StorageTek Automated Cartridge System Library Software

관리자 설명서

릴리스 8.4

E68228-01

2015년 10월

ORACLE

StorageTek Automated Cartridge System Library Software
관리자 설명서

E68228-01

차례

그림 목록

표 목록

예 목록

머리말

StorageTek ACSLS(Automated Cartridge System Library Software)는 Oracle StorageTek 자동화된 테이프 라이브러리를 제어하는 StorageTek UNIX 서버 소프트웨어입니다. StorageTek ACS 제품군은 완전히 자동화된 테이프 카트리지 기반의 데이터 스토리지 및 검색 시스템으로 구성됩니다. StorageTek ACSLS는 다양한 운영체제에서 실행되는 워크스테이션, 메인프레임, 슈퍼컴퓨터에 이르는 다양한 클라이언트 시스템에 네트워크 액세스를 지원합니다.

이 설명서는 StorageTek ACSLS 관리를 담당하는 개인을 대상으로 합니다. 다음에 대한 실무 지식이 있는 것으로 가정합니다.

- UNIX 파일 및 디렉토리 구조
- 현재 플랫폼의 UNIX 명령 및 유틸리티를 사용하는 방법
- UNIX 시스템 파일
- UNIX 응용 프로그램에 루트로 로그온하고 사용자 액세스를 설정하는 등 일반적인 UNIX 시스템 관리자 작업을 수행하는 방법

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.

1

1장. 개요

이 장에서는 ACSLS에 대한 개요를 제공합니다.

ACSLS에 대한 정보

ACSLS(Automated Cartridge System Library Software)는 StorageTek 테이프 라이브러리를 제어하는 Oracle StorageTek 서버 소프트웨어입니다. ACS(Automated Cartridge System)는 PTP(pass-thru-ports)를 통해 연결된 테이프 라이브러리 그룹입니다. ACSLS는 네트워크에서 명령 처리를 통해 하나 이상의 ACS에 저장된 정보에 액세스하고 이를 관리합니다. 이 소프트웨어에는 클라이언트 시스템 응용 프로그램에 대한 시스템 관리 구성 요소 및 인터페이스, 라이브러리 관리 기능이 포함되어 있습니다.

ACSLS HA에 대한 정보

ACSLS HA는 구성 요소 또는 부속 시스템 오류가 발생할 경우 무중단 테이프 라이브러리 제어 서비스를 보장하기 위해 이중 중복성, 자동 복구 및 자동 페일오버 복구를 제공하는 하드웨어 및 소프트웨어 구성입니다.

ZFS 파일 시스템을 사용하는 Solaris 11에서 ACSLS 8.4를 실행하는 데 대한 자세한 내용은 *ACSLS-HA* 설치, 구성 및 작업 안내서를 참조하십시오. 이 버전은 모든 사용자 정의된 파일 시스템에서 ACSLS 소프트웨어 설치를 지원합니다.

acssa 및 acsss 사용자 ID

이 절에서는 acssa 및 acsss 사용자 ID에 대해 설명합니다.

- *acssa*로 로그인하면 *cmd_proc*, 라이브러리 제어 작업을 위한 콘솔 사용자 인터페이스 및 ACSLS 유틸리티의 제한된 세트에 액세스할 수 있습니다.

  *acssa*의 일반적인 셸 환경에는 *cmd_proc*를 실행하는 하나 이상의 창과 ACSLS 이벤트 로그의 실행 끝부분을 모니터링하는 창이 포함됩니다. *acssa* 로그인 환경에서는 다음 리소스 모두에 액세스할 수 있습니다.

  ```
  $ cmd_proc
  $ acs_tail $LOG_PATH/acsss_event.log
  ```

- *acsss*로 로그인하면 일반 유지 관리, 구성, 데이터베이스 백업 및 복원, 셸 유틸리티 및 일반 진단을 위한 이러한 유틸리티와 기타 관리 유틸리티에 모두 액세스할 수 있습니다.

acsss 매크로

acsss 명령은 ACSLS 응용 프로그램과 연결된 여러 서비스를 조작하는 시작, 중지 및 상태 매크로입니다. 2장. *ACSLS* 시작 및 모니터링 및 “acsss 매크로 ” 절을 참조하십시오.

cmd_proc 사용

이 절에서는 *cmd_proc*를 설명합니다.

cmd_proc 창

다음 예제는 *acssa*로 로그인하면 표시되는 *cmd_proc* 창을 보여줍니다. curses 모드에서 *cmd_proc* 창은 분할된 화면으로서, 윗부분은 메시지 영역이고 아랫부분은 명령 영역입니다. 프롬프트에서 ACSLS 명령을 입력합니다.

명령을 허용하려면 ACSLS가 실행 중이어야 합니다. "*-q*" 옵션을 사용하여 *cmd_proc* 명령을 시작할 때 초기 쿼리 서버 요청을 억제할 수 있습니다.

```
cmd_proc -q
-------------------------ACSLS x.x.x--------------------------
ACSSA>query server
2008-01-23 15:41:42

Server Status
Identifier    State   Free Cell Audit Mount Dismount Enter Eject
                        Count    C/P   C/P    C/P      C/P   C/P
               run       234     0/0   0/0    0/0      0/0   0/0
```

cmd_proc 사용: Curses 모드와 라인 모드 비교

ACSLS *cmd_proc*는 고유 요청을 처리하는 동안 일반 서버 상태 정보를 계속 알리는 사용하기 쉬운 인터페이스입니다. *cmd_proc*의 기본 모드는 curses입니다. 이 인터페이스는 대부분의 터미널 유형에서 작동하는 다기능 인터페이스이며 표준 24행 x 80자 창을 사용합니다. Curses 인터페이스는 화면을 두 개의 섹션으로 나눕니다. 여기서 *STDERR*를 향한 메시지는 창의 윗부분으로 전송되고 *STDOUT*를 향한 메시지는 아랫부분으로 전송됩니다.

기본 Curses 모드에서 ACSLS *cmd_proc*를 사용하는 경우, 일반 서버 상태 메시지는 창의 윗부분에 표시되는 반면 사용자 고유 상호작용은 아래에 표시됩니다.

Curses 모드에서 보유되는 내역 제한

Curses의 한 가지 단점은 ACSLS 서버와 사용자의 상호작용 내역을 보유하는 기능이 제한된다는 것입니다. 이러한 상호작용에 사용하는 공간은 24행 창의 아래쪽 절반으로 제한됩니다.

라인 모드에서 *cmd_proc*를 사용하는 경우 이 단점을 해결할 수 있습니다.

```
cmd_proc -l
```

라인 모드에서는 사용자가 상호작용 내역이 스크롤 가능한 터미널 버퍼까지 이동하는 스크롤 창을 이용할 수 있습니다. 이 창은 버퍼 크기로만 제한됩니다.

라인 모드에서 명령과 혼합된 상태 메시지

라인 모드 작업의 가장 큰 단점은 *STDOUT*와 *STDERR*를 개별 공간으로 분할할 수 없다는 것입니다. 두 소스 모두의 출력 텍스트가 화면의 동일한 지점, 즉 요청을 작성하려고 시도하는 터미널의 단일 커서 행으로 전송됩니다.

cmd_proc 세션이 시스템의 유일한 세션인 경우에는 문제가 되지 않습니다. ACSLS를 사용하여 활성 작업을 진행 중인 사용량이 많은 프로덕션 환경에서, ACSLS 요청을 작성하는 라인에 상태 정보가 인쇄되는 창에서 작업하는 것은 매우 어려울 수 있습니다.

입력 중인 라인에서 시스템 상태 알림을 무시하는 것이 안전하지만 이 알림의 경로를 다른 위치로 재지정할 수 있습니다. 시스템 메시지의 경로를 다른 대상으로 재지정하려면 다음 방식으로 라인 모드 *cmd_proc*를 실행할 수 있습니다.

```
cmd_proc -l 2> /tmp/SysChatter.out
The expression 2> instructs the shell to redirect STDERR to another location. In this
 example, the status messages are sent to a file in the /tmp directory.
```

작업하면서 시스템 상태 정보를 보려면 두번째 셸 창을 열고 상태 메시지를 보낸 파일의 실행 증적을 볼 수 있습니다.

```
tail -f /tmp/SysChatter.out
```

원하는 *cmd_proc* 작업을 수행하려면 *STDERR*의 경로를 */dev/null*로 재지정할 수 있습니다.

```
cmd_proc -l 2> /dev/null
```

Curses 모드의 cmd_proc가 80자를 초과하는 라인을 표시할 수 없음

Curses 모드의 *cmd_proc* 명령은 80자보다 긴 행은 표시할 수 없으며, 80자보다 긴 행을 표시하려고 하면 *cmd_proc* 창이 중단됩니다.

이 경우 *Control+c* 및 *Control+d*를 사용하여 *cmd_proc* 창을 해제할 수 있습니다.

모든 *query* 및 기타 명령의 출력은 행당 80자 미만이며 display 명령을 통해 모든 레코드에 대해 보고된 기본 필드는 80자 미만이어야 합니다. 그러나 여러 선택적 필드를 표시하면 행의 문자가 80자가 넘을 수 있습니다.

여러 선택적 필드를 표시하는 경우에는 라인 모드에서 *cmd_proc*를 시작하는 것이 좋습니다(*-l* 옵션 사용). 예제: *cmd_proc -l*로 시작되는 *cmd_proc*를 사용하는 *display drive * -f volume type state serial_num wwn*

cmd_proc를 일시 중단하고 재시작하는 방법

*cmd_proc*를 일시중지하여 UNIX 명령을 수행한 다음 *cmd_proc*를 재시작할 수 있습니다. *cmd_proc*를 수동으로 시작해야 합니다. *cmd_proc*에서 시작하여 진행 중인 요청은 *cmd_proc*가 일시 중단된 동안에도 완료될 때까지 계속 진행됩니다.

cmd_proc 일시 중단 및 재시작:

1. *cmd_proc*를 실행 중인 동안 *Control+z*를 누릅니다.
2. UNIX 셸 프롬프트가 열립니다.

   원하는 UNIX 작업을 수행합니다.
3. *cmd_proc*를 다시 시작하려면 fg UNIX 명령을 입력합니다.

cmd_proc 종료

1. *cmd_proc*를 실행하는 동안 진행 중인 모든 활동이 완료되고 *ACSSA*> 프롬프트가 반환될 때까지 기다립니다.
2. *cmd_proc*를 종료하려면 다음과 같이 *logoff* 명령을 입력합니다.

   ```
   logoff
   ```

3. *cmd_proc* 세션이 종료됩니다.

cmd_proc 시작

*/etc/termcap*에 정의된 터미널 유형에서 c*md_proc*를 시작할 수 있습니다. Curses 모드에서 실행할 때 터미널의 표시 크기는 24x80 이상이어야 합니다.

cmd_proc 세션은 ACSLS와는 별개인 모드로 실행됩니다. ACSLS를 시작하지 않고 *cmd_proc* 세션을 시작하면 명령에 응답하지 않습니다. ACSLS가 실행 중이지 않은 동안 명령을 실행하려고 할 때 *cmd_proc*에서 소켓 통신 오류가 표시될 수 있습니다.

원격으로 로그인

SSH 클라이언트가 있는 모든 시스템에서 ACSLS 서버에 원격으로 액세스할 수 있습니다. *ssh* 클라이언트는 대부분의 POSIX 규격 운영체제(예: Solaris, Linux 및 MacOS)에서 셸을 사용하는 표준 기능입니다. Windows 환경의 경우 putty, WinSCP 또는 비슷한 상용 응용 프로그램 등의 SSH 클라이언트 소프트웨어를 설치해야 합니다.

acssa, 사용자로 ACSLS 서버에 원격으로 액세스하려면 다음 명령을 입력합니다.

```
$ ssh acssa@hostname
```

여기서 *hostname*은 ACSLS 서버의 호스트 ID입니다.

acssa의 일반적인 제거 환경에는 *cmd_proc*을 실행 중인 하나 이상의 SSH 로그인 셸과 ACSLS 이벤트 로그의 실행 증적을 모니터링하는 다른 셸이 포함됩니다.

```
$ acs_tail $LOG_PATH/acsss_event.log
```

cmd_proc 키보드 바로 가기 키

다음 표에서는 <CTRL>+ 키 입력 조합인 *cmd_proc* 키보드 바로 가기 키에 대해 설명합니다.

표 1.1. cmd_proc 키보드 바로 가기 키

키 조합	작업	참고 사항
Control+c	마지막 *cmd_proc* 명령을 취소합니다.	*Control+c*는 *cancel* 명령의 키보드 바로 가기 키입니다. *cancel* 명령에 대한 자세한 내용은 "cancel "의 내용을 참조하십시오.
Control+d	*cmd_proc* 프롬프트로 돌아갑니다.	현재 명령이 완료된 경우 *Control+d*는 적용되지 않습니다. 현재 명령을 처리 중인 경우 명령이 완료되지만 *cmd_proc*에서 응답 메시지를 표시하지 않습니다. ACSSS 프롬프트에 현재 명령을 입력하지 않은 경우 *Control+d*가 명령을 삭제합니다.
Control+h	명령줄에서 이전 문자를 삭제합니다.	대부분의 키보드에서 Enter 또는 백스페이스 키도 사용할 수 있습니다.
Control+i	*cmd_proc* 표시를 새로 고칩니다.	이 기능은 현재 *cmd_proc* 표시가 통신 회선의 잡음으로 인해 손상된 경우 유용합니다.
Control+r	현재 명령줄을 새로 고칩니다.	이 기능은 현재 명령줄 표시가 통신 회선의 잡음으로 인해 손상된 경우 유용합니다.
Control+r	현재 명령줄을 삭제합니다.	N/A
Control+z	*cmd_proc*를 일시 중단하고 셸 환경으로 이스케이프합니다.	*C shell fg* 명령을 입력하여 *cmd_proc*를 다시 시작합니다.

cmd_proc 입력 및 출력의 경로 재지정

*cmd_proc*를 시작할 때 입력 파일을 사용하여 명령을 자동으로 입력할 수 있습니다. 예를 들어, 카트리지를 마운트하고 마운트 해제하여 다음 입력 파일이 ACSLS를 검증합니다.

```
query drive 0,0,0,0
query volume JB1400
mount JB1400 0,0,0,0
dismount JPB1400 0,0,0,0 force
logoff
```

추가 cmd_proc 창에 입력 파일 지정

*cmd_proc*를 시작하려면 다음 명령을 입력합니다.

cmd_proc -q < filename

*cmd_proc*를 시작하고 입력 파일을 지정한 다음 출력 파일의 경로를 다른 파일로 재지정할 수도 있습니다. 입력 및 출력 파일을 사용하면 *cmd_proc* 시작 시 명령 세트를 실행하고 결과를 볼 수 있습니다. 예를 들어, 다음 파일은 입력 파일만 사용하는 *cmd_proc*를 표시하는 이전 예에서 실행되는 명령의 결과를 보여줍니다.

```
ACSSA> query drive 0,0,0,0
1998-06-30 18:23:08
Identifier State   Status     Cartridge Type
```

```
0,0,0,0    online  available  9840
ACSSA> query volume JPL1400
1998-06-30 18:23:09
Identifier Status Current location
JB1400     home   0,0,3,0,0
ACSSA> mount JPL1400 0,0,0,0
ACSSA> Mount: JB1400 mounted on 0,0,0,0
ACSSA> dismount JPL1400 0,0,0,0 force
ACSSA> Dismount: Forced dismount of JB1400 from 0,0,0,0
ACSSA> logoff
ACSSA
```

출력의 경로를 추가 cmd_proc 창으로 재지정:

추가 *cmd_proc*를 시작하려면 다음과 같이 입력 파일을 지정하고 출력의 경로를 재지정합니다.

1. *acssa* 또는 *acsss*로 로그인한 상태에서 UNIX 터미널 창을 엽니다.
2. cmd_proc를 시작하려면 다음 명령을 입력합니다.

   cmd_proc -q < file1 > file2

   여기서 *file1*은 입력 파일이고 *file2*는 출력이 전송되는 파일입니다.

   기본적으로 *cmd_proc* 표시 영역 메시지는 *stderr*에 쓰지만, 이러한 메시지의 경로를 재지정할 수도 있습니다. 예:

   cmd_proc -q < file1 > file2 2>> file2

ACSLS를 유휴 상태로 이동

이 절차를 사용하여 ACSLS를 유휴 상태로 만들어 요청 처리를 일시 중단합니다. 일반적으로 이 절차는 ACSLS를 종료하기 전에 사용하지만, ACSLS 요청 처리를 임시로 중지하는 데도 사용할 수 있습니다.

ACSLS를 유휴 상태로 이동:

*cmd_proc*에서 idle 명령을 입력합니다.

ACSLS가 현재 요청을 모두 처리하고, 새 요청을 모두 거부하며, 유휴 상태로 이동합니다.

ACSLS 다시 시작

이 절차를 사용하여 ACSLS를 실행 상태로 만들어 요청 처리를 다시 시작합니다. 일반적으로 ACSLS를 다시 시작하면 유휴 상태에서 제거됩니다.

ACSLS를 다시 시작하려면 다음을 수행합니다.

*cmd_proc*에서 다음 명령을 입력합니다.

```
start
```

ACSLS가 요청 처리를 다시 시작합니다.

ACSLS 디렉토리 구조

다음 표는 ACSLS 디렉토리 구조의 디렉토리, 하위 디렉토리 및 가장 일반적으로 사용되는 파일 및 셸 스크립트를 보여줍니다.

세 가지 변수가 ACSLS 경로에 사용됩니다. 다음과 같습니다.

- *$installDir*

  이 디렉토리는 기본 설치 디렉토리이며 기본적으로 */export/home/*입니다.
- *$ACS_HOME*

  *$installDir/ACSSS/*에 있는 이 디렉토리는 *acsss* 사용자 ID의 홈 디렉토리이며 ACSLS 제품이 설치되어 있습니다.

  *$ACS_HOME*은 기본적으로 */export/home/ACSSS*입니다.
- *$ACSDB_BKUP*

  이 디렉토리는 ACSLS 백업이 저장되는 디렉토리입니다.

표 1.2. ACSLS 디렉토리 구조

디렉토리	내용
$installDir (by default /export/home/)	기본 설치 디렉토리입니다.
$installDir/SSLM	ACSLS GUI 및 SMCE(논리 라이브러리 작업)를 포함하는 ACSLS java 구성 요소의 홈
$installDir/SSLM/AcslsDomain	ACSLS 웹 기반 GUI 응용 프로그램의 홈 디렉토리입니다.
$installDir/wlinstall	번들로 제공되는 WebLogic 응용 프로그램 서버 패키지 및 관련 설치 스크립트입니다.
$installDir/Oracle	번들로 제공되지 않는 WebLogic 홈 디렉토리입니다.
$installDir/acsls_thirdPartySoftware	타사 라이센스 정보 및 재발행된 관련 소스 코드의 모음입니다.
$ACS_HOME ($installDir/ACSSS) (기본값은 /export/home/ACSSS/)	*acsss* 사용자 ID의 홈 디렉토리입니다. ACSLS 홈 디렉토리이기도 합니다. (*ACS_HOME* 환경 변수는 이 디렉토리를 나타냅니다.)
$ACSDB_BKUP (기본값은 /export/backup/)	데이터베이스 백업
$ACS_HOME/config/	ACSLS 구성 파일을 포함합니다.
$ACS_HOME/data/external/	액세스 제어, 혼합 매체 및 카트리지 보고에서 사용하는 사용자 정의된 파일을 포함합니다.
$ACS_HOME/data/external/access_control/	액세스 제어 샘플 및 사용자 정의된 파일을 포함합니다.
$ACS_HOME/data/internal/	ACSLS 내부 구성 파일. 수정하지 마십시오.
$ACS_HOME/diag/bin	진단 파일 및 셸 스크립트를 포함합니다.
$ACS_HOME/lib/	런타임 시 필요한 ACSLS 설치 공유 라이브러리를 포함합니다.
$ACS_HOME/log/	ACSLS 이벤트 로그 및 유틸리티 이벤트 로그 파일을 포함합니다.

디렉토리	내용
$ACS_HOME ($installDir/ACSSA/) (기본값은 /export/home/ACSSA/)	*acssa* 홈 디렉토리입니다.
$installDir/ascdb/ (기본값은 /export/home/acsdb/)	데이터베이스 홈 디렉토리입니다.
$LOG_PATH	*$ACS_HOME/log*와 동일합니다. 이 디렉토리에는 *acsss_event.log* 및 ACSLS 작업에 속한 기타 유용한 로그가 포함되어 있습니다.

2

2장. ACSLS 시작 및 모니터링

ACSLS가 설치되고 연결된 라이브러리가 구성된 경우, *acsss enable* 명령을 사용하여 응용 프로그램을 사용으로 설정할 수 있습니다. *acsss* 매크로는 ACSLS와 연결된 여러 서비스를 조작하고, 적절한 순서로 서비스를 시작 및 종료하며, 전체 시스템 상태의 상위 레벨 보기를 제공합니다.

