* 파일에서 오류를 검사합니다. */opt/SUNWsamfs/sbin/sam-fsd* 명령을 사용하고 발견된 모든 오류를 수정합니다.

   sam-fsd 명령은 Oracle HSM 구성 파일을 읽고 파일 시스템을 초기화합니다. 만일 오류가 발생하면 중지됩니다. 예제에서는 *sam1mds-node1* 호스트에서 *mcf* 파일을 검사합니다.

```
[sam1mds-node2]root@solaris:~# sam-fsd
...
Would start sam-archiverd()
Would start sam-stagealld()
Would start sam-stagerd()
Would start sam-amld()
[sam1mds-node2]root@solaris:~#
```

4. 파일 시스템을 만듭니다. */opt/SUNWsamfs/sbin/sammkfs -S family-set-name* 명령을 사용합니다. 여기서 *family-set-name*은 */etc/opt/SUNWsamfs/mcf* 파일에서 파일 시스템 장비에 지정된 패밀리 세트 이름입니다.

   sammkfs 명령은 *hosts.family-set-name* 및 *mcf* 파일을 읽고 지정된 등록 정보를 가진 Oracle HSM 파일 시스템을 만듭니다.

```
[sam1mds-node2]root@solaris:~# sammkfs sam1
Building 'sam1' will destroy the contents of devices:
  ...
Do you wish to continue? [y/N]yes ...
[sam1mds-node2]root@solaris:~#
```

5. 텍스트 편집기에서 운영체제의 */etc/vfstab* 파일을 열고 새 파일 시스템에 대한 라인을 추가합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   이 예에서는 *vi* 편집기를 사용합니다.

```
[sam1mds-node2]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       -
...
sam1       -        /global/ha-sam/sam1 samfs   -     no       shared
:wq
[sam1mds-node2]root@solaris:~#
```

6. 보조 노드에서 고가용성 공유 파일 시스템에 대한 마운트 지점을 만듭니다.

```
[sam1mds-node2]root@solaris:~# mkdir -p /global/ha-sam/sam1
```

7. 보조 노드에 고가용성 공유 파일 시스템을 마운트합니다.

```
[sam1mds-node2]root@solaris:~# mount /global/ha-sam/sam1
```

8. 이제 HA-SAM 클러스터 리소스 그룹 만들기를 수행합니다.

HA-SAM 클러스터 리소스 그룹 만들기

HA-SAM 솔루션에 대한 고가용성 리소스를 관리할 리소스 그룹을 만듭니다.

1. HA-SAM 클러스터의 주 클러스터 노드에 *root*로 로그인합니다.

   예제에서는 주 노드가 *sam1mds-node1*입니다.

```
[sam1mds-node1]root@solaris:~#
```

2. HA-SAM 솔루션 리소스를 관리할 Solaris Cluster 리소스 그룹을 만듭니다. *clresourcegroup create -n node1,node2 groupname* 명령을 사용합니다. 설명:
   - *node1*은 주 클러스터 노드의 호스트 이름입니다.
   - *node2*는 보조 클러스터 노드의 호스트 이름입니다.
   - *groupname*은 HA-SAM 리소스 그룹에 사용하도록 선택한 이름입니다.

예제에서는 *has-rg*라는 리소스 그룹을 만들고 *sam1mds-node1* 및 *sam1mds-node2* 호스트를 포함합니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[sam1mds-node1]root@solaris:~# clresourcegroup create /
-n sam1mds-node1,sam1mds-node2 has-rg
```

3. 이제 Oracle HSM 구성 파일을 포함하도록 고가용성 로컬 파일 시스템 구성을 수행합니다.

Oracle HSM 구성 파일에 대한 고가용성 로컬 파일 시스템 구성

페일오버 후에 성공적으로 복구하려면 Oracle HSM 소프트웨어가 페일오버 발생 시 실행하고 있던 아카이빙 작업을 다시 시작해야 합니다. 아카이빙 작업을 다시 시작하려면 소프트웨어가 일반적으로 활성 메타데이터 서버의 로컬 파일 시스템에 저장되어 있는 시스템 구성 및 상태 정보에 액세스할 수 있어야 합니다. 따라서 클러스터의 두 노드에서 모두 항상 액세스할 수 있는 고가용성 로컬 파일 시스템으로 필요한 정보를 이동해야 합니다.

필요한 파일 시스템을 만들려면 다음과 같이 하십시오.

1. HA-SAM 클러스터의 주 노드에 *root*로 로그인합니다.

   예제에서는 주 노드가 *sam1mds-node1*입니다.

```
[sam1mds-node1]root@solaris:~#
```

2. 주 클러스터 노드에서 전역 장치의 사용 가능한 슬라이스에 UFS 파일 시스템을 만듭니다. *newfs /dev/global/dsk/dXsY* 명령을 사용합니다. 여기서 *X*는 전역 장치의 DID(장치 식별자) 번호이고 *Y*는 슬라이스 번호입니다.

   예제에서는 */dev/global/dsk/d10s0*에 새 파일 시스템을 만듭니다.

```
[sam1mds-node1]root@solaris:~# newfs /dev/global/dsk/d10s0
newfs: construct a new file system /dev/global/dsk/d10s0: (y/n)? y
/dev/global/dsk/d10s0: 1112940 sectors in 1374 cylinders of 15 tracks,
54 sectors 569.8MB in 86 cyl groups (16 c/g, 6.64MB/g, 3072 i/g)
super-block backups(for fsck -b #) at:
32, 13056, 26080, 39104, 52128, 65152, 78176, 91200, 104224, . . .
[sam1mds-node1]root@solaris:~#
```

3. 주 클러스터 노드에서 운영체제의 */etc/vfstab* 파일을 텍스트 편집기에서 엽니다. 새 UFS 파일 시스템에 대한 라인을 추가합니다. 파일을 저장하고 편집기를 닫습니다.

   새 라인은 */dev/global/dsk/dXsY /dev/global/dsk/dXsY /global/mount_point ufs 5 no global* 형식의 공백으로 구분된 목록이어야 합니다. 설명:

- *X*는 파일 시스템을 보관하는 전역 장치의 DID(장치 식별자) 번호입니다.
- *Y*는 파일 시스템을 보관하는 슬라이스의 번호입니다.
- */dev/global/dsk/dXsY*는 마운트할 파일 시스템 장치의 이름입니다.
- */dev/global/dsk/dXsY*는 *fsck* 명령에서 검사할 파일 시스템 장치의 이름입니다.
- *mount_point*는 UFS 파일을 마운트할 하위 디렉토리의 이름입니다.
- *ufs*는 파일 시스템 유형입니다.
- *5*는 권장되는 *fsck* 전달 번호입니다.
- *no*는 운영체제에 시작 시 파일 시스템을 마운트하지 않도록 지시합니다.
- *global*은 두 노드가 모두 액세스할 수 있도록 파일 시스템을 마운트합니다.

예제에서는 *vi* 편집기를 사용합니다. 파일 시스템 이름은 */dev/global/dsk/d10s0*이고 마운트 지점은 */global/hasam_cfg*입니다(파일 시스템은 한 라인으로 입력하며 페이지에 맞추기 위해 줄바꿈이 삽입되고 백슬래시 문자로 이스케이프됨).

```
[sam1mds-node1]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       -
...
sam1       -        /global/ha-samsam1  samfs   -     no       shared
/dev/global/dsk/d10s0   /dev/global/rdsk/d10s0  /global/hasam_cfg  ufs  5 /
     no   global
:wq
[sam1mds-node2]root@solaris:~#
```

4. 주 클러스터 노드에서 고가용성 로컬 파일 시스템에 대한 마운트 지점을 만듭니다. *mkdir -p /global/mount_point* 명령을 사용합니다. 여기서 *mount_point*는 선택한 마운트 지점 디렉토리입니다.

   예제에서는 */global/hasam_cfg* 디렉토리를 만듭니다.

```
[sam1mds-node1]root@solaris:~# mkdir -p /global/hasam_cfg
```

5. 보조 클러스터 노드에 *root*로 로그인합니다.

   예제에서 보조 노드는 *sam1mds-node2*입니다. *ssh*를 사용하여 로그인합니다.

```
[sam1mds-node1]root@solaris:~# ssh root@sam1mds-node2
Password:
[sam1mds-node2]root@solaris:~#
```

6. 보조 노드에서 운영체제의 */etc/vfstab* 파일을 텍스트 편집기에서 엽니다. 새 UFS 파일 시스템에 대한 동일한 항목을 추가합니다. 파일을 저장하고 편집기를 닫습니다.

```
[sam1mds-node1]root@solaris:~# ssh root@sam1mds-node2
Password:
[sam1mds-node2]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       -
...
sam1       -        /global/ha-samsam1  samfs   -     no       shared
/dev/global/dsk/d10s0   /dev/global/rdsk/d10s0  /global/hasam_cfg  ufs  5  /
    no   global
:wq
[sam1mds-node1]root@solaris:~#
```

7. 보조 노드에서 동일한 마운트 지점을 만듭니다.

   예제에서는 */global/hasam_cfg* 디렉토리를 만듭니다. 그런 다음 *ssh* 세션을 닫고 주 노드에서 작업을 재개합니다.

```
[sam1mds-node2]root@solaris:~# mkdir -p /global/hasam_cfg
[sam1mds-node2]root@solaris:~# exit
[sam1mds-node1]root@solaris:~#
```

8. 주 노드에서 고가용성 로컬 파일 시스템을 마운트합니다. *mount /global/mount_point* 명령을 사용합니다. 여기서 *mount_point*는 선택한 마운트 지점 디렉토리입니다.

   이 명령은 두 노드 모두에서 UFS 파일 시스템을 마운트합니다. 예제에서는 */global/hasam_cfg*에 파일 시스템을 마운트합니다.

```
[sam1mds-node1]root@solaris:~# mount /global/hasam_cfg
[sam1mds-node1]root@solaris:~#
```

9. 주 노드에서 Oracle HSM 스테이징 정보를 보관할 하위 디렉토리를 만듭니다. *mkdir -p /global/mount_point/catalog* 명령을 사용합니다. 여기서 *mount_point*는 선택한 마운트 지점 디렉토리입니다.

```
[sam1mds-node1]root@solaris:~# mkdir /global/hasam_cfg/catalog
[sam1mds-node1]root@solaris:~#
```

10. 주 노드에서 Oracle HSM 아카이브 카탈로그를 보관할 하위 디렉토리를 만듭니다. *mkdir -p /global/mount_point/stager* 명령을 사용합니다. 여기서 *mount_point*는 선택한 마운트 지점 디렉토리입니다.

```
[sam1mds-node1]root@solaris:~# mkdir /global/hasam_cfg/stager
[sam1mds-node1]root@solaris:~#
```

11. 이제 Oracle HSM 구성 파일을 고가용성 로컬 파일 시스템으로 재배치를 수행합니다.

Oracle HSM 구성 파일을 고가용성 로컬 파일 시스템으로 재배치

1. HA-SAM 클러스터의 주 노드에 *root*로 로그인합니다.

   예제에서는 주 노드가 *sam1mds-node1*입니다.

```
[sam1mds-node1]root@solaris:~#
```

2. 주 노드에서 *catalog/* 및 *stager/* 디렉토리를 */var/opt/SUNWsamfs/*의 해당 기본 위치에서 임시 위치로 복사합니다.

   예제에서는 디렉토리를 */var/tmp/*에 재귀적으로 복사합니다(아래의 첫번째 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[sam1mds-node1]root@solaris:~# cp -r /var/opt/SUNWsamfs/catalog /
/var/tmp/catalog
[sam1mds-node1]root@solaris:~# cp -r /var/opt/SUNWsamfs/stager /var/tmp/stager
[sam1mds-node1]root@solaris:~#
```

3. 주 노드에서 *catalog/* 및 *stager/* 디렉토리를 */var/opt/SUNWsamfs/*에서 삭제합니다.

```
[sam1mds-node1]root@solaris:~# rm -rf /var/opt/SUNWsamfs/catalog
[sam1mds-node1]root@solaris:~# rm -rf /var/opt/SUNWsamfs/stager
[sam1mds-node1]root@solaris:~#
```

4. 주 노드에서 카탈로그 정보의 기본 위치와 고가용성 UFS 로컬 파일 시스템의 새 위치 간에 심볼릭 링크를 만듭니다. *ln -s /global/mount_point/catalog /var/opt/SUNWsamfs/catalog* 명령을 사용합니다. 설명:
   - *mount_point*는 고가용성 로컬 파일 시스템이 노드의 루트 파일 시스템에 연결되는 하위 디렉토리의 이름입니다.
   - */var/opt/SUNWsamfs/catalog*가 기본 위치입니다.

   심볼릭 링크는 카탈로그 정보 요청을 새 위치로 자동으로 재지정합니다. 예제에서는 새 위치 */global/hasam_cfg/catalog*를 가리키는 *catalog* 링크를 만듭니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[sam1mds-node1]root@solaris:~# ln -s /global/hasam_cfg/catalog /
/var/opt/SUNWsamfs/catalog
[sam1mds-node1]root@solaris:~#
```

5. 주 노드에서 스테이징 정보의 기본 위치와 고가용성 UFS 로컬 파일 시스템의 새 위치 간에 심볼릭 링크를 만듭니다. *ln -s /global/mount_point/stager /var/opt/SUNWsamfs/stager* 명령을 사용합니다. 설명:
   - *mount_point*는 고가용성 로컬 파일 시스템이 노드의 루트 파일 시스템에 연결되는 하위 디렉토리의 이름입니다.
   - */var/opt/SUNWsamfs/stager*가 기본 위치입니다.

   심볼릭 링크는 스테이저 정보 요청을 새 위치로 자동으로 재지정합니다. 예제에서는 새 위치 */global/hasam_cfg/stager*를 가리키는 *stager* 링크를 만듭니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[sam1mds-node1]root@solaris:~# ln -s /global/hasam_cfg/stager / /var/opt/SUNWsamfs/stager
[sam1mds-node1]root@solaris:~#
```

6. 주 노드에서 기본 */var/opt/SUNWsamfs/catalog* 및 */var/opt/SUNWsamfs/stager* 위치가 심볼릭 링크로 대체되었는지 확인합니다. 또한 링크가 고가용성 파일 시스템의 새 위치를 가리키는지 확인합니다.

   예제에서는 링크가 올바릅니다.

```
[sam1mds-node1]root@solaris:~# ls -l /var/opt/SUNWsamfs/catalog
lrwxrwxrwx 1 root other ... /var/opt/SUNWsamfs/catalog -> /global/hasam_cfg/catalog
[sam1mds-node1]root@solaris:~# ls -l /var/opt/SUNWsamfs/stager
lrwxrwxrwx 1 root other ... /var/opt/SUNWsamfs/stager -> /global/hasam_cfg/stager
[sam1mds-node1]root@solaris:~#
```

7. 임시 위치의 *catalog/* 및 *stager/* 디렉토리 내용을 고가용성 파일 시스템에 복사합니다.

   예제에서는 */var/tmp/*의 *catalog/* 및 *stager/* 디렉토리를 새 위치 */global/hasam_cfg/stager*에 복사합니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[sam1mds-node1]root@solaris:~# cp -rp /var/tmp/catalog/* /
/var/opt/SUNWsamfs/catalog
[sam1mds-node1]root@solaris:~# cp -rp /var/tmp/stager/* /
/var/opt/SUNWsamfs/stager
[sam1mds-node1]root@solaris:~#
```

8. HA-SAM 클러스터의 보조 노드에 *root*로 로그인합니다.

예제에서는 *ssh*(보안 셸)를 사용하여 보조 노드인 *sam1mds-node2*에 로그인합니다.

```
[sam1mds-node1]root@solaris:~# ssh root@sam1mds-node2
Password:
[sam1mds-node2]root@solaris:~#
```

9. 보조 노드에서 카탈로그 정보의 기본 위치와 고가용성 UFS 로컬 파일 시스템의 새 위치 간에 심볼릭 링크를 만듭니다. *ln -s /global/mount_point/catalog /var/opt/SUNWsamfs/catalog* 명령을 사용합니다. 설명:
   - *mount_point*는 고가용성 로컬 파일 시스템이 노드의 루트 파일 시스템에 연결되는 하위 디렉토리의 이름입니다.
   - */var/opt/SUNWsamfs/catalog*가 기본 위치입니다.

   심볼릭 링크는 카탈로그 정보 요청을 새 위치로 자동으로 재지정합니다. 예제에서는 새 위치 */global/hasam_cfg/catalog*를 가리키는 *catalog* 링크를 만듭니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[sam1mds-node2]root@solaris:~# ln -s /global/hasam_cfg/catalog /
/var/opt/SUNWsamfs/catalog
[sam1mds-node2]root@solaris:~#
```

10. 보조 노드에서 스테이징 정보의 기본 위치와 고가용성 UFS 로컬 파일 시스템의 새 위치 간에 심볼릭 링크를 만듭니다. *ln -s /global/mount_point/stager /var/opt/SUNWsamfs/stager* 명령을 사용합니다. 설명:
    - *mount_point*는 고가용성 로컬 파일 시스템이 노드의 루트 파일 시스템에 연결되는 하위 디렉토리의 이름입니다.
    - */var/opt/SUNWsamfs/stager*가 기본 위치입니다.

    심볼릭 링크는 스테이저 정보 요청을 새 위치로 자동으로 재지정합니다. 예제에서는 새 위치 */global/hasam_cfg/stager*를 가리키는 *stager* 링크를 만듭니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[sam1mds-node2]root@solaris:~# ln -s /global/hasam_cfg/stager / /var/opt/SUNWsamfs/stager
[sam1mds-node2]root@solaris:~#
```

11. 보조 노드에서 기본 */var/opt/SUNWsamfs/catalog* 및 */var/opt/SUNWsamfs/stager* 위치가 심볼릭 링크로 대체되었는지 확인합니다. 또한 링크가 고가용성 파일 시스템의 새 위치를 가리키는지 확인합니다.

    예제에서는 링크가 올바릅니다. 따라서 *ssh* 세션을 닫고 주 노드에서 작업을 재개합니다.

```
[sam1mds-node2]root@solaris:~# ls -l /var/opt/SUNWsamfs/catalog
lrwxrwxrwx 1 root other ... /var/opt/SUNWsamfs/catalog -> /global/hasam_cfg/catalog
```

```
[sam1mds-node2]root@solaris:~# ls -l /var/opt/SUNWsamfs/stager
lrwxrwxrwx 1 root other ... /var/opt/SUNWsamfs/stager -> /global/hasam_cfg/stager
[sam1mds-node2]root@solaris:~# exit
[sam1mds-node1]root@solaris:~#
```

12. 이제 고가용성 로컬 파일 시스템을 사용하도록 HA-SAM 클러스터 구성을 수행합니다.

고가용성 로컬 파일 시스템을 사용하도록 HA-SAM 클러스터 구성

1. HA-SAM 클러스터의 주 노드에서 *SUNW.HAStoragePlus* 리소스 유형을 클러스터 구성의 일부로 등록합니다. Solaris Cluster 명령 *clresourcetype register SUNW.HAStoragePlus*를 사용합니다.

```
[sam1mds-node1]root@solaris:~# clresourcetype register SUNW.HAStoragePlus
[sam1mds-node1]root@solaris:~#
```

2. 주 노드에서 *SUNW.HAStoragePlus* 리소스 유형의 새 인스턴스를 만들어 Solaris Cluster 리소스 그룹과 연관시킵니다. *clresource create -g groupname -t SUNW.HAStoragePlus -x FilesystemMountPoints=mountpoint -x AffinityOn=TRUE resourcename* 명령을 사용합니다. 설명:
   - *groupname*은 HA-SAM 리소스 그룹에 사용하도록 선택한 이름입니다.
   - *SUNW.HAStoragePlus*는 로컬 파일 시스템의 페일오버를 지원하는 Solaris Cluster 리소스 유형입니다.
   - *mountpoint*는 카탈로그 및 스테이저 파일을 보관할 고가용성 로컬 파일 시스템에 대한 마운트 지점입니다.
   - *resourcename*은 리소스 자체에 사용하도록 선택한 이름입니다.

   예제에서는 *SUNW.HAStoragePlus* 유형의 *has-cfg*라는 리소스를 만듭니다. 새 리소스를 리소스 그룹 *has-rg*에 추가합니다. 그런 다음 리소스 확장 속성을 구성합니다. *FilesystemMountPoints*를 */global/hasam_cfg*로 설정하고 *AffinityOn*을 *TRUE*로 설정합니다(아래 명령은 각각 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[sam1mds-node1]root@solaris:~# clresource create -g has-rg /
-t SUNW.HAStoragePlus -x FilesystemMountPoints=/global/hasam_cfg /
-x AffinityOn=TRUE has-cfg
[sam1mds-node1]root@solaris:~#
```

3. 이제 QFS 파일 시스템 메타데이터 서버의 페일오버 구성을 수행합니다.

QFS 파일 시스템 메타데이터 서버의 페일오버 구성

Oracle HSM 소프트웨어에 의해 정의된 리소스 유형인 *SUNW.qfs* 클러스터 리소스를 만들어 메타데이터 서버의 페일오버를 구성합니다(자세한 내용은 *SUNW.qfs* 매뉴얼 페이지 참조). HA-SAM 구성에 대한 리소스를 만들고 구성하려면 다음과 같이 하십시오.

1. HA-SAM 클러스터의 주 클러스터 노드에 *root*로 로그인합니다.

   예제에서는 주 노드가 *sam1mds-node1*입니다.

```
[sam1mds-node1]root@solaris:~#
```

2. Solaris Cluster 소프트웨어에 대한 리소스 유형 *SUNW.qfs*를 정의합니다. *clresourcetype registerSUNW.qfs* 명령을 사용합니다.

```
[sam1mds-node1]root@solaris:~# clresourcetype register SUNW.qfs
[qfs1mds-node1]root@solaris:~#
```

3. 등록 파일을 찾을 수 없어 등록에 실패한 경우 */opt/SUNWsamfs/sc/etc/* 디렉토리에 대한 심볼릭 링크를 Solaris Cluster가 리소스 유형 등록 파일을 유지하는 디렉토리인 */opt/cluster/lib/rgm/rtreg/*에 배치합니다.