설치에 따라 ACSLS 응용 프로그램은 Solaris 또는 Linux 시스템에 설치된 최대 7개의 서비스로 구성된 집계입니다.

- *acsdb* - ACSLS 라이브러리 데이터베이스를 유지 관리합니다.
- *acsls* - 라이브러리 작업을 실행하는 라이브러리 제어 소프트웨어입니다.
- *weblogic* - ACSLS GUI용 웹 서버입니다.
- *surrogate* - java 서비스와 acsls 사이의 통신 링크입니다.
- *rmi-registry* - 명명된 java 객체 및 방법에 대한 조회 서비스입니다.
- *smce* - 논리 라이브러리의 SCSI 매체 교환기 에뮬레이션입니다.
- *stmf* - 논리적 라이브러리의 대상 모드 프레임워크입니다.

처음 두 서비스는 모든 설치에 공통됩니다. ACSLS GUI가 설치된 경우 *weblogic*, *surrogate* 및 *rmi-registry* 서비스가 있습니다. *smce* 및 *stmf* 서비스는 논리 라이브러리 지원이 구성된 Solaris 시스템에서 제공됩니다. 이러한 모든 서비스는 ACSLS 사용자가 하나의 매크로, *acsss*를 사용하여 처리합니다.

ACSLS 시작

*root*로서 다음을 실행하여 ACSLS를 시작합니다.

```
acsss enable
```

이 명령은 ACSLS를 시작하는 기본적인 방법입니다. 종속성을 검사하고 다양한 ACSLS 서비스와 ACSLS GUI를 적절한 순서로 활성화합니다. 서비스는 시스템 재부트 후 자동으로 다시 시작되도록 구성됩니다.

ACSLS 모니터링

다양한 ACSLS 서비스에 대한 빠른 상태 보고를 보려면 다음 명령을 실행합니다.

```
acsss status
```

ACSLS 중지

ACSLS 중지는 완전히 종료되는 것이 아니며 acsls 및 smce 서비스가 사용 안함으로 설정된 후 데이터베이스 및 GUI 로그인 세션이 유지 관리 작업에 대해 활성 상태로 유지되도록 합니다. 이 절차를 사용하여 ACSLS와 데이터베이스를 종료합니다.

ACSLS를 중지하려면 다음 명령을 사용합니다.

```
acsss disable
```

Solaris에서 SMF 시간 초과

Solaris SMF 유틸리티는 각 서비스가 완전히 사용 가능하도록 일정한 시간을 지정합니다. *acsls* 서비스를 위해 이 시간 한계는 라이브러리 구성(LSM 수, 드라이브 수 및 CAP 수)에 따라 계산됩니다. 라이브러리 구성이 크면 구성이 작은 경우보다 ACSLS를 복구하는 데 시간이 오래 걸리므로, 구성이 크면 더 긴 SMF 시간 초과가 지정됩니다.

드문 경우지만 LSM에 결함이 있으면 SMF 시간 제한에서 허용하는 시간보다 오랜 시간이 걸릴 수 있습니다. 시간 초과 기간이 만료되면 SMF가 작업을 다시 시작합니다. 이 작업으로 인해 시작 순서가 무한 루프에 빠질 수 있으므로, 어려운 시작 조건에서 ACSLS가 절대 복구하지 못할 수 있습니다.

*acsls_startup_policy*라는 특수 파일은 이러한 경우에 사용하도록 고안되었습니다. *$ACS_HOME/data/external* 디렉토리에 있는 이 파일이 구성된 경우 시작 복구를 위한 시간이 추가되거나 SMF 시작 순서 중에 특정 ACS가 복구되지 못합니다. 자세한 구성 지침은 *acsls_startup_policy*의 헤더 설명에 포함되어 있습니다. 이 파일에서 시작 매개변수를 조정하면 비정상 라이브러리 시작 조건으로 인한 ACSLS 시작 문제를 방지할 수 있습니다.

자세한 내용은 “ACSLS 시작 문제 진단”의 내용을 참조하십시오.

ACSLS 시작 정책

이 파일은 ACSLS를 시작할 때 적용되는 정상 시작 매개변수를 변경합니다. Oracle ACSLS 소프트웨어 지원과 상의하거나 신중하게 분석하지 않은 경우 기본 시작 값을 변경하지 않는 것이 좋습니다.

추가 시작 시간

이 매개변수는 Solaris에서 *acsls* 서비스에 대한 SMF 시작 시간 초과에 적용됩니다. *acsls* 시작 시간 초과는 현재 라이브러리 구성을 통해 자동으로 계산합니다. LSM, 드라이브 및 CAP가 더 많은 라이브러리에 더 긴 시간 초과가 지정됩니다. 이 시간 초과는 라이브러리 구성이 변경됨에 따라 자동으로 조정되며, 다음 명령을 통해 계산된 값을 볼 수 있습니다.

```
acsss timeout
```

자동으로 계산된 시간 초과가 충분하지 않으면 이전 시작 순서가 완료될 수 있도록 충분한 시간이 경과되기 전에 SMF 기능이 *acsls* 서비스를 다시 시작하도록 개입할 수 있습니다.

시작 순서에 시간을 더 지정하면 아무 손상 없이 SMF 개입과 같은 동작을 방지할 수 있습니다. 너무 많은 시간을 추가하면 주의해야 하는 문제의 구성 요소가 가려질 수 있습니다. 일반 시간 초과 기간을 확장하면 운영자에게 심각하거나 복구 불가능한 시작 문제를 알리는 SMF의 기능이 지연됩니다.

acsls start 순서가 완료되도록 추가 시간(분)을 부여하려면 다음 행에서 '=' 다음에 정수 값을 입력합니다.

```
additional_startup_time=0 # Minutes
```

ACS의 원하는 (오프라인) 시작 상태

ACSLS가 시작되면 모든 라이브러리 리소스가 마지막으로 설정된 원하는 상태가 됩니다. 원하는 상태가 온라인이면 ACS를 온라인으로 만드는 프로세스를 수행할 때, 지정된 ACS의 물리적 라이브러리 리소스를 구성의 데이터베이스 이미지와 비교하여 확인하고 검증하는 복구 기간이 필요합니다. 이 프로세스는 라이브러리 구성의 크기 및 이례적인 상황 발생 여부에 따라 1분 미만에서 몇 분까지 수행될 수 있습니다.

ACS의 원하는 상태와 관련 포트를 오프라인으로 만들어 ACS의 복구 시간을 무시할 수 있습니다. 이러한 조치로 인해 온라인 상태의 *acsls* SMF 서비스 속도가 빨라지면, 실제 ACS와 포트를 온라인으로 변경하기 위한 후속 수동 작업이 필요합니다.

ACS 및 해당 포트의 원하는 시작 상태를 오프라인으로 설정하려면 *$ACS_HOME/data/external/* 디렉토리의 *acsls_startup_policy* 파일에 있는 해당 행의 시작 부분에서 설명 문자(*#*)를 제거합니다.

예를 들어,

```
# ACS0_desired_startup_state_is_offline
```

명령을 다음으로 변경합니다.

```
ACS0_desired_startup_state_is_offline
```

3

3장. ACSLS 라이센스 키

StorageTek ACSLS 버전 7.3.1부터 소프트웨어 사용권 라이센스가 ACSLS에 더 이상 적용되지 않으며 ACSLS가 유효한 라이센스 키를 더 이상 검사하지 않습니다. 곧 만료될 라이센스 키나 라이브러리 용량 라이센스와 관련된 메시지가 시스템 콘솔이나 *acsss_event.log*에 더 이상 나타나지 않습니다.

다음 유틸리티는 유효한 라이센스 키를 설정하고 검사할 목적으로 더 이상 작동하지 않습니다.

- *licensekey.sh*
- *get_license_info.sh*

4

4장. ACSLS GUI

ACSLS 8.0에 도입된 웹 기반 ACSLS GUI(그래픽 사용자 인터페이스)는 라이브러리 작업을 조작 및 모니터링하고 논리적 라이브러리를 관리 및 작업하기 위한 그래픽 콘솔을 제공합니다. GUI 사용에 대한 절차와 자세한 내용은 온라인 도움말을 참조하십시오.

GUI에는 다음이 제공됩니다.

- 유연성이 대폭 향상되고 사용이 편리한 대체 라이브러리 콘솔. 기존 *cmd_proc*에서 사용 가능한 대부분의 작업을 제공합니다(예외는 “ACSLS 8.4 GUI에 포함된 시스템 작업 ”에 나와 있음).
- 논리적 라이브러리 지원

  논리적 라이브러리에 대한 자세한 내용은 “논리적 라이브러리 정보”를 참조하십시오.
- 라이브러리 관리 및 작업 지원. 논리적 라이브러리 관리와 관련된 새로운 작업과 함께 대부분의 기존 *cmd_proc* 작업을 수행하는 기능을 제공합니다.
- 테이프 라이브러리 구성 요소의 실시간 모니터링
- 물리적/논리적 구성을 탐색할 수 있는 트리 브라우저
- 각 화면에서 볼 수 있는 실시간 경보

  경보는 하드웨어, 데이터 또는 응용 프로그램 소프트웨어를 손상시킬 수 있는 조건에 대해 알려주며, 항상 관련 정보 앞에 표시됩니다.
- 사용자 지정 조건에 따라 볼륨 및 드라이브 표시를 필터링하는 기능
- 시스템 이벤트 및 시스템 로그에 대한 사용자 정의 보기
- 온라인 도움말

논리적 라이브러리를 만들고 관리하거나 삭제하려면 ACSLS GUI(그래픽 사용자 인터페이스)를 사용해야 합니다. GUI는 ACSLS 8.4 설치 패키지에 자동으로 포함됩니다. 논리적 라이브러리에 대한 자세한 내용은 부록 H. 논리적 라이브러리 지원을 참조하십시오.

ACSLS GUI 시작

ACSSS 서비스를 사용으로 설정하면 ACSLS GUI를 제어하는 소프트웨어가 자동으로 시작됩니다. ACSLS GUI는 Solaris 플랫폼의 WebLogic 내에 다른 응용 프로그램과 함께 배치됩니다.

ACSLS GUI에 로그인하려면 다음을 수행합니다.

1. 브라우저를 열고 서버 호스트 이름 또는 IP 주소가 포함된 URL을 제출합니다.

   https://myAcslsHostName.myDomainName:7002/SlimGUI/faces/Slim.jsp

   또는

   http://127.99.99.99:7001/SlimGUI/faces/Slim.jsp

   호스트 시스템의 전체 호스트 이름 또는 IP 주소를 사용하는 것이 가장 좋습니다. WebLogic에서 URL을 완전히 분석할 수 없는 경우 ACSLS 도움말 페이지를 포함한 일부 페이지가 제대로 표시되지 않을 수 있습니다.

   http와 7001 포트를 사용하는 경우 WebLogic이 https와 7002 포트에 자동으로 경로를 다시 지정합니다.

   WebLogic은 보안 https 프로토콜을 사용하므로 브라우저에 사이트 보안 인증서가 등록되어 있지 않으면 신뢰할 수 없다는 경고가 나타날 수 있습니다. URL이 로컬 ACSLS 서버라고 확신하면 계속 진행해도 안전합니다. 이 시점에 로그인 화면이 표시됩니다. 브라우저에 보안 인증서가 필요한 경우 “HTTPS에 자체 지정 디지털 인증서 구성 ”을 참조하십시오.
2. 로그인 화면에서 유효한 ACSLS 사용자 ID와 해당 암호를 입력합니다. WebLogic을 설치할 때 또는 *userAdmin.sh*를 통해 설정하는 사용자 ID(예: *acsls_admin*)입니다.
3. 로그인이 성공하면 ACSLS GUI Welcome 페이지가 표시됩니다.

GUI 사용자 및 암호 관리

userAdmin.sh 메뉴 방식 유틸리티를 사용하여 ACSLS GUI 사용자와 암호를 추가하고 관리합니다. “userAdmin.sh ”를 참조하십시오.

GUI 개요

다음 절에서는 GUI의 기본 개요에 대해 설명합니다.

GUI Welcome 화면

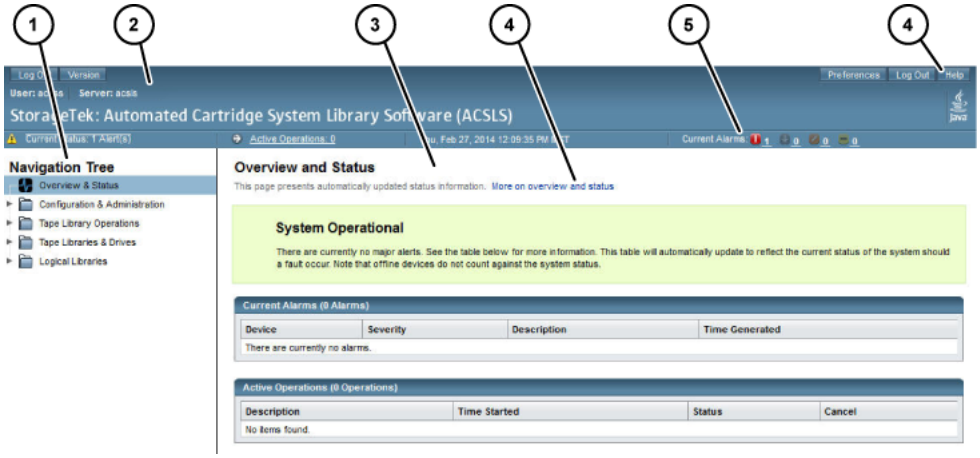
ACSLS GUI Welcome 화면에는 다음과 같은 창 3개가 표시됩니다.

- 최상위 프레임의 마스트헤드
- 왼쪽 프레임의 탐색 트리
- 오른쪽 프레임의 Overview and Status 페이지

  GUI를 브라우저에 처음으로 로드한 후 오른쪽 프레임이 표시되지 않는 경우 브라우저를 새로 고칩니다.

다음 그림에서는 ACSLS GUI를 보여줍니다.

그림 4.1. ACSLS 그래픽 사용자 인터페이스



범례:

1. 탐색 트리
2. 마스트헤드
3. 페이지
4. 도움말 링크
5. 알람

마스트헤드

마스트헤드는 다음과 같은 버튼으로 구성됩니다.

- Log Out

  Log Out 버튼을 누르면 GUI에서 로그아웃됩니다.

  보안을 강화하기 위해 GUI에서 로그아웃할 때마다 검색 기록을 지우는 것이 좋습니다.
- Version

  Version 버튼을 누르면 현재 ACSLS 버전과 저작권 통지가 표시됩니다.
- Preferences

  Preferences 버튼을 누르면 GUI 시스템 환경 설정을 지정할 수 있는 페이지가 열립니다.
- Logout

  Logout 버튼을 누르면 현재 세션이 종료됩니다.
- Help

  Help 버튼을 누르면 온라인 도움말 시스템이 시작됩니다.
- Active Operations

Active Operations 버튼을 누르면 진행 중인 ACSLS 작업 수가 표시됩니다. 알람 및 활성 작업에 대한 세부정보를 보여주는 Overview and Status 페이지에 대한 하이퍼링크입니다.

- Current Status

  Current Status 버튼을 누르면 활성 경보 수가 표시됩니다. Overview and Status 페이지로 연결되는 하이퍼링크입니다.

- Date and Time of Login

  Date and Time of Login은 현재 GUI 세션에 원래 로그인 시간을 보여주는 시간 기록입니다.

- Current Alarms

  Current Alarms 버튼은 현재 시스템 알람의 수와 심각도를 보여주는 대시보드 보기입니다. 현재 알람에 대한 세부정보가 Overview and Status 페이지에 요약됩니다.

알람 색상 코드

다음 표에서는 알람 색상 코드에 대해 설명합니다.

색상	심각도 레벨	영향 받는 장치
빨간색	위험	일반 ACSLS 작업을 심각하게 방해하는 조건을 의미합니다. 위험 알람은 근본적인 문제에 즉각적인 관심이 요구되는 경우에 표시됩니다. 결과적으로 시스템 전반에서 일반 ACSLS 라이브러리 작업이 중지될 수 있습니다.
검정색	작동 중지	이 알람은 ACSLS 내의 부속 시스템 중 하나 이상이 오프라인 상태임을 나타냅니다. 오프라인 상태는 의도적 수동 조작에 따른 것일 수도 있고 다른 요인으로 인해 발생한 것일 수도 있습니다. 일반적으로 기술적 요인이 연관된 경우 노란색(차요) 알람 또는 주황색(주요) 알람 아이콘이 표시됩니다.
주황색	주요	주요 알람은 ACSLS 또는 GUI를 작동하도록 하는 기본 구성 요소 또는 소프트웨어의 오작동이 감지된 경우에 나타납니다.
노란색	차요	이 색상은 시스템이 완전하게 작동하지 않음을 나타냅니다. 시스템이 초기화 중이거나 하나 이상의 개별 부속 시스템이 작동하지 않을 수 있습니다. 이 중단은 일시적으로 간주되거나, 표시된 기능이 전체 라이브러리 제어 작업에 중요하지 않은 별개의 기능인 경우입니다.

마스트헤드 알람은 해당 범주 내에 하나 이상의 활성 알람이 있는 경우에만 색상으로 표시됩니다. 실제 알람 수는 색상 아이콘 옆에 숫자 텍스트로 표시됩니다. 활성 알람이 없는 경우 숫자 0이 표시됩니다.

색맹인 사용자를 위해 마우스 커서를 각 색상 아이콘 위로 가져가면 설명이 표시되는 "도구 설명" 기능이 있습니다. 작은 팝업 창에 해당 그래픽 아이콘의 의미가 일반 텍스트로 표시됩니다.

System Preferences

이 페이지는 마스트헤드에서 Preferences 버튼을 눌러 액세스할 수 있습니다. 이 페이지에는 개인 환경 설정에 따라 시스템 동작을 변경할 수 있는 다양한 드롭다운 메뉴가 제공됩니다.

- Default Tree Menu - 확장되거나 축소된 메뉴 형식을 지정합니다.
- Log Page Size - Log Viewer의 페이지에 표시할 이벤트 수를 지정합니다.
- Alert Update Interval - 브라우저에서 서버의 시스템 경보를 프로브할 빈도를 지정합니다. GUI 응답 시간이 느린 경우 이 간격을 늘릴 수 있습니다.
- Number of Days to Retain Events - System Events 페이지에 내역을 얼마나 표시할지를 지정합니다.
- Enable Fast Load for SCSI Clients - 유효한 마운트 요청 수신 후 즉시 SCSI 클라이언트에 성공 응답을 반환하려면 (yes)를 지정하고, 로봇 마운트 작업이 실제로 완료될 때까지 기다리려면 (no)를 지정합니다.

Navigation Tree

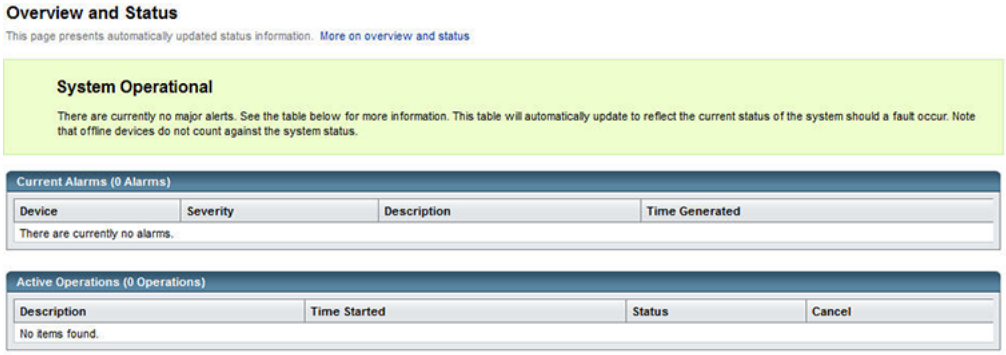
Navigation Tree는 항상 ACSLS GUI의 왼쪽 프레임에 있습니다. 축소된 트리에는 다음에 대한 탐색 링크가 제공됩니다.

- Overview and Status 페이지
- Configuration and Administration 페이지
- Tape Library Operations 페이지
- Tape Libraries and Drives 페이지
- Logical Libraries 페이지

Overview and Status

이 페이지는 마스트헤드에서 알람 아이콘으로 표시된 서비스에 대한 특정 오작동 또는 중단을 식별하려는 경우 처음으로 이동하는 페이지입니다. 이 페이지의 상단에는 전체 시스템 조건의 심각도를 반영하는 상태 요약이 제공됩니다.

그림 4.2. ACSLS Overview and Status 페이지



상태 요약 아래에는 두 개의 상자가 표시됩니다. 다음과 같습니다.

- Current Alarms

  Current Alarms에는 영향 받는 장치, 심각도 레벨, 설명 및 이벤트가 발생한 시간이 표시됩니다. 심각도 레벨은 다음과 같습니다.