   Oracle HSM 소프트웨어를 설치하기 전에 Oracle Solaris Cluster 소프트웨어를 설치하지 않은 경우 등록이 실패합니다. 일반적으로 Oracle HSM는 설치 도중 Solaris Cluster를 감지한 경우 *SUNW.qfs* 등록 파일의 위치를 자동으로 제공합니다. 예제에서는 수동으로 링크를 만듭니다.

```
[qfs1mds-node1]root@solaris:~# cd /opt/cluster/lib/rgm/rtreg/
[qfs1mds-node1]root@solaris:~# ln -s /opt/SUNWsamfs/sc/etc/SUNW.qfs SUNW.qfs
[qfs1mds-node1]root@solaris:~#
```

4. 새 리소스 그룹에서 활성 메타데이터 서버에 대한 가상 호스트 이름을 설정합니다. Solaris Cluster 명령 *clreslogicalhostname create -g group-name virtualMDS*를 사용합니다. 설명:
   - *group-name*은 QFS 리소스 그룹의 이름입니다.
   - *virtualMDS*는 가상 호스트 이름입니다.

   공유 파일 시스템에 대한 hosts 파일에 사용한 것과 같은 가상 호스트 이름을 사용합니다. 예제에서는 가상 호스트 이름 *sam1mds*를 *has-rg* 리소스 그룹에 추가합니다.

```
[sam1mds-node1]root@solaris:~# clreslogicalhostname create -g has-rg sam1mds
[qfs1mds-node1]root@solaris:~#
```

5. Oracle HSM 파일 시스템 리소스를 리소스 그룹에 추가합니다. *clresource create -g groupname -t SUNW.qfs -x QFSFileSystem=mount-point* 명령을 사용합니다. 설명:
   - *groupname*은 HA-SAM 리소스 그룹에 사용하도록 선택한 이름입니다.
   - *SUNW.qfs*는 QFS 파일 시스템 메타데이터 서버의 페일오버를 지원하는 Solaris Cluster 리소스 유형입니다.

- *mount-point*는 시스템 루트 디렉토리 바로 아래에 있지 않은 하위 디렉토리인 클러스터의 파일 시스템에 대한 마운트 지점입니다.

  공유 QFS 파일 시스템을 루트 바로 아래에 마운트하면 *SUNW.qfs* 리소스 유형 사용 시에 페일오버 문제가 발생할 수 있습니다.
- *resource-name*은 리소스 자체에 사용하도록 선택한 이름입니다.

예제에서는 *SUNW.qfs* 유형의 *has-qfs*라는 리소스를 리소스 그룹 *has-rg*에서 만듭니다. *SUNW.qfs* 확장 등록 정보 *QFSFileSystem*을 */global/ha-sam/sam1* 마운트 지점으로 설정합니다. 표준 등록 정보 *Resource_dependencies*를 활성 메타데이터 서버를 나타내는 가상 호스트 이름인 *sam1mds*로 설정합니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[sam1mds-node1]root@solaris:~# clresource create -g has-rg -t SUNW.qfs /
-x QFSFileSystem=/global/ha-sam/sam1 -y Resource_dependencies=sam1mds has-qfs
[sam1mds-node1]root@solaris:~#
```

6. 이제 Oracle Hierarchical Storage Manager 응용 프로그램의 페일오버 구성을 수행합니다.

Oracle Hierarchical Storage Manager 응용 프로그램의 페일오버 구성

Oracle HSM *SUNW.hasam* 리소스를 만들어 Oracle Hierarchical Storage Manager 응용 프로그램의 페일오버를 구성합니다. 이 리소스 유형은 Oracle HSM 프로세스의 정상적인 종료 및 다시 시작을 조정합니다.

Oracle HSM 응용 프로그램의 페일오버를 구성하려면 다음과 같이 하십시오.

1. HA-SAM 클러스터의 주 클러스터 노드에 *root*로 로그인합니다.

   예제에서는 주 노드가 *sam1mds-node1*입니다.

```
[sam1mds-node1]root@solaris:~#
```

2. Solaris Cluster 소프트웨어에 대한 리소스 유형 *SUNW.hasam*을 정의합니다. *clresourcetype register SUNW.hasam* 명령을 사용합니다.

```
[sam1mds-node1]root@solaris:~# clresourcetype register SUNW.hasam
[sam1mds-node1]root@solaris:~#
```

3. Oracle HSM *SUNW.hasam* 리소스를 리소스 그룹에 추가합니다. *clresource create -g groupname -t SUNW.hasam -x QFSName=fs-name -x CatalogFileSystem=mount-point resource-name* 명령을 사용합니다. 설명:
   - *groupname*은 HA-SAM 리소스 그룹에 사용하도록 선택한 이름입니다.

- *SUNW.hasam*은 Oracle Hierarchical Storage Manager 응용 프로그램의 페일오버를 지원하는 Solaris Cluster 리소스 유형입니다.
- *mount-point*는 Oracle HSM 아카이브 카탈로그를 보관하는 전역 파일 시스템의 마운트 지점입니다.
- *resource-name*은 리소스 자체에 사용하도록 선택한 이름입니다.

예제에서는 *SUNW.hasam* 유형의 *has-sam*이라는 리소스를 리소스 그룹 *has-rg*에서 만듭니다. *SUNW.hasam* 확장 등록 정보 *QFSName*을 *mcf* 파일에 지정된 QFS 파일 시스템 이름인 *sam1*로 설정합니다. *SUNW.hasam* 확장 등록 정보 *CatalogFileSystem*을 */global/hasam_cfg* 마운트 지점으로 설정합니다.

```
[sam1mds-node1]root@solaris:~# clresource create -g has-rg -t SUNW.hasam /
-x QFSName=sam1 -x CatalogFileSystem=/global/hasam_cfg has-sam
[sam1mds-node1]root@solaris:~#
```

4. 이제 HA-SAM 솔루션에 대한 클러스터 리소스 종속성 정의를 수행합니다.

HA-SAM 솔루션에 대한 클러스터 리소스 종속성 정의

1. HA-SAM 클러스터의 주 클러스터 노드에 *root*로 로그인합니다.

   예제에서는 주 노드가 *sam1mds-node1*입니다.

```
[sam1mds-node1]root@solaris:~#
```

2. 고가용성 로컬 파일 시스템을 사용할 수 없는 경우 QFS 파일 시스템이 시작되면 안됩니다. 따라서 *SUNW.qfs* 리소스가 *SUNW.HAStoragePlus* 리소스에 종속되게 합니다. Solaris Cluster 명령 *clresource set -p Resource_dependencies=dependency resource-name*을 사용합니다. 설명:
   - *dependency*는 *SUNW.HAStoragePlus* 리소스의 이름입니다.
   - *resource-name*은 *SUNW.qfs* 리소스의 이름입니다.

   예제에서는 *SUNW.qfs* 리소스가 *SUNW.HAStoragePlus* 리소스인 *has-cfg*에 종속되게 합니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[sam1mds-node1]root@solaris:~# clresource set /
-p Resource_dependencies=has-cfg has-qfs
[sam1mds-node1]root@solaris:~#
```

3. QFS 활성 메타데이터 서버가 온라인 상태가 아니면 클러스터가 가상 호스트 이름을 사용할 수 있게 하면 안됩니다. 따라서 가상 호스트 이름이 *SUNW.qfs* 리소스에 종속되게 합니다. Solaris Cluster 명령 *clresource set -p Resource_dependencies=virtualMDS resource-name*을 사용합니다. 설명:

- *virtualMDS*는 활성 Oracle HSM 메타데이터 서버를 나타내는 가상 호스트 이름입니다.
- *resource-name*은 *SUNW.qfs* 리소스의 이름입니다.

예제에서 *SUNW.qfs* 리소스를 설정할 때 만든 가상 호스트 이름은 *sam1mds*입니다. 리소스 자체의 이름은 *has-qfs*입니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[sam1mds-node1]root@solaris:~# clresource set /
-p Resource_dependencies=sam1mds has-qfs
[sam1mds-node1]root@solaris:~#
```

4. 이제 HA-SAM 리소스 그룹을 온라인 상태로 전환 및 구성 테스트를 수행합니다.

HA-SAM 리소스 그룹을 온라인 상태로 전환 및 구성 테스트

1. HA-SAM 클러스터의 주 클러스터 노드에 *root*로 로그인합니다.

   예제에서는 주 노드가 *sam1mds-node1*입니다.

```
[sam1mds-node1]root@solaris:~#
```

2. 자원 그룹을 온라인으로 전환합니다. Solaris Cluster 명령 *clresourcegroup manage groupname* 및 *clresourcegroup online -emM groupname*을 사용합니다. 여기서 *groupname*은 HA-SAM 리소스 그룹의 이름입니다.

   예제에서는 *has-rg* 리소스 그룹을 온라인 상태로 전환합니다.

```
[sam1mds-node1]root@solaris:~# clresourcegroup manage has-rg
[sam1mds-node1]root@solaris:~# clresourcegroup online -emM has-rg
[sam1mds-node1]root@solaris:~#
```

3. HA-SAM 리소스 그룹이 온라인 상태인지 확인합니다. Solaris Cluster *clresourcegroup status* 명령을 사용합니다.

   예제에서는 *has-rg* 리소스 그룹이 주 노드 *sam1mds-node1*에서 *online* 상태입니다.

```
[sam1mds-node1]root@solaris:~# clresourcegroup status
=== Cluster Resource Groups ===
Group Name  Node Name       Suspended   Status
----------  -------------   ---------   ------
has-rg      sam1mds-node1   No          Online
            sam1mds-node2   No            Offline
[sam1mds-node1]root@solaris:~#
```

4. 다음에는 리소스 그룹이 제대로 페일오버되는지 확인합니다. 리소스 그룹을 보조 노드로 이동합니다. Solaris Cluster 명령 *clresourcegroup switch -n node2 groupname*을 사용합니다. 여기서 *node2*는 보조 노드의 이름이고 *groupname*은 HA-SAM 리소스 그룹에 사용하도록 선택한 이름입니다. 그런 다음 *clresourcegroup status*를 사용하여 결과를 검사합니다.

   예제에서는 *has-rg* 리소스 그룹을 *sam1mds-node2*로 이동하고 이 리소스 그룹이 지정된 노드에서 온라인 상태가 되는지 확인합니다.

```
[sam1mds-node1]root@solaris:~# clresourcegroup switch -n sam1mds-node2 has-rg
[sam1mds-node1]root@solaris:~# clresourcegroup status
=== Cluster Resource Groups ===
Group Name  Node Name       Suspended   Status
----------  -------------   ---------   ------
has-rg      sam1mds-node1   No          Offline
            sam1mds-node2   No          Online
[sam1mds-node1]root@solaris:~#
```

5. 리소스 그룹을 다시 주 노드로 이동합니다. Solaris Cluster 명령 *clresourcegroup switch -n node1 groupname*을 사용합니다. 여기서 *node1*은 주 노드의 이름이고 *groupname*은 HA-SAM 리소스 그룹에 사용하도록 선택한 이름입니다. 그런 다음 *clresourcegroup status*를 사용하여 결과를 확인합니다.

   예제에서는 *has-rg* 리소스 그룹을 *sam1mds-node1*로 성공적으로 다시 이동합니다.

```
[sam1mds-node1]root@solaris:~# clresourcegroup switch -n sam1mds-node1 has-rg
[sam1mds-node1]root@solaris:~# clresourcegroup status
=== Cluster Resource Groups ===
Group Name  Node Name       Suspended   Status
----------  -------------   ---------   ------
has-rg      sam1mds-node1   No          Online
            sam1mds-node2   No          Offline
[sam1mds-node1]root@solaris:~#
```

6. 필요한 경우 HA-NFS(고가용성 네트워크 파일 시스템) 공유를 지금 구성합니다.

   HA-NFS 설정에 대한 자세한 지침은 *Oracle Solaris Cluster* 온라인 설명서 라이브러리에 포함된 *Oracle Solaris Cluster Data Service for Network File System (NFS) Guide*에 나와 있습니다.
7. 사이드밴드 데이터베이스 기능을 사용할 계획인 경우 10장. 보고 데이터베이스 구성을 참조하십시오.
8. 그렇지 않은 경우 11장. 알림 및 로깅 구성으로 이동합니다.

고가용성 QFS 공유 파일 시스템 및 Oracle RAC

SC-RAC(Solaris Cluster-Oracle Real Application Cluster) 구성에서 Solaris Cluster 소프트웨어는 QFS 공유 파일 시스템을 또한 Oracle 데이터베이스 및 Oracle RAC(Real Application Cluster) 소프트웨어를 호스팅하는 노드에 마운트된 *SUNW.qfs* 리소스로 관리합니다. 모든 노드는 하나의 활성 메타데이터 서버 및 다른 잠재적 메타데이터 서버가 있는 QFS 서버로 구성됩니다. 활성 메타데이터 서버 노드가 실패할 경우 Solaris Cluster 소프트웨어는 정상적인 노드에서 잠재적 메타데이터 서버를 자동으로 활성화하고 페일오버를 시작합니다. Oracle RAC를 통해 I/O가 조정되며 QFS 파일 시스템이 공유되고 이미 모든 노드에 마운트되어 있습니다. 따라서 데이터에 중단 없이 계속 액세스할 수 있습니다.

SC-RAC 구성에서 RAC 소프트웨어는 I/O 요청을 조정하고 작업 로드를 분배하며 클러스터의 노드에서 실행되는 여러 Oracle 데이터베이스 인스턴스에 대한 일관된 단일 데이터베이스 파일 세트를 유지 관리합니다. 파일 시스템 무결성이 RAC에서 보장되므로 QFS 잠재적 메타데이터 서버는 공유 파일 시스템의 클라이언트로 I/O를 수행할 수 있습니다. 자세한 내용은 *Oracle Solaris Cluster Online Documentation Library*에서 Oracle Real Application Clusters에 대한 Oracle Solaris Cluster 데이터 서비스 설명서를 참조하십시오.

SC-RAC 파일 시스템을 구성하려면 아래 작업을 수행합니다.

- 모든 SC-RAC 클러스터 노드에서 QFS 공유 파일 시스템 hosts 파일 만들기
- HA-COTC 클러스터 외부의 QFS 서버 및 클라이언트에서 로컬 hosts 파일 만들기
- 주 SC-RAC 클러스터 노드에서 활성 QFS 메타데이터 서버 구성 또는 소프트웨어 RAID 스토리지를 사용하여 SC-RAC 노드에서 QFS 메타데이터 서버 구성
- 나머지 SC-RAC 클러스터 노드에서 잠재적 QFS 메타데이터 서버 구성
- SC-RAC 메타데이터 서버의 페일오버 구성
- 필요한 경우 "NFS 및 SMB/CIFS를 사용하여 여러 호스트에서 파일 시스템 액세스"에 설명된 대로 NFS(네트워크 파일 시스템) 공유를 구성합니다. HA-NFS(고가용성 NFS)는 지원되지 않습니다.

모든 SC-RAC 클러스터 노드에서 QFS 공유 파일 시스템 hosts 파일 만들기

QFS 공유 파일 시스템에서는 모든 호스트가 파일 시스템의 메타데이터에 액세스할 수 있도록 메타데이터 서버에서 hosts 파일을 구성해야 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. SC-RAC 클러스터의 주 클러스터 노드에 *root*로 로그인합니다.

   예제에서는 주 노드가 *qfs1rac-node1*입니다.

```
[qfs1rac-node1]root@solaris:~#
```

2. 클러스터 구성을 표시합니다. */usr/global/bin/cluster show* 명령을 사용합니다. 출력에서 각 *Node Name*에 대한 레코드를 찾아서 *privatehostname*과 각 네트워크 어댑터의 *Transport Adapter* 이름 및 *ip_address* 등록 정보를 메모합니다.

   예제에서는 각 노드에 2개의 네트워크 인터페이스인 *qfe3* 및 *hme0*이 있습니다.

   - *hme0* 어댑터는 클러스터가 노드 간의 내부 통신에 사용하는 개인 네트워크의 IP 주소를 가집니다. Solaris Cluster 소프트웨어는 각 개인 주소에 해당하는 *privatehostname*을 지정합니다.

     기본적으로 주 노드의 개인 호스트 이름은 *clusternode1-priv*이고 보조 노드의 개인 호스트 이름은 *clusternode2-priv*입니다.
   - *qfe3* 어댑터는 클러스터가 데이터 전송에 사용하는 공용 IP 주소 및 공용 호스트 이름인 *qfs1rac-node1* 및 *qfs1rac-node2*를 가집니다.

   표시는 줄임표(. . .)를 사용하여 축약되어 있습니다.

```
[qfs1rac-node1]root@solaris:~# cluster show
...
  === Cluster Nodes ===
  Node Name:                                    qfs1rac-node1...
    privatehostname:                               clusternode1-priv...
    Transport Adapter List:                        qfe3, hme0...
    Transport Adapter:                          qfe3...
      Adapter Property(ip_address):                172.16.0.12...
    Transport Adapter:                          hme0...
      Adapter Property(ip_address):                10.0.0.129...
  Node Name:                                    qfs1rac-node2...
    privatehostname:                               clusternode2-priv...
    Transport Adapter List:                        qfe3, hme0...
      Adapter Property(ip_address):                172.16.0.13...
    Transport Adapter:                          hme0
      Adapter Property(ip_address):                10.0.0.122...
  Node Name:                                    qfs1rac-node3...
    privatehostname:                               clusternod3-priv...
    Transport Adapter List:                        qfe3, hme0...
      Adapter Property(ip_address):                172.16.0.33...
    Transport Adapter:                          hme0
      Adapter Property(ip_address):                10.0.0.092
```

3. 텍스트 편집기를 사용하여 */etc/opt/SUNWsamfs/hosts.family-set-name* 파일을 만듭니다. 여기서 *family-set-name*은 */etc/opt/SUNWsamfs/mcf* 파일에서 파일 시스템 장비에 지정되는 패밀리 세트 이름입니다.

예제에서는 *vi* 텍스트 편집기를 사용하여 *hosts.qfs1rac* 파일을 만듭니다. 일부 선택적 머리글을 추가하여 호스트 테이블의 열을 표시하고 각 라인을 해시 기호(#)로 시작하여 주석이라는 것을 나타냅니다.

```
[qfs1rac-node1]root@solaris:~# vi /etc/opt/SUNWsamfs/hosts.qfs1rac
# /etc/opt/SUNWsamfs/hosts.qfs1rac
#                                          Server   On/  Additional
#Host Name     Network Interface            Ordinal  Off  Parameters
#------------  -------------------------------  -------  ---  ----------
```

4. 테이블의 첫번째 열에 주 및 보조 메타데이터 서버 노드의 호스트 이름과 그 뒤에 약간의 공백을 입력합니다. 각 항목을 별개의 라인에 배치합니다.

   hosts 파일에서 라인은 행(레코드)이고 공백은 열(필드) 구분자입니다. 예제에서 처음 두 행의 Host Name 열에는 클러스터 노드의 호스트 이름인 *qfs1rac-node1*, *qfs1rac-node2* 및 *qfs1rac-node3*이 나열됩니다.

```
#                                          Server   On/  Additional
#Host Name     Network Interface            Ordinal  Off  Parameters
#------------  -------------------------------  -------  ---  ----------
qfs1rac-node1
qfs1rac-node2
qfs1rac-node3
```

5. 각 라인의 두번째 열에서 *Host Name* 호스트에 대한 *Network Interface* 정보 제공을 시작합니다. 각 SC-RAC 클러스터 노드의 Solaris Cluster 개인 호스트 이름 또는 개인 네트워크 주소와 그 뒤에 콤마를 입력합니다.

   SC-RAC 서버 노드는 고가용성 클러스터 내의 서버-서버 통신에 개인 호스트 이름을 사용합니다. 예제에서는 Solaris Cluster 소프트웨어에 의해 지정된 기본 이름인 개인 호스트 이름 *clusternode1-priv*, *clusternode2-priv* 및 *clusternode3-priv*를 사용합니다.

```
#                                          Server   On/  Additional
#Host Name     Network Interface            Ordinal  Off  Parameters
#------------  -------------------------------  -------  ---  ----------
qfs1rac-node1  clusternode1-priv,
qfs1rac-node2  clusternode2-priv,
qfs1rac-node3  clusternode3-priv,
```

6. 각 라인의 두번째 열에서 콤마 뒤에 활성 메타데이터 서버에 대한 공용 호스트 이름과 그 뒤에 공백을 입력합니다.

SC-RAC 서버 노드는 모두 클러스터 외부에 상주하는 클라이언트와 통신하기 위해 공용 데이터 네트워크를 사용합니다. 페일오버 도중 활성 메타데이터 서버의 IP 주소 및 호스트 이름이 변경되므로(예를 들어, *qfs1rac-node1*에서 *qfs1rac-node2*로 변경) 가상 호스트 이름 *qfs1rac-mds*를 사용하여 활성 서버를 나타냅니다. 나중에 *qfs1rac-mds*에 대한 요청을 현재 활성 메타데이터 서버를 호스팅하는 노드로 항상 경로를 지정하도록 Solaris Cluster 소프트웨어를 구성할 것입니다.

```
#                                          Server   On/  Additional
#Host Name     Network Interface            Ordinal  Off  Parameters
#------------  ------------------------------  -------  ---  ----------
qfs1rac-node1  clusternode1-priv,qfs1rac-mds
qfs1rac-node2  clusternode2-priv,qfs1rac-mds
qfs1rac-node3  clusternode3-priv,qfs1rac-mds
```

7. 각 라인의 세번째 열에 서버의 순서 번호(활성 메타데이터 서버의 경우 *1*이고 잠재적 메타데이터 서버의 경우 *2*)와 그 뒤에 공백을 입력합니다.

   예제에서는 주 노드 *qfs1rac-node1*이 활성 메타데이터 서버입니다. 따라서 순서 *1*입니다. 두번째 노드 *qfs1rac-node2*는 순서 *2*입니다.