  - System Critical은 빨간색 아이콘에 빨간 배경색으로 표시됩니다. 일반 ACSLS 작업을 심각하게 방해하는 조건을 의미합니다.
  - System Degraded는 노란색 아이콘에 노란 배경색으로 표시됩니다. 시스템이 완전하게 작동하지 않음을 나타냅니다. 시스템이 초기화 중이거나 하나 이상의 개별 부속 시스템이 작동하지 않을 수 있습니다.
  - System Operational은 녹색 배경색으로 표시됩니다. 모든 부속 시스템이 작동하고 시스템 구성 요소에서 오류가 감지되지 않았음을 나타냅니다.
- Active Alarms

  Active Operations 테이블에는 현재 진행 중인 라이브러리 작업이 나열됩니다. 각 작업은 간략한 설명, 작업 시작 시간, 현재 상태, 작업 취소를 위한 라디오 버튼 옵션과 함께 나열됩니다.

Configuration and Administration

Configuration and Administration 섹션에는 ACSLS 시스템을 모니터링할 수 있는 페이지에 대한 링크와 논리적 라이브러리 구성을 관리할 수 있는 다른 페이지에 대한 링크가 포함되어 있습니다. 트리의 이 리프를 확장하면 다음과 같은 하위 메뉴가 표시됩니다.

- Log Viewer
- System Events
- Logical Library Configuration

Log Viewer

기본 Log Viewer 페이지에는 *acsss_event.log*, *smce_trace.log*를 비롯하여 시스템 이벤트를 모니터하는 실행 중인 다양한 로그의 끝 부분이 표시됩니다. 이 페이지에는 로그의 내역을 시작 부분까지 다시 스크롤할 수 있는 탐색 버튼이 제공됩니다.

System Events

모든 개별 라이브러리 작업이 System Events 로그에 기록됩니다. 이 로그의 각 레코드에는 이벤트 시간 기록, 이벤트 유형, 이벤트 설명이 포함됩니다.

Logical Library Configuration

트리의 이 리프를 확장하면 논리적 라이브러리를 구성 및 관리할 수 있는 다양한 옵션이 열립니다.

- 논리적 라이브러리 만들기
- 논리적 라이브러리 편집
- 논리적 라이브러리에 볼륨 지정
- 논리적 라이브러리에서 볼륨 지정 해제
- 클라이언트 연결 관리

논리적 라이브러리에 대한 자세한 내용은 부록 H. 논리적 라이브러리 지원을 참조하십시오.

Tape Library Operations

테이프 작업을 사용하여 다음을 수행할 수 있습니다.

- ACS, LSM 또는 패널 감사
- 볼륨 마운트
- 볼륨 마운트 해제
- 볼륨 넣기
- 볼륨 꺼내기

Tape Libraries & Drives

테이프 라이브러리 및 드라이브에 대해 다음 세부정보가 제공됩니다.

- 처리를 중지하거나 시작할 수 있는 라이브러리 작업
- 다음 항목을 볼 수 있는 기능 제공:
  - 라이브러리 서버의 물리적 상태
  - 지난 1시간 동안의 마운트와 지난 24시간 동안의 마운트를 그래픽으로 보여주는 현재 작업
  - 다음을 포함하는 물리적 구성 요소:
  - 총 ACS 수
  - 용량
  - 총 볼륨
  - 유형별 총 볼륨
  - 총 사용 가능한 셀
  - 셀 사용에 대한 그래픽 보기
  - 총 LSM
  - 유형별 총 LSM
  - 총 드라이브
  - 유형별 총 드라이브
  - 총 CAP 수
  - 논리적 라이브러리 목록

ACSLS 8.4 GUI에 포함된 시스템 작업

대부분의 설치 유틸리티, 진단 및 기타 셸 명령 유틸리티는 ACSLS GUI 콘솔의 범위를 벗어납니다. ACSLS GUI 릴리스 8.4에서 구현되지 않은 기존 *cmd_proc*에는 다음과 같은 일부 작업이 포함되어 있습니다.

- 풀 정의, 삭제, 질의 또는 표시
- 패널 표시
- 서버 유휴 상태 전환 또는 시작

- 볼륨 잠금 또는 잠금 해제
- 드라이브 잠금 또는 잠금 해제
- 스크래치 볼륨 마운트
- LMU 질의 또는 전환
- 질의 마운트 수행
- 스크래치 볼륨 설정
- 잠금 설정 또는 지우기
- 포트 전환, 질의 또는 표시
- venter

방화벽 및 GUI

ACSLS 서버에서 방화벽 보호를 사용으로 설정한 경우 방화벽 소프트웨어를 사용하여 정책을 명시적으로 구성하지 않으면 ACSLS GUI에 대한 원격 사용자 액세스에 영향을 줍니다.

Solaris

ipf 및 *ipfilter* 매뉴얼 페이지를 참조하십시오. *root*가 다음 명령을 사용하여 *ipfilter* 방화벽을 사용 또는 사용 안함으로 설정합니다.

svcadm enable ipfilter (svcadm disable ipfilter)

- ipfilter의 현재 상태를 확인하려면 다음을 수행합니다.

  svcs ipfilter

방화벽 정책은 */etc/ipf/ipf.conf* 파일에서 정의할 수 있습니다. 로컬 호스트에서 구성 요소 간(예: ACSLS와 WebLogic 간)에 자유로운 통신을 허용하려면 아래 예와 같이 명령문을 포함합니다.

```
pass in quick from 127.0.0.1 to 127.0.0.1, or
pass in quick from 127.0.0.1 to all
```

- 원격 웹 기반 브라우저에서 ACSLS GUI에 액세스하도록 허용하는 정책을 제공하려면 *7001* 및 *7002* 포트를 열어야 합니다.

  ```
  pass in quick from any to any port = 7001
  pass in quick from any to any port = 7002
  ```

ACSLS ACSAPI 클라이언트의 경우 ACSLS에서 사용 중인 포트를 검색해야 합니다. UNIX 셸에서 다음 명령을 사용하십시오.

```
rpcinfo -p | egrep "300031 | 536871166"
```

화면의 마지막 필드에 포트 ID가 나열됩니다. 각각에 'pass in quick' 문을 추가해야 합니다. RPC portmapper 포트 111에 대해 'pass in quick' 문을 포함해야 할 수도 있습니다.

ACSLS의 방화벽에 대한 자세한 내용은 부록 M. 방화벽 보안 옵션을 참조하십시오.

제안된 규칙 세트의 마지막 명령문인 *block in from any*는 이전 명령문에서 특별히 허용되지 않는 한 트래픽이 호스트에 도달할 수 없는 것으로 간주합니다.

Linux

iptables 매뉴얼 페이지를 참조하십시오. *root* 사용자가 다음 명령을 사용하여 iptables 방화벽을 사용 또는 사용 안함으로 설정합니다.

service iptables start (service iptables stop)

- iptables의 상태를 확인하려면 다음을 수행합니다.

  service iptables status

iptables에 대한 정책 파일은 */etc/sysconfig/iptables*입니다. ACSLS GUI에 원격 http/https 액세스를 허용하는 정책을 포함하려면 아래 예와 같이 명령문을 사용하여 7001 및 7002 포트에 대한 예외가 포함되도록 파일을 업데이트해야 합니다.

```
-A input -p tcp --dport 7001 -j ACCEPT
-A input -p tcp --dport 7002 -j ACCEPT
```

iptables에 대한 자세한 내용은 "GUI가 작동 중인지 확인"을 참조하십시오.

ACSLS ACSAPI 클라이언트의 경우 ACSLS에서 사용 중인 포트를 검색해야 합니다. Linux 셸에서 다음 명령을 사용하십시오.

rpcinfo -p | egrep "300031 | 536871166"

화면의 마지막 필드에 포트 ID가 나열됩니다. iptables 정책에서 각 포트에 대한 예외를 추가해야 합니다. RPC portmapper 포트 111에 대해 예외 명령문을 제공해야 할 수도 있습니다.

ACSLS의 방화벽에 대한 자세한 내용은 부록 M. 방화벽 보안 옵션을 참조하십시오.

HTTPS에 자체 지정 디지털 인증서 구성

ACSLS 서버에 WebLogic을 설치하는 경우 클라이언트 브라우저와의 기본 https 교환을 지원하기 위한 단순 512비트 공개 키를 자동으로 사용할 수 있게 됩니다. 일반적으로 추가 구성은 필요하지 않습니다. 하지만 일부 브라우저의 경우 특히 Microsoft Internet Explorer의 경우 1024비트 미만의 키가 필요합니다.

Internet Explorer 및 FireFox 버전 39 이상에서는 타사 디지털 서명 기관에서 확인한 인증서를 사용하지 않는 https 서버에서 사용하기 위한 WebLogic 설정 절차가 필요합니다.

절차는 *ACSLS 8.4 Installation Guide*의 "Configuring a Self-Assigned Digital Certificate for HTTPS"를 참조하십시오.

5

5장. 라이브러리 하드웨어 설치 및 구성

라이브러리 하드웨어를 설치 및 구성하는 작업은 다음과 같습니다.

- “라이브러리 하드웨어에 대한 연결 설치 ”

  라이브러리 하드웨어에 대한 연결 설치 수행:
  - SCSI mchanger 장치 드라이버 추가
  - SL8500 또는 SL3000에 대해 이중 TCP/IP 지원을 선택적으로 사용
- “acsss_config를 사용하여 라이브러리 하드웨어 구성 ”

  수행 가능한 작업:
  - 라이브러리 통신 설정
  - 지원할 라이브러리 수 구성 및 분할 여부 확인
  - 라이브러리 연결이 SCS/광 섬유, TCP/IP, 직렬 또는 SCSI/광 섬유인지 여부 및 사용할 형식 설정
  - 라이브러리 하드웨어 재구성

> **주:**
>
> 지원되는 라이브러리, 드라이브 유형, 매체 유형 및 드라이브와 매체 간 호환성에 대한 최신 목록은 *ACSLS Product Information Guide*를 참조하십시오.

라이브러리 하드웨어에 대한 연결 설치

이 절에서는 라이브러리 하드웨어에 대한 연결 설치에 대해 설명합니다.

SCSI mchanger 장치 드라이버 추가

SCSI 매체 교환기(mchanger)는 Solaris 운영체제에서 ACSLS와 SCSI 또는 광 섬유 연결 라이브러리 간에 통신하는 장치 드라이버입니다. Linux의 ACSLS에서는 고유 sg 드라이버를 사용합니다.

또한 Solaris와 Linux 모두에 대해 ACSLS는 ACSLS에 대한 라이브러리 하드웨어를 구성할 때 지정하는 `/dev/mchanger` 링크를 만듭니다.

SCSI 라이브러리에 대한 지원은 설치 프로세스 중에 선택적으로 추가할 수 있습니다. 하지만 기존 ACSLS 설치에 새 SCSI 라이브러리를 추가해야 하는 경우도 있거나 SCSI 지원만 추가하면 되는 경우도 있습니다. 언제든지 필요한 드라이버를 설치하고 mchanger 링크를 만들 수 있습니다.

1. ACSLS 서버에 *root* 사용자로 로그인하고 암호를 입력합니다.
2. 설치 디렉토리로 이동합니다.

   ```
   cd $ACS_HOME/install
   ```
3. SCSI 라이브러리가 작동하고 ACSLS 서버에 물리적으로 연결되는지 확인합니다.

   주:

   (Solaris) SL500 라이브러리에 대해 다중 경로 하드웨어를 의도적으로 구성하는 경우를 제외하고, 라이브러리 연결 경로에 사용되는 상위 장치 드라이버(예: "fp" 드라이버)에 대해 다중 경로 I/O를 사용 안함으로 설정해야 합니다. SL500에 성공적으로 연결할 수 없는 경우 *<driver>.conf* 파일(일반적으로 */kernel/drv/fp.conf*)에서 다중 경로 I/O가 사용 안함으로 설정되어 있는지 확인해야 합니다.

   ```
   mpxio-disable="yes"
   ```
4. 적절한 드라이버 설치 스크립트를 호출합니다.
   - Solaris

     ```
     ./install_scsi_sol.sh
     ```
   - Linux

     ```
     ./install_scsi_Linux.sh
     ```

SL8500 또는 SL3000 이중 TCP/IP 지원 사용

SL8500 또는 SL3000 라이브러리를 설치했으며 이중 TCP/IP 지원을 사용으로 설정하려면 “이중 TCP/IP 지원 ”을 참조하십시오. 또한 이 절에서는 만들어야 하는 사용자 정의 경로 지정 테이블 항목에 대해 설명합니다.

라이브러리 하드웨어 구성 또는 재구성

다음과 같은 두 가지 방법으로 라이브러리 하드웨어(신규 또는 변경된 라이브러리 및/또는 테이프 드라이브)를 구성 또는 재구성할 수 있습니다.

- *acsss_config*

  이 명령은 ACSLS 작동이 중지된 상태에서 실행해야 합니다. 다음의 경우 *acsss_config*를 사용합니다.
  - 라이브러리 하드웨어 초기(처음) 구성
  - ACS 제거
  - 라이브러리에 대한 포트 연결 변경 또는 제거
  - 광 섬유 연결 또는 SCSI 연결 라이브러리 재구성
- 동적 구성(*config*) 유틸리티

  ACSLS가 실행 중인 동안 이 유틸리티를 실행합니다. 위의 경우를 제외하고 모든 변경에 대해 *config*를 사용합니다. *config* 유틸리티로 다음을 수행합니다.

- 새 라이브러리 추가
- TCP/IP 연결 라이브러리 구성 업데이트
- 포트 연결, LSM, CAP 및 드라이브 추가

acsss_config를 사용하여 라이브러리 하드웨어 구성

*acsss_config*를 사용하여 라이브러리 하드웨어를 구성 또는 재구성할 수 있습니다.

- 각 ACS에 하나 이상의 CAP가 있어야 합니다. 다른 분할 영역과 공유되는 CAP일 수 있습니다.
- 전체 ACSLS 시스템에 대해 하나 이상의 드라이브가 구성되어 있어야 합니다.

  예를 들어, ACSLS에서 4개의 라이브러리를 지원하는 경우 이 중 3개의 라이브러리에는 드라이브가 없을 수 있습니다. 하지만 네번째 라이브러리에는 하나 이상의 드라이브가 포함되어야 합니다.
- 라이브러리 하드웨어를 구성합니다.

  라이브러리 하드웨어를 처음으로 구성하려면 *acsss_config* 유틸리티를 사용해야 합니다. *acsss_config* 유틸리티는 메뉴 방식이며, 옵션 8을 선택하여 하드웨어 처음 구성, 라이브러리 삭제, 라이브러리에 대한 포트 연결 변경 등과 같은 작업을 수행할 수 있습니다.

  주:

  라이브러리를 구성하거나 재구성할 때 ACSLS에서 ACS 번호를 건너뛸 수 있습니다(옵션 6 참조). *acsss_config*와 동적 구성 모두 ACS 번호 지정 및 건너뛰기를 지원합니다. 예를 들어, 9310의 ACS 0에서 SL8500의 ACS 1로 마이그레이션한 후 나머지 SL8500 ACS 1에서 드라이브와 볼륨 ID의 번호를 다시 지정하지 않고 ACS 0을 제거할 수 있습니다.
- 라이브러리 하드웨어 재구성

  *acsss_config*를 사용하여 모든 라이브러리에 대한 구성 정보를 업데이트하기 전에 현재 구성을 기록합니다. 현재 라이브러리에 지정된 ACS 번호를 변경하면 모든 볼륨이 존재하지 않는 것으로 표시되고 모든 드라이브의 주소가 변경됩니다. *query lmu all* 명령의 출력을 저장하여 현재 ACS 번호와 포트 연결을 기록합니다.

  주:

  기존의 물리적 ACS에 대한 논리적 라이브러리가 클라이언트에 연결된 상태에서 *acsss_config*를 사용하여 해당 ACS를 구성에서 제거할 경우 SMCE 부속 시스템은 유지 관리 상태로 전환될 수 있습니다.

  이러한 상황을 방지하려면 *acsss_config*를 사용하여 라이브러리 구성에서 ACS를 제거하기 전에 ACSLS GUI 또는 *lib_cmd* CLI(ACSLS 8.2 이상 릴리스)를 사용하여 연관된 논리적 라이브러리를 삭제해야 합니다. 연관된 라이브러리를 먼저 삭제하면 모든 관련 정보가 올바르게 제거됩니다.

  초기 라이브러리 구성 후에 *config* 유틸리티를 사용하여 ACSLS를 중지하지 않고 ACS, LSM 또는 테이프 드라이브를 동적으로 추가하거나 재구성할 수 있습니다.

  자세한 내용과 절차는 “acsss 매크로 ”를 참조하십시오.

주:

라이브러리의 물리적 구성이 변경될 때마다 *acsss_config* 또는 *config*를 사용하여 ACSLS 데이터베이스를 업데이트해야 합니다. 데이터베이스에 정의된 구성이 라이브러리에 정의된 구성과 일치하지 않을 경우 ACSLS가 올바르게 실행되지 않습니다.

예:

데이터베이스를 업데이트해야 하는 구성 변경은 다음과 같습니다.

- ACS, LSM(SCSI 연결 LSM(예: SL500) 포함), PTP(전달 포트), 전송, 대기 LMU 추가 또는 제거

주의:

새 ACS 또는 LSM을 설치한 후 ACSLS를 재구성하기 전에 각 연결 라이브러리와 LSM이 완전히(하드웨어) 구성되고, 전원이 켜지고, 준비가 되어 있는지 확인하십시오. 그렇지 않으면 *acsss_config* 또는 *config*를 통해 라이브러리가 올바르게 구성되지 않습니다.

주:

ACSLS HA가 추가된 라이브러리를 모니터하도록 하려면 *ha_acs_list.txt* 파일에 항목을 작성해야 합니다. 자세한 내용은 “ACSLS HA에 대한 정보 ”를 참조하십시오.

- 서버 시스템과 LMU 사이의 포트 연결 추가 또는 제거

*acsss_config*를 시작하려면 다음을 수행합니다.

1. *acsss*로 로그인하십시오.
2. 라이브러리 하드웨어를 재구성하려면 다음 단계를 따릅니다.

   ACSLS를 중지하기 전에 *query lmu all cmd_proc* 명령을 사용하여 기존 ACS 번호, 분할 영역 ID 및 포트 연결을 표시하고 이 정보를 기록합니다.
3. ACSLS(실행 중인 경우)를 종료합니다.
4. 다음 구성 유틸리티를 실행합니다.

   acsss_config

   ACSLS 기능 구성 화면이 나타납니다.

주:

모든 라이브러리, LSM 및 전송이 완전히 구성되고, 전원이 켜져 있고, 준비되어 있다면 *acsss_config* 구성 유틸리티는 라이브러리를 올바르게 구성합니다.

다음 메뉴가 나타납니다.

```
ACSLS feature configuration
Please enter the number followed by Return for your choice from the following menu
 to configure product behavior in that area.
Press? followed by the Return key for help.
```

```
1: Set CSI tuning variables
2: Set event logging variables
3: Set general product behavior variables
4: Set access control variables
5: Set automatic backup parameters
6: Rebuild Access Control information
7: Event Notification settings
8: Define or Change Library Hardware Configuration
E: Exit
Menu choice:
```

주:

옵션 1-7에 대한 설명은 6장. *ACSLS* 동작을 제어하는 변수 설정을 참조하십시오.

5. 옵션 8을 선택합니다.

   다음 프롬프트는 구성을 설정하는 과정을 안내합니다.

   - 프롬프트: *`Configure library communications? (y/n):`*

     ACSLS 서버와 라이브러리 간의 통신을 설정하거나 업데이트하려면 *y*를 입력합니다. 여기에는 라이브러리에 대한 포트 연결 추가, 삭제 또는 변경과 라이브러리 분할 영역 추가, 삭제 또는 변경이 포함됩니다.

     *n*을 선택하면 현재 라이브러리의 구성을 새로 고칩니다.

     ACS를 추가 또는 제거하거나 라이브러리에 대한 연결을 추가, 변경 또는 제거할 필요가 없는 경우 라이브러리 통신을 구성하지 않고 하드웨어 구성을 새로 고치는 것이 가장 쉽고 적합한 옵션입니다. 라이브러리에 대한 모든 현재 연결 재정의를 건너뛰고 ACSLS 데이터베이스에 기록된 라이브러리 구성만 새로 고칩니다. 그러면 라이브러리에 대한 연결이 누락되거나 잘못 지정될 위험이 없습니다.

   - 프롬프트: *`Library server database exists and will be overwritten, continue? (y or n): y`*

     이 프롬프트는 라이브러리 통신이 이미 설정되어 있고 기존 데이터베이스가 있는 경우에만 표시됩니다.

     이 프롬프트에 *y*를 입력하면 변경사항이 구성에 적용됩니다.

6. 지원할 ACS 수를 지정합니다.

   - 프롬프트: *`Number of ACSs to be supported:`*

     사이트에서 지원할 ACS 수(1 ~ 32)를 입력합니다. ACS가 "1"개 이상 있어야 합니다. 하나 이상의 ACS를 지원하려면 하드웨어가 설치되어 있어야 합니다.

     주:

     전달 포트로 연결된 L700e 라이브러리 쌍은 하나의 ACS로 계산됩니다.