```
#                                          Server   On/  Additional
#Host Name     Network Interface            Ordinal  Off  Parameters
#------------  ------------------------------  -------  ---  ----------
qfs1rac-node1  clusternode1-priv,qfs1rac-mds     1
qfs1rac-node2  clusternode2-priv,qfs1rac-mds     2
qfs1rac-node3  clusternode3-priv,qfs1rac-mds     3
```

8. 각 라인의 네번째 열에 *0*(영)과 그 뒤에 공백을 입력합니다.

   네번째 열의 *0*, *-*(하이픈) 또는 공백 값은 호스트가 *on*으로 구성되었고 공유 파일 시스템에 액세스할 수 있음을 나타냅니다. *1*(숫자 1)은 호스트가 파일 시스템에 액세스할 수 있도록 구성되지 않은 *off* 상태라는 것을 나타냅니다(공유 파일 시스템을 관리할 때 이러한 값을 사용하는 방법에 대한 자세한 내용은 *samsharefs* 매뉴얼 페이지 참조).

```
#                                          Server   On/  Additional
#Host Name     Network Interface            Ordinal  Off  Parameters
#------------  ------------------------------  -------  ---  ----------
qfs1rac-node1  clusternode1-priv,qfs1rac-mds     1        0
qfs1rac-node2  clusternode2-priv,qfs1rac-mds     2        0
qfs1rac-node3  clusternode3-priv,qfs1rac-mds     3        0
```

9. 주 노드의 라인에서 다섯번째 열에 *server* 키워드를 입력합니다. 파일을 저장하고 편집기를 닫습니다.

server 키워드는 기본 활성 메타데이터 서버를 식별합니다.

```
#                                          Server   On/  Additional
#Host Name     Network Interface            Ordinal  Off  Parameters
#------------  ---------------------------  -------  ---  ----------
qfs1rac-node1  clusternode1-priv,qfs1rac-mds  1        0    server
qfs1rac-node2  clusternode2-priv,qfs1rac-mds  2        0
qfs1rac-node3  clusternode3-priv,qfs1rac-mds  2        0
:wq
[qfs1rac-node1]root@solaris:~#
```

10. 전역 */etc/opt/SUNWsamfs/hosts.family-set-name* 파일의 복사본을 SC-RAC 클러스터의 각 노드에 배치합니다.
11. 이제 주 SC-RAC 클러스터 노드에서 활성 QFS 메타데이터 서버 구성을 수행합니다.

주 SC-RAC 클러스터 노드에서 활성 QFS 메타데이터 서버 구성

1. SC-RAC 클러스터에 대한 주 노드 및 QFS 공유 파일 시스템에 대한 활성 메타데이터 서버 둘 다로 사용될 클러스터 노드를 선택합니다. *root*로 로그인합니다.

   예제에서는 주 노드가 *qfs1rac-node1*입니다.

```
[qfs1rac-node1]root@solaris:~#
```

2. QFS 파일 시스템에 사용할 전역 스토리지 장치를 선택합니다. */usr/global/bin/cldevice list -v* 명령을 사용합니다.

   Solaris Cluster 소프트웨어는 클러스터 노드에 연결되는 모든 장치에 고유 DID(장치 식별자)를 지정합니다. 전역 장치는 클러스터의 모든 노드에서 액세스할 수 있고 로컬 장치는 해당 장치를 마운트하는 호스트에서만 액세스할 수 있습니다. 전역 장치는 페일오버 후에 계속 액세스할 수 있습니다. 로컬 장치는 그렇지 않습니다.

   예제에서는 *d1*, *d2*, *d6*, *d7* 및 *d8* 장치를 일부 노드에서 액세스할 수 없습니다. 따라서 고가용성 QFS 공유 파일 시스템을 구성할 때 *d3*, *d4* 및 *d5* 장치 중에서 선택합니다.

```
[qfs1rac-node1]root@solaris:~# cldevice list -v
DID Device          Full Device Path
----------          ----------------
d1                  qfs1rac-node1:/dev/rdsk/c0t0d0
d2                  qfs1rac-node1:/dev/rdsk/c0t6d0
d3                  qfs1rac-node1:/dev/rdsk/c1t1d0
d3                  qfs1rac-node2:/dev/rdsk/c1t1d0
d3                  qfs1rac-node3:/dev/rdsk/c1t1d0
d4                  qfs1rac-node1:/dev/rdsk/c1t2d0
d4                  qfs1rac-node2:/dev/rdsk/c1t2d0
```

```
d4                  qfs1rac-node3:/dev/rdsk/c1t2d0
d5                  qfs1rac-node1:/dev/rdsk/c1t3d0
d5                  qfs1rac-node2:/dev/rdsk/c1t3d0
d5                  qfs1rac-node3:/dev/rdsk/c1t3d0
d6                  qfs1rac-node2:/dev/rdsk/c0t0d0
d7                  qfs1rac-node2:/dev/rdsk/c0t1d0
d8                  qfs1rac-node3:/dev/rdsk/c0t1d0
```

3. *mr* 데이터 장치를 사용하는 고성능 공유 *ma* 파일 시스템을 만듭니다. 텍스트 편집기에서 */etc/opt/SUNWsamfs/mcf* 파일을 엽니다.

   예제에서는 파일 시스템 *qfs1rac*를 구성합니다. *d3* 장치를 메타데이터 장치로 구성하고(장비 유형 *mm*) *d4* 및 *d5* 장치를 데이터 장치로 사용합니다(장비 유형 *mr*).

```
[qfs1rac-node1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment           Equipment  Equipment  Family   Device   Additional
# Identifier          Ordinal    Type       Set      State    Parameters
#-----------------   ---------  ---------  -------  ------   --------------
qfs1rac               100        ma         qfs1rac  -
 /dev/did/dsk/d3s0    101        mm         qfs1rac  -
 /dev/did/dsk/d4s0    102        mr         qfs1rac  -
 /dev/did/dsk/d5s0    103        mr         qfs1rac  -
...
```

4. */etc/opt/SUNWsamfs/mcf* 파일에서 파일 시스템 항목의 *Additional Parameters* 열에 *shared* 매개변수를 입력합니다. 파일을 저장합니다.

```
[qfs1rac-node1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# Equipment           Equipment  Equipment  Family   Device   Additional
# Identifier          Ordinal    Type       Set      State    Parameters
#-----------------   ---------  ---------  -------  ------   --------------
qfs1rac               100        ma         qfs1rac  -        shared
 /dev/did/dsk/d3s0     101        mm         qfs1rac  -
 /dev/did/dsk/d4s0     102        mr         qfs1rac  -
 /dev/did/dsk/d5s0     103        mr         qfs1rac  -
...
:wq
[qfs1rac-node1]root@solaris:~#
```

5. *mcf* 파일에서 오류를 검사합니다. */opt/SUNWsamfs/sbin/sam-fsd* 명령을 사용하고 발견된 모든 오류를 수정합니다.

   sam-fsd 명령은 Oracle HSM 구성 파일을 읽고 파일 시스템을 초기화합니다. 만일 오류가 발생하면 중지됩니다. 예제에서는 *qfs1rac-node1* 호스트에서 *mcf* 파일을 검사합니다.

```
[qfs1rac-node1]root@solaris:~# sam-fsd
...
Would start sam-archiverd()
Would start sam-stagealld()
Would start sam-stagerd()
Would start sam-amld()
[qfs1rac-node1]root@solaris:~#
```

6. 파일 시스템을 만듭니다. */opt/SUNWsamfs/sbin/sammkfs -S family-set-name* 명령을 사용합니다. 여기서 *family-set-name*은 파일 시스템에 대한 장비 식별자입니다.

   sammkfs 명령은 *hosts.family-set-name* 및 *mcf* 파일을 읽고 지정된 등록 정보를 가진 공유 파일 시스템을 만듭니다.

```
[qfs1rac-node1]root@solaris:~# sammkfs -S qfs1rac
Building 'qfs1rac' will destroy the contents of devices:
  ...
Do you wish to continue? [y/N]yes ...
[qfs1rac-node1]root@solaris:~#
```

7. 텍스트 편집기에서 운영체제의 */etc/vfstab* 파일을 열고 새 파일 시스템에 대한 라인을 시작합니다. 첫번째 열에 파일 시스템 이름과 공백을 입력하고 두번째 열에 하이픈과 추가 공백을 입력합니다.

   예제에서는 *vi* 텍스트 편집기를 사용합니다. *qfs1rac* 파일 시스템에 대한 라인을 시작합니다. 하이픈은 운영체제에서 UFS 도구를 사용하여 파일 시스템 무결성을 검사하려는 시도를 방지합니다.

```
[qfs1rac-node1]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       -
...
qfs1rac    -
```

8. */etc/vfstab* 파일의 세번째 열에 클러스터를 기준으로 파일 시스템의 마운트 지점을 입력합니다. 시스템 루트 디렉토리의 바로 아래에 있지 않은 하위 디렉토리를 지정합니다.

공유 QFS 파일 시스템을 루트 바로 아래에 마운트하면 *SUNW.qfs* 리소스 유형 사용 시에 페일오버 문제가 발생할 수 있습니다. 예제에서는 *qfs1rac* 파일 시스템에 대한 마운트 지점이 */global/sc-rac/qfs1rac*입니다.

```
#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       -
...
qfs1rac    -        /global/sc-rac/qfs1rac
```

9. 파일 시스템 유형 *samfs*를 네번째 열에 입력하고 -(하이픈)을 다섯번째 열에 입력하고 *no*를 여섯번째 열에 입력합니다.

```
#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       -
...
qfs1rac    -        /global/sc-rac/qfs1rac  samfs   -     no
:wq
[qfs1rac-node1]root@solaris:~#
```

10. */etc/vfstab* 파일의 일곱번째 열에 아래 나열된 마운트 옵션을 입력합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

    다음 마운트 옵션은 SC-RAC 클러스터 구성에 권장됩니다. 이러한 마운트 옵션을 여기에서 */etc/vfstab*에 지정하거나 더 편리한 경우 */etc/opt/SUNWsamfs/samfs.cmd* 파일에 지정할 수 있습니다.

    - *shared*
    - *stripe=1*
    - *sync_meta=1*
    - *mh_write*
    - *qwrite*
    - *forcedirectio*
    - *notrace*
    - *rdlease=300*
    - *wrlease=300*

- *aplease=300*

예제에서는 페이지 레이아웃에 맞게 목록이 축약되었습니다.

```
#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      -
...
qfs1rac   -       /global/sc-rac/qfs1rac samfs  -    no      shared,...=300
:wq
[qfs1rac-node1]root@solaris:~#
```

11. 고가용성 공유 파일 시스템에 대한 마운트 지점을 만듭니다.

```
[qfs1rac-node1]root@solaris:~# mkdir -p /global/sc-rac/qfs1rac
[qfs1rac-node1]root@solaris:~#
```

12. 주 노드에 고가용성 공유 파일 시스템을 마운트합니다.

```
[qfs1rac-node1]root@solaris:~# mount /global/sc-rac/qfs1rac
[qfs1rac-node1]root@solaris:~#
```

13. 이제 나머지 SC-RAC 클러스터 노드에서 잠재적 QFS 메타데이터 서버 구성을 수행합니다.

나머지 SC-RAC 클러스터 노드에서 잠재적 QFS 메타데이터 서버 구성

클러스터의 나머지 노드는 잠재적 메타데이터 서버로 사용됩니다. 잠재적 메타데이터 서버는 메타데이터 장치에 액세스하고 메타데이터 서버의 역할을 담당할 수 있는 호스트입니다. 따라서 주 노드의 활성 메타데이터 서버가 실패할 경우 Solaris Cluster 소프트웨어는 보조 노드로 페일오버하여 잠재적 메타데이터 서버를 활성화할 수 있습니다.

SC-RAC 클러스터의 각 나머지 노드에 대해 다음과 같이 하십시오.

1. 노드에 *root*로 로그인합니다.

   예제에서는 현재 노드가 *qfs1rac-node2*입니다.

```
[qfs1rac-node2]root@solaris:~#
```

2. */etc/opt/SUNWsamfs/mcf* 파일을 주 노드에서 현재 노드로 복사합니다.
3. *mcf* 파일에서 오류를 검사합니다. */opt/SUNWsamfs/sbin/sam-fsd* 명령을 실행하고 발견된 모든 오류를 수정합니다.

sam-fsd 명령은 Oracle HSM 구성 파일을 읽고 파일 시스템을 초기화합니다. 만일 오류가 발생하면 중지됩니다. 예제에서는 *qfs1rac-node2* 호스트에서 *mcf* 파일을 검사합니다.

```
[qfs1rac-node2]root@solaris:~# sam-fsd
...
Would start sam-archiverd()
Would start sam-stagealld()
Would start sam-stagerd()
Would start sam-amld()
[qfs1rac-node2]root@solaris:~#
```

4. 텍스트 편집기에서 운영체제의 */etc/vfstab* 파일을 열고 새 파일 시스템에 대한 라인을 시작합니다.

   이 예에서는 *vi* 편집기를 사용합니다.

```
[qfs1rac-node2]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       -
...
qfs1rac    -        /global/sc-rac/qfs1rac samfs   -     no
```

5. */etc/vfstab* 파일의 일곱번째 열에 아래 나열된 마운트 옵션을 입력합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   다음 마운트 옵션은 SC-RAC 클러스터 구성에 권장됩니다. 이러한 마운트 옵션을 여기에서 */etc/vfstab*에 지정하거나 더 편리한 경우 */etc/opt/SUNWsamfs/samfs.cmd* 파일에 지정할 수 있습니다.

   - *shared*
   - *stripe=1*
   - *sync_meta=1*
   - *mh_write*
   - *qwrite*
   - *forcedirectio*
   - *notrace*
   - *rdlease=300*
   - *wrlease=300*

- *aplease=300*

예제에서는 페이지 레이아웃에 맞게 목록이 축약되었습니다.

```
#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      -
...
qfs1rac   -       /global/sc-rac/qfs1rac samfs   -     no      shared,...=300
:wq
[qfs1rac-node2]root@solaris:~#
```

6. 보조 노드에서 고가용성 공유 파일 시스템에 대한 마운트 지점을 만듭니다.

```
[qfs1rac-node2]root@solaris:~# mkdir -p /global/sc-rac/qfs1rac
[qfs1rac-node2]root@solaris:~#
```

7. 보조 노드에 고가용성 공유 파일 시스템을 마운트합니다.

```
[qfs1rac-node2]root@solaris:~# mount /global/sc-rac/qfs1rac
[qfs1rac-node2]root@solaris:~#
```

8. 이제 SC-RAC 메타데이터 서버의 페일오버 구성을 수행합니다.

SC-RAC 메타데이터 서버의 페일오버 구성

Solaris Cluster 소프트웨어가 관리하는 클러스터에서 Oracle HSM 공유 파일 시스템을 호스팅할 경우 Oracle HSM 소프트웨어에 의해 정의된 리소스 유형인 *SUNW.qfs* 클러스터 리소스를 만들어 메타데이터 서버의 페일오버를 구성합니다(자세한 내용은 *SUNW.qfs* 매뉴얼 페이지 참조). SC-RAC 구성에 대한 리소스를 만들고 구성하려면 다음과 같이 하십시오.

1. SC-RAC 클러스터의 주 노드에 *root*로 로그인합니다.

   예제에서는 주 노드가 *qfs1rac-node1*입니다.

```
[qfs1rac-node1]root@solaris:~#
```

2. Solaris Cluster 소프트웨어에 대한 QFS 리소스 유형 *SUNW.qfs*를 정의합니다. *clresourcetype registerSUNW.qfs* 명령을 사용합니다.

```
[qfs1rac-node1]root@solaris:~# clresourcetype registerSUNW.qfs
[qfs1rac-node1]root@solaris:~#
```

3. 등록 파일을 찾을 수 없어 등록에 실패한 경우 */opt/SUNWsamfs/sc/etc/* 디렉토리에 대한 심볼릭 링크를 Solaris Cluster가 리소스 유형 등록 파일을 유지하는 디렉토리인 */opt/cluster/lib/rgm/rtreg/*에 배치합니다.

   Oracle HSM 소프트웨어를 설치하기 전에 Oracle Solaris Cluster 소프트웨어를 설치하지 않았습니다. 일반적으로 Oracle HSM는 설치 도중 Solaris Cluster를 감지한 경우 *SUNW.qfs* 등록 파일의 위치를 자동으로 제공합니다. 따라서 링크를 수동으로 만들 필요가 없습니다.

```
[qfs1rac-node1]root@solaris:~# cd /opt/cluster/lib/rgm/rtreg/
[qfs1rac-node1]root@solaris:~# ln -s /opt/SUNWsamfs/sc/etc/SUNW.qfs SUNW.qfs
[qfs1rac-node1]root@solaris:~#
```

4. QFS 메타데이터 서버에 대한 리소스 그룹을 만듭니다. Solaris Cluster 명령 *clresourcegroup create -n node-list group-name*을 사용합니다. 여기서 *node-list*는 콤마로 구분된 클러스터 노드 목록이고 *group-name*은 리소스 그룹에 사용할 이름입니다.

   예제에서는 SC-RAC 서버 노드가 멤버로 포함된 리소스 그룹 *qfsracrg*를 만듭니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[qfs1rac-node1]root@solaris:~# clresourcegroup create /
-n qfs1rac-node1,qfs1rac-node2 qfsracrg
[qfs1rac-node1]root@solaris:~#
```

5. 새 리소스 그룹에서 활성 메타데이터 서버에 대한 가상 호스트 이름을 설정합니다. Solaris Cluster 명령 *clreslogicalhostname create -g group-name*을 사용합니다. 여기서 *group-name*은 QFS 리소스 그룹의 이름이고 *virtualMDS*는 가상 호스트 이름입니다.

   공유 파일 시스템에 대한 hosts 파일에 사용한 것과 같은 가상 호스트 이름을 사용합니다. 예제에서는 *qfsracrg* 리소스 그룹에 가상 호스트 *qfs1rac-mds*를 만듭니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[qfs1rac-node1]root@solaris:~# clreslogicalhostname create /
-g qfsracrg qfs1rac-mds
[qfs1rac-node1]root@solaris:~#
```

6. QFS 파일 시스템 리소스를 리소스 그룹에 추가합니다. *clresource create -g group-name -t SUNW.qfs -x QFSFileSystem=mount-point -y Resource_dependencies=virtualMDS resource-name* 명령을 사용합니다. 설명:
   - *group-name*은 QFS 리소스 그룹의 이름입니다.
   - *mount-point*는 시스템 루트 디렉토리 바로 아래에 있지 않은 하위 디렉토리인 클러스터의 파일 시스템에 대한 마운트 지점입니다.

공유 QFS 파일 시스템을 루트 바로 아래에 마운트하면 *SUNW.qfs* 리소스 유형 사용 시에 페일오버 문제가 발생할 수 있습니다.

- *virtualMDS*는 활성 메타데이터 서버의 가상 호스트 이름입니다.
- *resource-name*은 리소스에 제공할 이름입니다.

예제에서는 *SUNW.qfs* 유형의 *scrac*라는 리소스를 리소스 그룹 *qfsracrg*에서 만듭니다. *SUNW.qfs* 확장 등록 정보 *QFSFileSystem*을 */global/sc-rac/qfs1rac* 마운트 지점으로 설정합니다. 표준 등록 정보 *Resource_dependencies*를 활성 메타데이터 서버의 논리적 호스트인 *qfs1rac-mds*로 설정합니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[qfs1rac-node1]root@solaris:~# create -g qfsracrg -t SUNW.qfs /
-x QFSFileSystem=/global/sc-rac/qfs1rac /
-y Resource_dependencies=qfs1rac-mds scrac
[qfs1rac-node1]root@solaris:~#
```

7. 자원 그룹을 온라인으로 전환합니다. Solaris Cluster 명령 *clresourcegroup manage group-name* 및 *clresourcegroup online -emM group-name*을 사용합니다. 여기서 *group-name*은 QFS 리소스 그룹의 이름입니다.

   예제에서는 *qfsracrg* 리소스 그룹을 온라인 상태로 만듭니다.

```
[qfs1rac-node1]root@solaris:~# clresourcegroup manage qfsracrg
[qfs1rac-node1]root@solaris:~# clresourcegroup online -emM qfsracrg
[qfs1rac-node1]root@solaris:~#
```

8. QFS 리소스 그룹이 온라인 상태인지 확인합니다. Solaris Cluster 명령 *clresourcegroup status*를 사용합니다.

   예제에서는 *qfsracrg* 리소스 그룹이 주 노드 *qfs1rac-node1*에서 *online* 상태입니다.

```
[qfs1rac-node1]root@solaris:~# clresourcegroup status
=== Cluster Resource Groups ===
Group Name  Node Name       Suspended   Status
----------  -------------   ---------   ------
qfsracrg    qfs1rac-node1   No          Online
            qfs1rac-node2   No          Offline
            qfs1rac-node3   No          Offline
[qfs1rac-node1]root@solaris:~#
```

9. 리소스 그룹이 제대로 페일오버되는지 확인합니다. 리소스 그룹을 보조 노드로 이동합니다. Solaris Cluster 명령 *clresourcegroup switch -n node2 group-name*을 사용합니다. 여기서 *node2*는 보조 노드의 이름이고 *group-name*은 HA-SAM 리소스 그룹에

사용하도록 선택한 이름입니다. 그런 다음 *clresourcegroup status*를 사용하여 결과를 검사합니다.

예제에서는 *qfsracrg* 리소스 그룹을 *qfs1rac-node2* 및 *qfs1rac-node3*으로 이동하고 이 리소스 그룹이 지정된 노드에서 온라인 상태가 되는지 확인합니다.

```
[qfs1rac-node1]root@solaris:~# clresourcegroup switch -n qfs1rac-node2 qfsracrg
[qfs1rac-node1]root@solaris:~# clresourcegroup status
=== Cluster Resource Groups ===
Group Name  Node Name       Suspended   Status
----------  -------------   ---------   ------
qfsracrg    qfs1rac-node1   No          Offline
            qfs1rac-node2   No          Online
            qfs1rac-node3   No          Offline
[qfs1rac-node1]root@solaris:~# clresourcegroup switch -n qfs1rac-node3 qfsracrg
[qfs1rac-node1]root@solaris:~# clresourcegroup status
=== Cluster Resource Groups ===
Group Name  Node Name       Suspended   Status
----------  -------------   ---------   ------
qfsracrg    qfs1rac-node1   No          Offline
            qfs1rac-node2   No          Offline
            qfs1rac-node3   No          Online
[qfs1rac-node1]root@solaris:~#
```

10. 리소스 그룹을 다시 주 노드로 이동합니다. Solaris Cluster 명령 *clresourcegroup switch -n node1 group-name*을 사용합니다. 여기서 *node1*은 주 노드의 이름이고 *group-name*은 HA-SAM 리소스 그룹에 사용하도록 선택한 이름입니다. 그런 다음 *clresourcegroup status*를 사용하여 결과를 검사합니다.

    예제에서는 *qfsracrg* 리소스 그룹을 *qfs1rac-node1*로 성공적으로 다시 이동합니다.

```
[qfs1rac-node1]root@solaris:~# clresourcegroup switch -n qfs1rac-node1 qfsracrg
[qfs1rac-node1]root@solaris:~# clresourcegroup status
=== Cluster Resource Groups ===
Group Name  Node Name       Suspended   Status
----------  -------------   ---------   ------
samr        qfs1rac-node1   No          Online
            qfs1rac-node2   No          Offline
            qfs1rac-node3   No          Offline
[qfs1rac-node1]root@solaris:~#
```

11. 사이드밴드 데이터베이스 기능을 사용할 계획인 경우 10장. 보고 데이터베이스 구성을 참조하십시오.
12. 그렇지 않은 경우 11장. 알림 및 로깅 구성으로 이동합니다.

소프트웨어 RAID 스토리지를 사용하여 SC-RAC 노드에서 QFS 메타데이터 서버 구성

고가용성 파일 시스템은 데이터 및 메타데이터를 중복 주 스토리지 장치에 저장해야 합니다. 중복 디스크 배열 하드웨어는 메타데이터를 위한 RAID-1 또는 RAID-10 및 데이터를 위한 RAID-5를 사용하여 이 중복성을 제공할 수 있습니다. 그러나 일반 이중 포트 SCSI 디스크 장치 또는 JBOD(*Just a Bunch Of Disks*) 배열을 주 스토리지로 사용해야 할 경우 소프트웨어에서 필요한 중복성을 제공해야 합니다.

이러한 이유로 SC-RAC 구성은 Oracle SVM(Solaris Volume Manager) 다중 소유자 디스크 세트에 기초한 소프트웨어 RAID 구성을 지원합니다. 이 절에서는 이러한 변형된 SC-RAC 파일 시스템 구성을 설정할 때 수행해야 하는 기본 단계에 대해 설명합니다.

중복 스토리지 배열을 관리하기 위해서만 Solaris Volume Manager를 사용해야 합니다. 별개의 장치에서 스토리지를 연결하지 마십시오. 이 경우 I/O가 구성 요소 장치에 비효율적으로 분산되고 QFS 파일 시스템 성능이 저하됩니다.

다음 작업을 수행합니다.

- Solaris 11+에 Solaris Volume Manager 설치
- Solaris Volume Manager 다중 소유자 디스크 그룹 만들기
- QFS 데이터 및 메타데이터에 대한 미러링된 볼륨 만들기
- 미러링된 볼륨을 사용하여 SC-RAC 클러스터에서 QFS 공유 파일 시스템 만들기

Solaris 11+에 Solaris Volume Manager 설치

Solaris 11부터 SVM(Solaris Volume Manager)은 더 이상 Solaris에 포함되지 않습니다. 그러나 Solaris Cluster 4 소프트웨어는 Solaris Volume Manager를 계속해서 지원합니다. 따라서 이 소프트웨어를 사용하려면 Solaris 10 9/10 릴리스에 포함된 버전을 다운로드하여 설치해야 합니다. 클러스터의 각 노드에 대해 다음과 같이 하십시오.

1. 노드에 *root*로 로그인합니다.

   아래 예제에서는 Solaris IPS(이미지 패키징 시스템)를 사용하여 클러스터 노드 *qfs2rac-node1*을 구성합니다.

```
[qfs2rac-node1]root@solaris:~#
```

2. 로컬로 사용 가능한 SVM(Solaris Volume Manager) 패키지를 확인합니다. *pkg info svm* 명령을 사용합니다.

```
[qfs2rac-node1]root@solaris:~# pkg info svm
pkg: info: no packages matching the following patterns you specified are
installed on the system.  Try specifying -r to query remotely:
        svm
[qfs2rac-node1]root@solaris:~#
```

3. 패키지를 로컬로 찾을 수 없는 경우 Solaris IPS(이미지 패키징 시스템) 저장소를 확인합니다. *pkg -r svm* 명령을 사용합니다.

```
[qfs2rac-node1]root@solaris:~# pkg -r svm
          Name: storage/svm
       Summary: Solaris Volume Manager
   Description: Solaris Volume Manager commands
      Category: System/Core
         State: Not installed
     Publisher: solaris
       Version: 0.5.11
 Build Release: 5.11
        Branch: 0.175.0.0.0.2.1
Packaging Date: October 19, 2011 06:42:14 AM
          Size: 3.48 MB
          FMRI: pkg://solaris/storage/svm@0.5.11,5.11-0.175.0.0.0.2.1:20111019T064214Z
[qfs2rac-node1]root@solaris:~#
```

4. 패키지를 설치합니다. *pkg install storage/svm* 명령을 사용합니다.

```
[qfs2rac-node1]root@solaris:~# pkg install storage/svm
           Packages to install:   1
       Create boot environment:  No
Create backup boot environment: Yes
            Services to change:   1
DOWNLOAD      PKGS       FILES    XFER (MB)
Completed      1/1     104/104      1.6/1.6
PHASE            ACTIONS
Install Phase    168/168
PHASEITEMS
Package State Update Phase         1/1
Image State Update Phase           2/2
[qfs2rac-node1]root@solaris:~#
```

5. 설치가 완료되면 *metadb*의 위치를 확인합니다. *which metadb* 명령을 사용합니다.

```
[qfs2rac-node1]root@solaris:~# which metadb
/usr/sbin/metadb
[qfs2rac-node1]root@solaris:~#
```

6. 설치를 확인합니다. *metadb* 명령을 사용합니다.

```
[qfs2rac-node1]root@solaris:~# metadb
[qfs2rac-node1]root@solaris:~#
```

7. *metadb*에서 오류가 반환될 경우 *kernel/drv/md.conf* 파일이 있는지 확인합니다.

```
[qfs2rac-node1]root@solaris:~# metadb
metadb: <HOST>: /dev/md/admin: No such file or directory
[qfs2rac-node1]root@solaris:~# ls -l /kernel/drv/md.conf
-rw-r--r--   1 root     sys          295 Apr 26 15:07 /kernel/drv/md.conf
[qfs2rac-node1]root@solaris:~#
```

8. *kernel/drv/md.conf* 파일이 없을 경우 만듭니다. *root*를 파일 소유자로 만들고 *sys*를 그룹 소유자로 만듭니다. 권한을 *644*로 설정합니다.

   예제에서는 *vi* 편집기를 사용하여 파일을 만듭니다. 파일의 내용은 다음과 같아야 합니다.

```
[qfs2rac-node1]root@solaris:~# vi kernel/drv/md.conf
#ident   "@(#)md.conf    2.1   00/07/07 SMI"
#
# Copyright (c) 1992-1999 by Sun Microsystems, Inc.
# All rights reserved.
#
name="md" parent="pseudo" nmd=128 md_nsets=4;
:wq
[qfs2rac-node1]root@solaris:~# chown root:sys kernel/drv/md.conf
[qfs2rac-node1]root@solaris:~# chmod 644
[qfs2rac-node1]root@solaris:~#
```

9. *md.conf* 파일을 동적으로 다시 스캔하고 장치 트리가 업데이트되었는지 확인합니다. *update_drv -f md* 명령을 사용합니다.

   예제에서는 장치 트리가 업데이트되었습니다. 따라서 Solaris Volume Manager가 설치되었습니다.

```
[qfs2rac-node1]root@solaris:~# update_drv -f md
[qfs2rac-node1]root@solaris:~# ls -l  /dev/md/admin
lrwxrwxrwx   1 root root 31 Apr 20 10:12 /dev/md/admin -> ../../devices/pseudo/md@0:admin
[qfs2rac-node1]root@solaris:~#
```

10. 이제 Solaris Volume Manager 다중 소유자 디스크 그룹 만들기를 수행합니다.

Solaris Volume Manager 다중 소유자 디스크 그룹 만들기

1. SC-RAC 구성의 모든 노드에 *root*로 로그인합니다.

예제에서는 *qfs2rac-node1* 노드에 로그인합니다. 그런 다음 새 터미널 창을 열고 *ssh*를 사용하여 *qfs2rac-node2* 및 *qfs2rac-node3* 노드에 로그인합니다.

```
[qfs2rac-node1]root@solaris:~#

[qfs2rac-node1]root@solaris:~# ssh root@qfs2rac-node2
Password:
[qfs2rac-node2]root@solaris:~#

[qfs2rac-node1]root@solaris:~# ssh root@qfs2rac-node3
Password:
[qfs2rac-node3]root@solaris:~#
```

2. Oracle Solaris Cluster 4.*x*를 Solaris 11.*x* 이상에서 사용하는 중이고 Solaris Volume Manager 설치를 아직 수행하지 않은 경우 계속하기 전에 각 노드에서 설치를 수행합니다.

   Solaris 11부터 Solaris Volume Manager는 기본적으로 설치되지 않습니다.
3. 각 노드에서 새 상태 데이터베이스 장치를 연결하고 상태 데이터베이스 복제본 3개를 만듭니다. *metadb -a -f -c3 device-name* 명령을 사용합니다. 여기서 *device-name*은 *cXtYdYsZ* 형태의 물리적 장치 이름입니다.

   Solaris Cluster DID(장치 식별자)를 사용하지 마십시오. 물리적 장치 이름을 사용합니다. 예제에서는 3개의 클러스터 노드 모두에서 상태 데이터베이스 장치를 만듭니다.

```
[qfs2rac-node1]root@solaris:~# metadb -a -f -c3 /dev/rdsk/c0t0d0

[qfs2rac-node2]root@solaris:~# metadb -a -f -c3 /dev/rdsk/c0t6d0

[qfs2rac-node3]root@solaris:~# metadb -a -f -c3 /dev/rdsk/c0t4d0
```

4. 하나의 노드에 Solaris Volume Manager 다중 소유자 디스크 그룹을 만듭니다. *metaset -sdiskset -M -a -h host-list* 명령을 사용합니다. 여기서 *host-list*는 공백으로 구분된 소유자 목록입니다.

   Solaris Volume Manager는 디스크 세트당 최대 4개의 호스트를 지원합니다. 예제에서는 *qfs2rac-node1*에 디스크 그룹 *datadisks*를 만들고 3개 노드 *qfs2rac-node1*, *qfs2rac-node2* 및 *qfs2rac-node3*을 소유자로 지정합니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[qfs2rac-node1]root@solaris:~# metaset -s datadisks -M -a -h qfs2rac-node1 /
qfs2rac-node2 qfs2rac-node3
```

5. 노드 중 하나에 있는 장치를 나열합니다. Solaris Cluster 명령 *cldevice list -n -v*를 사용합니다.

```
[qfs2rac-node1]root@solaris:~# cldevice list -n -v
DID Device  Full Device Path
----------  ----------------
d13         qfs2rac-node1:/dev/rdsk/c6t600C0FF00000000000332B62CF3A6B00d0
d14         qfs2rac-node1:/dev/rdsk/c6t600C0FF0000000000876E950F1FD9600d0
d15         qfs2rac-node1:/dev/rdsk/c6t600C0FF0000000000876E9124FAF9C00d0
...
[qfs2rac-node1]root@solaris:~#
```

6. *cldevice list -n -v* 명령의 출력에서 미러링할 장치를 선택합니다.

   예제에서는 4개의 미러를 위한 네 쌍의 장치인 *d21* 및 *d13*, *d14* 및 *d17*, *d23* 및 *d16*, *d15* 및 *d19*를 선택합니다.

```
[qfs2rac-node1]root@solaris:~# cldevice list -n -v
DID Device  Full Device Path
----------  ----------------
d13         qfs2rac-node1:/dev/rdsk/c6t600C0FF00000000000332B62CF3A6B00d0
d14         qfs2rac-node1:/dev/rdsk/c6t600C0FF0000000000876E950F1FD9600d0
d15         qfs2rac-node1:/dev/rdsk/c6t600C0FF0000000000876E9124FAF9C00d0
d16         qfs2rac-node1:/dev/rdsk/c6t600C0FF00000000000332B28488B5700d0
d17         qfs2rac-node1:/dev/rdsk/c6t600C0FF000000000086DB474EC5DE900d0
d18         qfs2rac-node1:/dev/rdsk/c6t600C0FF0000000000876E975EDA6A000d0
d19         qfs2rac-node1:/dev/rdsk/c6t600C0FF000000000086DB47E331ACF00d0
d20         qfs2rac-node1:/dev/rdsk/c6t600C0FF0000000000876E9780ECA8100d0
d21         qfs2rac-node1:/dev/rdsk/c6t600C0FF000000000004CAD5B68A7A100d0
d22         qfs2rac-node1:/dev/rdsk/c6t600C0FF000000000086DB43CF85DA800d0
d23         qfs2rac-node1:/dev/rdsk/c6t600C0FF000000000004CAD7CC3CDE500d0
d24         qfs2rac-node1:/dev/rdsk/c6t600C0FF000000000086DB4259B272300d0
....
[qfs2rac-node1]root@solaris:~#
```

7. 선택한 장치를 동일한 노드의 디스크 세트에 추가합니다. *metaset -a devicelist* 명령을 사용합니다. 여기서 *devicelist*는 공백으로 구분된 하나 이상의 클러스터 장치 식별자 목록입니다.

   예제에서는 나열된 디스크를 다중 소유자 디스크 세트 *dataset1*에 추가합니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[qfs2rac-node1]root@solaris:~# metaset -s dataset1 -M -a -h /dev/did/rdsk/d21 / /dev/did/rdsk/d13 /dev/did/
rdsk/d14 /dev/did/rdsk/d17 /dev/did/rdsk/d23 /
/dev/did/rdsk/d16 /dev/did/rdsk/d15 /dev/did/rdsk/d19
```

```
[qfs2rac-node1]root@solaris:~#
```

8. 이제 QFS 데이터 및 메타데이터에 대한 미러링된 볼륨 만들기를 수행합니다.

QFS 데이터 및 메타데이터에 대한 미러링된 볼륨 만들기

1. 구성 요소 간의 관계를 명확하게 유지하기 위해 만들려는 RAID-0 논리 볼륨 및 RAID-1 미러에 대한 이름 지정 체계를 결정합니다.

   일반적으로 RAID-1 미러는 *dn*으로 이름이 지정되고 여기서 *n*은 정수입니다. RAID-1 미러를 구성하는 RAID-0 볼륨은 *dnX*로 이름이 지정되고 여기서 *X*는 미러 내에서 장치의 위치를 나타내는 정수입니다(일반적으로 양방향 미러의 경우 *0* 또는 *1*).

   이 절차의 예제에서는 RAID-0 논리 볼륨 쌍에서 양방향 RAID-1 미러를 만듭니다. 따라서 미러의 이름을 *d1*, *d2*, *d3*, *d4* 등으로 지정합니다. 그런 다음 RAID-0 볼륨을 포함하는 RAID-1 미러에 대한 각 쌍의 RAID-0 볼륨 이름을 *d10* 및 *d11*, *d20* 및 *d21*, *d30* 및 *d31*, *d40* 및 *d41* 등으로 지정합니다.
2. 다중 소유자 디스크 세트를 만든 노드에 로그인합니다. *root*로 로그인합니다.

   위 예제에서는 *qfs2rac-node1*에서 디스크 세트를 만들었습니다.

```
[qfs2rac-node1]root@solaris:~#
```

3. 첫번째 RAID-0 논리 볼륨을 만듭니다. *metainit -s diskset-name device-name number-of-stripes components-per-stripe component-names* 명령을 사용합니다. 여기서 각 항목은 다음과 같습니다.
   - *diskset-name*은 디스크 세트에 사용하도록 선택한 이름입니다.
   - *device-name*은 RAID-0 논리 볼륨에 사용하도록 선택한 이름입니다.
   - *number-of-stripes*는 *1*입니다.
   - *components-per-stripe*는 *1*입니다.
   - *component-name*은 RAID-0 볼륨에서 사용할 디스크 세트 구성 요소의 장치 이름입니다.

   예제에서는 다중 소유자 디스크 세트 *dataset1*의 클러스터(DID) 장치 */dev/did/dsk/d21s0*을 사용하여 RAID-0 논리 볼륨 *d10*을 만듭니다.

```
[qfs2rac-node1]root@solaris:~# metainit -s dataset1 d10 1 1 /dev/did/dsk/d21s0
[qfs2rac-node1]root@solaris:~#
```

4. 나머지 RAID-0 논리 볼륨을 만듭니다.

```
[qfs2rac-node1]root@solaris:~# metainit -s dataset1 d11 1 1 /dev/did/dsk/d13s0
[qfs2rac-node1]root@solaris:~# metainit -s dataset1 d20 1 1 /dev/did/dsk/d14s0
[qfs2rac-node1]root@solaris:~# metainit -s dataset1 d21 1 1 /dev/did/dsk/d17s0
[qfs2rac-node1]root@solaris:~# metainit -s dataset1 d30 1 1 /dev/did/dsk/d23s0
```

```
[qfs2rac-node1]root@solaris:~# metainit -s dataset1 d31 1 1 /dev/did/dsk/d16s0
[qfs2rac-node1]root@solaris:~# metainit -s dataset1 d40 1 1 /dev/did/dsk/d15s0
[qfs2rac-node1]root@solaris:~# metainit -s dataset1 d41 1 1 /dev/did/dsk/d19s0
...
[qfs2rac-node1]root@solaris:~#
```

5. 첫번째 RAID-1 미러를 만듭니다. *metainit -s diskset-name RAID-1-mirrorname -m RAID-0-volume0* 명령을 사용합니다. 여기서 각 항목은 다음과 같습니다.
   - *diskset-name*은 다중 소유자 디스크 세트의 이름입니다.
   - *RAID-1-mirrorname*은 RAID-1 미러링된 볼륨의 이름입니다.
   - *RAID-0-volume0*은 미러에 추가하는 중인 첫번째 RAID-0 논리 볼륨입니다.

   예제에서는 *d1* 미러를 만들고 미러의 첫번째 RAID-0 볼륨 *d10*을 추가합니다.

```
[qfs2rac-node1]root@solaris:~# metainit -s dataset1 d1 -m d10
[qfs2rac-node1]root@solaris:~#
```

6. 나머지 RAID-0 볼륨을 첫번째 RAID-1 미러에 추가합니다. *metattach -s diskset-name RAID-1-mirrorname RAID-0-volume* 명령을 사용합니다. 여기서 각 항목은 다음과 같습니다.
   - *diskset-name*은 다중 소유자 디스크 세트의 이름입니다.
   - *RAID-1-mirrorname*은 RAID-1 미러링된 볼륨의 이름입니다.
   - *RAID-0-volume*은 미러에 추가하는 중인 RAID-0 논리 볼륨입니다.

   예제에서는 *d1*이 이중 미러이므로 단일 RAID-0 볼륨 *d11*을 추가합니다.

```
[qfs2rac-node1]root@solaris:~# metattach -s dataset1 d11 d1
[qfs2rac-node1]root@solaris:~#
```

7. 나머지 미러를 만듭니다.

   예제에서는 *d2*, *d3*, *d4* 등의 미러를 만듭니다.

```
[qfs2rac-node1]root@solaris:~# metainit -s dataset1 d2 -m d20
[qfs2rac-node1]root@solaris:~# metattach -s dataset1 d21 d2
[qfs2rac-node1]root@solaris:~# metainit -s dataset2 d3 -m d30
[qfs2rac-node1]root@solaris:~# metattach -s dataset2 d31 d3
[qfs2rac-node1]root@solaris:~# metainit -s dataset2 d4 -m d40
[qfs2rac-node1]root@solaris:~# metattach -s dataset2 d41 d4
...
[qfs2rac-node1]root@solaris:~#
```

8. QFS 파일 시스템 메타데이터를 보관할 미러를 선택합니다.

아래 예제에서는 *d1* 및 *d2* 미러를 선택합니다.

9. 선택한 미러에서 QFS 메타데이터를 보관할 소프트 분할 영역을 만듭니다. 각 미러에 대해 *metainit -s diskset-name partition-name -p RAID-1-mirrorname size* 명령을 사용합니다. 설명:
   - *diskset-name*은 다중 소유자 디스크 세트의 이름입니다.
   - *partition-name*은 새 분할 영역의 이름입니다.
   - *RAID-1-mirrorname*은 미러의 이름입니다.
   - *size*는 분할 영역의 크기입니다.

   예제에서는 2개의 500GB 분할 영역을 만듭니다. 즉, *d1* 미러에서 *d53*을 만들고 *d2* 미러에서 *d63*을 만듭니다.

```
[qfs2rac-node1]root@solaris:~# metainit -s dataset1 d53 -p d1 500g
[qfs2rac-node1]root@solaris:~# metainit -s dataset1 d63 -p d2 500g
```

10. 이제 미러링된 볼륨을 사용하여 SC-RAC 클러스터에서 QFS 공유 파일 시스템 만들기를 수행합니다.

미러링된 볼륨을 사용하여 SC-RAC 클러스터에서 QFS 공유 파일 시스템 만들기

1. 아직 수행하지 않은 경우 “모든 SC-RAC 클러스터 노드에서 QFS 공유 파일 시스템 hosts 파일 만들기” 절차를 수행합니다. 완료되면 여기로 돌아옵니다.
2. SC-RAC 클러스터에 대한 주 노드 및 QFS 공유 파일 시스템에 대한 활성 메타데이터 서버 둘 다로 사용될 클러스터 노드를 선택합니다. *root*로 로그인합니다.