라이브러리를 구성 또는 재구성할 때 ACS 번호를 순서대로 지정하지 않고 ACS 번호 지정을 건너뛸 수 있습니다.

7. 각 *ACS* 번호를 지정합니다*.*
   - 프롬프트: *Please enter the first ACS number [default: 0]:*

     이 ACS에 대한 번호를 입력합니다.

     주의:

     기존 ACS를 재구성할 경우 현재 지정된 것과 동일한 ACS 번호를 지정하십시오.

     그러면 각 ACS에 대한 장치 연결을 정의하라는 프롬프트가 표시됩니다.
8. 라이브러리 분할 영역을 지정하고 정의합니다.
   - 프롬프트: *Is ACS #n in a partitioned SL8500 or SL3000? (y or n)*
     - *y*를 입력하면 ACS에 대한 분할 영역 ID를 묻는 메시지가 표시됩니다.

       이 분할 영역 ID는 SLConsole의 분할 영역 ID와 일치해야 합니다.
     - 분할된 라이브러리가 아니거나 SCSI/광 섬유 연결 라이브러리인 경우 *n*을 입력합니다.

     ACSLS는 분할된 SCSI/광 섬유 연결 라이브러리(예: SL500)를 지원하지 않습니다. 또한 분할된 SCSI/광 섬유 연결 라이브러리에는 분할 영역 ID가 없습니다.
9. 각 ACS에 대한 장치 연결(포트)을 정의합니다.
   - 프롬프트: *Number of connections to ACS #n*

     이 프롬프트는 구성된 각 ACS에 대해 나타나며 각 ACS에 대한 통신 포트 수를 설정합니다. 연결은 다음과 같습니다.
     - SCSI 연결

       주:

       전달 포트로 연결된 L700e SCSI 라이브러리 쌍을 포함하는 L1400에는 각 L700e에 하나씩 두 개의 연결이 있습니다.
     - ACSLS와 라이브러리 사이의 직렬 또는 TCP/IP 연결. 두 개 이상의 연결이 권장됩니다.

       여러 SL8500 라이브러리에 연결하려는 경우 최대 15개의 연결이 허용됩니다.

     ACS를 물리적으로 연결해야 합니다. 1 - 15 사이의 10진수를 입력합니다.
   - 프롬프트: *Device or host - ACS#n, connection #n:*

     각 연결에 대한 장치 또는 호스트를 입력합니다.

     주의:

     올바른 호스트 이름 또는 IP 주소를 지정해야 합니다. 잘못된 라이브러리에 연결하지 마십시오.

ACS에 대해 지정된 모든 연결이 실제로 동일한 ACS에 연결되는지 확인합니다.

TCP/IP로 연결된 LMU에 대한 연결을 지정하려면 다음 중 하나를 입력합니다.

- IP 주소
- 호스트 이름
- 전체 호스트 이름

> **주:**
>
> 호스트 이름이 지정되면 이 호스트 이름은 SL8500 또는 SL3000 라이브러리나 9330 LMU에 입력된 IP 주소에 매핑되어야 합니다. 이 호스트 이름-IP 주소 매핑은 사이트에만 한정됩니다. 일반적으로 이 작업은 */etc/hosts* 파일, DNS(도메인 이름 서버), NIS 또는 NIS+를 통해 수행됩니다.

예:

SL8500, SL3000 또는 9300 라이브러리에 대한 샘플 TCP/IP 장치 이름

```
Device or host - ACS #0, connection #0: hostname1
Device or host - ACS #0, connection #1: hostname2
Device or host - ACS #1, connection #0: fully_qualified_hostname
Device or host - ACS #2, connection #0: 192.168.174.31
```

> **주:**
>
> ACSLS에서는 SL3000에 대한 TCP/IP 연결만 지원하고, 광 섬유 연결 SL3000은 지원하지 않습니다. 9330 ACS에 대한 연결은 직렬 또는 TCP/IP 연결이거나 둘 다일 수 있습니다.

예:

4400 또는 9300 라이브러리에 대한 샘플 직렬 장치 이름

```
Device or host - ACS #0, device #0: /dev/ttya
Device or host - ACS #0, device #1: /dev/ttyb
```

광 섬유 또는 SCSI 연결 라이브러리는 mchanger 장치를 통해 연결됩니다.

예:

SCSI 라이브러리에 대한 샘플 장치 이름

```
Device or host - ACS #1, connection #0: /dev/mchanger2
```

전달 포트로 연결된 L700e SCSI 라이브러리 쌍을 포함하는 L1400에는 각 L700e에 하나씩 두 개의 연결이 있습니다.

예:

L700e 라이브러리 쌍에 대한 샘플 장치 이름

```
Device or host - ACS #1, connection #0: /dev/mchanger2
Device or host - ACS #1, connection #1: /dev/mchanger3
```

정의할 ACS가 더 있는 경우 ACS 번호를 지정하라는 메시지가 표시됩니다(6단계 참조).

10. ACSLS 데이터베이스에서 구성을 만들거나 업데이트합니다.
    - 프롬프트: *`This step builds a database image of your complete library hardware configuration. Before proceeding, make sure that your library is completely configured, that all subsystems are functional and powered on. Build/Verify library configuration? (y or n):`*

      *y*를 입력합니다.

      계속하기 전에 라이브러리가 완전히 구성되고, 모든 부속 시스템이 작동하고 전원이 켜져 있는지 확인합니다.

      이 단계에서는 포트 연결 추가 또는 제거를 제외하고 모든 구성 변경사항으로 데이터베이스를 업데이트합니다.
    - 프롬프트: *`Library server database exists and will be overwritten, continue? (y or n):`*

      *y*를 입력합니다. 스크립트에서 라이브러리 구성을 작성할 때 각 LSM에 각 패널에 대한 다음 메시지가 표시됩니다.

      `ACS # n, LSM # nn, PANEL # nn, created`

      또한 이 스크립트는 라이브러리 구성 보고서를 생성하여 다음 파일에 첨부합니다.

      `$ACS_HOME/log/acsss_config.log`
11. 선택적으로 Solaris에서 TCP/IP가 아닌 클라이언트를 구성합니다.
    - 프롬프트: *`Configure client system interfaces? (y or n):`*

      클라이언트 시스템 인터페이스를 구성할지 여부를 묻는 메시지가 표시되면 (*y* 또는 *n*)으로 응답합니다.

      ICL 클라이언트를 OSLAN 프로토콜과 함께 사용하지 않을 경우 *n*을 입력합니다.

      *y*를 입력하면 다음 메시지가 표시됩니다.

      ```
      CSI SELECTION TABLE
      ```

```
1) OSLAN CSI Not Selected
2) ONC/RPC CSI Always Selected

Do you want to change the CSI selection table (n):
```

*y*를 입력하면 다음 메시지가 표시됩니다.

Select OSLAN CSI (n):

ONC/RPC는 항상 선택됩니다.

주:

*acsss_config*는 데이터베이스를 자동으로 백업한 다음 종료됩니다.

12. *acsss enable*을 실행하여 서버를 시작합니다.

    이벤트 로그에서 모든 것이 작동 및 실행되고 있는지 확인할 수 있습니다.

    서버 시스템이 재부트되면 ACSLS가 자동으로 시작됩니다.
13. 라이브러리 감사를 수행합니다.

    감사에서 ACSLS 데이터베이스를 라이브러리 카트리지의 실제 인벤토리와 일치하도록 업데이트합니다.

    이제 ACSLS가 라이브러리 작업을 수행할 준비가 되었습니다. ACSLS 명령을 입력하려면 *acssa*로 로그인합니다.

ACS 번호 다시 지정

기존 ACS의 번호를 변경해야 하거나 변경하려면 “acs_renumber.sh ”를 참조하십시오.

6

6장. ACSLS 동작을 제어하는 변수 설정

ACSLS에는 정적 변수 및 동적 변수가 모두 있습니다.

- 정적 변수

  정적 변수는 변수를 설명하고 새 값을 설정할 수 있는 프롬프트의 문장으로 알 수 있습니다.

  예: *Changes to <variable_name> will not take effect until product is restarted.*

  > **주:**
  >
  > ACSLS 정적 변수를 변경한 후 새 값을 사용하려면 ACSLS를 중지하고 다시 시작해야 합니다.

- 동적 변수

  ACSLS 동적 변수의 변경사항은 변수가 ACSLS 작업 중에 다음에 참조될 때 적용됩니다.

ACSLS 변수를 표시하고 업데이트하는 간편한 방법:

- 모든 ACSLS 변수의 현재 설정 표시(정적 및 동적):

  dv_config -d
  - 현재 변수 설정을 파일에 저장:

    dv_config -d > filename
  - 파일 보기:

    vi filename
- 동적 옵션 값 출력:

  dv_print
- 변수를 설명하고 설정을 변경할 수 있게 하는 프롬프트와 함께 하나의 변수의 현재 설정 표시:

  dv_config –p <variable_name> -u

  설명:
  - *-p*

    프롬프트를 통해 변수에 대한 새 값을 지정할 수 있습니다.

- *<variable_name>*

  여기에 원하는 변수 이름을 삽입합니다.

- *-u*

  변수를 변경한 경우 공유 메모리에 동적 변수 값을 업데이트합니다. *-u* 옵션은 정적 변수에서 사용되지 않습니다.

  주:

  변수에 관한 자세한 설명을 보려면 프롬프트에 물음표(?)를 입력하십시오. 프롬프트가 변수를 변경할 수 있도록 새로 고쳐집니다.

또한, *acsss_config* 메뉴를 사용하여 ACSLS 동작을 제어하는 변수를 설정할 수 있습니다. 옵션 1~8로 다음을 수행합니다.

- "CSI 조정 변수 설정"
- "이벤트 로깅 변수 설정"
- "일반 제품 동작 변수 설정"
- "액세스 제어 변수 설정"
- "자동 백업 변수 설정"
- "액세스 제어 정보 재구성"
- "이벤트 알림 설정 정의"
- "라이브러리 하드웨어 구성 또는 업데이트"
- "시스템 이벤트의 전자 메일 알림 등록"

ACSLS 기능 구성 메뉴 액세스

ACSLS를 설치하거나 업그레이드할 때 대부분의 사용자 환경을 기반으로 하는 시스템 기본값이 이미 설정되어 있습니다. 그러나 필요한 경우 옵션 1~8을 사용하여 이러한 설정을 변경할 수 있습니다. 옵션 1~7을 통해 동적 및 정적 변수를 변경할 수 있습니다. 정적 변수를 적용하려면 ACSLS를 다시 시작해야 합니다.

라이브러리 하드웨어를 추가 및 정의하려면 옵션 8을 사용하십시오. 처음으로 ACSLS를 이제 막 설치한 경우 즉, 예를 들어, 새 라이브러리를 추가해야 할 때 이 옵션을 사용하십시오.

각 프롬프트에 대한 도움말을 보려면 *?*를 선택하십시오.

*acsss_config*를 시작하려면:

1. CDE 로그인에 도달할 때까지 CDE를 종료합니다.
2. *acsss*로 로그인하십시오.
3. 구성 스크립트를 실행합니다.

   acsss_config

   ACSLS 기능 구성 화면이 나타납니다.

모든 LMU, LSM 및 이동이 완전히 구성되고 켜져 있고 준비되어 있지 않으면 구성 스크립트 *acsss_config*가 실패합니다.

다음 메뉴가 나타납니다.

```
ACSLS Feature Configuration

Enter the number followed by Return for your choice from the following menu to
 configure product behavior in that area.

Press ? followed by the Return key for help.

1: Set CSI tuning variables
2: Set event logging variables
3: Set general product behavior variables
4: Set access control variables
5: Set automatic backup parameters
6: Rebuild Access Control information
7: Event Notification settings
8: Define or Change Library Configuration

E: Exit

Menu choice:
```

동적 및 정적 변수 확인 및 변경

옵션 1~8을 통해 다음 동적 및 정적 변수를 변경할 수 있습니다.

CSI 조정 변수 설정

CSI(클라이언트 시스템 인터페이스)는 ACSLS와 다른 서버의 클라이언트 사이의 통신을 처리합니다. CSI에서는 각 클라이언트에 대한 통신 처리 방법을 설정합니다. 클라이언트와의 통신이 끊어질 경우 다른 클라이언트는 영향을 받지 않고 중단 없이 통신이 계속됩니다. 다중 CSI는 ACSLS에서 실행할 수 있습니다.

각 프롬프트에 대한 도움말을 보려면 *?*를 선택하십시오.

옵션 1을 사용하면 다음을 설정 또는 변경할 수 있습니다.

- *CSI_CONNECT_AGETIME*

  프롬프트: *CSI* 요청 대기열에서 보류 중인 요청의 최대 기간*(초) [172800]*

  동적 변수, 이 설정에서는 ACSLS가 응답하지 않은 클라이언트 요청에 대한 보류 기간을 결정합니다.

유효한 입력은 600~315360000초입니다. 기본값은 172800초입니다.

- *CSI_RETRY_TIMEOUT*

  프롬프트: 연속하는 재시도 간격*(*초*) [4]*

  기본값은 4초입니다.

  동적 변수, 이 옵션에서는 CSI가 네트워크 연결 설정 시도 사이에 기다려야 하는 최소 시간(초)을 지정합니다.

  CSC와 CSI 사이에 타이밍 문제가 발생하는 경우 이 값을 수정해야 합니다.

- *CSI_RETRY_TRIES*

  프롬프트: 시간 초과 현상 발생 전 *CSI* 재시도 횟수 *[5]*.

  동적 변수, 이 옵션에서는 CSI가 메시지를 전송하기 위해 시도해야 할 횟수를 지정합니다. 지정된 재시도 횟수 안에 연결을 설정할 수 없는 경우 보류 중인 메시지가 삭제됩니다. 기본 재시도 횟수는 5회입니다.

- *CSI_TCP_RPCSERVICE*

  프롬프트: *TCP* 프로토콜 사용을 바꾸는 변경사항은 제품을 다시 시작해야 적용됩니다*. TCP* 프로토콜을 사용하는 *RPC*에 대한 *CSI* 지원이 사용으로 설정되어 있습니다*[TRUE].*

  정적 옵션, 이 옵션에서는 CSI가 TCP RPC 서버로 동작할 것인지 지정합니다. 기본값은 true입니다.

- *CSI_UDP_RPCSERVICE*

  프롬프트: *UDP* 프로토콜 사용을 바꾸는 변경사항은 제품을 다시 시작해야 적용됩니다*. UDP* 프로토콜을 사용하는 *RPC*에 대한 *CSI* 지원이 사용으로 설정되어 있습니다 *[TRUE].*

  정적 옵션, 이 옵션에서는 CSI가 UDP RPC 서버로 동작할 것인지 지정합니다. UDP를 통해 ACSLS와 통신하는 클라이언트에 대한 기본값을 수락해야 합니다. 기본값은 TRUE입니다.

  이 옵션을 적용하려면 ACSLS를 다시 시작해야 합니다.

- *CSI_MULTI_HOMED_CL*

  프롬프트: 서버 플랫폼의 *CSI*에서 다중 홈 클라이언트*(*즉*, IP* 주소가 둘 이상인 시스템의 클라이언트*)*의 요청 패킷을 처리하도록 설정합니다*.* 또한*, NAT(*네트워크 주소 변환*)* 또는 *VPN*에서 수정된 *IP* 주소에 응답합니다*. (TRUE/FALSE) [FALSE]*

  이 옵션을 사용하면 서버의 CSI에서 패킷 헤더의 IP 주소가 패킷을 보낸 IP 주소와 동일하지 않은 들어오는 패킷을 처리할 수 있습니다. 클라이언트가 ACSLS에 보낸 요청 패킷의 주소는 클라이언트 응용 프로그램에서 호출한 'hostname'에서 파생된 후 *gethostbyname*이 조회됩니다. 이로 인해 로컬 호스트 IP 주소가 요청 패킷에 입력됩니다.

일반적으로, 이 옵션은 작동합니다. 하지만, 예외가 있습니다.

- 대부분의 경우 ACSLS에 대한 클라이언트에는 고정된 단일 호스트 주소가 있습니다. 그러나, 클라이언트 시스템에 둘 이상의 네트워크 인터페이스가 있어서 ACSLS에 요청을 제출하는 데 모든 다중 IP 포트를 사용할 수 있습니다.
- 클라이언트 시스템은 외부에서 액세스할 수 없는 내부 개인 IP 주소를 가지고 NAT 뒤에 있을 수 있습니다.
- 클라이언트가 VPN을 통해 ACSLS 서버에 연결된 경우 클라이언트의 IP 주소는 ACSLS가 응답해야 하는 IP 주소가 아닙니다.

이러한 경우 ACSLS는 클라이언트 응용 프로그램에서 localhost 주소가 아닌 IP 주소에 응답해야 합니다. CSI_MULTI_HOMED_CL 변수는 이러한 상황을 처리하도록 설계되어 있습니다. 유효한 섹션은 다음과 같습니다.

- FALSE

  ACSLS 작업에 대한 기본 설정입니다. 이 설정을 하면 ACSLS에서 클라이언트에 대한 응답을 반환할 때 localhost의 IP 주소에 항상 응답합니다.
- TRUE

  ACSLS는 요청 패킷의 IP 주소를 무시합니다. 대신, ACSLS가 클라이언트에 응답할 때 ACSLS는 ACSAPI 요청 패킷과 연관된 들어오는 RPC 데이터그램의 IP 헤더에 있는 주소를 사용합니다. 이 주소는 요청을 제출한 IP 주소입니다. 액세스 제어가 사용으로 설정된 경우 이 대체 IP 주소가 internet.addresses 파일에 지정되어 있어야 합니다.

- *CSI_USE_PORTMAPPER*

  프롬프트: 포트 매퍼 사용을 바꾸는 변경사항은 제품을 다시 시작해야 적용됩니다. 포트 매퍼 사용으로 설정*: (ALWAYS / NEVER /IF_DUAL_LAN_NOT_ENABLED) [IF_DUAL_LAN_NOT_ENABLED].*

  기본값은 *IF_DUAL_LAN_NOT_ENABLED*입니다. 정적 옵션, 유효한 옵션은 다음과 같습니다.

  - ALWAYS - 포트 매퍼는 CSI가 클라이언트에 메시지를 보낼 수 없을 때 항상 조사받아야 합니다.
  - NEVER - 포트 매퍼는 CSI가 클라이언트에 메시지를 보낼 수 없을 때 조사받지 않아야 합니다. 클라이언트가 포트 매퍼를 지원하지 않는 경우 이 옵션을 선택하십시오.
  - *IF_DUAL_LAN_NOT_ENABLED* - 포트 매퍼는 이중 LAN 지원이 사용으로 설정되지 않은 경우에만 조사받아야 합니다. 이중 LAN 지원이 사용으로 설정된 경우 클라이언트가 포트 매퍼를 지원하지 않는 것으로 가정합니다. 이 옵션을 선택하면 제품 동작에 역방향 호환성을 제공합니다.

- *SURROGATE_PROCESSES*

  프롬프트: 시작해야 하는 *ACSSURR* 연속 프로세스의 수*[0]:*

  유효한 입력은 0 또는 1입니다. LM 게이트웨이가 설치되지 않은 경우 0을 입력하십시오.

- *SURROGATE_PORT*

프롬프트: *ACSLS Surrogate (ACSSURR)* 소켓이 게이트웨이 시스템의 요청을 수신하는 *TCP/IP* 포트 번호입니다*[50300*].

이 변수는 LM(라이브러리 관리) 게이트웨이에만 적용됩니다. 유효한 입력은 50300~99999입니다.

- *SURROGATE_TIMEOUT*

  프롬프트: *Surrogate/*게이트웨이 소켓에서 읽어야 할 데이터 패킷을 기다리는 시간*(초):*

  이 변수는 LM(라이브러리 관리) 게이트웨이에만 적용됩니다. 유효한 입력은 1~600입니다.

- *SURROGATE_QUEUE_AGE*

  프롬프트: 사용되지 않는 대기열 항목을 삭제하기 전에 기다리는 시간*(분)*입니다*[5]*.

  이 변수는 LM(라이브러리 관리) 게이트웨이에만 적용됩니다. 유효한 입력은 5~60입니다.

- *START_CSCI_PROCESS*

  프롬프트: *ACSLS*를 시작할 때 자동으로 *CSCI* 시작*(TRUE/FALSE) [FALSE]:*

  이 변수는 ACSLS를 시작하는 동안 CSCI 프로세스를 자동으로 시작할 것인지를 결정합니다. 기본값은 FALSE이며, CSCI는 ACSLS와 함께 시작되지 않음을 의미합니다. ACSLS와 함께 CSCI 프로세스를 시작하려면 이 변수를 TRUE로 설정하십시오.

- *CSI_FIREWALL_SECURE*

  프롬프트: *CSI*가 방화벽 뒤에서 사용되도록 설정*(*사용자 정의 인바운드 포트*) (TRUE/FALSE) [FALSE]:*

  동적 변수, 이 설정을 사용하면 ACSLS로 들어오는 요청에 대해 사용자 정의된 단일 포트의 방화벽 보안 정의를 사용 또는 사용 안함으로 설정할 수 있습니다. 값은 다음과 같습니다.

  - False - ACSLS 서버의 포트를 제한하지 않을 경우 지정합니다.
  - True - 이 값이 기본값입니다. ACSLS 서버가 안전한 방화벽 뒤에서 작동합니다.

- *CSI_INET_PORT*

  프롬프트: 들어오는 *ACSLS* 요청을 수신하기 위해 *CSI*에서 사용되는 포트 번호입니다 *[30031].*

  이 변수는 하나 이상의 클라이언트에서 들어오는 TCP 요청에 대해 CSI에서 사용되는 사용자 정의된 단일 포트를 지정합니다.

  이 변수는 *CSI_FIREWALL_SECURE*를 True로 설정하여 방화벽 보안 CSI를 사용으로 설정한 경우에만 사용됩니다.

  기본값: 30031

유효한 입력: 1024~65535(50003 제외)

- *CSI_FAILED_RESPONSE_LIMIT*

  프롬프트: *CSI*에서 요청에 대한 나머지 모든 응답을 삭제하기 전에 *CSI*가 클라이언트에 반환할 수 없는 감사, 넣기, 꺼내기 또는 이벤트 등록 요청에 대한 최근 응답 수*. [5]*

  대부분의 요청에 대해 CSI가 클라이언트에 하나의 응답을 보낼 수 없는 경우 해당 요청에 대한 나머지 모든 응답이 삭제됩니다. 이렇게 하면 응답을 수신할 수 없는 클라이언트에 응답을 보내려는 시도를 차단하여 CSI를 보호합니다.

  감사, 넣기, 꺼내기 또는 이벤트 등록 요청에 대해 CSI에서는 첫번째 실패 후 계속해서 요청에 대한 몇 가지 응답을 반환하려고 시도합니다. *CSI_FAILED_RESPONSE_LIMIT*는 CSI에서 요청에 대한 남은 모든 응답을 취소하기 전에 요청에 대한 응답 반환에 실패한 시도 횟수를 지정합니다. 이렇게 하면 통신 문제가 있거나 클라이언트가 더는 활성 상태가 아닌 경우 CSI 가용성을 보호합니다.

  클라이언트와 CSI 사이에 타이밍 문제가 발생하는 경우 이러한 값만 수정해야 합니다. 자세한 내용은 *ACSLS* 관리 안내서를 참조하십시오.

  1에서 9 사이의 값을 입력하십시오.

이벤트 로깅 변수 설정

각 프롬프트에 대한 도움말을 보려면 *?*를 선택하십시오.

옵션 2를 사용하면 다음 이벤트 로그를 설정 또는 변경할 수 있습니다.

- *EVENT_FILE_NUMBER*

  프롬프트: 보존할 이벤트 로그 파일 수 *[9]:*

  이 옵션은 보존할 추가 이벤트 로그 파일 수를 지정합니다. 이 옵션이 사용으로 설정되어 있고 현재 이벤트 로그 파일 크기가 임계값 크기에 도달하면 로그가 자동으로 다른 파일에 복사됩니다. 지정된 파일 수에 도달하면 가장 오래된 파일의 데이터가 겹쳐집니다.

  추가 이벤트 로그 파일을 보존하지 않으려면 **0**을 입력하십시오. 단일 이벤트 로그 파일은 수동으로 잘릴 때까지 커집니다. 이런 경우, 이벤트 로그 파일 크기가 임계값 크기에 도달하면 이벤트 로그 이름을 바꾸거나 이벤트 로그를 삭제할 때까지 원치 않는 "Event log full" 메시지가 주기적으로 표시됩니다. ACSLS가 설치 또는 업그레이드되는 경우 기본값은 구(9)입니다.

  이 기능을 사용으로 설정하고 보존할 로그 파일 수를 지정하려면 **1**에서 **9** 사이(범위 값 포함)의 수를 입력하십시오. 이벤트 로그가 임계값 크기에 도달하면 데이터가 *event0.log* 파일로 이동합니다. 다음번에 임계값 크기에 도달하면 *event0.log* 데이터가 event1.log로 이동하고 이벤트 로그 데이터가 *event0.log*로 이동합니다. 이 주기는 지정된 보존된 파일 수에 도달할 때, 즉 데이터가 가장 오래된 파일에서 삭제되는 시점까지 계속됩니다.

- *LOG_PATH*

프롬프트: 로깅 디렉토리 변경사항은 제품을 다시 시작해야 적용됩니다. 로깅 정보를 배치해야 할 디렉토리 위치 *[$ACS_HOME/log]:*

로그 파일을 배치할 디렉토리를 선택하십시오. 설치된 대로 ACSLS는 *$ACS_HOME/log* 디렉토리에 정보를 기록합니다. 일반적인 사용에서 이 변수의 값은 변경되지 않습니다. *$ACS_HOME*을 포함하는 파일 시스템에 디스크 공간 문제가 있는 경우 대체 경로를 지정할 수 있습니다. 지정된 경로가 절대 경로여야 합니다(*/* 또는 *$ACS_HOME*으로 시작되는 경로).

이 변수를 적용하려면 ACSLS 제품을 다시 시작해야 합니다.

- *LOG_SIZE*

  프롬프트: 최대 라이브러리 서버 이벤트 로그 크기*(KB) (=1000*바이트*).*

  이 옵션에서는 이벤트 로그의 임계값 크기(KB, 1,000바이트)를 지정합니다. 음수가 아닌 숫자를 입력하십시오. ACSLS가 설치 또는 업그레이드되는 경우 기본값은 500입니다.

  유효한 입력: 32~2147483

- *TIME_FORMAT*

  프롬프트: 모든 로그의 날짜/시간 형식 *[%Y-%m-%d%H:%M:%S]:*

  이 옵션에서는 이벤트 로그 및 추적 로그의 날짜 및 시간 정보를 출력하는 데 사용할 형식을 지정합니다. 형식은 C 언어 함수에서 사용된 것(*strftime*)과 동일합니다.

- *LM_RP_TRAIL*

  프롬프트: 라이브러리 관리자 요청 프로세스 감사 기능을 켜야 합니까*? [TRUE]:*

  TRUE로 설정하면 이 옵션을 통해 모든 ACSLS 요청의 감사 추적을 사용으로 설정합니다. 각 요청이 시작된 클라이언트 또는 사용자 인터페이스의 시간 기록 및 이름과 함께 각 요청이 기록됩니다. 결과 로그는 *rpTrail.log* 파일에 있습니다.

- *RP_TRAIL_LOG_SIZE*

  프롬프트: *rpTrail* 로그에 대한 최대 크기*(KB) [1000]:*

  이 옵션에서는 KB로 표현된 rpTrail 로그에 대한 임계값 크기를 지정합니다. 음수는 입력할 수 없습니다. ACSLS가 설치 또는 업그레이드되는 경우 1000KB(1MB)가 기본값입니다. *rpTrail.log*가 이 크기를 초과하면 이 로그 파일이 압축되고 아카이브되며 새 *rpTrail.log*가 후속 요청 감사를 위해 만들어집니다.

  10에서 10000 사이의 값을 입력하십시오.

- *RP_TRAIL_FILE_NUM*

  프롬프트: 보존할 *rpTrail* 아카이브 파일 수 *[10]:*

  이 옵션에서는 보존할 아카이브된 rpTrail 로그 파일 수를 지정합니다. 현재 *rpTrail.log* 크기가 임계값 크기를 초과하면 로그가 자동으로 압축되고 아카이브되며, 압축된 시점의

시간 기록이 포함됩니다. 아카이브된 파일은 "*log_archives*" 디렉토리에 저장됩니다. 지정된 아카이브된 로그 수에 도달하고 새 로그가 추가되는 경우 가장 오래된 파일이 제거됩니다.

1에서 999개 사이의 아카이브된 로그 파일을 보존할 수 있습니다.

- *RP_TRAIL_DIAG*

  프롬프트: 내부 요청 처리에 대한 진단 시간 기록을 기록합니까*? [FALSE]:*

  이 옵션에서는 감사 추적에 진단 정보를 포함할 것인지를 지정합니다.

  TRUE로 설정하면 ”QUEUED AT”, ”FORKED/WRITTEN TO AT” 및 ”FINAL RESPONSE SENT TO <recipient” AT”에 대한 시간 기록이 추적 로그에 포함됩니다.

- *XAPI_LOG_SIZE*

  프롬프트: *XAPI* 로그 크기 변경사항은 *XAPI* 서버가 다시 시작될 때까지 적용되지 않습니다*.* 최대 *XAPI* 로그 크기*(MB) (=1048576*바이트*) [20].*

  이 옵션에서는 MB로 표현된 XAPI 로그에 대한 임계값 크기를 지정합니다(여기서는 "1048576바이트"로 정의됨). 음수가 아닌 숫자를 입력하십시오. 이 옵션의 기본값은 20입니다.

  이 변수를 적용하려면 XAPI 서버를 다시 시작해야 합니다.

- *XAPI_LOG_FILE_NUM*

  프롬프트: 보존할 *XAPI* 로그 아카이브 파일 수 *[10].*

  이 옵션에서는 보존할 아카이브된 XAPI 로그 파일 수를 지정합니다. 현재 vlog.file 크기가 임계값 크기를 초과하면 해당 로그 파일은 0-n 접미사가 붙은 이름으로 바뀝니다. 0은 가장 최근 파일이며, n은 가장 오래된 파일입니다. 아카이브된 파일은 *API_WORK_PATH* 디렉토리에 저장됩니다. 지정된 아카이브된 로그 수에 도달하는 경우 새 파일이 아카이브 디렉토리에 추가될 때마다 가장 오래된 파일이 이 디렉토리에서 제거됩니다. 1개에서 99개 사이의 아카이브된 파일을 보존할 수 있습니다. 보존할 아카이브된 로그 파일 수를 지정하려면 1에서 99 사이의 숫자를 입력하십시오.

  이 변수를 적용하려면 XAPI 서버를 다시 시작해야 합니다.

- *XAPI_TRACE_SIZE*

  프롬프트: *XAPI* 추적 크기 변경사항은 *xapi* 서버를 다시 시작해야 적용됩니다*.* 최대 *XAPI* 추적 크기*(MB) (=1048576*바이트*) [50].*

  이 옵션은 XAPI 추적에 대한 임계값 크기(MB)를 지정합니다(여기서는 "1048576바이트"로 정의됨). 음수가 아닌 숫자를 입력하십시오. 이 옵션의 기본값은 50입니다.

  이 변수를 적용하려면 XAPI 서버를 다시 시작해야 합니다.

- *XAPI_TRACE_FILE_NUM*

프롬프트: 보존할 *XAPI* 추적 아카이브 파일 수.

이 옵션에서는 보존할 아카이브된 XAPI 추적 파일 수를 지정합니다. 현재 vtrace.file 크기가 임계값 크기를 초과하면 해당 추적 파일은 0-n 접미사가 붙은 이름으로 바뀝니다. 0은 가장 최근 파일이며, n은 가장 오래된 파일입니다. 아카이브된 파일은 *XAPI_WORK_PATH* 디렉토리에 저장됩니다. 지정된 아카이브된 로그 수에 도달하는 경우 새 파일이 아카이브 디렉토리에 추가될 때마다 가장 오래된 파일이 이 디렉토리에서 제거됩니다. 1개에서 99개 사이의 아카이브된 파일을 보존할 수 있습니다. 보존할 아카이브된 로그 파일 수를 지정하려면 1에서 99 사이의 숫자를 입력하십시오.

이 변수를 적용하려면 XAPI 서버를 다시 시작해야 합니다.

일반 제품 동작 변수 설정

각 프롬프트에 대한 도움말을 보려면 *?*를 선택하십시오.

옵션 3을 사용하면 다음을 설정 또는 변경할 수 있습니다.

- *LIB_VOL_STATS*

  프롬프트: 라이브러리 볼륨 통계 수집을 사용으로 설정*: (ON / OFF) [OFF]:*

  이 옵션에서는 넣기, 꺼내기, 마운트 해제, 마운트 및 감사 작업에서 LVSTATS(라이브러리 볼륨 통계) 정보를 수집할 것인지 지정합니다. LVSTATS 수집 및 보고 세션을 시작하려면 ON을 입력하십시오. LVSTATS 수집 및 보고 세션을 종료하려면 OFF를 입력하십시오. 세션이 시작되거나 종료되는 경우 해당하는 메시지가 *acsss_stats.log* 로그 파일에 출력됩니다. 옵션을 적절히 사용하면 위에 나열된 개별 작업에 대한 로그 파일 브래키팅이 수행됩니다.
- *VOL_STATS_FILE_NUM*

  프롬프트: 보존할 *acsss_stats* 로그 파일 수 *[9]:*

  이 옵션에서는 보존할 추가 *acsss_stats* 로그 파일 수를 지정합니다. 이 옵션이 사용으로 설정되어 있고 현재 acsss_stats 로그 파일 크기가 임계값 크기에 도달하면 로그가 자동으로 다른 파일에 복사됩니다. 지정된 파일 수에 도달하면 가장 오래된 파일의 데이터가 겹쳐집니다.

  추가 로그 파일을 보존하지 않으려면 0을 입력하십시오. 단일 acsss_stats 로그 파일은 수동으로 잘릴 때까지 커집니다. 이런 경우, acsss_stats 로그 파일이 임계값 크기에 도달하면 *acsss_stats* 로그 이름을 바꾸거나 이 로그를 삭제할 때까지 원치 않는 "*acsss_stats* log full" 메시지가 이벤트 로그에 주기적으로 표시됩니다. ACSLS가 설치 또는 업그레이드되는 경우 기본값은 구(9)입니다.

  보존할 로그 파일 수를 지정하려면 1에서 9 사이(범위 값 포함)의 숫자를 입력하십시오. acsss_stats 로그가 임계값 크기에 도달하면 데이터가 *vol_stats0.log* 파일로 이동합니다. 다음번에 임계값 크기에 도달하면 vol_stats0.log 데이터가 *vol_stats1.log*로 이동하고 acsss_stats 로그 데이터가 vol_stats0.log로 이동합니다. 이 주기는 지정된 보존

된 파일 수에 도달할 때, 즉 데이터가 가장 오래된 파일에서 삭제되는 시점까지 계속됩니다.

- *VOL_STATS_FILE_SIZE*

  프롬프트: 최대 라이브러리 서버 *access_stats* 로그 크기*(KB) (=1000*바이트*) [500]:*

  이 옵션에서는 KB로 표현되는 *acsss_stats* 로그에 대한 임계값 크기를 지정합니다(여기서는 "1000바이트"로 정의됨). 32 이상의 값을 입력하십시오. ACSLS가 설치 또는 업그레이드되는 경우 이 옵션의 크기에 대한 기본값은 500KB입니다.

  유효한 입력: 32~10000

- *UNIFORM_CLEAN_USE*

  프롬프트: 청소 카트리지 순서 지정 방법 선택 *[VOLID_SORT]:*

  유효한 옵션은 다음과 같습니다.

  - *VOLID_SORT* - 청소 카트리지 순서를 볼륨 식별자별로 지정합니다. 이 옵션은 청소 카트리지를 다음으로 이동하기 전에 사용합니다. 이 옵션을 선택하면 ACSLS가 이전 릴리스의 ACSLS 소프트웨어와 같은 순서로 청소 카트리지를 사용하고 반환합니다. ACSLS가 설치 또는 업데이트되는 경우 이 옵션이 기본값입니다.
  - *LEAST_USED* - 청소 카트리지 순서를 사용률별로 지정합니다. 이 옵션을 선택하면 ACSLS가 볼륨 목록을 사용률의 역순으로 정렬하고 가장 적게 사용한 카트리지를 먼저 반환합니다. 이 옵션을 통해 청소 카트리지를 균일하게 사용할 수 있습니다.
  - *MOST_CAPACITY* - 청소 카트리지 순서를 남은 사용 횟수별로 지정합니다. 이 옵션을 사용하면 ACSLS에서는 청소 카트리지의 남은 사용 횟수에 따라 볼륨 목록을 정렬하고 가장 사용 횟수가 많이 남은 카트리지를 먼저 반환합니다. 이 옵션은 모든 청소 카트리지를 거의 동시에 사용합니다.

- *AUTO_CLEAN*

  프롬프트: 운송 장비 자동 청소 옵션을 사용으로 설정*(TRUE/FALSE) [TRUE]:*

  유효한 옵션은 다음과 같습니다.

  - TRUE - 운송 장비에 청소가 필요한 경우 ACSLS에서는 자동으로 다음 마운트 전에 운송 장비를 청소합니다.

    SCSI 또는 광 섬유 연결 라이브러리의 드라이브 청소 작업은 ACSLS가 아닌 라이브러리 펌웨어에서 관리됩니다. LSM 컨트롤 패널의 드라이브 청소가 사용으로 설정되어 있는지 확인하십시오. 설정되지 않은 경우 필요에 따라 주기적으로 ACSLS에서 청소 카트리지를 수동 명령으로 마운트할 수 있습니다.
  - FALSE - 자동 청소를 사용으로 설정하지 않음

- *AUTO_START*

  프롬프트: 라이브러리 서버 시작 상태*(RUN/IDLE) [RUN]:*

  이 옵션에서는 복구가 서버 소프트웨어를 시작하는 동안 완료된 후 초기 ACSLS 상태를 지정합니다. 유효한 옵션은 다음과 같습니다.

- RUN - 복구가 완료된 후 바로 사용자 요청이 처리됩니다.
- IDLE - 복구가 완료된 후 바로 사용자 요청이 처리되지 않게 합니다.

이 옵션을 IDLE로 설정하면 사용자에 대한 서버 액세스를 허용하기 전에 장치 상태를 전환하거나 일부 작업 활동을 수행해야 할 경우에 유용할 수 있습니다.

- *MAX_ACSMT*

  프롬프트: *ACSLS*가 지원하는 마운트 프로세스 수 변경사항은 제품을 다시 시작해야 적용됩니다. 마운트 프로세스 수 *[2]:*

  유효한 입력은 1~5입니다.

  Oracle에서는 초기 구성에서 기본값을 수락하고 필요한 경우 그 값을 변경하는 것을 권장합니다. 이 값을 높이면 성능이 향상될 수 있습니다. 대규모 구성(8개보다 많은 LSM이 있는 구성)의 경우 마운트/마운트 해제 성능을 개선하기 위해 이 값을 높여야 할 수도 있습니다.

  주의:

  이 숫자를 너무 높게 설정하면 ACSLS가 제대로 시작되지 않을 수 있습니다. 이런 경우, 이 숫자를 낮추거나 사용자당 최대 허용 가능 프로세스 수를 높이십시오.

  기본값은 2입니다.

- *ENABLE_ACSQY*

  프롬프트: *QUERY* 연속 프로세스를 사용으로 설정*(TRUE/FALSE) [TRUE]:*

  이 옵션을 사용하면 하나 이상의 연속 프로세스로서 쿼리를 실행할 수 있습니다. 쿼리는 연속 또는 요청 프로세스로서 실행될 수 있습니다. 초기 구성에서 기본값을 수락하고 필요한 경우 그 값을 변경하십시오. 이 값을 높이면 성능이 향상될 수 있습니다. 유효한 옵션은 다음과 같습니다.

  - TRUE - 연속 쿼리 프로세스를 사용으로 설정합니다.

    주의:

    최소 구성 시스템에서 10개보다 많은 쿼리 연속 프로세스로 인해 시스템 리소스를 사용하여 ACSLS를 시작하지 못할 수 있습니다.

  - FALSE - 연속 쿼리 프로세스를 사용 안함으로 설정합니다. 쿼리가 요청 프로세스로서 호출됩니다.

- *MAX_ACSQY*

  프롬프트: *ACSLS*가 지원하는 쿼리 프로세스 수 변경사항은 제품을 다시 시작해야 적용됩니다. 쿼리 프로세스 수 *[2]:*

  이 옵션에서는 만들 연속 쿼리 프로세스 수를 지정합니다. 이 옵션은 위 단계에서 사용으로 설정한 경우에만 유효합니다. 유효한 숫자는 1~5입니다.

Oracle에서는 초기 구성에서 기본값 (2)를 수락하고 필요한 경우 그 값을 변경하는 것을 권장합니다. 대규모 구성(8개보다 많은 LSM)의 경우 성능을 개선하기 위해 이 값을 높여야 할 수도 있습니다.

주:

이 숫자를 너무 높게 설정하면 ACSLS가 제대로 시작될 수 없을 수 있습니다. 이 숫자를 낮추거나 사용자당 최대 허용 가능 프로세스 수를 높이십시오.

이 옵션을 적용하려면 ACSLS를 다시 시작해야 합니다.

- *MAX_ACS_PROCESSES*

  프롬프트: 최대 *ACSLS* 프로세스 수 변경사항은 제품을 다시 시작해야 적용됩니다. *ACSLS* 프로세스 수 *[40]:*

  유효한 숫자는 32~100입니다.

  임시 프로세스는 *`mount; dismount; lock; unlock; clear_lock`* 및 *`query_lock`*을 제외한 모든 요청을 만족합니다. 기본 40개의 프로세스는 대단히 많은 구성을 제외한 모든 ACSLS 프로세스에 대해 작동합니다. 기본값은 40입니다.