   예제에서는 *qfs2rac-node1* 노드를 선택합니다.

```
[qfs2rac-node1]root@solaris:~#
```

3. 주 노드에서 고성능 공유 *ma* 파일 시스템을 만듭니다. Solaris Volume Manager 미러링된 디스크 볼륨을 *mm* 메타데이터 장치 및 *mr* 데이터 장치로 사용합니다. 텍스트 편집기에서 */etc/opt/SUNWsamfs/mcf* 파일을 열고 필요한 대로 편집한 다음 파일을 저장합니다.

   예제에서는 *vi* 텍스트 편집기를 사용하여 *qfs2rac* 파일 시스템을 만듭니다. 미러링된 볼륨 *d1* 및 *d2*의 분할 영역은 파일 시스템의 *mm* 메타데이터 장치 2개인 *110* 및 *120*으로 사용됩니다. 미러링된 볼륨 *d3* 및 *d4*는 파일 시스템의 *mr* 데이터 장치 2개인 *130* 및 *140*으로 사용됩니다.

```
[qfs2rac-node1]root@solaris:~# vi /etc/opt/SUNWsamfs/mcf
# /etc/opt/SUNWsamfs/mcf file:
#
# Equipment              Equipment Equipment Family   Device  Additional
```

```
# Identifier               Ordinal   Type      Set       State   Parameters
# ----------------------- --------- --------  -------  ------  ----------
qfs2rac                    100        ma       qfs2rac  on      shared
 /dev/md/dataset1/dsk/d53  110        mm       qfs2rac  on
 /dev/md/dataset1/dsk/d63  120        mm       qfs2rac  on
 /dev/md/dataset1/dsk/d3   130        mr       qfs2rac  on
 /dev/md/dataset1/dsk/d4   140        mr       qfs2rac  on
:wq
[qfs2rac-node1]root@solaris:~#
```

4. *mcf* 파일에서 오류를 검사합니다. */opt/SUNWsamfs/sbin/sam-fsd* 명령을 사용하고 발견된 모든 오류를 수정합니다.

   sam-fsd 명령은 Oracle HSM 구성 파일을 읽고 파일 시스템을 초기화합니다. 만일 오류가 발생하면 중지됩니다. 예제에서는 *qfs2rac-node1* 호스트에서 *mcf* 파일을 검사합니다.

```
[qfs2rac-node1]root@solaris:~# sam-fsd
...
Would start sam-archiverd()
Would start sam-stagealld()
Would start sam-stagerd()
Would start sam-amld()
[qfs2rac-node1]root@solaris:~#
```

5. 파일 시스템을 만듭니다. */opt/SUNWsamfs/sbin/sammkfs -S family-set-name* 명령을 사용합니다. 여기서 *family-set-name*은 파일 시스템의 장비 식별자입니다.

   sammkfs 명령은 *hosts.family-set-name* 및 *mcf* 파일을 읽고 지정된 등록 정보를 가진 공유 파일 시스템을 만듭니다.

```
[qfs2rac-node1]root@solaris:~# sammkfs -S qfs2rac
Building 'qfs2rac' will destroy the contents of devices:
  ...
Do you wish to continue? [y/N]yes ...
[qfs2rac-node1]root@solaris:~#
```

6. 텍스트 편집기에서 운영체제의 */etc/vfstab* 파일을 열고 새 파일 시스템에 대한 라인을 시작합니다. 첫번째 열에 파일 시스템 이름과 공백을 입력하고 두번째 열에 하이픈과 추가 공백을 입력합니다.

   예제에서는 *vi* 텍스트 편집기를 사용합니다. *qfs2rac* 파일 시스템에 대한 라인을 시작합니다. 하이픈은 운영체제에서 UFS 도구를 사용하여 파일 시스템 무결성을 검사하려는 시도를 방지합니다.

```
[qfs2rac-node1]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      -
...
qfs2rac    -
```

7. */etc/vfstab* 파일의 세번째 열에 클러스터를 기준으로 파일 시스템의 마운트 지점을 입력합니다. 시스템 루트 디렉토리의 바로 아래에 있지 않은 마운트 지점 하위 디렉토리를 지정합니다.

   공유 QFS 파일 시스템을 루트 바로 아래에 마운트하면 *SUNW.qfs* 리소스 유형 사용 시에 페일오버 문제가 발생할 수 있습니다. 예제에서는 *qfs2rac* 파일 시스템에 대한 마운트 지점이 */global/sc-rac/qfs2rac*입니다.

```
#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      -
...
qfs2rac    -       /global/sc-rac/qfs2rac samfs  -    no
```

8. */etc/vfstab* 파일의 네번째 열에 파일 시스템 유형(*samfs*)을 입력합니다.

```
#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      -
...
qfs2rac    -       /global/sc-rac/qfs2rac samfs
```

9. */etc/vfstab* 파일의 다섯번째 열에 *fsck* 전달 옵션(-)을 입력합니다.

```
#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      -
...
qfs2rac   -       /global/sc-rac/qfs2rac samfs  -
```

10. */etc/vfstab* 파일의 여섯번째 열에 부트 시 마운트 옵션(*no*)을 입력합니다.

```
#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      -
...
qfs2rac   -       /global/sc-rac/qfs2rac samfs  -    no
```

11. */etc/vfstab* 파일의 일곱번째 열에 *sw_raid* 마운트 옵션 및 SC-RAC 구성에 권장되는 마운트 옵션을 입력합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

    다음 마운트 옵션이 권장됩니다. 이러한 마운트 옵션을 여기에서 */etc/vfstab*에 지정하거나 더 편리한 경우 */etc/opt/SUNWsamfs/samfs.cmd* 파일에 지정할 수 있습니다.

    - *shared*
    - *stripe=1*
    - *sync_meta=1*
    - *mh_write*
    - *qwrite*
    - *forcedirectio*
    - *notrace*
    - *rdlease=300*
    - *wrlease=300*
    - *aplease=300*

    예제에서는 페이지 레이아웃에 맞게 마운트 옵션 목록이 축약되었습니다.

```
#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      -
...
qfs2rac   -       /global/sc-rac/qfs2rac samfs  -    no      shared,...sw_raid
:wq
```

```
[qfs2rac-node1]root@solaris:~#
```

12. 고가용성 공유 파일 시스템에 대한 마운트 지점을 만듭니다.

```
[qfs2rac-node1]root@solaris:~# mkdir -p /global/sc-rac/qfs2rac
[qfs2rac-node1]root@solaris:~#
```

13. 주 노드에 고가용성 공유 파일 시스템을 마운트합니다.

```
[qfs2rac-node1]root@solaris:~# mount /global/sc-rac/qfs2rac
[qfs2rac-node1]root@solaris:~#
```

14. 두번째 노드를 설정합니다. “나머지 SC-RAC 클러스터 노드에서 잠재적 QFS 메타데이터 서버 구성 ” 절차를 사용하십시오.
15. 페일오버를 구성합니다. “SC-RAC 메타데이터 서버의 페일오버 구성” 절차를 사용하십시오. 그런 다음 여기로 돌아옵니다.

    Solaris Volume Manager 미러링된 볼륨을 사용하여 SC-RAC 구성을 성공적으로 구성했습니다.
16. 사이드밴드 데이터베이스 기능을 사용할 계획인 경우 10장. 보고 데이터베이스 구성을 참조하십시오.
17. 그렇지 않은 경우 11장. 알림 및 로깅 구성으로 이동합니다.

10

10장. 보고 데이터베이스 구성

Oracle HSM에서는 지정된 파일 시스템의 모든 파일에 대한 최신 메타데이터 정보를 저장하는 선택적 보고 데이터베이스를 지원합니다. 이 사이드밴드 데이터베이스는 파일 및 파일 시스템 작업을 관리하고 이 작업에 대해 보고하는 데 매우 유용할 수 있습니다.

Oracle HSM 사이드밴드 데이터베이스는 구현하기 쉽습니다. *samdb* 명령을 사용하여 제공된 스키마(또는 사용자 정의 대안) 및 *samfsdump* 명령을 통해 생성된 복구 지점 파일로 MySQL 데이터베이스를 만들고 구성합니다. 그러면 해당 파일 시스템이 변경될 때 Oracle HSM 데몬 프로세스에서 자동으로 데이터베이스를 업데이트합니다. 추가 *samdb* 명령을 사용하여 데이터베이스를 쿼리하고 관리할 수 있습니다. 명령 및 옵션에 대한 자세한 내용은 *samdb* 및 *samdb.conf* 매뉴얼 페이지를 참조하십시오.

사이드밴드 데이터베이스 기능을 사용하려면 다음 작업을 수행합니다.

- MySQL 서버 소프트웨어 설치 및 구성
- 데이터베이스 로드 파일 만들기
- 사이드밴드 데이터베이스 만들기

MySQL 서버 소프트웨어 설치 및 구성

samdb 보고 기능을 사용으로 설정하려면 MySQL 데이터베이스를 설치하고 구성해야 합니다. 다음과 같이 하십시오.

1. *http://dev.mysql.com/doc/*에서 *MySQL Reference Manual*을 다운로드합니다.

   samdb 보고를 사용으로 설정할 때 필요한 MySQL 작업을 식별하려면 아래 절차를 따릅니다. 하지만 아래 단계가 완전하다거나 신뢰할 수 있다는 의미는 아닙니다. *MySQL Reference Manual*을 참조할 때 길잡이로 삼으십시오.
2. MySQL 서버를 호스트하는 시스템을 *root*로 로그인합니다.

   Oracle HSM 메타데이터 서버 호스트나 독립 Solaris 또는 Linux 호스트에 MySQL 서버를 설치할 수 있습니다.

   예제에서는 MySQL을 Solaris 호스트 *samsql*에 설치합니다.

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

3. *MySQL Reference Manual*에 지시된 대로 MySQL 서버 소프트웨어를 다운로드하고 설치합니다. 자동 시작을 사용으로 설정합니다.

4. *mysql* 클라이언트 및 *root* 사용자 계정을 사용하여 MySQL 서버에 연결합니다. *mysql --user=root -p* 명령을 사용합니다. 메시지가 표시되면 설치 중 *root* 사용자에게 지정한 암호를 입력합니다.

   mysql 명령 셸이 시작합니다.

```
[samsql]root@solaris:~# mysql --user=root -p
Enter Password:
mysql>
```

5. Oracle HSM MySQL 사용자를 만듭니다. *CREATE USER 'user_name'@'host_name' IDENTIFIED BY 'user-password'* 명령을 사용합니다. 설명:
   - *user_name*은 Oracle HSM MySQL 사용자 이름입니다.
   - *host_name*은 MySQL이 Oracle HSM 메타데이터 서버 호스트에 설치된 경우 *localhost*이고, 그렇지 않은 경우 메타데이터 서버의 호스트 이름 또는 IP 주소입니다.
   - *user-password*는 Oracle HSM MySQL 사용자에 지정하는 암호입니다.

   예제에서는 Oracle HSM 메타데이터 서버 *samqfs1mds*에서 *samsql* 사용자를 만듭니다. 사용자 암호 *samsqluserpassw0rd*를 데모용으로 설정합니다(운용 데이터베이스용으로는 안전한 선택이 아님).

```
[samsql]root@solaris:~# mysql --user=root
Enter Password:
mysql> CREATE USER 'samsql'@'samqfs1mds' IDENTIFIED BY 'samsqluserpassw0rd'
mysql>
```

6. Oracle HSM 사용자에게 필요한 권한을 부여합니다. *GRANT CREATE,DROP,INDEX,SELECT,INSERT,UPDATE,DELETE ON host_name TO 'user_name'@'host_name'* 명령을 사용합니다.

   예제에서는 메타데이터 서버 *samqfs1mds*에서 *samsql* 사용자에게 권한을 부여합니다.

```
[samsql]root@solaris:~# mysql --user=root -p
Enter Password:
mysql> CREATE USER 'samsql'@'host_name' IDENTIFIED BY 'samsqluserpassw0rd'
mysql> GRANT CREATE,DROP,INDEX,SELECT,INSERT,UPDATE,DELETE ON samqfs1mds TO /
'samsql'@'samqfs1mds'
mysql>
```

7. MySQL 명령 인터페이스를 닫고 운영체제 명령 셸로 돌아갑니다. MySQL 명령 *QUIT*를 사용합니다.

```
[samsql]root@solaris:~# mysql --user=root -p
```

```
Enter Password:
mysql> CREATE USER 'samsql'@'host_name' IDENTIFIED BY 'samsqluserpassw0rd'
mysql> GRANT CREATE,DROP,INDEX,SELECT,INSERT,UPDATE,DELETE ON samqfs1mds TO /
'samsql'@'samqfs1mds'
mysql> QUIT
Bye
root@solaris:~#
```

8. 이제 데이터베이스 로드 파일 만들기를 수행합니다.

데이터베이스 로드 파일 만들기

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

   예제에서는 *samqfs1mds* 호스트에 로그인합니다.

```
[samqfs1mds]root@solaris:~#
```

2. 최신 복구 지점 파일이 이미 있으므로 복구 지점 파일의 내용에서 데이터베이스 로드 파일을 생성합니다. *samfsrestore -SZ output-path-name -f recoverypoint-file* 명령을 사용합니다. 설명:
   - *-f*는 *recoverypoint-file*을 입력 파일의 경로 및 파일 이름으로 지정합니다.
   - *-SZ*를 지정하면 명령에서 복구 지점 파일을 검색하고 *output-path-name*에서 지정한 경로 및 파일 이름을 사용하여 데이터베이스 로드 파일을 출력합니다.

   자세한 내용은 *samfsdump* 매뉴얼 페이지를 참조하십시오.

   예제에서는 *samqfs1* 파일 시스템을 구성할 때 예약한 일별 복구 지점 파일 */zfs1/hsmqfs1_recovery/140129*를 사용합니다("파일 시스템 보호 구성" 참조). 출력을 데이터베이스 로드 파일 */root/hsmqfs1dataload*로 보냅니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시 문자로 이스케이프됨).

```
[samqfs1mds]root@solaris:~# samfsrestore -SZ /root/hsmqfs1dataload -f /
/zfs1/hsmqfs1_recovery/140129
```

3. 최신 복구 지점 파일이 없는 경우 지금 데이터베이스 로드 파일을 만듭니다. Oracle HSM 파일 시스템의 루트 디렉토리로 변경합니다. 그런 다음 *samfsdump -SZ output-path-name* 명령을 사용합니다.

   자세한 내용은 *samfsdump* 매뉴얼 페이지를 참조하십시오. 예제에서는 */hsmqfs1* 디렉토리로 변경합니다. 출력을 데이터베이스 로드 파일 */root/hsmqfs1dataload*로 보냅니다.

```
[samqfs1mds]root@solaris:~# cd /hsmqfs1
[samqfs1mds]root@solaris:~# samfsdump -SZ /root/hsmqfs1dataload
```

4. 이제 사이드밴드 데이터베이스 만들기를 수행합니다.

사이드밴드 데이터베이스 만들기

1. MySQL 서버 호스트에 *root*로 로그인합니다.

   예제에서는 MySQL 호스트가 Solaris 호스트 *samqfs1mds*에서 호스트됩니다.

```
[samqfs1mds]root@solaris:~#
```

2. 텍스트 편집기에서 */etc/opt/SUNWsamfs/samdb.conf* 파일을 엽니다.

   예제에서는 *vi* 편집기를 사용합니다. 먼저 제목 행을 주석으로 추가합니다.

```
[samqfs1mds]root@solaris:~# vi /etc/opt/SUNWsamfs/samdb.conf
 #FS_NAME:HOST:USER:PASSWORD:NAME:PORT:CLIENT_FLAG:MOUNT_POINT
```

3. *samdb.conf* 파일의 첫번째 열에 파일 시스템의 패밀리 세트 이름을 입력한 후 콜론(*:*)을 열 구분자로 입력합니다.

   예제에서는 패밀리 세트 이름 *samqfs1*을 입력합니다.

```
[samqfs1mds]root@solaris:~# vi /etc/opt/SUNWsamfs/samdb.conf
# /etc/opt/SUNWsamfs/samdb.conf
 #FS_NAME:HOST:USER:PASSWORD:NAME:PORT:CLIENT_FLAG:MOUNT_POINT
samqfs1:
```

4. 두번째 열에 MySQL 데이터베이스 서버의 호스트 이름을 입력한 후 콜론(*:*)을 열 구분자로 입력합니다.

   예제에서는 Oracle HSM 메타데이터 서버 호스트 *samqfs1mds*에서 데이터베이스 서버를 공동 호스팅합니다. 따라서 호스트 이름 *localhost*를 입력합니다.

```
[samqfs1mds]root@solaris:~# vi /etc/opt/SUNWsamfs/samdb.conf
# /etc/opt/SUNWsamfs/samdb.conf
 #FS_NAME:HOST:USER:PASSWORD:NAME:PORT:CLIENT_FLAG:MOUNT_POINT
samqfs1:localhost:
```

5. 세번째 열에 Oracle HSM 소프트웨어가 MySQL 데이터베이스에 액세스할 때 사용하는 사용자 이름을 입력한 후 콜론(*:*)을 열 구분자로 입력합니다.

   예제에서는 데이터베이스에 로그인하는 용도로 *samqfs* 사용자를 만들었습니다.

```
[samqfs1mds]root@solaris:~# vi /etc/opt/SUNWsamfs/samdb.conf
# /etc/opt/SUNWsamfs/samdb.conf
```

```
 #FS_NAME:HOST:USER:PASSWORD:NAME:PORT:CLIENT_FLAG:MOUNT_POINT
samqfs1:localhost:samqfs:
```

6. 네번째 열에 Oracle HSM 소프트웨어가 MySQL 데이터베이스에 액세스할 때 사용하는 암호를 입력한 후 콜론(*:*)을 열 구분자로 입력합니다.

   예제에서는 더미 암호 *P^ssw0rd*를 사용합니다.

```
[samqfs1mds]root@solaris:~# vi /etc/opt/SUNWsamfs/samdb.conf
# /etc/opt/SUNWsamfs/samdb.conf
 #FS_NAME:HOST:USER:PASSWORD:NAME:PORT:CLIENT_FLAG:MOUNT_POINT
samqfs1:localhost:samqfs:P^ssw0rd:
```

7. 다섯번째 열에 MySQL 데이터베이스의 이름을 입력한 후 콜론(*:*)을 열 구분자로 입력합니다.

   예제에서는 데이터베이스 이름을 *samqfs1db*로 지정합니다.

```
[samqfs1mds]root@solaris:~# vi /etc/opt/SUNWsamfs/samdb.conf
# /etc/opt/SUNWsamfs/samdb.conf
 #FS_NAME:HOST:USER:PASSWORD:NAME:PORT:CLIENT_FLAG:MOUNT_POINT
samqfs1:localhost:samqfs:P^ssw0rd:samqfs1db:
```

8. 여섯번째 열에 데이터베이스 서버의 TCP/IP 포트를 입력한 후 콜론(*:*)을 열 구분자로 입력합니다.

   예제에서는 *0*을 입력합니다. 원격 서버를 사용한 경우 0 값을 입력하거나 비워 두면 기본 포트 *3306*이 지정됩니다. 하지만 여기서는 *localhost*를 사용하므로 0은 단순히 자리 표시자로만 사용됩니다.

```
[samqfs1mds]root@solaris:~# vi /etc/opt/SUNWsamfs/samdb.conf
# /etc/opt/SUNWsamfs/samdb.conf
 #FS_NAME:HOST:USER:PASSWORD:NAME:PORT:CLIENT_FLAG:MOUNT_POINT
samqfs1:localhost:samqfs:P^ssw0rd:samqfs1db:0:
```

9. 일곱번째 열에 MySQL 클라이언트 플래그를 입력한 후 콜론(*:*)을 열 구분자로 입력합니다.

   MySQL 클라이언트 플래그는 일반적으로 *0*으로 설정됩니다. 하지만 다양한 값 조합을 설정하여 특정 MySQL 기능을 사용으로 설정할 수 있습니다. 자세한 내용은 *mysql_real_connect()* 함수에 대한 MySQL 설명서를 참조하십시오.

   예제에서는 *0*을 입력합니다.

```
[samqfs1mds]root@solaris:~# vi /etc/opt/SUNWsamfs/samdb.conf
# /etc/opt/SUNWsamfs/samdb.conf
```

```
 #FS_NAME:HOST:USER:PASSWORD:NAME:PORT:CLIENT_FLAG:MOUNT_POINT
samqfs1:localhost:samqfs:Pˆssw0rd:samqfs1db:0:0:
```

10. 여덟번째인 마지막 열에 Oracle HSM 파일 시스템의 마운트 지점을 입력합니다. 파일을 저장하고 편집기를 닫습니다.

    예제에서는 파일 시스템이 */hsmqfs/hsmqfs1*에 마운트됩니다.

```
[samqfs1mds]root@solaris:~# vi /etc/opt/SUNWsamfs/samdb.conf
# /etc/opt/SUNWsamfs/samdb.conf
 #FS_NAME:HOST:USER:PASSWORD:NAME:PORT:CLIENT_FLAG:MOUNT_POINT
samqfs1:localhost:samqfs:Pˆssw0rd:samqfs1db:0:0:/hsmqfs/hsmqfs1
:wq
[samqfs1mds]root@solaris:~#
```

11. 새 데이터베이스 및 연관된 테이블을 만듭니다. *samdb create family_set* 명령을 사용합니다. 여기서 *family_set*는 */etc/opt/SUNWsamfs/mcf* 파일에서 Oracle HSM 파일 시스템에 지정된 패밀리 세트 이름입니다.

    기본 데이터베이스 스키마는 */opt/SUNWsamfs/etc/samdb.schema*입니다. 명령을 *samdb create family_set -s schema*로 입력하여 대체 스키마를 지정할 수 있습니다. 여기서 *schema*는 스키마 파일의 경로 및 이름입니다.

    예제에서는 기본 스키마를 사용하여 파일 시스템 패밀리 세트 *samqfs1*에 대한 데이터베이스를 만듭니다.

```
[samqfs1mds]root@solaris:~# samdb create samqfs1
```

12. 이전 절차에서 만든 데이터베이스 로드 파일에 포함된 데이터로 데이터베이스를 채웁니다. *samdb load family_set input_file* 명령을 사용합니다. 여기서 *family_set*는 */etc/opt/SUNWsamfs/mcf* 파일에서 파일 시스템에 대해 지정한 패밀리 세트이고 *input_file*은 데이터베이스 로드 파일의 경로 및 이름입니다.