  주:

  이 값을 변경하기 전에 고객지원센터에 문의하는 것이 좋습니다.

  이 옵션을 적용하려면 ACSLS를 다시 시작해야 합니다.

- *TRACE_ENTER*

  프롬프트: 자동 넣기 추적 기능을 사용으로 설정*(TRUE/FALSE) [FALSE]:*

  이 옵션에서는 자동 넣기 작업의 결과를 이벤트 로그에 기록할 것인지 지정합니다. 유효한 옵션은 다음과 같습니다.
  - TRUE - 각 자동 넣기 작업 끝에 이벤트 로그에 메시지를 작성하도록 설정합니다. 이 옵션은 자동 넣기 작업 중에 볼륨을 넣지 않은 이유를 결정하는 유일한 방법입니다.
  - FALSE - 이 기능을 사용 안함으로 설정하고, 이벤트 로그에 작성된 메시지 수를 최소화하도록 지원합니다.

- *TRACE_VOLUME*

  프롬프트: 볼륨 추적 기능을 사용으로 설정*(TRUE/FALSE) [FALSE]:*

  이 옵션에서는 데이터베이스에서 볼륨을 추가/삭제하는 경우 원치 않는 메시지가 표시되는지 지정합니다. 유효한 옵션은 다음과 같습니다.
  - TRUE - 볼륨이 데이터베이스에서 추가 또는 삭제될 때마다 원치 않는 메시지가 표시되도록 설정합니다. 이 메시지를 생성할 수 있는 작업에는 감사, 마운트, 마운트 해제, 넣기, 꺼내기, 복구 및 전환이 있습니다.
  - FALSE - 이 기능을 사용 안함으로 설정하고, 이벤트 로그에 작성된 메시지 수를 최소화하도록 지원합니다.

- *ABSENT_VOLUME_RETENTION_PERIOD*
- 프롬프트: 데이터베이스에서 존재하지 않음 또는 꺼냄으로 식별되는 볼륨의 보존 일수*. [5]:*

  동적 변수, 이 옵션에서는 데이터베이스에 존재하지 않거나 꺼낸 카트리지를 보관하는 일수(0~999)를 설정합니다. 볼륨을 다시 넣지 않고 보존 기간이 만료되는 경우 ACSLS는 자동으로 데이터베이스에서 볼륨을 삭제합니다.

  볼륨이 삭제되지 않게 하려면 999를 입력하십시오. 볼륨을 즉시 삭제하려면 0을 입력하십시오.

  주의:

  보존 기간을 길게 설정하면 대규모의 존재하지 않거나 꺼낸 볼륨을 보존할 수도 있습니다. 이런 경우 데이터베이스 공간을 더 많이 사용하며 더 빨리 채울 수 있습니다.

- *ENABLE_STATUS_VOLUME_ABSENT*

  프롬프트: 존재하지 않는 볼륨 및 꺼낸 볼륨을 *ACSAPI* 클라이언트에 보고합니다*. [FALSE]:*

  유효한 옵션은 다음과 같습니다.
  - TRUE - 존재하지 않거나 꺼낸 볼륨을 ACSAPI 클라이언트에 보고합니다.
  - FALSE - 존재하지 않거나 꺼낸 볼륨이 ACSAPI 클라이언트에 보고되지 않습니다(이전 ACSLS 버전에서 발생함).
- *ENABLE_STATUS_VOLUME_MISSING*

  프롬프트: 누락된 볼륨 상태가 *ACSAPI* 클라이언트에 보고됨*: TRUE*=누락됨*. FALSE*= 운반 중 *[FALSE]:*

  이 옵션에서는 누락된 볼륨에 대해 ACSAPI 클라이언트에 반환할 상태 코드를 설정합니다. 유효한 옵션은 다음과 같습니다.
  - TRUE - 누락된 카트리지를 ACSAPI 클라이언트에 보고합니다.
  - FALSE - 누락된 볼륨이 ACSAPI 클라이언트에 보고되지 않습니다(이전 ACSLS 버전에서 발생됨).

  ****ENABLE_INIT_ACSLM*이 TRUE여야 함****

  이 변수는 TRUE여야 하므로 GUI 및 논리 라이브러리에서 레거시 ACSLS 프로세스와 통신할 수 있습니다. [TRUE]:

  레거시 ACSLS 프로세스는 프로세스 간 통신을 위해 UNIX 기반 소켓만 사용했습니다. 이 변수를 통해 GUI 및 논리 라이브러리 지원을 위한 INET 기반 소켓 통신을 사용으로 설정합니다. ACSLS 8+ 기능을 사용으로 설정하려면 *ENABLE_INIT_ACSLM*이 true여야 합니다. 변경사항을 적용하려면 ACSLS 제품을 다시 시작해야 합니다.
- *ALPHANUM_VOL_RANGES*

프롬프트: 명령 및 유틸리티에 대한 영숫자 볼륨 범위를 지원합니다. 영숫자 범위에는 *ASCII* 조합 시퀀스의 모든 유효한 *vol_ids*가 포함됩니다. 주 - 영숫자 볼륨 범위 지원 변경 사항은 제품을 다시 시작해야 적용됩니다. *(TRUE/FALSE) [FALSE]*:

영숫자 볼륨 범위 지원이 명령 및 유틸리티에 대해 활성인지 지정합니다. 유효한 옵션은 다음과 같습니다.

- FALSE(기본값) - 볼륨 범위를 지정할 때 `vol_id` 범위의 첫번째 문자는 같아야 하고 마지막 가변 문자는 숫자여야 합니다. 예: AAA000-AAA999
- TRUE - 영숫자 볼륨 범위에 모든 유효한 볼륨 ID가 포함되어 있는 경우 범위에 대해 유효한 `vol_id`를 임의로 지정할 수 있습니다. 유효한 볼륨 범위는 숫자(0-9), 문자(A-Z), 달러 기호($), 파운드 기호(#) 및 공백(선행 및 후행)을 조합하여 구성됩니다.

예: A1Z27BC-G$123R

- *EJECT_RESPONSE_ON_CAP_FULL*

  프롬프트: 꺼내기 처리 중에, *CAP*가 채워지면 *MAX_ID* 볼륨을 꺼내도록 기다리지 않고 꺼낸 볼륨을 포함하는 중간 응답을 보냅니다. *(TRUE/FALSE) [FALSE].*

  이 옵션에서는 꺼내기 프로세스에서 CAP가 채워지면 중간 응답을 보내야 하는지, 아니면 *`MAX_ID`* 볼륨을 꺼내도록 기다려야 하는지를 지정합니다. CAP가 *`MAX_ID`* 셀보다 많은 구성에서는 중간 응답이 *`MAX_ID`* 볼륨을 꺼낼 때 생성되므로 이 옵션은 동작에 영향을 주지 않습니다. 이 설정은 *`cmd_proc`*에 영향을 미치지 않고 *`cmd_proc`*는 CAP가 채워지면 항상 응답을 수신합니다.

  주의:

  이 변수 값을 변경하면 중간 꺼내기 응답에 항상 MAX_ID 볼륨을 포함할 것을 기대하는 ACSAPI 클라이언트에 영향을 줍니다.

- *MOUNT_RETRY_DELAY*

  프롬프트: 라이브러리가 사용 중이거나 일시적으로 사용할 수 없는 경우 마운트 및 마운트 해제 실패를 방지하려면 요청을 대기열에 저장하고 재시도합니다. 큐에 저장된 마운트 및 마운트 해제 요청을 재시도하기 전에 시간이 지연*(분)*되거나 *ACSLS*에서 일시적으로 오프라인 라이브러리 또는 드라이브가 사용 가능한지 확인합니다. 범위 *1~6*분 *[2]:*

  라이브러리가 사용 중이거나 일시적으로 사용할 수 없기 때문에 마운트 및 마운트 해제 요청이 실패할 때 자동으로 큐에 저장됩니다. 요청을 주기적으로 재시도하거나 라이브러리 및 드라이브 가용성을 다시 확인합니다. 이 변수는 시퀀스를 재시도하려는 시도 사이의 시간 간격(분)을 지정합니다.

- *MOUNT_RETRY_TIME_LIMIT*

  프롬프트: 라이브러리가 사용 중이거나 일시적으로 사용할 수 없는 경우 마운트 및 마운트 해제 실패를 막으려면 요청을 대기열에 저장하고 재시도합니다. 마운트 및 마운트 해제 요청을 대기열에 저장하는 시간 제한*(분)*입니다. 이 시간 제한 후에는 요청이 실패합니다. 범위 *5~80*분 *[5]:*

라이브러리를 일시적으로 사용할 수 없을 때 마운트 및 마운트 해제 요청이 실패하지 않도록 자동으로 대기열에 저장됩니다. 요청을 주기적으로 재시도하거나 라이브러리 및 드라이브 가용성을 다시 확인합니다. 이 변수는 ACSLS가 마운트 또는 마운트 해제 요청을 계속 대기열에 저장하는 최대 시간입니다.

- *AUTO_CLEAN_RETRY_LIMIT*

  프롬프트: 드라이브 청소 시도를 중단하고 요청된 데이터 카트리지 마운트를 진행하기 전 자동 청소 작업 재시도 횟수입니다. 재시도 범위 *0~5*회 *[1].*

  드라이브에 청소가 필요하고 자동 청소가 사용으로 설정되어 있는 경우 ACSLS는 해당 드라이브에 대한 다음 마운트 전에 드라이브를 청소하려고 시도합니다. 청소 카트리지 최대 사용 횟수를 초과하거나 다른 이유로 사용할 수 없어서 청소 작업에 실패하면 ACSLS가 다른 청소 카트리지를 선택하고 마운트하여 드라이브를 청소하려고 시도합니다.

  *AUTO_CLEAN_RETRY_LIMIT*는 요청된 데이터 카트리지를 계속하고 마운트하기 전에 ACSLS가 청소 작업을 재시도하는 횟수를 제한합니다.

- *XAPI_PORT*

  프롬프트: *XAPI* 서버에 대한 사용자 정의 인바운드 포트 변경사항은 *XAPI* 서버를 다시 시작해야 적용됩니다. 들어오는 *XAPI* 요청을 수신하기 위해 *XAPI* 서버에서 사용되는 포트 번호입니다. 포트 *50003*을 지정하지 마십시오*[50020].*

  이 옵션은 클라이언트의 TCP 요청 수신을 위해 XAPI 서버에서 사용되는 포트를 지정합니다. XAPI 서버에서 사용되는 포트를 정의하려면 1024에서 65535 사이의 숫자를 입력하십시오. 포트 50003은 지정하지 **마십시오**.

- *XAPI_WORK_PATH*

  프롬프트: *XAPI* 작업 디렉토리 변경사항은 *XAPI* 서버를 다시 시작해야 적용됩니다. *XAPI* 로깅/추적 정보를 배치해야 할 위치 *[*$ACS_HOME/log/xapi*]*

  XAPI 서버 작업 파일이 배치되는 디렉토리를 선택하십시오. 설치되면 XAPI 서버가 정보를 *$ACS_HOME/log/xapi* 디렉토리에 기록합니다. 일반적인 사용에서 이 변수의 값은 변경되지 않습니다. *$ACS_HOME*을 포함하는 파일 시스템에 디스크 공간 문제가 있는 경우 대체 경로를 지정할 수 있습니다. 지정된 경로가 절대 경로여야 합니다(*/* 또는 *$ACS_HOME*으로 시작되는 경로).

  이 변수를 적용하려면 XAPI 서버를 다시 시작해야 합니다.

- *LIMIT_CAP_CONCURRENT_MOVES*

  프롬프트: 여러 라이브러리가 있는 *ACS*에서 꺼내기 및/또는 넣기에 많은 *CAP*를 사용하는 경우 마운트 및 마운트 해제를 위한 라이브러리 리소스를 예약하기 위해 *CAP*로 또는 *CAP*로부터 동시에 이동하는 횟수를 제한합니다. *(TRUE/FALSE).*

  각 꺼내기 또는 넣기 작업에 대해 ACSLS는 동시에 CAP로 또는 CAP로부터 여러 카트리지를 이동합니다. (각 CAP에 대한 동시 이동 횟수 기본값은 4회입니다.) 다중 연결된 라이브러리의 ACS(라이브러리 컴플렉스)에서 수많은 CAP가 동시에 꺼내기 및 넣기에 사용

중인 경우 이로 인해 마운트, 마운트 해제를 비롯한 기타 요청에 사용할 수 있는 라이브러리 리소스가 제한될 수 있습니다.

많은 CAP를 넣기 및 꺼내기에 동시에 사용하는 경우 *LIMIT_CAP_CONCURRENT_MOVES*를 TRUE로 설정하여 마운트, 마운트 해제 등에 필요한 라이브러리 리소스를 예약하십시오.

- *xapi_startup_file*

  프롬프트: *XAPI* 시작 파일 이름 변경사항은 *XAPI* 서버를 다시 시작해야 적용됩니다*. XAPI* 시작 파일의 이름 *[startup].*

  이 옵션은 XPI 시작 파일의 이름을 지정합니다. 이 파일은 *XAPI_WORK_PATH* 디렉토리에 있으며 XAPI 시작 매개변수를 포함합니다.

  이 변수를 적용하려면 XAPI 서버를 다시 시작해야 합니다.

- *BULK_CAP_EJECT_HANDLE*

  프롬프트: *SL8500* 대량 *CAP*의 카트리지를 꺼낼 때 각 *CAP* 매거진의 슬롯을 핸들로 사용될 수 있도록 비워두십시오*. (TRUE/FALSE)*

  *BULK_CAP_EJECT_HANDLE*을 TRUE로 설정하여 꺼내기 작업 중에 각 대량 CAP 매거진의 슬롯을 비워두십시오. 빈 슬롯은 핸들로 사용될 수 있습니다. 레일 높이에 따라, 아래쪽 슬롯 또는 위쪽 슬롯이 비어있는 상태가 됩니다.

- *ENTER_CLOSE_TO_DRIVES*

  프롬프트: 카트리지를 *SL8500* 라이브러리에 넣을 때 카트리지를 테이프 드라이브와 가까운 스토리지 셀로 이동합니다*. (TRUE/FALSE)*

  *ENTER_CLOSE_TO_DRIVES*가 *TRUE*이면 ACSLS에서는 SL8500에 넣는 중인 카트리지를 테이프 드라이브와 가까운 스토리지 셀로 이동합니다. 이렇게 하면 나중에 드라이브에 카트리지를 손쉽게 마운트할 수 있습니다.

  *ENTER_CLOSE_TO_DRIVES*가 *FALSE*이면 ACSLS에서는 연속 방식으로 넣는 중인 카트리지에 새 홈 셀을 할당하고 각 패널에 차례로 홈 셀을 할당합니다.

- *DISMOUNT_AWAY_FROM_DRIVES*

  프롬프트: *SL8500* 라이브러리에서 마운트 해제 시 카트리지를 새 홈 셀로 *Float*하는 경우 카트리지를 테이프 드라이브에서 멀리 있는 스토리지 셀로 이동합니다*. (TRUE/FALSE)*

  *DISMOUNT_AWAY_FROM_DRIVES*가 *TRUE*이면 ACSLS에서는 SL8500에서 마운트 해제 시 새 홈 셀에 Float 중인 카트리지를 테이프 드라이브에서 멀리 있는 스토리지 셀로 이동합니다. 이렇게 하면 테이프 드라이브에 새 카트리지를 위한 공간이 마련됩니다.

  *DISMOUNT_AWAY_FROM_DRIVES*가 *FALSE*이면 ACSLS에서는 연속 방식으로 마운트 해제 시 Float중인 카트리지에 새 홈 셀을 할당하고 각 패널에 차례로 빈 셀을 할당합니다.

- *XAPI_TAPEPLEX_NAME*

프롬프트: *XAPI Tapeplex* 이름 변경사항은 *XAPI* 서버를 다시 시작해야 적용됩니다*.*

*XAPI Tapeplex*의 이름 *[]*

이 옵션에서는 XAPI Tapeplex의 이름을 지정합니다. 이 변수를 적용하려면 XAPI 서버를 다시 시작해야 합니다.

텍스트 길이가 8을 초과하지 않도록 입력하십시오.

액세스 제어 변수 설정

각 프롬프트에 대한 도움말을 보려면 *?*를 선택하십시오.

옵션 4를 사용하면 다음 액세스 제어 변수를 설정 또는 변경할 수 있습니다.

- *AC_CMD_ACCESS*

  프롬프트: 명령 액세스 제어 변경사항은 제품을 다시 시작해야 적용됩니다*.* 액세스 제어는 명령에 대해 활성입니다*. (TRUE/FALSE) [FALSE].*

  액세스 제어가 명령에 대해 활성인지 지정합니다. 이 옵션을 사용하면 각 네트워크 클라이언트에서 실행하도록 허용되는 ACSLS 명령을 제어할 수 있습니다.

  기본 값은 FALSE이며 액세스 제어를 사용 안함으로 설정합니다.

  주:

  이 변경사항을 적용하려면 ACSLS를 중지한 다음 다시 시작해야 합니다.

- *AC_CMD_DEFAULT*

  프롬프트: 명령에 대한 기본 액세스 *(ACCESS/NOACCESS) [ACCESS].*

  관련 액세스 제어 목록이 없는 경우 명령에 대한 기본 액세스를 지정합니다.

  기본값 *ACCESS*는 액세스 제어 파일에서 제공된 구성 정보를 통해 거부된 경우가 아니면 명령 액세스를 허용합니다.

- *AC_VOL_ACCESS*

  프롬프트: 볼륨 액세스 제어 변경사항은 제품을 다시 시작해야 적용됩니다*.* 액세스 제어는 볼륨에 대해 활성입니다*. (TRUE/FALSE) [FALSE].*

  액세스 제어가 볼륨에 대해 활성인지 지정합니다. 이 옵션을 통해 특정 볼륨을 액세스할 수 있는 ACSLS 클라이언트를 제어할 수 있습니다. 이 옵션을 적용하려면 ACSLS를 다시 시작해야 합니다.

  기본 값은 FALSE이며 볼륨에 대한 액세스 제어를 사용 안함으로 설정합니다.

주:

이 변경사항을 적용하려면 ACSLS를 중지한 다음 다시 시작해야 합니다.

- *AC_VOL_DEFAULT*

  프롬프트: 볼륨에 대한 기본 액세스 *(ACCESS/NOACCESS) [NOACCESS].*

  관련 액세스 제어 목록이 없는 경우 볼륨에 대한 기본 액세스를 지정합니다.

  기본값 *NOACCESS*는 액세스 제어 파일에서 제공된 구성 정보를 통해 허용된 경우가 아니면 볼륨 액세스를 허용하지 않습니다.

- *AC_LOG_ACCESS*

  프롬프트: 명령 또는 볼륨에 대한 액세스가 거부되면 메시지가 기록됩니다*(TRUE/FALSE) [FALSE].*

  명령 또는 볼륨에 대한 액세스가 거부되는 경우 메시지를 이벤트 로그에 기록할 것인지 지정합니다. 명령 또는 볼륨 ID가 기록되고, 이 옵션을 사용으로 설정하면 액세스 ID 및 호스트 ID가 기록됩니다.

자동 백업 변수 설정

각 프롬프트에 대한 도움말을 보려면 *?*를 선택하십시오.

옵션 5를 사용하면 다음 자동 백업 변수를 설정 또는 변경할 수 있습니다.

- 프롬프트: 자동 백업 설정을 수정하시겠습니까*?(y* 또는 *n)*

  변수: 없음. 자동 백업 설정은 크론탭에 저장됩니다. *acsss_config*를 사용하여 이러한 설정을 수정하십시오.

  유효한 옵션은 다음과 같습니다.
  - Y - 예인 경우 매일 백업할 것인지 묻습니다. "예"라고 응답한 경우 시간을 *HH:MM* 형식으로 입력해야 합니다.

    아니라고 응답한 경우 백업할 주의 요일을 선택해야 합니다. 시간을 *HH:MM* 형식으로 입력하십시오.
  - N - SDM 관리 데이터베이스 백업 파일의 보존 기간과 관련된 다음 질문으로 이동합니다.

- *RETENTION_PERIOD*

  프롬프트: *SDM* 관리 데이터베이스 백업 파일의 보존 기간 일수를 설정합니다*.* 설명을 보려면 도움말을 참조하십시오*. [5].*

  이 옵션에서는 DSM에서 스토리지 카트리지의 데이터를 며칠간 추적한 후에 카트리지를 재사용 부적합으로 지정할 것인지를 설정합니다. 유효한 일수: 4~30

보존 기간은 백업을 보존하는 기간입니다. 트랜잭션 로그 파일이 각 백업과 함께 보존되어 백업 분할 영역에 필요한 공간이 늘어납니다. 보존 기간을 선택할 때 현장에 따라 다음 조건을 고려하십시오.

- 라이브러리의 크기 및 작업
- 지정된 기간에 수행된 자동 및 수동 백업 수입니다.