    예제에서는 데이터베이스 로드 파일 */root/hsmqfs1dataload*를 사용하여 파일 시스템 패밀리 세트 *samqfs1*에 대한 데이터베이스를 로드합니다.

```
[samqfs1mds]root@solaris:~# samdb load samqfs1 /root/hsmqfs1dataload
```

13. 데이터베이스가 일관성이 있는지 확인합니다. *samdb check family_set* 명령을 사용합니다. 여기서 *family_set*는 */etc/opt/SUNWsamfs/mcf* 파일에서 파일 시스템에 지정한 패밀리 세트 이름입니다.

    samdb check 명령은 데이터베이스 항목을 현재 파일 시스템 메타데이터와 비교합니다. 로드 프로세스 중 발생할 수 있는 불일치를 발견하고 가능하면 수정합니다.

예제에서는 데이터베이스 로드 파일 */root/hsmqfs1dataload*를 사용하여 파일 시스템 패밀리 세트 *samqfs1*에 대한 데이터베이스를 로드합니다.

```
[samqfs1mds]root@solaris:~# samdb check samqfs1
```

14. 이제 데이터베이스 지원을 사용으로 설정하고 Oracle HSM 파일 시스템 마운트를 수행합니다.

데이터베이스 지원을 사용으로 설정하고 Oracle HSM 파일 시스템 마운트

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

```
[samqfs1mds]root@solaris:~#
```

2. */etc/vfstab* 파일을 백업합니다.

```
[samqfs1mds]root@solaris:~# cp /etc/vfstab /etc/vfstab.backup
```

3. 텍스트 편집기에서 */etc/vfstab* 파일을 열고 데이터베이스를 만든 파일 시스템 항목까지 아래로 스크롤합니다.

   예제에서는 *vi* 편집기를 사용합니다. 여기서는 *samqfs1* 파일 시스템 항목까지 아래로 스크롤합니다.

```
[samqfs1mds]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       -
...
samqfs1    -       /hsmqfs1  samfs   -     yes      ... ,partial=64
```

4. */etc/vfstab* 파일의 마지막 열에서 *sam_db*를 파일 시스템의 마운트 옵션 목록에 추가합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   예제에서는 *samqfs1* 파일 시스템에서 사이드밴드 데이터베이스를 사용으로 설정합니다.

```
[samqfs1mds]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       -
...
samqfs1    -        /hsmqfs1  samfs   -     yes      ... ,partial=64,sam_db
:wq
root@solaris:~#
```

5. Oracle HSM 아카이빙 파일 시스템을 마운트합니다.

   파일 시스템이 *sam_db* 옵션을 사용하여 마운트되면 Oracle HSM 소프트웨어에서 사이드밴드 데이터베이스를 업데이트하는 프로세스를 시작합니다.

   예제에서는 파일 시스템 */hsmqfs1*을 마운트합니다.

```
[samqfs1mds]root@solaris:~# mount /hsmqfs1
```

6. 다음에는 11장. 알림 및 로깅 구성으로 이동합니다.

11

11장. 알림 및 로깅 구성

Oracle HSM 파일 시스템은 SNMP(Simple Network Management Protocol)를 사용하여 자동화된 원격 알림을 지원하고 포괄적으로 구성 가능한 로깅 기능을 제공합니다. 이 장에서는 다음 항목을 설명합니다.

- SNMP(Simple Network Management Protocol) 구성
- Oracle HSM 로깅 사용
- 장치 로깅 구성
- 로그 교체 구성
- 전자 메일 경보 사용.

SNMP(Simple Network Management Protocol) 구성

네트워크 관리 응용 프로그램은 SNMP(Simple Network Management Protocol)를 사용하여 Oracle HSM 파일 시스템을 모니터할 수 있습니다. SNMP 에이전트를 구성하여 결함 및 구성 변경사항을 네트워크 관리 스테이션에 알리는 트랩을 자동으로 보낼 수 있습니다.

Oracle HSM MIB(Management Information Base)는 SNMP 트랩에서 제공되는 정보 유형을 정의합니다. 구성 오류, SCSI *tapealert* 이벤트 및 다양한 유형의 이례적인 시스템 작업이 여기에 포함됩니다. 자세한 내용은 MIB(Management Information Base) 파일(*/var/snmp/mib/SUN-SAM-MIB.mib*)을 참조하십시오.

Oracle HSM 트랩 이벤트가 발생하면 Solaris 커널 시스템 이벤트 알림 데몬 *syseventd*가 */etc/opt/SUNWsamfs/scripts/sendtrap* 스크립트를 호출합니다. 그런 다음 스크립트가 트랩을 로컬 호스트나 지정된 관리 스테이션으로 보냅니다. 이 스크립트는 SNMP 표준 2c 버전을 지원합니다. 이 버전은 이전 버전의 표준과 호환됩니다. 2c 버전은 인증 자격 증명(커뮤니티 문자열) 및 관리 데이터를 일반 텍스트로 교환합니다. 자세한 내용은 *sendtrap* 매뉴얼 페이지를 참조하십시오.

SNMP 알림을 구성하려면 다음 작업을 수행합니다.

- 모든 SNMP 관리 스테이션이 **/etc/hosts** 파일에 나열되어 있는지 확인
- SNMP 지원 사용
- 관리 스테이션을 트랩 수신자로 지정하고 인증 구성.

이 절에는 언제든 SNMP 지원 사용 안함의 경우 지침도 포함되어 있습니다.

모든 SNMP 관리 스테이션이 `/etc/hosts` 파일에 나열되어 있는지 확인

1. Oracle HSM 서버에 *root*로 로그인합니다.

   예제에서 Oracle HSM 서버 호스트는 *samqfs1mds*입니다.

```
[samqfs1mds]root@solaris:~#
```

2. */etc/hosts* 파일을 텍스트 편집기에서 엽니다. SNMP 관리 스테이션으로 사용하려는 각 호스트에 대한 항목이 포함되어 있는지 확인합니다.

   예제에서는 *vi* 편집기를 사용합니다. 원하는 관리 스테이션 중 *management1*은 나열되지만, *management2*는 나열되지 않습니다.

```
[samqfs1mds]root@solaris:~# vi /etc/hosts
# Internet host table
::1 localhost
127.0.0.1 localhost loghost
10.0.0.10 server1
10.0.0.20 management1
```

3. */etc/hosts* 파일에 원하는 SNMP 관리 스테이션 호스트 중 일부 또는 모두에 대한 항목이 없는 경우 필요한 항목을 추가하고 파일을 저장합니다.

   예제에서는 *vi* 편집기를 사용하여 누락된 관리 스테이션 *management2*를 추가합니다.

```
[samqfs1mds]root@solaris:~# vi /etc/hosts
# Internet host table
::1 localhost
127.0.0.1 localhost loghost
10.0.0.10 server1
10.0.0.20 management1
10.0.0.30 management2
```

4. */etc/hosts* 파일에 원하는 모든 SNMP 관리 스테이션 호스트에 대한 항목이 있는 경우 편집기를 닫습니다.

```
[samqfs1mds]root@solaris:~# vi /etc/hosts
...
10.0.0.20 management1
10.0.0.30 management2
:wq
[samqfs1mds]root@solaris:~#
```

5. 이제 SNMP 지원 사용을 수행합니다.

SNMP 지원 사용

기본적으로 SNMP 알림 지원은 사용으로 설정되므로 중간에 SNMP 지원을 사용 안함으로 설정한 경우가 아니면 별도의 작업이 필요하지 않습니다. SNMP 지원을 사용으로 다시 설정해야 하는 경우 다음과 같이 하십시오.

1. Oracle HSM 서버에 *root*로 로그인합니다.

   예제에서 Oracle HSM 서버 호스트는 *samqfs1mds*입니다.

```
[samqfs1mds]root@solaris:~#
```

2. */etc/opt/SUNWsamfs/defaults.conf* 파일을 텍스트 편집기에서 엽니다. *alerts* = *off* 라인을 찾습니다.

   alerts = *off* 지시어는 SNMP 지원을 사용 안함으로 설정합니다. 예제에서는 *vi* 편집기에서 파일을 열고 행을 찾습니다.

```
[samqfs1mds]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.
...
alerts = off
```

3. SNMP 알림 지원을 사용으로 설정하려면 *alerts* 지시어의 값을 *on*으로 변경합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

```
[samqfs1mds]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.
...
alerts = on
:wq
[samqfs1mds]root@solaris:~#
```

4. Oracle HSM 서비스에 *defaults.conf* 파일을 다시 읽고 그에 따라 재구성하도록 지시합니다. *samd config* 명령을 사용합니다.

```
[samqfs1mds]root@solaris:~# 1mds]samd config
[samqfs1mds]root@solaris:~#
```

5. 이제 관리 스테이션을 트랩 수신자로 지정하고 인증 구성을 수행합니다.

관리 스테이션을 트랩 수신자로 지정하고 인증 구성

1. Oracle HSM 서버에 *root*로 로그인합니다.

   예제에서 Oracle HSM 서버 호스트는 *samqfs1mds*입니다.

```
[samqfs1mds]root@solaris:~#
```

2. */etc/opt/SUNWsamfs/scripts/sendtrap* 파일을 텍스트 편집기에서 열고 *TRAP_DESTINATION=*으로 시작하는 라인을 찾습니다.

   sendtrap 파일은 구성 가능한 셸 스크립트입니다. 예제에서는 *vi* 편집기에서 파일을 엽니다.

```
[samqfs1mds]root@solaris:~# vi /etc/opt/SUNWsamfs/scripts/sendtrap
# /etc/opt/SUNWsamfs/scripts/sendtrap#!/usr/bin/sh
# sendtrap:
#  This script gets invoked by the sysevent configuration file.
#  This is not expected to be run as a stand-alone program
...
# CONFIGURATION PARAMETERS:
TRAP_DESTINATION=`hostname`
```

3. *TRAP_DESTINATION=`* 라인에서 작은 따옴표 안의 텍스트를 하나 이상의 수신자의 공백으로 구분된 목록(각각 *hostname*:*port* 형식)으로 바꿉니다. 여기서 *hostname*은 */etc/hosts*에 나열되는 관리 스테이션의 호스트 이름이고, *port*는 호스트에서 트랩을 수신하는 포트입니다.

   기본적으로 *localhost*의 UDP 포트 *161*로 트랩을 보냅니다. 예제에서는 *management1* 및 *management2* 호스트를 기본값인 *localhost*에 추가합니다. *localhost* 및 *management1*에서는 기본 포트를 사용하고, *management2*에서는 사용자 정의 포트 *1161*을 사용합니다.

```
...
# CONFIGURATION PARAMETERS:
TRAP_DESTINATION=`localhost:161 management1:161 management1:1161`
```

4. 커뮤니티 문자열 *COMMUNITY="public"*을 설정하는 행까지 아래로 스크롤합니다.

   커뮤니티 문자열은 SNMP 버전 2c에서 에이전트 및 관리 스테이션을 인증하는 일반 텍스트로 된 공유 암호입니다. 기본값은 SNMP 표준인 *public*입니다.

```
...
# CONFIGURATION PARAMETERS:
TRAP_DESTINATION=`localhost:161 management1:161 management1:1161`
...
```

```
COMMUNITY="public"
```

5. *COMMUNITY=""* 지시어를 관리 스테이션에 사용되는 값으로 설정합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   파일에서 그 외에 아무것도 편집하지 마십시오. *COMMUNITY=""* 및 *TRAP_DESTINATION=``* 매개변수만 편집할 수 있습니다.

   기본 SNMP 커뮤니티 문자열 *public*은 안전하지 않습니다. 따라서 네트워크 관리자가 보다 안전한 문자열을 강제로 선택할 수 있습니다. SNMP 버전 2c에서는 최대 32자의 영숫자 문자를 사용할 수 있습니다. 예제에서는 커뮤니티 문자열을 *Iv0wQh2th74bVVt8of16t1m3s8it4wa9*로 설정합니다.

```
...
# CONFIGURATION PARAMETERS:
TRAP_DESTINATION=`localhost:161 management1:163 management1:1162`
...
COMMUNITY="Iv0wQh2th74bVVt8of16t1m3s8it4wa9"
:wq
[samqfs1mds]root@solaris:~#
```

6. 이제 Oracle HSM 응용 프로그램 로깅 사용을 수행합니다.

SNMP 지원 사용 안함

원격 알림 기능은 기본적으로 사용으로 설정됩니다. 원격 알림을 사용 안함으로 설정하려면 다음과 같이 하십시오.

1. Oracle HSM 서버에 *root*로 로그인합니다.

   예제에서 Oracle HSM 서버 호스트는 *samqfs1mds*입니다.

```
[samqfs1mds]root@solaris:~#
```

2. 텍스트 편집기에서 */etc/opt/SUNWsamfs/defaults.conf* 파일을 엽니다. *#alerts = on* 라인을 찾습니다.

   이 예에서는 *vi* 편집기를 사용합니다.

```
[samqfs1mds]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.
...
#alerts = on
[samqfs1mds]root@solaris:~#
```

3. SNMP 알림 지원을 사용 안함으로 설정하려면 해시 문자(*#*)를 삭제하여 행을 주석 해제하고 *alerts* 값을 *off*로 변경합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

```
[samqfs1mds]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.
...
alerts = off
:wq
[samqfs1mds]root@solaris:~#
```

4. Oracle HSM 서비스에 *defaults.conf* 파일을 다시 읽고 그에 따라 재구성하도록 지시합니다. *samd config* 명령을 사용합니다.

```
[samqfs1mds]root@solaris:~# samd config
[samqfs1mds]root@solaris:~#
```

5. 여기서 중지합니다. SNMP 지원이 사용 안함으로 설정됩니다.

Oracle HSM 로깅 사용

/var/adm/sam-log 파일에는 Oracle HSM 응용 프로그램과 해당 구성 요소 데몬 및 프로세스에 대한 상태 및 오류 정보가 기록됩니다. 로깅 프로세스를 설정하려면 다음과 같이 하십시오.

Oracle HSM 응용 프로그램 로깅 사용

1. Oracle HSM 서버에 *root*로 로그인합니다.

   예제에서 Oracle HSM 서버 호스트는 *samqfs1mds*입니다.

```
[samqfs1mds]root@solaris:~#
```

2. */etc/syslog.conf* 파일을 텍스트 편집기에서 엽니다.

   예제에서는 *vi* 편집기에서 파일을 엽니다.

```
[samqfs1mds]root@solaris:~# vi /etc/syslog.conf
# syslog configuration file ...
*.err;kern.notice;auth.notice                  /dev/sysmsg
*.err;kern.debug;daemon.notice;mail.crit       /var/adm/messages
*.alert;kern.err;daemon.err                    operator
*.alert                                        root
...
```

3. */etc/syslog.conf* 파일에서 문자열 *local7.debug*, 하나 이상의 탭 문자 및 경로 문자열 */var/adm/sam-log*로 구성된 행을 추가합니다. 그런 다음 파일을 저장하고 편집기를 닫습니다.

   예제에서는 설명도 추가합니다.

```
[samqfs1mds]root@solaris:~# vi /etc/syslog.conf
# syslog configuration file ...
*.err;kern.notice;auth.notice                  /dev/sysmsg
*.err;kern.debug;daemon.notice;mail.crit       /var/adm/messages
*.alert;kern.err;daemon.err                    operator
*.alert                                        root
...
# Oracle HSM logging
local7.debug    /var/adm/sam-log
:wq
[samqfs1mds]root@solaris:~#
```

4. */var/adm/sam-log* 로그 파일을 만듭니다. *touch /var/adm/sam-log* 명령을 사용합니다.

```
[samqfs1mds]root@solaris:~# touch /var/adm/sam-log
[samqfs1mds]root@solaris:~#
```

5. Solaris *syslogd* 데몬에 구성 파일을 다시 읽고 Oracle HSM 로깅을 시작하도록 지시합니다. *pkill -HUP syslogd* 명령을 사용합니다.

   HUP 신호를 받을 때마다 *syslogd* 로깅 서비스는 */etc/syslog.conf* 구성 파일을 다시 읽고 열려 있는 로그 파일을 모두 닫은 다음 *syslog.conf*에 나열된 로그 파일을 엽니다. 명령이 실행되면 Oracle HSM 로깅이 사용으로 지정됩니다.

```
[samqfs1mds]root@solaris:~# pkill -HUP syslogd
[samqfs1mds]root@solaris:~#
```

6. 장치 로깅 구성으로 이동합니다.

장치 로깅 구성

장치 로깅 기능은 개별 하드웨어 장치와 관련된 오류 정보를 제공합니다(소프트 매체 오류는 수집하지 않음). 각 장치에는 해당 장비 순서 번호로 이름이 지정되고 */var/opt/SUNWsamfs/devlog/* 디렉토리에 저장되는 자체 로그 파일이 있습니다.

장치 로그는 빠르게 커질 수 있습니다. 기본적으로 *err*, *retry*, *syserr*, *date* 등의 제한된 이벤트 데이터 세트만 기록됩니다. 나중에 문제가 발생할 경우 *samset* 명령을 사용하여 장

치별로 추가 이벤트를 로깅할 수 있습니다(자세한 내용은 *samset* 매뉴얼 페이지의 *devlog* 절 참조).

`defaults.conf` 파일에서 정치 로그 사용

기본 장치 로깅을 사용으로 설정하려면 다음과 같이 하십시오.

1. Oracle HSM 서버에 *root*로 로그인합니다.

   예제에서 Oracle HSM 서버 호스트는 *samqfs1mds*입니다.

```
[samqfs1mds]root@solaris:~#
```

2. 텍스트 편집기에서 */etc/opt/SUNWsamfs/defaults.conf*를 엽니다.

   예제에서는 *vi* 편집기에서 파일을 엽니다.

```
[samqfs1mds]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.
...
```

3. *defaults.conf* 파일에서 필요한 장치 로깅의 기본 레벨을 정의하는 행을 추가합니다. *devlog equipment-number loggable-events* 지시어를 입력합니다. 설명:
   - *equipment-number*는 */etc/opt/SUNWsamfs/mcf* 파일에 정의된 모든 장비를 나타내는 키워드 *all*이거나 *mcf*에 정의된 장비의 특정 부분을 식별하는 장치 순서입니다.
   - *loggable-events*는 기본값의 공백으로 구분된 목록(*err retry syserr date*)입니다.

     전체 이벤트 유형 목록은 *samset* 매뉴얼 페이지의 *devlog* 절을 참조하십시오. 그러나 로그 크기를 최소화하려면 기본 선택을 사용합니다. 진단을 위해 *samset* 명령은 필요에 따라 선택적으로 추가 이벤트를 사용으로 설정할 수 있습니다.

   예제에서는 기본 로깅 레벨을 사용하여 *all* 장치에 대한 장치 로깅을 사용으로 설정합니다.

```
[samfs-mds1]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.
...
devlog all err retry syserr date
```

4. *defaults.conf* 파일을 저장하고 편집기를 닫습니다.

```
[samfs-mds1]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.
...
devlog all err retry syserr date
:wq
[samfs-mds1]root@solaris:~#
```

5. Oracle HSM 서비스에 *defaults.conf* 파일을 다시 읽고 그에 따라 재구성하도록 지시합니다. *samd config* 명령을 사용합니다.

```
[samfs-mds1]root@solaris:~# samd config
[samfs-mds1]root@solaris:~#
```

6. 이제 Oracle HSM 로그 파일에 대한 자동 교체 설정을 수행합니다.

로그 교체 구성

로그 파일이 빠르게 커져 많은 공간을 차지하고 로그를 사용하기 어려워질 수 있습니다. 따라서 Oracle HSM 로그에 대한 자동 로그 교체를 구성해야 합니다. 소프트웨어에는 이러한 용도의 *log_rotate.sh* 스크립트가 포함되어 있습니다. Solaris *crontab* 파일에서 이 스크립트를 실행할 수 있습니다.

교체할 각 로그에 대해 *crontab* 항목을 두 개 만듭니다. 첫번째 항목은 원하는 시간에 *log_rotate.sh* 스크립트를 실행합니다. 대상 로그 파일이 지정된 최대 크기(기본값: *100000*바이트)에 도달하면 스크립트는 로그 파일의 이름을 바꾸고 가장 오래된 기존 복사본(항상 7개의 복사본이 유지됨)을 삭제합니다. 두번째 *crontab* 항목은 Solaris 로깅 데몬 *syslogd*에 새 로그 파일로 로깅을 다시 시작하도록 지시합니다.

Oracle HSM 로그 파일에 대한 자동 교체 설정

다음 로그 교체를 고려합니다.

- */etc/syslog.conf* 파일에서 지정된 위치에 있는 Oracle HSM 로그 파일 *sam-log*.
- */var/opt/SUNWsamfs/devlog/* 디렉토리에 있는 장치 로그 파일.
- */etc/opt/SUNWsamfs/stager.cmd* 파일에 지정된 스테이저 로그 파일.
- */etc/opt/SUNWsamfs/releaser.cmd* 파일에 지정된 릴리서 로그 파일.
- */etc/opt/SUNWsamfs/recycler.cmd* 파일에 지정된 리사이클러 로그 파일.

아카이버 로그 파일은 교체하면 안됩니다. 로그 정보는 분석 및 파일 시스템 복구에 중요합니다. 아카이버 로그의 적절한 처리를 위해 "파일 시스템 보호 구성"을 참조하십시오.

교체할 로그를 결정한 후 각 로그에 대해 다음과 같이 하십시오.

1. Oracle HSM 서버에 *root*로 로그인합니다.

   예제에서 Oracle HSM 서버 호스트는 *samqfs1mds*입니다.

```
[samqfs1mds]root@solaris:~#
```

2. 스크립트 파일 *log_rotate.sh*를 설치 제거된 위치 */opt/SUNWsamfs/examples/*에서 */etc/opt/SUNWsamfs/scripts/*로 복사합니다.

   아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시로 이스케이프됩니다.

```
[samfs-mds1]root@solaris:~# cp /opt/SUNWsamfs/examples/log_rotate.sh /  /etc/opt/SUNWsamfs/scripts/
```

3. 편집을 위해 *root* 사용자의 *crontab* 파일을 엽니다. *crontab -e* 명령을 사용합니다.

   crontab 명령은 *root* 사용자의 *crontab* 파일의 편집 가능 복사본을 *EDITOR* 환경 변수에 지정된 텍스트 편집기에서 엽니다(자세한 내용은 Solaris *crontab* 매뉴얼 페이지 참조). 이 예에서는 *vi* 편집기를 사용합니다.

```
[samfs-mds1]root@solaris:~# crontab -e
#ident "%Z%%M% %I% %E% SMI"
# Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
# The root crontab should be used to perform accounting data collection.
10 3 * * * /usr/sbin/logadm
...
30 0,9,12,18,21 * * * /usr/lib/update-manager/update-refresh.sh
```

4. 새 행에서 *minutes hour * * day-of-the-week*를 입력하여 로그 파일을 교체할 시간을 지정합니다. 설명:
   - *minutes*는 작업이 시작되는 분을 지정하는 [*0-59*] 범위의 정수입니다.
   - *hour*는 작업이 시작되는 시간을 지정하는 [*0-23*] 범위의 정수입니다.
   - *(별표)는 사용되지 않는 값을 지정합니다.

     매일 실행되는 작업의 경우 일 [*1-31*] 및 월 [*1-12*] 값이 사용되지 않습니다.
   - *day-of-the-week*는 일요일(*0*)부터 시작하는 범위 [*0-6*] 이내의 정수입니다.
   - 시간 지정에서 필드는 공백으로 구분됩니다.

   예제에서는 매주 일요일 3:10 AM에 로그 교체를 시작하도록 예약합니다.

```
[samfs-mds1]root@solaris:~# crontab -e
#ident "%Z%%M% %I% %E% SMI"
# Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
# The root crontab should be used to perform accounting data collection.
```

```
10 3 * * * /usr/sbin/logadm
...
30 0,9,12,18,21 * * * /usr/lib/update-manager/update-refresh.sh
10 3 * * 0
```

5. 계속해서 동일한 행에 Oracle HSM 로그를 교체하는 셸 스크립트 파일의 경로와 이름을 입력하고(*/etc/opt/SUNWsamfs/scripts/log_rotate.sh*) 한 칸 띄웁니다.

```
[samfs-mds1]root@solaris:~# crontab -e
#ident "%Z%%M% %I% %E% SMI"
# Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
# The root crontab should be used to perform accounting data collection.
10 3 * * * /usr/sbin/logadm
...
30 0,9,12,18,21 * * * /usr/lib/update-manager/update-refresh.sh
10 3 * * 0 /etc/opt/SUNWsamfs/scripts/log_rotate.sh
```

6. 계속해서 동일한 행에 교체할 로그의 이름과 교체할 최소 파일 크기 파일을 입력합니다. *samfslog* [*minimum-size*] 텍스트를 입력합니다. 여기서 *samfslog*는 Oracle HSM 로그 파일의 경로이고 [*minimum-size*]는 스크립트가 교체되는 최소 파일 크기(바이트)를 지정하는 선택적 정수입니다(기본값: *100000*).

   예제에서는 */var/adm/sam-log*를 교체해야 합니다. 기본 최소 크기를 수락합니다.

```
[samfs-mds1]root@solaris:~# crontab -e
#ident "%Z%%M% %I% %E% SMI"
# Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
# The root crontab should be used to perform accounting data collection.
10 3 * * * /usr/sbin/logadm
...
30 0,9,12,18,21 * * * /usr/lib/update-manager/update-refresh.sh
10 3 * * 0 /etc/opt/SUNWsamfs/scripts/log_rotate.sh /var/adm/sam-log
```

7. 새 행을 시작합니다. *log_rotate.sh* 스크립트보다 10분 후에 시작되는 *crontab* 항목을 만듭니다. 이 항목은 Solaris *syslogd* 데몬에 이전 로그 파일을 닫고 새 파일에 로깅을 재개하도록 지시합니다. *minutes hour * * day-of-the-week /bin/kill -HUP `/bin/cat /etc/syslog.pid`* 라인을 입력합니다. 여기서 *minutes hour * * day-of-the-week*는 이전 단계에서 지정한 시간보다 10분 늦은 시간을 지정합니다.

   예제의 항목은 일요일 3:20 AM에 Oracle HSM 로깅을 다시 시작합니다.

```
[samfs-mds1]root@solaris:~# crontab -e
#ident "%Z%%M% %I% %E% SMI"
```

```
# Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
# The root crontab should be used to perform accounting data collection.
10 3 * * * /usr/sbin/logadm
...
30 0,9,12,18,21 * * * /usr/lib/update-manager/update-refresh.sh
10 3 * * 0 /etc/opt/SUNWsamfs/scripts/log_rotate.sh /var/adm/sam-log
20 3 * * 0  /bin/kill -HUP `/bin/cat /etc/syslog.pid`
```

8. 파일을 저장하고 편집기를 닫습니다.

```
[samfs-mds1]root@solaris:~# crontab -e
#ident "%Z%%M% %I% %E% SMI"
# Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
# The root crontab should be used to perform accounting data collection.
10 3 * * * /usr/sbin/logadm
...
30 0,9,12,18,21 * * * /usr/lib/update-manager/update-refresh.sh
10 3 * * 0 /etc/opt/SUNWsamfs/scripts/log_rotate.sh /var/adm/sam-log
20 3 * * 0  /bin/kill -HUP `/bin/cat /etc/syslog.pid`
:wq
[samfs-mds1]root@solaris:~#
```

9. 이 절차를 반복하여 모든 필요한 로그에 대한 로그 교체를 구성합니다.
10. 이제 필요한 경우 전자 메일 경보 사용을 수행합니다.
11. 그렇지 않은 경우 13장. *Oracle HSM* 구성 백업으로 이동합니다.

전자 메일 경보 사용

전자 메일 경보는 Oracle HSM Manager 그래픽 사용자 인터페이스에서 가장 원활하게 설정됩니다. 자세한 내용은 온라인 도움말을 참조하십시오.

명령줄 인터페이스에서 전자 메일 경보를 구성해야 하는 경우 *defaults.conf*, *archiver.sh*, *dev_down.sh*, *load_notify.sh*, *recycler.sh*, *archiver.cmd*, *recycler.cmd* 및 *notify.cmd* 매뉴얼 페이지를 참조하십시오.

이제 Oracle HSM 시스템이 구성되었습니다. 하지만 시스템을 사용하기 전에 작업을 보호하십시오. 지침은 13장. *Oracle HSM* 구성 백업을 참조하십시오.

12

12장. 특수한 요구 사항에 맞게 I/O 특성 조정

이전 장에서 설명한 기본 파일 시스템 구성 단계를 수행하면 대개의 경우 균형 잡힌 최적의 성능을 얻을 수 있습니다. 따라서 응용 프로그램 작동 방식을 잘 모를 경우 이 절의 설정을 기본값으로 두는 것이 좋습니다. 그러나 응용 프로그램에서 특이하게 일정하거나 큰 I/O 요청을 만드는 경우 파일 시스템이 물리적 I/O를 처리하는 방식을 조정하거나 변경하면 전반적인 성능이 향상될 수 있습니다.

읽기 및 쓰기가 전부 또는 대부분 디스크 섹터의 512바이트 경계에서 정확히 시작하고 종료하는 경우 물리적 I/O가 가장 효율적입니다. 디스크 I/O는 섹터 크기 청크로만 발생할 수 있습니다. 따라서 I/O 요청에 섹터 경계에 걸쳐 있는 경우 시스템에서는 추가 작업을 수행하여 응용 프로그램 데이터를 동일한 섹터의 관련 없는 데이터와 분리합니다. 이 과정에서 관련 없는 데이터가 손상되지 않도록 해야 합니다. 최악의 경우 여러 섹터에 쓸 때 파일 시스템은 섹터를 읽고 메모리의 섹터 데이터를 수정한 다음 섹터를 다시 디스크에 써야 합니다. 이러한 읽기-수정-쓰기 작업은 추가되는 기계적 작업만으로도 성능 면에서 비용이 많이 듭니다.

대부분의 응용 프로그램은 섹터 경계에 제대로 정렬되지 않은 다양한 크기의 데이터를 읽고 써야 합니다. 이 때문에 대부분의 파일 시스템처럼 Oracle HSM은 기본적으로 페이징된 *I/O*를 사용합니다. 파일 시스템에서는 Solaris 페이징된 메모리의 데이터 캐시를 읽거나 이 캐시에 써서 응용 프로그램의 I/O 요청을 즉시 처리합니다. 파일 시스템에서는 더욱 효율적으로 크기가 지정되고 제대로 정렬된 물리적 읽기 및 쓰기를 통해 비동기적으로 캐시를 업데이트합니다. 디스크에서 데이터를 읽을 때마다 동일한 작업에서 예정된 읽기를 예측하고 해당하는 데이터를 캐시로 로드하여 물리적 I/O의 대부분을 수행할 수 있습니다. 따라서 대부분의 I/O 요청을 가상 메모리 페이지에 캐시된 데이터를 사용하여 충족할 수 있으므로, 추가되는 물리적 디스크 작업이 없습니다. 페이징된 I/O로 인해 메모리가 사용되고 시스템 CPU에 로드가 가해지지만 대부분의 경우 이러한 비용은 물리적 I/O 효율 향상으로 상쇄되는 것보다 많습니다.

그러나 이 I/O의 이점으로도 페이징된 I/O 관련 추가 오버헤드를 상쇄할 수 없는 경우도 있습니다. 잘 정렬된 I/O를 항상 수행하는 응용 프로그램과 이렇게 하도록 조정할 수 있는 응용 프로그램에는 페이지 캐싱이 아무 도움도 되지 않습니다. 엄청나게 큰 I/O를 수행하는 응용 프로그램도 첫번째와 마지막 섹터만 제대로 정렬되지 않고 큰 I/O는 어떤 경우든 너무 커서 캐시에 보관할 수 없기 때문에 페이지 캐싱으로 얻을 수 있는 이점이 거의 없습니다. 마지막으로 원격 측정 데이터, 감시 비디오 또는 다른 유형의 실시간 정보를 스트리밍하는 응용 프로그램은 쓰기를 비휘발성 스토리지로 즉시 커밋하지 않는 경우 복구할 수 없는 데이터가 손실될 위험이 있습니다. 이러한 경우 직접 *I/O*를 사용하는 것이 더 좋을 수 있습니다. 직접 I/O를 지정하는 경우 파일 시스템에서는 페이지 캐시를 건너뛰고 응용 프로그램 메모리와 디스크 장치 간에 직접 데이터를 전송합니다.

Oracle HSM에서는 I/O 캐싱 동작을 선택하고 조정할 때 상당한 유연성을 제공합니다. 응용 프로그램의 I/O 특성을 파악하고 "예상 파일 시스템 I/O에 대한 Solaris 시스템 및 드라이버 매개변수 조정"에 설명된 작업을 수행했으면 다음과 같이 접근 방법을 선택합니다.

- 응용 프로그램에서 계속 작고, 크기가 다양하며 제대로 정렬되지 않은 I/O 요청을 만드는 경우 Oracle HSM 기본 설정을 적용합니다. 이 절에서 아무 것도 변경하지 마십시오.
- 응용 프로그램에서 크기가 다양하지만 평균보다 크고 잘못 정렬된 I/O 요청을 만드는 경우 대량 데이터 전송에 맞게 페이징된 I/O 최적화를 수행합니다.
- 응용 프로그램에서 제대로 정렬되거나 아주 큰 I/O 요청 및 작고 잘못 정렬된 요청을 함께 만드는 경우 페이징된 I/O 및 직접 I/O 간 전환 사용을 수행합니다.
- 응용 프로그램에서 제대로 정렬되거나 매우 큰 I/O 요청을 계속 만드는 경우 직접 I/O만 사용하도록 파일 시스템 구성을 수행합니다.
- 공유 파일 시스템 클라이언트에서 실행 중인 응용 프로그램이 계속 많은 수의 파일을 여는 경우 Directory Name Lookup Cache 크기 늘리기를 수행합니다.

대량 데이터 전송에 맞게 페이징된 I/O 최적화

응용 프로그램 및 하드웨어 특성에 더 잘 맞게 페이징된 I/O를 조정할 수 있습니다. 읽기 캐시와 쓰기 캐시가 충분해서 응용 프로그램이 전송하는 평균 데이터 양이든 물리적 스토리지에서 전송할 수 있는 최대 데이터 양이든 전송할 수 있어야 합니다. 어느 쪽 양이든 페이지 캐싱을 적절하게 조정하지 못하면 캐시가 제대로 활용되지 않아 응용 프로그램 I/O 요청에 물리적 I/O가 더 필요하게 되고 전반적인 시스템 성능이 떨어집니다.

예를 들어 단일 디스크 볼륨에서 구현된 *md* 데이터 장치와 3+1 RAID 5 볼륨 그룹으로 구현된 *md* 장치 간의 차이점을 고려해 보십시오. 64킬로바이트 DAU(디스크 할당 단위) 하나를 캐시에서 RAID 그룹에 써서 응용 프로그램의 쓰기 요청을 각각 처리하고 다중 디스크 장치에서 발생할 수 있는 추가 대역폭을 무시할 경우 RAID 장치는 작지만 효율성이 낮은 21(및 22)킬로바이트 조각 3개로 I/O를 분할한 다음 RAID 그룹의 데이터 디스크 3개에 데이터를 써야 합니다. 따라서 이 구성을 사용하여 응용 프로그램의 64킬로바이트 I/O 요청을 수행하려면 페이지 캐시를 사용하여 요청을 하나의 3-DAU, 192킬로바이트 I/O로 조합할 경우보다 필요한 작업이 상당히 더 많습니다. 응용 프로그램이 장치 대역폭의 몇 배 즉, 192, 384 또는 576킬로바이트의 I/O 요청을 만들 수 있거나 이렇게 하도록 조정할 수 있는 경우 더 많은 데이터를 캐시하고 각 물리적 I/O에 더 많이 전송하여 오버헤드를 줄이고 성능을 향상할 수 있습니다.

따라서 응용 프로그램의 I/O 요구 사항을 식별하고 하드웨어의 I/O 등록 정보를 파악합니다. 그런 다음 다음과 같이 진행하십시오.

1. 파일 시스템 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. 운영체제의 */etc/vfstab* 파일을 백업합니다.

```
root@solaris:~# cp /etc/vfstab /etc/vfstab.backup
root@solaris:~#
```

3. 텍스트 편집기에서 */etc/vfstab* 파일을 열고 조정이 필요한 파일 시스템의 행을 찾습니다.

   이 예제에서 파일 시스템의 이름은 *qfsma*입니다.

```
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       -
...
qfsma      -        /qfsma    samfs   -     yes      ...
```

4. 파일 시스템의 *Mount Options* 필드에서 *writebehind=n* 마운트 옵션을 추가합니다. 여기서 *n*은 *8*킬로바이트의 배수입니다. 콤마를 공백 없이 사용하여 마운트 옵션을 구분합니다. 파일을 저장하고 편집기를 닫습니다.

   writebehind 옵션에 따라 캐시가 디스크로 플러시되기 전에 지정된 파일이 페이지 캐시에 대기될 수 있는 양이 결정됩니다. 대기열이 크면 여러 개의 작은 응용 프로그램 쓰기를 더 적은 수의 크고 더 효율적인 물리적 I/O로 통합하므로 매개변수를 더 높은 값으로 설정하면 성능이 향상됩니다. 매개변수를 더 작은 값으로 설정하면 변경사항이 비휘발성 스토리지에 더 빨리 기록되므로 데이터 보호가 향상됩니다.

   기본값은 *512*킬로바이트(8개의 64킬로바이트 DAU)이며, 일반적으로 큰 블록의 순차적 I/O에 적합합니다. 하지만 이 예제에서는 패밀리 세트에 스트라이프 파일 할당을 사용하는 *md* 디스크 장치가 2개 포함되어 있습니다. 두 *md* 장치에 128킬로바이트를 쓰기 위해 스트라이프 너비는 하나의 64킬로바이트 DAU입니다. *md* 장치는 3+1 RAID 5 그룹입니다. 따라서 세 개의 데이터 스핀들 각각에 최소 128킬로바이트를 쓰려고 하므로 총 쓰기는 *768*킬로바이트(각각 8킬로바이트 그룹 96개) 이상입니다.

```
#File
#Device    Device   Mount     System  fsck  Mount    Mount
#to Mount  to fsck  Point     Type    Pass  at Boot  Options
#--------  -------  --------  ------  ----  -------  --------------------
/devices   -        /devices  devfs   -     no       -
...
qfsma      -        /qfsma    samfs   -     yes      ...,writebehind=768
:wq
root@solaris:~#
```

5. 파일 시스템의 I/O 성능을 테스트하고 필요한 경우 *writebehind* 설정을 조정합니다.
6. 텍스트 편집기에서 */etc/vfstab* 파일을 다시 엽니다. 파일 시스템의 *Mount Options* 필드에서 *readahead=n* 마운트 옵션을 추가합니다. 여기서 *n*은 8킬로바이트의 배수입

니다. 콤마를 공백 없이 사용하여 마운트 옵션을 구분합니다. 파일을 저장하고 편집기를 닫습니다.

readahead 옵션에 따라 단일 물리적 읽기 동안 캐시로 읽어들이는 데이터 양이 결정됩니다. 응용 프로그램에서 순차적으로 읽는 것 같으면 파일 시스템에서는 각 물리적 읽기 동안 파일 데이터의 예정 블록을 캐시합니다. 그런 다음 일련의 응용 프로그램 읽기 요청을 캐시 메모리에서 처리하여, 여러 응용 프로그램 읽기 요청을 하나의 물리적 I/O 요청으로 통합합니다.

기본값은 *1024*킬로바이트(16개의 64킬로바이트 DAU)이며, 일반적으로 큰 블록의 순차적 I/O에 적합합니다. 데이터베이스 또는 소형 응용 프로그램에서 자체적으로 미리 읽기를 수행하는 경우에는 Oracle HSM *readahead*를 *0*으로 설정하여 충돌을 방지합니다. 그렇지 않은 경우 일반적으로 *readahead*는 단일 물리적 I/O에서 전송할 수 있는 최대 데이터를 캐시하도록 설정해야 합니다. *readahead* 설정이 응용 프로그램에서 일반적으로 요청하고 장치가 제공할 수 있는 데이터 양보다 작은 경우 응용 프로그램 I/O 요청을 수행하려면 물리적 I/O가 필요 이상으로 필요합니다. 그러나 *readahead*를 너무 높게 설정하는 경우 메모리를 너무 많이 사용하여 전체 시스템 성능이 떨어질 수 있습니다. 예제에서는 *readahead*를 *736*킬로바이트(36개 64킬로바이트 DAU)로 설정합니다.

```
#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       -
...
qfsma      -        /qfsma    samfs   -     yes      ...,readahead=736
:wq
root@solaris:~#
```

7. 파일 시스템의 I/O 성능을 테스트하고 필요한 경우 *readahead* 설정을 조정합니다.

   readahead 매개변수 크기를 늘리면 큰 파일 전송의 성능이 어느 정도만 증가합니다. 따라서 *readahead* 크기를 재설정한 후 시스템의 성능을 테스트합니다. 그런 다음 전송 속도가 더 이상 향상되지 않을 때까지 *readahead* 크기를 위로 조정합니다.

페이징된 I/O 및 직접 I/O 간 전환 사용

응용 프로그램의 I/O 동작에 더 잘 맞는 경우 페이징된 I/O와 직접 I/O 간을 전환할 수 있도록 Oracle HSM 파일 시스템을 구성할 수 있습니다. 직접 I/O가 유용할 수 있는 읽기 및 쓰기의 섹터 맞춤 및 최소 크기 특성을 지정한 다음 전환을 트리거할 적격 읽기 및 쓰기 수를 설정합니다. 다음과 같이 하십시오.

1. 파일 시스템 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. 운영체제의 */etc/vfstab* 파일을 백업합니다.

```
root@solaris:~# cp /etc/vfstab /etc/vfstab.backup
root@solaris:~#
```

3. 텍스트 편집기에서 */etc/vfstab* 파일을 열고 구성할 파일 시스템의 행을 찾습니다.

   이 예제에서 파일 시스템의 이름은 *qfsma*입니다.

```
root@solaris:~# vi /etc/vfstab
#File     Device                      Mount
#Device   to     Mount    System fsck at    Mount
#to Mount fsck   Point    Type   Pass Boot  Options
#-------- ------ -------- ------ ---- ----- ----------------------------
/devices  -      /devices devfs  -    no    -
/proc     -      /proc    proc   -    no    -
...
qfsma     -      /qfsma   samfs  -    yes   stripe=1
```

4. 512바이트 섹터 경계에 제대로 정렬되는 읽기 요청에 대해 직접 I/O를 시작하기 위한 임계값 크기를 설정하려면 파일 시스템의 *Mount Options* 필드에 *dio_rd_form_min=n* 마운트 옵션을 추가합니다. 여기서 *n*은 킬로바이트 수입니다. 콤마를 공백 없이 사용하여 마운트 옵션을 구분합니다.

   기본적으로 *dio_rd_form_min=256*킬로바이트입니다. 예제에서는 응용 프로그램이 512킬로바이트 이상의 읽기를 요청하지 않는 한 제대로 정렬된 읽기를 지속적으로 만들지 않습니다. 따라서 제대로 정렬된 직접 읽기에 대한 임계값 크기를 *512*로 변경합니다.

```
#File     Device                      Mount
#Device   to     Mount    System fsck at    Mount
#to Mount fsck   Point    Type   Pass Boot  Options
#-------- ------ -------- ------ ---- ----- ----------------------------
/devices  -      /devices devfs  -    no    -
/proc     -      /proc    proc   -    no    -
...
qfsma     -      /qfsma   samfs  -    yes   stripe=1,dio_rd_form_min=512
```

5. 512바이트 섹터 경계에 제대로 정렬되는 쓰기 요청에 대해 직접 I/O를 시작하기 위한 임계값 크기를 설정하려면 파일 시스템의 *Mount Options* 필드에 *dio_wr_form_min=n* 마운트 옵션을 추가합니다. 여기서 *n*은 킬로바이트 수입니다. 콤마를 공백 없이 사용하여 마운트 옵션을 구분합니다.