  주:

  bdb.acsss 유틸리티를 사용하여 수동 백업을 실행하는 경우 로컬 디스크에 자동 백업이 수행되므로 백업 분할 영역의 백업 수가 증가합니다.

  보존 기간은 백업 간격보다 커야 합니다. 예를 들어, 보존 기간이 4일인 경우 자동 또는 수동 백업은 일주일에 3번 이상 수행해야 하며 백업 간격이 3일보다 크지 않아야 합니다.

- 백업과 트랜잭션 로그 파일을 보존하려는 기간의 경우 보존 기간이 길면 백업 분할 영역에 필요한 공간이 증가합니다.

액세스 제어 정보 재구성

옵션 6은 액세스 제어가 활성 상태이거나 변경된 경우 액세스 제어 테이블을 재구성합니다.

이벤트 알림 설정 정의

각 프롬프트에 대한 도움말을 보려면 *?*를 선택하십시오.

옵션 7을 사용하면 이벤트 알림 설정을 정의할 수 있습니다.

- *CLIENT_CHECK_MESSAGE_COUNT*

  프롬프트: *ACSLS*에서 두 개의 연속하는 클라이언트 확인 사이에 클라이언트에 보낸 이벤트 메시지 수를 설정합니다*. [100]*

  이 옵션에서는 ACSLS에서 등록된 클라이언트의 활성 상태 검증을 확인하기 전에 클라이언트에 보낼 이벤트 알림 메시지 수와 다음 질문으로 설정되는 최소 확인 간격을 지정합니다.

  유효한 입력: 10~1000개 메시지
- *CLIENT_CHECK_MIN_INTERVAL*

  프롬프트: *ACSLS*에서 수행한 두 개의 연속하는 클라이언트 확인 사이의 최소 간격*(분)*을 입력합니다*. [30]*

  ACSLS에서는 등록된 클라이언트가 계속 활성 상태인 경우 주기적으로 확인합니다. 이 확인은 최소 간격 및 이벤트 알림 메시지 수를 초과하는 경우 수행됩니다.

  유효한 입력: 1~600분
- *CLIENT_CHECK_RESPONSE_TIME*

프롬프트: 클라이언트가 확인 등록 요청이 있는 클라이언트 확인에 응답해야 하는 시간 *(초)*을 입력합니다. *[30]:*

ACSLS에서 주기적으로 등록된 클라이언트가 계속 활성 상태인지 확인할 경우 클라이언트는 지정된 응답 시간 내에 확인 등록 요청과 함께 응답해야 합니다. 클라이언트가 이 기간 내에 응답하지 않으면 클라이언트의 이벤트 알림 등록이 취소됩니다.

유효한 입력: 5~300초

- *`CLIENT_CHECK_POLLING_INTERVAL`*

  프롬프트: 이벤트 알림 클라이언트 등록을 확인하는 폴링 간격*(분)* 입력 *[60]:*

  ACSLS에서는 등록된 클라이언트가 계속 활성 상태인 경우 주기적으로 확인합니다. 폴링 간격이 지정되면 ACSLS는 등록, 등록 취소 또는 이 폴링 간격 내에 등록 요청 확인을 하지 않은 모든 클라이언트에게 클라이언트 확인 응답을 보냅니다.

  유효한 입력은 10~1440분입니다.

  정적 변수 설정을 완료하면 *`acsls`* 서비스를 다시 시작해야 합니다. *`acsss disable`* 후에 *`acsss enable`*을 사용하십시오.

  동적 변수의 경우에는 이 작업을 수행하지 않아도 됩니다.

라이브러리 하드웨어 구성 또는 업데이트

처음 라이브러리 하드웨어를 만들거나 새 라이브러리를 추가하거나 기존 라이브러리를 재구성하려면 *`acsss_config`*(옵션 8)를 사용해야 합니다. 초기 라이브러리 구성 후에 *`config`* 유틸리티를 사용하여 동적으로 LSM 및 테이프 드라이브를 추가, 재구성 또는 제거하거나, ACS를 추가 또는 재구성하거나, 라이브러리에 대한 포트 연결을 추가할 수 있습니다. 자세한 정보 및 절차를 알아보려면 [71]6장. *ACSLS* 동작을 제어하는 변수 설정을 참조하십시오.

자세한 내용은 5장. 라이브러리 하드웨어 설치 및 구성 을 참조하십시오.

시스템 이벤트의 전자 메일 알림 등록

관리 책임이 있는 사용자는 시스템 부트 이벤트 및 ACSLS-HA 시스템 클러스터 페일오버 이벤트를 비롯한 시스템 이벤트의 자동 전자 메일 알림을 등록할 수 있습니다.

이러한 이벤트를 등록하려면 사용자가 자신의 전자 메일 주소를 다음 디렉토리 아래의 해당 파일에 추가해야 합니다.

```
$ACS_HOME/data/external/email_notification/
  boot_notification
  ha_failover_notification
```

헤더 설명 아래에 각 받는 사람에 대한 전자 메일 주소를 한 라인에 입력합니다. 그런 다음에는 시스템이 부트되거나 HA 클러스터가 대기 노드로 페일오버를 수행할 때마다 등록된 각 사용자에게 전자 메일 알림이 전송됩니다.

이 기능은 메일 전송 서비스가 ACSLS 서버에 사용으로 설정되었고 네트워크 방화벽 제약 조건에서 데이터 센터로부터의 전자 메일 통신이 허용된다고 가정합니다.

7

7장. 액세스 제어

액세스 제어는 다음을 제공합니다.

- 볼륨 액세스 제어 - 볼륨을 하나의 클라이언트 응용 프로그램에 지정할 수 있습니다. 다른 클라이언트는 클라이언트의 볼륨에 액세스할 수 있습니다.
- 명령 액세스 제어 - 관리자가 특정 ACSLS 명령을 특정 클라이언트에 지정할 수 있습니다.

볼륨 액세스 제어 및 명령 액세스 제어는 모두 ACSAPI를 통해 요청을 제출하는 클라이언트 응용 프로그램의 사용자에게 적용됩니다.

액세스 제어는 *cmd_proc* 또는 ACSLS GUI를 사용해서 라이브러리 요청을 제출하는 관리 사용자로 액세스를 제한하지 않습니다.

볼륨 액세스 제어

이 기능이 사용으로 설정된 경우, 특정 사용자가 소유하는 볼륨은 해당 사용자 또는 신뢰할 수 있는 다른 사용자만 액세스할 수 있습니다.

ACSLS에서 볼륨 액세스 제어를 처음 구성할 때는 다음 단계를 수행합니다.

1. ACSLS에서 *volume access* 제어를 사용으로 설정합니다.
2. 클라이언트 응용 프로그램을 사용자 이름과 연관시킵니다.
3. 사용자 볼륨에 액세스할 수 있는 다른 사용자를 정의합니다.
4. 볼륨 소유권을 설정합니다.

볼륨 액세스 제어 사용으로 설정

ACSLS에서 *volume access* 제어를 사용으로 설정하려면 다음을 수행합니다.

1. 구성 유틸리티 *acsss_config*를 실행합니다.

   기본 메뉴가 표시됩니다.
2. 옵션 4 - Set Access Control Variables를 선택합니다.

   각 변수는 한 번에 하나씩 나열되며 해당 현재 설정이 표시됩니다.
3. **Enter**를 눌러서 현재 또는 기본 설정을 수락합니다.
4. 유틸리티에 *Access control is active for volumes* 메시지가 표시되면 *[TRUE]*를 선택하고 **Enter**를 누릅니다.

5. 유틸리티에 *Default access for volumes [ACCESS/NOACCESS]*... 메시지가 표시되면 다음 중 하나를 선택합니다.
   - 특정 사용자에 대한 액세스를 금지하고 다른 모든 사용자에 대한 액세스를 허용하려면 [*ACCESS]*를 선택합니다.

     이렇게 하려면 특정 사용자가 *users.ALL.disallow* 파일 또는 특정 users .COMMAND.disallow 파일에 나열되어 있어야 합니다. "사용자 볼륨에 액세스하도록 허용된 다른 사용자 정의 "를 참조하십시오.
   - 특정 사용자에 대한 액세스를 허용하고 모든 사용자에 대한 액세스를 금지하려면 [*NOACCESS*]를 선택합니다.

     이렇게 하려면 특정 사용자가 users.ALL.allow 파일 또는 특정 *users .COMMAND.allow* 파일에 나열되어 있어야 합니다. "사용자 볼륨에 액세스하도록 허용된 다른 사용자 정의 "를 참조하십시오.

     볼륨 액세스가 거부되는 인스턴스를 기록하려면 해당 프롬프트에 대한 응답으로 [TRUE]를 선택합니다.

     볼륨 액세스를 사용 또는 사용 안함으로 설정할 때마다 해당 변경사항을 적용하려면 ACSLS를 다시 시작해야 합니다.

```
Associating a client identity with a user name
```

클라이언트 응용 프로그램이 모두 ACSLS 요청 패킷에 사용자 ID를 전달하는 것은 아닙니다. 사용자 이름으로 클라이언트를 식별할 수 없는 경우에는 사용자 ID를 지정할 수 있습니다.

1. *access_control* 구성 디렉토리로 이동합니다.

```
$ACS_HOME/data/external/access_control.
```

2. internet.addresses 이름으로 파일을 만들거나 *internet.addresses.SAMPLE* 파일을 복사합니다.
3. 이 파일에서 각 클라이언트에 대해 레코드를 만듭니다. 각 레코드에는 최소 두 개의 필드(클라이언트 IP 주소와 해당 사용자 이름)가 포함됩니다. 설명을 위해 추가 필드를 포함시킬 수 있습니다.

   다음 예제에 표시된 것처럼 공백 또는 탭으로 필드를 구분합니다.

```
192.0.2.1  ulyssis   payroll department
```

   클라이언트-사용자 연관은 클라이언트 응용 프로그램 수만큼 만들 수 있습니다.
   - 클라이언트 응용 프로그램이 ACSLS 요청에 사용자 이름을 전달하는 경우에는 internet.addresses 파일이 지정된 IP 주소로 사용자 이름을 인증하고, 두 필드가 모두 요청 패킷의 값과 일치하지 않는 경우에는 액세스를 거부합니다. 공통 플랫폼에서

여러 클라이언트가 호스트되는 경우에는 동일한 IP 주소가 이 파일에 여러 번 포함될 수 있으며, 해당 IP 주소에 올바르게 적용되는 한 이 주소를 여러 사용자 이름과 연관시킬 수 있습니다.

- 클라이언트 응용 프로그램이 요청에 사용자 이름을 전달하지 않는 경우에는 *internet.addresses* 파일이 해당 클라이언트에 대해 사용자 이름을 설정합니다. 이 경우에는 사용자 이름 하나만 클라이언트 주소와 연관시킬 수 있습니다.

4. *internet.addresses* 파일에 대한 모든 업데이트를 저장합니다.
   - *acsss_config*를 실행합니다.
   - 옵션 6 - "Rebuild Access Control Information"을 선택합니다.

   ACSLS는 변경사항을 동적으로 인식합니다.

   TCP/IP를 사용하지 않는 SNA 및 OSLAN 클라이언트는 *access_control* 디렉토리에 있는 *lu62.names* 또는 *adi.names* 파일을 참조하십시오.

사용자 볼륨에 액세스하도록 허용된 다른 사용자 정의

특정 사용자 소유의 볼륨에 다른 사용자가 액세스하도록 권한을 부여하려면 다음을 수행합니다.

1. *access_control* 디렉토리에서 *users.ALL.allow* 또는 *users.ALL.disallow*를 만듭니다.

   *users.*SAMPLE*.allow* 또는 *users.*SAMPLE*.disallow* 템플리트를 복사할 수 있습니다.
2. 각 소유자에 대해 파일에 레코드를 추가하여 왼쪽 여백에 소유자의 사용자 ID를 배치합니다.
3. 각 소유자와 동일한 라인에 영향을 받는 사용자를 지정합니다.
4. 다음 예제에 표시된 것처럼 공백 또는 탭으로 사용자 이름을 구분합니다.

```
owner_john   user-Allie   user-andre
```

   users.allow 및 *users.disallow* 파일에 나열된 사용자 이름은 대소문자 구분없이 고유해야 합니다. 사용자 이름에 사용된 문자의 대소문자는 무시됩니다.

   소유자와 동일한 라인에 나열되지 않은 사용자에게는 소유자 볼륨에 대한 기본(*ACCESS* 또는 *NOACCESS*) 관계가 부여됩니다.

> **주:**
>
> 동일한 명령 또는 ALL에 대해 동일한 *owner_ID* 및 *user_ID* 쌍을 users.COMMAND.allow 및 users.COMMAND.disallow 파일 모두에 지정할 수 없습니다. 또한 중복된 *owner_ID* 및 *user_ID* 쌍을 동일한 users.COMMAND.allow 및 users.COMMAND.disallow 파일에 지정할 수 없습니다. 여기에는 동일한 라인의 동일 *user_ID* 반복이 포함됩니다.
>
> 한 소유자에 대해 허용되는 사용자가 한 라인에 들어가는 것보다 많으면 허용된 사용자 목록을 후속 라인에 계속 지정할 수 있습니다. 각 라인은 소유자 ID로 시작해야 합니다.

5. 선택적으로 사용자가 정의한 볼륨 액세스 정책에 대해 예외사항을 설정할 수 있습니다.

   일반적으로 사용자에게는 액세스 제어 하에 있는 볼륨에 대해 전체 액세스 권한이 허용되거나 액세스 권한이 전혀 허용되지 않습니다. 하지만 다른 사용자의 볼륨에 대해 특정 제한된 액세스 권한을 사용자에게 허용할 수 있습니다.

   예를 들어, 특정 사용자가 소유하는 볼륨에 대해 모든 사용자가 해당 볼륨을 마운트하거나 마운트 해제할 수 없더라도 질의할 수 있도록 허용하는 정책을 설정할 수 있습니다. 예외사항은 액세스 제어로 영향을 받는 모든 명령에 적용할 수 있습니다.

   특정 명령에 대해 볼륨 액세스 정책 예외사항을 구성하려면 다음을 수행합니다.

   - *users.COMMAND.allow* 또는 *users.COMMAND.disallow*파일을 만들어야 합니다. 여기서 COMMAND는 권한을 부여하거나 제한하려는 특정 명령으로 대체합니다.

     users.COMMAND.allow 및 *users.COMMAND.disallow* 파일은 아래에 나열된 것처럼 지정된 이름과 정확히 동일한 명령 구성 요소 이름을 포함해야 하며, 명령 이름을 대문자로 표시해야 합니다. 명령의 다른 변형 형태에 대한 액세스 제어(예: *QUERY_VOLUME*)는 지원되지 않습니다.

     ```
     DISMOUNT
     EJECT
     LOCK
     MOUNT (1)
     MOUNT_READONLY (2)
     QUERY
     REGISTER
     SET_CLEAN
     SET_SCRATCH
     UNLOCK
     ```

     주:

     - MOUNT (1) - *MOUNT* 정책은 mount scratch에도 적용됩니다. mount readonly에는 정책이 적용되지 않습니다.
     - *MOUNT_READOLNY* (2) - mount readonly에 대한 정책은 mount와 별도로 정의됩니다.
     - 위에서 비중복 소유자 ID 및 허용되는 사용자 ID 쌍과 후속 라인으로 이어지는 허용되는 사용자 연속 목록에 대한 고려 사항은 허용되지 않는 사용자 목록에도 적용됩니다.
   - 각 소유자에 대해 소유자 이름을 왼쪽 여백에 배치하고 정책이 적용되는 사용자를 그 뒤에 표시합니다.
6. 정의한 정책에 대한 모든 업데이트를 저장합니다.
   - *acsss_config*를 실행합니다.
   - 옵션 6 - "Rebuild Access Control Information"을 선택합니다.

ACSLS는 변경사항을 동적으로 인식합니다.

볼륨 소유권 설정

볼륨 액세스 제어는 명시적 소유권이 있는 볼륨에만 적용됩니다. 라이브러리에서 소유자가 없는 볼륨은 모든 사용자가 액세스할 수 있습니다. 볼륨 소유권을 명시적으로 설정하려면 `cmd_proc` 인터페이스를 사용합니다.

```
ACSSA>set owner "daffy" volume V00100-V00199
Set: owner set for volumes V00100-V00199
Set: Set completed, Success.
```

빈 문자열을 사용해서 비슷한 방식으로 소유권을 제거할 수 있습니다.

```
ACSSA> set owner "" volume V00100-V00199
Set: owner set for volumes V00100-V00199
```

이 작업은 해당 범위의 모든 볼륨에서 소유권을 지웁니다. 자세한 내용은 “set owner ”을 참조하십시오.

볼륨 소유권은 `watch_vols` 유틸리티에서 자동으로 설정할 수 있습니다. 자세한 내용은 “watch_vols ”를 참조하십시오.

소유권 정책

또한 소유권 자동 설정 및 제거 정책은 ACSLS에서 정의할 수 있습니다. 예를 들어, 마운트된 스크래치 볼륨이 볼륨을 마운트한 사용자 소유가 되는 정책을 설정할 수 있습니다. 그런 다음에는 볼륨이 해당 사용자의 소유가 됩니다. 동일 정책을 조금 더 발전시키면 볼륨이 스크래치 상태로 돌아왔을 때마다 소유권을 제거할 수 있습니다. 넣은 모든 볼륨이 기본 사용자에게 지정되거나, 넣기를 요청한 사용자에게 지정되도록 또는 볼륨이 이전에 소유된 경우, 이전 소유자에게 지정되도록 정책을 작성할 수 있습니다. 이 기능에는 상당한 유연성이 제공됩니다.

소유권 정책은 access_control 디렉토리에 있는 *`ownership.assignments`* 파일에 정의됩니다. 이 파일에서 각 *`enter or automatic enter, set scratch`* 또는 *`mount scratch`* 작업으로 소유권을 자동으로 지정 또는 지정 해제하도록 정책을 설정할 수 있습니다. *`ownership.assignments`* 파일을 사용하면 기본 소유자를 정의할 수 있습니다. 볼륨에 이러한 작업이 수행될 때마다 해당 소유권을 다음 항목에 지정할 수 있습니다.

- *`Owner_default`*(기본 소유자)
- 동일 소유자(이전 소유자)
- 요청자(현재 요청을 실행한 사용자)
- 소유 해제(볼륨에서 소유권 철회)

> **주:**
>
> 소유권 정책 정의 지침은 *`ownership.assignments`* 파일에 자세히 설명되어 있습니다. 이 파일에는 볼륨 소유권 설정을 위해 사용할 수 있는 전체 명령 목록이 포함되어 있습니다.

- 정의한 정책에 대한 모든 업데이트를 저장합니다.
  - *acsss_config*를 실행합니다.
  - 옵션 6 - Rebuild Access Control Information을 선택합니다.

  ACSLS는 변경사항을 동적으로 인식합니다.

소유권 확인

소유권을 확인하려면 *owner_id.volrpt* 템플리트를 사용해서 *volrpt*를 실행할 수 있습니다.

```
cd ~acsss/data/external/volrpt
volrpt -f owner_id.volrpt
```

이렇게 하면 라이브러리의 모든 볼륨이 연관된 소유자와 함께 표시됩니다.

볼륨 액세스 요약

볼륨 액세스 제어에서는 다음 명령이 지원됩니다.

```
dismount*
display
eject
enter
lock
set_clean
set_scratch
mount
query_mount
query_scratch
query_volume
unlock
```

force 옵션은 StorageTek ACSLS가 볼륨 ID를 무시하고 무조건 볼륨을 마운트 해제하도록 지시하기 때문에 *dismount force*에는 액세스 제어가 적용되지 않습니다.

다음 표에서는 *volume access control*이 사용으로 설정되었을 때 적용되는 컨텍스트를 요약해서 보여줍니다.

표 7.1. 볼륨 액세스가 사용으로 설정됨

볼륨에 대한 기본 액세스가 **ACCESS**인 경우	액세스 허용	액세스 거부
액세스가 *cmd_proc*를 통해 수행됨	X	
지정된 볼륨이 소유 해제되어 있음	X	
사용자가 볼륨 소유자임	X	
사용자가 *users.ALL.disallow*의 소유자와 연관됨		X
사용자가 *users.ALL.disallow*의 소유자와 연관되지 않음	X	

표 7.2. 볼륨 액세스가 사용으로 설정됨

볼륨에 대한 기본 액세스가 *NOACCESS*인 경우	액세스 허용	액세스 거부
액세스가 *cmd_proc*를 통해 수행됨	X	
지정된 볼륨이 소유 해제되어 있음	X	
사용자가 볼륨 소유자임	X	
사용자가 *users.ALL.allow*의 소유자와 연관됨	X	
사용자가 *users.ALL.allow*의 소유자와 연관되지 않음		X

명령 액세스 제어

명령 액세스 제어를 사용하면 ACSLS 관리자가 특정 명령 클래스를 네트워크에 있는 특정 응용 프로그램 또는 특정 사용자로 제한할 수 있습니다. 제어된 액세스는 ACSAPI를 통해 제출된 사용자 명령에만 적용되며 cmd_proc를 사용해서 명령을 제출하는 로컬 사용자에게 적용되지 않습니다.