   기본적으로 *dio_wr_form_min=256*킬로바이트입니다. 예제에서는 응용 프로그램이 1메가바이트 이상의 쓰기를 요청하지 않는 한 제대로 정렬된 쓰기를 지속적으로 만들지

않습니다. 따라서 제대로 정렬된 직접 쓰기에 대한 임계값 크기를 *1024*킬로바이트로 변경합니다.

```
#File     Device                       Moun
#Device   to     Mount    System fsck at    Mount
#to Mount fsck   Point    Type   Pass Boot  Options
#-------- ------ -------- ------ ---- ----- ----------------------------
/devices  -      /devices devfs  -    no    -
/proc     -      /proc    proc   -    no    -
...
qfsma     -      /qfsma   samfs  -    yes   ...,dio_wr_form_min=1024
```

6. 512바이트 섹터 경계에 제대로 정렬되지 않는 읽기 요청에 대해 직접 I/O를 시작하기 위한 임계값 크기를 설정하려면 파일 시스템의 *Mount Options* 필드에 *dio_rd_ill_min=n* 마운트 옵션을 추가합니다. 여기서 *n*은 킬로바이트 수입니다. 콤마를 공백 없이 사용하여 마운트 옵션을 구분합니다.

   기본적으로 *dio_rd_ill_min=0*킬로바이트이므로 잘못 정렬된 읽기에 직접 I/O가 사용되지 않습니다. 예제에서는 응용 프로그램이 일반적으로 작은 데이터 청크에 대해 잘못 정렬된 읽기 요청을 만듭니다. 이 데이터는 대부분 순차적으로 다시 읽습니다. 따라서 이 읽기에는 페이지 캐싱을 사용하는 것이 더 좋습니다. 직접 I/O로 전환하면 불필요하게 물리적 I/O가 추가되고 성능이 떨어집니다. 따라서 기본값을 적용하고 *vfstab* 파일을 변경하지 않습니다.

```
#File     Device                       Mount
#Device   to     Mount    System fsck at    Mount
#to Mount fsck   Point    Type   Pass Boot  Options
#-------- ------ -------- ------ ---- ----- ----------------------------
/devices  -      /devices devfs  -    no    -
/proc     -      /proc    proc   -    no    -
...
qfsma     -      /qfsma   samfs  -    yes   ...,dio_wr_form_min=1024
```

7. 512바이트 섹터 경계에 제대로 정렬되지 않는 쓰기 요청에 대해 직접 I/O를 시작하기 위한 임계값 크기를 설정하려면 파일 시스템의 *Mount Options* 필드에 *dio_wr_ill_min=n* 마운트 옵션을 추가합니다. 여기서 *n*은 킬로바이트 수입니다. 콤마를 공백 없이 사용하여 마운트 옵션을 구분합니다.

   기본적으로 *dio_wr_ill_min=0*킬로바이트이므로 잘못 정렬된 쓰기에 직접 I/O가 사용되지 않습니다. 잘못 정렬된 쓰기는 시스템에서 섹터를 읽고, 수정하고, 써야 하므로 성능 측면에서 특히 비용이 많이 들 수 있습니다. 그러나 예제에서는 응용 프로그램이 경우에 따라 섹터 경계에 맞지 않는 하나의 큰 쓰기 요청을 한다는 점을 알고 있습니다. 읽기-쓰기-수정 작업이 큰 순차 섹터 블록의 처음과 끝에서만 수행되므로 직접 I/O가 페이징된 I/O보다 더 이롭습니다. 따라서 *dio_wr_ill_min=2048*킬로바이트를 설정합니다.

예제에서는 맞지 않는 데이터 쓰기 작업 동안 직접 I/O를 사용하기 위한 기본 임계값을 *2048*킬로바이트로 변경합니다.

```
#File     Device                      Mount
#Device   to     Mount    System fsck at    Mount
#to Mount fsck   Point    Type   Pass Boot  Options
#-------- ------ -------- ------ ---- ----- ----------------------------
/devices  -      /devices devfs  -    no    -
/proc     -      /proc    proc   -    no    -
...
qfsma     -      /qfsma   samfs  -    yes   ...,dio_wr_ill_min=2048
```

8. 읽기에 직접 I/O를 사용하도록 설정하려면 *Mount Options* 필드에 *dio_rd_consec=n* 마운트 옵션을 추가합니다. 여기서 *n*은 직접 I/O로의 전환을 트리거하기 위해 위에 저장된 크기 및 맞춤 요구 사항을 충족해야 하는 연속 I/O 전송의 수입니다. 직접 I/O가 유용한 응용 프로그램 작업에 대해 선택하는 값을 선택합니다. 콤마를 공백 없이 사용하여 마운트 옵션을 구분합니다.

   기본적으로 *dio_rd_consec=0*이므로 I/O 전환이 사용 안함으로 설정됩니다. 예제에서는 응용 프로그램이 *dio_rd_form_min*에 지정된 최소 크기 512킬로바이트 이상이고 잘 정렬된 연속 읽기 3개를 요청하는 경우 직접 I/O가 적합해질 때까지 이러한 요청이 계속됩니다. *dio_rd_form_min*에 지정된 최소 크기가 기본값인 *0*이므로 직접 I/O를 사용으로 설정해도 맞지 않는 읽기 요청에는 영향을 주지 않습니다. 따라서 *dio_rd_consec=3*을 설정합니다.

```
#File     Device                      Mount
#Device   to     Mount    System fsck at    Mount
#to Mount fsck   Point    Type   Pass Boot  Options
#-------- ------ -------- ------ ---- ----- ----------------------------
/devices  -      /devices devfs  -    no    -
/proc     -      /proc    proc   -    no    -
...
qfsma     -      /qfsma   samfs  -    yes   ...,dio_rd_consec=3
```

9. 쓰기에 직접 I/O를 사용하도록 설정하려면 *Mount Options* 필드에 *dio_wr_consec=n* 마운트 옵션을 추가합니다. 여기서 *n*은 직접 I/O로의 전환을 트리거하기 위해 위에 저장된 크기 및 맞춤 요구 사항을 충족해야 하는 연속 I/O 전송의 수입니다. 직접 I/O가 유용한 응용 프로그램 작업에 대해 선택하는 값을 선택합니다. 콤마를 공백 없이 사용하여 마운트 옵션을 구분합니다.

   기본적으로 *dio_wr_consec=0*이므로 I/O 전환이 사용 안함으로 설정됩니다. 예제에서는 응용 프로그램이 *dio_wr_form_min*에 지정된 최소 크기 1024킬로바이트 이상이고 제대로 정렬된 연속 쓰기 2개를 요청하는 경우 직접 I/O가 적합해질 때까지 이러한 요청이 계속됩니다. 또한 잘못 정렬된 연속 쓰기 2개가 *dio_wr_form_min*에 지정된 2048

킬로바이트보다 크면 잘못 정렬된 상태가 비교적 문제되지 않습니다. 따라서 *dio_wr_consec=2*를 설정합니다.

```
#File     Device                      Mount
#Device   to     Mount    System fsck at    Mount
#to Mount fsck   Point    Type   Pass Boot  Options
#-------- ------ -------- ------ ---- ----- ----------------------------
/devices  -      /devices devfs  -    no    -
/proc     -      /proc    proc   -    no    -
...
qfsma     -      /qfsma   samfs  -    yes   ...,dio_wr_consec=2
```

10. *vfstab* 파일을 저장하고 편집기를 닫습니다.

```
#File     Device                      Mount
#Device   to     Mount    System fsck at    Mount
#to Mount fsck   Point    Type   Pass Boot  Options
#-------- ------ -------- -----  ---- ----- ----------------------------
/devices  -      /devices devfs  -    no    -
/proc     -      /proc    proc   -    no    -
...
qfsma     -      /qfsma   samfs  -    yes   ...,dio_wr_consec=2
:wq
root@solaris:~#
```

11. 수정된 파일 시스템을 마운트합니다.

```
root@solaris:~# mount /qfsms
root@solaris:~#
```

직접 I/O만 사용하도록 파일 시스템 구성

응용 프로그램의 I/O 특성으로 인해 직접 I/O만 사용하는 것이 좋은 경우 *forcedirectio* 마운트 옵션을 사용하여 전체 파일 시스템을 마운트할 수 있습니다(개별 파일 또는 디렉토리에 대해 직접 I/O를 지정하는 방법에 대한 자세한 내용은 Oracle HSM *setfa* 매뉴얼 페이지 참조).

직접 I/O만 사용하도록 파일 시스템을 마운트하려면 다음과 같이 하십시오.

1. 파일 시스템 호스트에 *root*로 로그인합니다.

```
root@solaris:~#
```

2. 운영체제의 */etc/vfstab* 파일을 백업합니다.

```
root@solaris:~# cp /etc/vfstab /etc/vfstab.backup
```

```
root@solaris:~#
```

3. 텍스트 편집기에서 */etc/vfstab* 파일을 열고 직접 I/O를 사용할 파일 시스템의 행을 찾습니다.

   이 예제에서 파일 시스템의 이름은 *qfsma*입니다.

```
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       -
...
qfsma      -        /qfsma    samfs   -     yes      stripe=1
```

4. 파일 시스템의 *Mount Options* 필드에 *forcedirectio* 마운트 옵션을 추가합니다. 콤마를 공백 없이 사용하여 마운트 옵션을 구분합니다. 파일을 저장하고 편집기를 닫습니다.

```
#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       -
...
qfsma      -        /qfsma    samfs   -     yes      stripe=1,forcedirectio
:wq
root@solaris:~#
```

5. 수정된 파일 시스템을 마운트합니다.

```
root@solaris:~# mount /qfsms
root@solaris:~#
```

Directory Name Lookup Cache 크기 늘리기

공유 파일 시스템의 클라이언트가 여러 파일을 동시에 여는 경우 메타데이터 서버에서 Oracle Solaris DNLC(Directory Name Lookup Cache)의 기본 크기가 적합하지 않을 수 있습니다. 메타데이터 서버가 모든 클라이언트 대신 파일 이름을 조회하므로 이런 상태에서는 파일 시스템 성능이 저하될 수 있습니다.

이러한 종류의 작업 로드가 예상되는 경우 Directory Name Lookup Cache 크기 매개변수 `ncsize` 값을 기본 크기의 2~3배로 변경합니다. 지침은 *Oracle Solaris Information Library*에서 *Oracle Solaris Tunable Parameters Reference Manual*을 참조하십시오(머리말의 "사용 가능한 설명서" 절 참조).

13

13장. Oracle HSM 구성 백업

Oracle Hierarchical Storage Manager and StorageTek QFS Software 구성을 완료하고 나면 구성 파일 및 관련 정보를 백업하여 투자를 보호합니다. 다음 작업을 수행합니다.

- Oracle HSM 구성을 위한 백업 위치 만들기
- **`samexplorer`** 실행 및 안전하게 보고서 저장
- 수동으로 Oracle HSM 구성 백업.

Oracle HSM 구성을 위한 백업 위치 만들기

다음과 같이 하십시오.

1. 파일 시스템 호스트에 *`root`*로 로그인합니다.

```
root@solaris:~#
```

2. Oracle HSM 구성의 백업 복사본을 위한 스토리지 위치를 선택합니다. 파일 시스템 호스트에 마운트할 수 있는 독립된 파일 시스템을 선택합니다.
3. 선택한 파일 시스템이 물리적 장치를 아카이빙 파일 시스템과 공유하지 않는지 확인합니다.

   보호해야 하는 파일 시스템에 복구 지점 파일을 저장하지 마십시오. 또한 아카이빙 파일 시스템도 호스팅하는 물리적 장치에 상주하는 분할 영역 또는 LUN과 같은 논리적 장치에 복구 지점 파일을 저장하지 마십시오.
4. 선택한 파일 시스템에서 구성 정보를 보유하기 위한 디렉토리를 만듭니다. *`mkdir mount-point/path`* 명령을 사용합니다. 여기서 *`mount-point`*는 선택한 독립된 파일 시스템의 마운트 지점이고 *`path`*는 선택한 디렉토리의 경로 및 이름입니다.

   예제에서는 독립된 파일 시스템 */zfs1*에서 */zfs1/sam_config* 디렉토리를 만들었습니다.

```
root@solaris:~# mkdir /zfs1/sam_config
```

5. 이제 *`samexplorer`* 실행 및 안전하게 보고서 저장을 수행합니다.

samexplorer 실행 및 안전하게 보고서 저장

*samexplorer*는 Oracle HSM 소프트웨어 및 파일 시스템에 대한 포괄적인 구성 및 상태 정보를 캡처 및 보고하는 진단 도구입니다. Oracle은 문제 해결 시에 서비스 직원이 출력을 사용하도록 지원합니다. 따라서 Oracle HSM 소프트웨어 및 파일 시스템을 구성 또는 재구성할 때마다 기준 *samexplorer* 보고서를 만드는 것이 좋습니다.

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

   예제에서 호스트 이름은 *samqfs1mds*입니다.

```
[samqfs1mds]root@solaris:~#
```

2. 백업 구성 정보를 보유한 디렉토리에서 *samexplorer* 보고서를 위한 하위 디렉토리를 만듭니다. *mkdir mount-point/path* 명령을 사용합니다. 여기서 *mount-point*는 선택한 독립된 파일 시스템의 마운트 지점이고 *path*는 선택한 디렉토리의 경로 및 이름입니다.

   예제에서는 */zfs1/sam_config/explorer* 디렉토리를 만듭니다.

```
[samqfs1mds]root@solaris:~# mkdir /zfs1/sam_config/explorer
[samqfs1mds]root@solaris:~#
```

3. 선택한 디렉토리에 *samexplorer* 보고서를 만듭니다. *samexplorer path/hostname.YYYYMMDD.hhmmz.tar.gz* 명령을 사용합니다. 여기서 *path*는 선택한 디렉토리의 경로이고 *hostname*은 Oracle HSM 파일 시스템 호스트의 이름이며 *YYYYMMDD.hhmmz*는 날짜 및 시간 기록입니다.

   기본 파일 이름은 */tmp/SAMreport.hostname.YYYYMMDD.hhmmz.tar.gz*입니다.

   예제에서는 */zfs1/sam_config/explorer/* 디렉토리에 *samhost1.20140130.1659MST.tar.gz* 파일을 만듭니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시로 이스케이프됨).

```
[samqfs1mds]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

[samqfs1mds]root@solaris:~#
```

4. 파일 시스템을 대폭 재구성할 때마다 이 절차를 반복합니다.
5. 이제 수동으로 Oracle HSM 구성 백업을 수행합니다.

수동으로 Oracle HSM 구성 백업

samexplorer 유틸리티가 많은 Oracle HSM 구성 정보를 캡처하지만 전체 중복성을 위해 주요 구성 작업 후에도 다음 절차를 수행해야 합니다.

1. 파일 시스템 호스트에 *root*로 로그인합니다.

   예제에서 호스트 이름은 *samqfs1mds*입니다.

```
[samqfs1mds]root@solaris:~#
```

2. 백업 구성 정보를 보유하는 디렉토리에서 Oracle HSM 구성의 수동 백업 복사본을 위한 하위 디렉토리를 만듭니다. *mkdir mount-point/path* 명령을 사용합니다. 여기서 *mount-point*는 선택한 독립된 파일 시스템의 마운트 지점이고 *path*는 선택한 디렉토리의 경로 및 이름입니다.

   예제에서는 아카이빙 파일 시스템 */hsmqfs1*에 대한 복구 지점을 구성하는 중입니다. 따라서 */zfs1/sam_config/samconfig* 디렉토리를 만들었습니다.

```
[samqfs1mds]root@solaris:~# mkdir /zfs1/sam_config/samconfig
[samqfs1mds]root@solaris:~#
```

3. 선택한 디렉토리에서 현재 Oracle HSM 구성에 대한 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 날짜를 사용하여 하위 디렉토리의 이름을 지정합니다.

```
samqfs1mdsroot@solaris:~# mkdir /zfs1/sam_config/samconfig/20140127
[samqfs1mds]root@solaris:~#
```

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
```

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*에 대한 카탈로그 데이터를 독립된 NFS 마운트 파일 시스템 *zfs1*의 디렉토리에 있는 *library1cat.dump* 파일로 덤프합니다(아래 명령은 한 라인으로 입력하며 줄바꿈이 백슬래시로 이스케이프됨).

```
samqfs1mdsroot@solaris:~# dump_cat -V /var/opt/SUNWsamfs/catalog/library1cat /
> /zfs1/sam_config/20140513/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 만들기 및 로그 교체를 관리하기 위해 *crontab* 항목을 만든 경우 복사본을 지금 저장합니다.
8. Oracle HSM, Solaris 및 Solaris Cluster(적용 가능한 경우)를 포함한 현재 설치된 소프트웨어의 개정 수준을 기록하고 *readme* 파일 정보의 복사본을 선택한 하위 디렉토리에 저장합니다.
9. 선택한 하위 디렉토리에서 다운로드한 Oracle HSM, Solaris 및 Solaris Cluster 패키지의 복사본을 저장하여 필요할 때 소프트웨어를 빨리 복원할 수 있도록 합니다.
10. 여기서 중지합니다. 구성이 백업되었으며 파일 시스템을 사용할 준비가 되었습니다.

부록 A

부록 A. 장비 유형 용어집

마스터 구성 파일(*mcf*)의 *Equipment Type* 필드 값은 Oracle Hierarchical Storage Manager and StorageTek QFS Software 내에서 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 해당하는 *Equipment Identifier* 필드에는 ACSLS 인터페이스에 대한 매개변수 파일의 경로가 포함되어야 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. 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 C

부록 C. 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 끝납니다.

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 지정된 복사 매개변수를 정의된 모든 아카이브 세트에 적용하는 특수한 지시어 *allsets*에 의해 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 아카이브 매체에 기록할 때 아카이브 파일을 *number-blocks*dev_blksize*로 고정하는 버퍼 크기를 설정합니다. 여기서 *number-blocks*는 버퍼링된 테이프 블록의 수로서 [*2-32*] 범위의 정수이고, *dev*

*_blksize*는 *defaults.conf* 파일에서 매체 유형에 대해 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 사용

할 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 버퍼를 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
- 전역 및/또는 파일 시스템 특정 지시어
- 샘플 **preview.cmd** 파일

전역 지시어

다음은 순수한 전역 지시어입니다.

- **vsn_priority**: 볼륨 우선순위 조정
- **age_priority**: 대기열에서의 대기 소비 시간에 대한 우선순위 조정

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 아카이

브 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 D

부록 D. 예제

/opt/SUNWsamfs/examples/ 하위 디렉토리에는 다양한 요구 사항에 대한 다양한 기능과 솔루션을 보여주는 샘플 Oracle HSM 구성 파일, 셸 스크립트 및 *dtrace* 프로그램이 포함되어 있습니다. 여기에는 다음 파일이 포함됩니다.

```
01_example.archiver.cmd.simple.txt
01_example.mcf.simple.txt
01_example.vfstab.txt
02_example.archiver.cmd.disk.tape.txt
02_example.diskvols.conf.NFS.txt
02_example.mcf.shared.txt
02_example.vfstab.disk.archive.NFS.txt
03_example.archiver.cmd.dk.9840.9940.txt
03_example.diskvols.conf
03_example.mcf.dk.9840.9940.txt
03_example.stk.9840C_parms.txt
03_example.stk.9940B_parms.txt
03_example.vfstab.disk.archive.txt
04_example.archiver.cmd.9840.LTO.txt
04_example.mcf.ma.9840.LTO.txt
04_example.stk50c.txt
05_example.archiver.cmd.9840.9940.T10k.txt
05_example.mcf.veritas.9840.9940.T10K.txt
05_example.stk_params9840.txt
05_example.stk_params9940.txt
05_example.stk_paramsT10K.txt
05_example.vstab.txt
06_example.archiver.cmd.samremote.client.txt
06_example.local.copy.samremote.client.stk50.txt
06_example.mcf.samremote.client.txt
06_example.mcf.samremote.server.txt
06_example.samremote.client.setup.stk100.txt
06_example.samremote.client.vfstab.txt
06_example.samremote.configuration.samremote.server.txt
07_example.mcf.distio.client.txt
07_example.mcf.distio.mds.txt
archiver.sh
defaults.conf
dev_down.sh
dtrace/fs_mon
dtrace/ino_mon
hosts.shsam1
hosts.shsam1.local.client
hosts.shsam1.local.server
inquiry.conf
load_notify.sh
log_rotate.sh
metadata_config_samfs.xml
nrecycler.sh
preview.cmd
recover.sh
```

```
recycler.sh
restore.sh
samdb.conf
samfs.cmd
samst.conf
save_core.sh
sendtrap
ssi.sh
st.conf_changes
stageback.sh
syslog.conf_changes
tarback.sh
verifyd.cmd
```

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. 볼륨, 테이프 또는 매체 조각이라고도 합니다. **volume(볼륨)**, **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. **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. 재활용은 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: SSD(Solid State Disk) 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)**를 참조하십시오.

qfsrestore **samfsrestore(qfsrestore)**를 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. **samfsdump(qfsdump)**, **samfsrestore(qfsrestore)**를 참조하십시오.

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 아카이브 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. 이전 버전의 Oracle Hierarchical Storage Manager 제품에 대한 일반 약어입니다.

2. 아카이브용으로 구성되고 Oracle HSM 소프트웨어에서 관리되는 **QFS** 파일 시스템을 설명하는 형용사입니다.

SAM-Remote client(SAM-Remote 클라이언트)

여러 의사 장치를 포함하는 클라이언트 데몬으로 구성된 Oracle HSM 시스템이며, 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. **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(검색)

랜덤 액세스 I/O 작업 중에 디스크 장치의 읽기/쓰기 헤드를 특정 디스크 위치에서 다른 위치로 이동하는 것입니다.

sftp

Secure File Transfer Protocol의 약어로 **ftp**를 기반으로 하는 **ssh**의 보안 구현입니다.

shared hosts file(공유 호스트 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)

SCSI를 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

기호

ㅅ