*command access control*에 대한 ACSLS 구성 프로세스에는 세 가지 단계가 포함됩니다.

ACSLS에서 명령 액세스 제어를 처음 구성할 때는 다음 단계를 수행합니다.

1. ACSLS에서 명령 액세스 제어를 사용으로 설정합니다.
2. 클라이언트 ID를 사용자 이름과 연관시킵니다.
3. 사용자에게 제공되는 명령을 정의합니다.

명령 액세스 제어 사용으로 설정

ACSLS에서 명령 액세스 제어를 사용으로 설정하려면 다음을 수행합니다.

1. 구성 유틸리티 *acsss_config*를 실행합니다.

   기본 메뉴가 표시됩니다.
2. 옵션 4 - Set Access Control Variables를 선택합니다.

   각 변수는 한 번에 하나씩 나열되며 해당 현재 설정이 표시됩니다.
3. **Enter**를 눌러서 현재 또는 기본 설정을 수락합니다.
4. 유틸리티에 *Access control is active for commands* 메시지가 표시되면 TRUE를 선택하고 Enter를 누릅니다.
5. "Default access for commands" 메시지가 표시되면 다음을 수행합니다.
   - 모든 명령에 대해 모든 사용자 액세스를 허용하려면 *ACCESS*를 선택합니다.

     특정 사용자가 명령을 실행하지 않도록 차단하려면 이러한 사용자가 *command.*ALL*.disallow* 파일 또는 특정 command.XXX.disallow 파일에 나열되어 있어야 합니다.

     XXX는 액세스 제어 대상이 되는 명령입니다.

- 명령에 대한 사용자 액세스를 거부하려면 [*NOACCESS*]를 선택합니다.

  특정 사용자가 명령을 실행하도록 허용하려면 이러한 사용자가 *command.*ALL*.allow* 파일 또는 특정 *command.*XXX*.allow* 파일에 나열되어 있어야 합니다.

  주:

  명령 액세스가 거부되는 인스턴스를 기록하려면 해당 프롬프트에 대한 응답으로 "TRUE"를 입력합니다.

  주:

  명령 액세스를 사용 또는 사용 안함으로 설정할 때마다 변경사항을 적용하려면 ACSLS를 다시 시작해야 합니다.

클라이언트 ID를 사용자 이름과 연관

???의 절차를 참조하십시오.

사용자에게 제공되는 명령 정의

이 프로세스는 명령 액세스 제어를 사용으로 설정할 때 선택한 기본 동작에 따라 달라집니다. *$ACS_HOME/data/external/access_control* 디렉토리에 정책 파일을 만들어야 합니다.

- 위에서 정의한 기본 동작이 [NOACCESS]인 경우 모든 ACSLS 명령에 대한 액세스 권한을 갖는 각 클라이언트의 사용자 ID가 포함된 *command.ALL.allow* 파일을 만들어야 합니다. 각 사용자 ID는 파일에서 별도의 라인에 나열됩니다.

  특정 사용자에게 특정 명령만 권한을 부여하려면 사용자가 실행하도록 허용된 각 명령에 대해 *command.XXX.allow* 파일을 만들어야 합니다. 예를 들어, 특정 사용자가 라이브러리에 볼륨을 넣을 수 있도록 권한을 부여하려면 *command.ENTER.allow*라는 이름으로 파일을 만들고 이 파일의 개별 라인에 적격한 각 'enter' 사용자의 ID를 나열합니다.
- 위에서 정의한 기본 동작이 [ACCESS]인 경우 모든 ACSLS 명령에 대한 액세스 권한을 갖지 않는 각 클라이언트의 사용자 ID가 포함된 *command.ALL disallow* 파일을 만들어야 합니다. 각 사용자 ID는 파일에서 별도의 라인에 나열됩니다.

  주:

  동일한 명령 또는 ALL에 대해 동일한 user_ID를 *command.XXX.allow* 및 *command.XXX.disallow* *command.XXX* 파일 모두에 지정할 수 없습니다.

명령 액세스 제어 allow 및 disallow 파일에 대한 명령 이름

command.XXX.allow 및 *command.XXX.disallow* 파일은 아래에 나열된 것처럼 지정된 이름과 정확히 동일한 명령 구성 요소 이름을 포함해야 하며 명령 이름을 대문자로 표시해야 합니다. 명령의 다른 변형 형태에 대한 액세스 제어(예: *QUERY_VOLUME*)는 지원되지 않습니다.

AUDIT

```
CANCEL
CHECK_REGISTRATION
CLEAR_LOCK
DEFINE_POOL
DELETE_POOL
DISMOUNTDISMOUNT_FORCE
DISPLAY
EJECT
ENTER      (1)
IDLE
LOCK
MOUNT      (2)
QUERY
QUERY_LOCK
REGISTER
SET_CAP
SET_CLEAN
SET_OWNER
SET_SCRATCH
START
UNLOCK
UNREGISTER
VARY
```

주:

ENTER (1) - 가상 넣기 및 수동 넣기에 정책이 적용되며 자동 넣기에는 적용되지 않습니다. MOUNT (2) - *`mount scratch`* 및 *`mount readonly`*에도 정책이 적용됩니다.

다음 표는 명령 액세스를 허용할 경우를 확인하기 위한 빠른 참조로 활용할 수 있습니다.

표 7.3. 명령 액세스가 사용으로 설정됨

명령에 대한 기본 액세스가 **NOACCESS**인 경우	액세스 허용	액세스 거부
요청이 *`cmd_proc`*에서 입력됨	X	
user_ID가 *`command.COMMAND.allow`*에 나열됨	X	
user_ID가 *`command.ALL.allow`*에 나열됨	X	
- - 기타 모든 조건 - -		X

표 7.4. 명령 액세스가 사용으로 설정됨

명령에 대한 기본 액세스가 **ACCESS**인 경우	액세스 허용	액세스 거부
요청이 *`cmd_proc`*에서 입력됨	X	
user_ID가 *`command.COMMAND.disallow`*에 나열됨		X
user_ID가 *`command.ALL.disallow`*에 나열됨		X
- - 기타 모든 조건 - -	X	

- 정의한 정책에 대한 모든 업데이트를 저장합니다.
  - *`acsss_config`*를 실행합니다.
  - 옵션 6 - "Rebuild Access Control Information"을 선택합니다.

ACSLS는 변경사항을 동적으로 인식합니다.

액세스 제어 메시지 기록

사용자에게 액세스가 거부되었기 때문에 실패한 모든 트랜잭션을 기록하도록 정책을 설정할 수 있습니다. 이 메시지에는 사용자 이름 및 시도된 명령이 표시됩니다.

액세스 제어 로깅을 사용으로 설정하려면 다음을 수행합니다.

1. *`acsss_config`*를 실행하고 옵션 4 - "Set Access Control Variables"를 선택합니다.
2. "Messages will be logged when access to commands or volumes is denied." 프롬프트에서 [FALSE]를 [TRUE]로 변경합니다.
3. 옵션 6 - "Rebuild access control information"을 선택합니다.

명령 요청이 거부될 때마다 ACSLS가 변경사항을 인식하고 로깅을 시작합니다.

8

8장. 라이브러리 관리

라이브러리 관리는 다음과 같은 작업으로 구성될 수 있습니다.

ACS 번호 지정

ACSLS에서는 모든 ACS 번호를 순서대로 지정하지 않고 라이브러리를 구성하거나 재구성할 수 있습니다.

예:

9310 라이브러리에서 SL8500으로 마이그레이션할 수 있습니다. 9310 ACS는 이제 ACS 0이고 SL8500 ACS는 ACS 1입니다.

ACS 번호를 건너뛰면 SL8500 ACS의 번호를 재지정하지 않아도 모든 카트리지와 드라이브를 SL8500 ACS로 마이그레이션하고 9310 ACS를 제거할 수 있습니다.

SL8500 ACS의 번호를 다시 지정하면 모든 볼륨이 삭제된 다음 후속 감사에 다시 추가됩니다. 그러면 모든 *drive_ids*도 변경됩니다.

기존 ACS의 번호를 변경하려면 “acs_renumber.sh ”의 내용을 참조하십시오.

테이프 드라이브를 이동, 추가 또는 제거할 때 ACSLS 재구성

위치 간에 테이프 드라이브가 스왑되고 라이브러리의 테이프 드라이브를 다른 테이프 드라이브로 바꿀 때마다 ACSLS 데이터베이스의 드라이브 일련 번호와 드라이브 테이프를 업데이트하도록 ACSLS를 재구성해야 합니다. 라이브러리에 삽입하거나 제거할 때 테이프 드라이브를 추가하거나 제거하도록 재구성합니다. 매체 검증 풀에 테이프 드라이브를 추가하면 ACSLS에서 액세스하지 못하도록 제거되며, 매체 검증 풀에서 테이프 드라이브를 제거하면 ACSLS에 추가됩니다.

테이프 드라이브가 기존 드라이브를 대체하는 경우, ACSLS가 라이브러리에서 테이프 드라이브 상태를 읽을 때까지 드라이브 유형과 일련 번호가 업데이트되지 않습니다. 이 동작은 다음 시점에 발생합니다.

- ACSLS 시작
- ACS 또는 LSM이 준비되지 않음 상태가 된 후 다시 준비됨 상태가 되어 ACSLS에서 라이브러리를 복구하도록 할 때
- 고객이 ACS, LSM 또는 테이프 드라이브를 오프라인으로 변경한 후 다시 오프라인으로 변경할 때
- 고객이 테이프 드라이브, LSM 또는 ACS를 재구성할 때

테이프 드라이브를 추가하거나 제거할 때, ACSLS 데이터베이스의 드라이브를 추가하거나 삭제하려면 ACSLS 구성을 업데이트해야 합니다.

ACSLS 테이프 드라이브 구성을 업데이트하면 마운트 시 오류가 방지되고 잘못된 테이프 드라이브에 카트리지가 마운트되지 않습니다.

ACSLS에 구성된 테이프 드라이브 업데이트

라이브러리에서 테이프 드라이브를 이동하거나 대체한 경우 동적 구성을 사용하여 드라이브 테이프와 일련 번호를 업데이트합니다. 한 테이프 드라이브가 동일한 위치에 있는 다른 테이프 드라이브를 대체하는 경우 업데이트할 때 고객의 확인이 필요하지 않습니다. 테이프 드라이브를 라이브러리에서 제거하거나 라이브러리에 삽입하는 경우 고객이 구성 변경을 확인해야 합니다.

주:

- *`config`* 요청을 발행하기 전에 영향받는 모든 구성 요소가 준비되었는지 확인합니다.
- ACSLS를 사용으로 설정한 경우 동적 구성을 사용하여 ACSLS 데이터베이스 업데이트가 완료됩니다. 동적 구성은 중단되지 않으며, 구성을 업데이트하는 동안 ACSLS가 요청 처리를 계속할 수 있습니다.
- *`config lsm`* 또는 *`config acs`*를 발행한 다음 영향받는 LSM 또는 ACS를 감사하는 것이 좋습니다.

Unix 명령 프롬프트에서 동적 구성 유틸리티 명령을 사용하여 테이프 드라이브 구성을 업데이트합니다. *`acsss`*로 로그인해야 합니다.

- *`config drive <panel_id>`*

변경이 단일 패널 또는 SL8500 레일에 있는 테이프 드라이브에만 영향을 미치는 경우 *config drive <panel_id>*를 사용하여 패널에 있는 모든 테이프 드라이브의 드라이브 구성을 업데이트합니다.

- *config lsm <lsm_id>*

  *config lsm <lsm_id>*를 사용하여 두 개의 드라이브 패널이 있는 SL3000의 모든 테이프 드라이브에 대한 드라이브 구성을 업데이트합니다.

  주:

  *config lsm <lsm_id>*가 LSM의 CAPS 및 스토리지 용량도 업데이트한 다음 LSM을 감사해야 합니다.

- *config acs <acs_id>*

  *config acs <acs_id>*를 사용하여 모든 ACS 요소(예: SL8500 라이브러리 복합기)의 구성을 업데이트합니다.

  주:

  *config acs <acs_id>*도 전체 ACS의 CAPS 및 스토리지 용량도 업데이트한 다음 ACS를 감사해야 합니다.

라이브러리 감사

감사에서 ACSLS 데이터베이스를 라이브러리 카트리지의 실제 인벤토리와 일치하도록 업데이트합니다.

감사 작동 방식

감사 시 중복되고 잘못된 볼륨을 꺼냅니다. 카트리지에는 다음이 있습니다.

- 이미 스캔된 레이블과 중복되는 외부 레이블
- 누락되거나 읽을 수 없는 외부 레이블, 가상 레이블 없음
- 잘못된 매체 유형
- 잘못된 볼륨 ID

ACSLS는 이벤트 로그의 감사에서 데이터베이스 변경을 기록하고, 감사 중에 *cmd_proc* 메시지도 표시합니다. 감사는 테이프 드라이브나 CAP가 아니라 LSM 스토리지 셀에만 적용됩니다. 감사 실행에 대한 정보는 “audit ”의 내용을 참조하십시오.

감사 실행 시기

감사를 실행하여 다음을 수행합니다.

- 새로 구성된 라이브러리의 볼륨 정보를 데이터베이스에 작성합니다.
- CAP를 통해 카트리지를 입력하지 않은 경우 데이터베이스에 볼륨을 추가합니다.

예제: 라이브러리에 LSM을 추가하고, LSM 도어를 연 다음, LSM에 수동으로 카트리지를 추가했습니다.

- 라이브러리와 데이터베이스 간의 불일치를 해결합니다.

  예제: CAP를 통해 카트리지를 꺼내지 않고, LSM 도어를 열어 수동으로 카트리지를 제거한 경우 감사를 실행합니다. 그 결과 볼륨이 없음으로 표시되거나, 제거된 볼륨이 데이터베이스에서 삭제됩니다.

ACSLS에 대해 구성 또는 다시 구성한 다음 SL3000을 감사합니다. 라이브러리는 ACSLS가 이를 감사할 때 ACSLS에서 액세스할 수 없는 잠재적인 셀 위치를 보고합니다. 액세스할 수 없는 셀 위치에는 다음이 포함됩니다.

- CAP, 드라이브 및 운영자 패널이 설치된 위치.
- 로봇이 액세스할 수 없는 셀 위치.
- 활성화되지 않은 셀 위치.
- 이 파티션에 없는 셀 위치.

감사 간격

감사 간격은 ACSLS 구성, 라이브러리가 데이터베이스에서 셀 컨텐츠를 보고하는지 아니면 각 셀을 조사하여 보고하는지 여부, 라이브러리 구성, 필요한 데이터베이스 변경 수 및 감사 범위 등의 여러 요인에 따라 달라집니다. 다음 표는 감사 범위의 차이가 감사 간격에 미치는 영향을 설명합니다.

표 8.1. 감사 범위가 감사 간격에 영향을 미치는 방식

시간이 덜 소요되는 감사	시간이 더 소요되는 감사
진단 ACS/LSM	온라인 ACS/LSM
감사 전용 ACS/LSM	다른 요청을 처리하는 ACS/LSM
전체 패널	(부분적 또는 완전히) 빈 패널
드라이브 패널	표준 패널
내부 패널	외부 패널

또한 다음 표에 표시된 대로 감사하는 구성 요소의 LSM 유형을 고려하십시오. 4410, 9310 및 L5500의 경우, 모든 셀이 가득 찬 경우 시간이 가장 단축되며 셀이 비어 있으면 시간이 가장 많이 소요됩니다.

표 8.2. 지원되는 LSM의 평균 감사 시간

LSM 유형	구성 요소	평균 감사 시간 (최소 및 최대)
4410	LSM	3 - 8시간
9310/L5500	LSM	1.2 - 6시간
9360	LSM	5분
97xx/L700/180	LSM	1분
SL500	LSM	3분

LSM 유형	구성 요소	평균 감사 시간 (최소 및 최대)
SL8500 또는 SL3000	LSM	5분(ACSLS 감사 전에 SL8500 물리적 감사가 완료되는 경우)

주:

이러한 시간은 기타 활성 프로세스가 없다고 가정합니다. 감사되는 셀의 사용 비율이 높은 경우 4410 및 9310 감사 시간이 줄어들 수 있습니다.

SCSI 또는 섬유 연결 라이브러리를 ACSLS에 추가

SCSI 매체 교환기(mchanger)는 ACSLS와 SCSI 라이브러리 간에 통신하는 장치 드라이버입니다. ACSLS에 연결된 SCSI 또는 섬유 연결 라이브러리마다 mchanger를 작성해야 합니다.

자세한 내용은 “SCSI mchanger 장치 드라이버 추가 ”의 내용을 참조하십시오.

Extended Store 기능 사용

다음 절에서는 Extended Store 기능 사용에 대한 정보를 제공합니다.

주:

이 기능은 전달 포트가 없는 단일 LSM에 적용되지 않습니다.

카트리지가 마운트된 경우 ”홈 위치”가 카트리지가 마운트된 스토리지 셀입니다. 일반적으로 카트리지가 마운트 해제되고 홈 위치가 드라이브와 다른 LSM에 있는 경우, ACSLS에서 가장 가까운 LSM(테이프 드라이브에서 가장 짧은 전달 거리)에 새 홈 위치를 지정하려고 합니다. 이것은 "Floating"이라고 하며 새 홈 위치의 카트리지입니다.

홈 디렉토리가 Extended Store 기능에 사용 설정된 LSM에 있는 경우 ACSLS에서는 마운트 해제 후에 홈 위치로 카트리지를 반환하려고 시도합니다.

주:

볼륨의 홈 LSM이 오프라인이거나 마운트 해제 시 레이블 불일치 등의 이벤트로 인해 ACSLS에서 카트리지를 홈 위치로 반환하지 않을 수 있습니다. 이러한 이벤트가 발생하는 경우 카트리지가 마운트 해제된 LSM으로부터 최대한 가까운 위치에 저장됩니다.

카트리지 아카이브에 사용하는 Extended Store LSM이 있는 경우 이 기능을 사용하면 해당 LSM의 카트리지를 해당 LSM의 홈 위치로 반환할 수 있습니다. 예를 들어, Extended Store LSM 3이 이 기능에 대해 사용으로 설정되어 있으며 LSM 1에 연결된 드라이브에 LSM 3의 카트리지가 마운트되는 경우, 카트리지의 마운트가 해제된 후 ACSLS에서 LSM 3의 홈 위치로 카트리지를 반환하려고 시도합니다. LSM 3을 사용 안함으로 설정한 경우 ACSLS에서 LSM 1의 새 셀에 카트리지를 저장하려고 합니다.

이 기능에 대해서는 전체 LSM만 사용으로 설정할 수 있습니다. 패널 또는 개별 셀과 같은 LSM 하위 구성 요소를 사용으로 설정할 수 없습니다. 이 기능에 대해 전체 ACS를 사용으로 설정하려면 ACS에서 각 LSM을 사용으로 설정해야 합니다.

주:

Extended Store 기능을 사용으로 설정하면 카트리지를 마운트 해제할 때 전달 활동이 증가합니다. 그러면 라이브러리 성능이 크게 저하될 수 있습니다.

Extended Store 기능에 대해 LSM을 사용으로 설정하려면 *lsm_fixed_volume* 파일 (*$ACS_HOME/data/external/fixed_volume* 디렉토리에 있음)을 수정합니다. 파일을 수정한 다음 지정된 LSM을 사용으로 설정하도록 ACSLS를 중지하고 다시 시작합니다.

샘플 파일을 수정할 때 다음 규약을 사용합니다.

- 파일 전체에서 설명과 빈 행을 사용할 수 있습니다.
- 각 LSM 식별자 행은 ACS 번호, 쉼표 및 LSM 번호 순으로 구성됩니다. ACS 번호와 LSM 번호 사이에는 공백이나 탭을 사용할 수 없습니다.
- 각 LSM 식별자는 개별 행에 있어야 합니다.
- LSM 식별자 행을 명시적으로 정렬하지 않아도 됩니다.
- 명시적 파일 끝 분리자가 필요하지 않습니다.

Extended Store 기능 제어 파일의 예제:

```
# This lsm_fixed_volume file must be found in the
# $ACS_HOME/data/external/fixed_volume
# directory. This is a sample lsm_fixed_volume file that may be
# edited your particular configuration.
# Comments may appear anywhere in this file, but must include a
# pound sign in the first column.
# Blanks lines are also allowed throughout the file for # readability, and
# will be ignored.
# For all the LSM identifiers found in this file, a ”best” attempt
# will be
# made to return the volume to its home location at dismount.
# A valid LSM identifier consists of the ACS number, separated bya comma,
# and followed by the LSM number. Leading or trailing blanks are
# ignored.
#    ACS,LSM
#    0,0
#    0,1
# ACS 1, LSM 0 through 3
#    ACS,LSM
#    1,0
#    1,1
#    1,2
#   1,3
```

다음 예제는 수정된 제어 파일입니다. 이 예제에서는 LSM 0.0 및 0.1이 사용으로 설정됨을 지정하기 위해 굵은체로 표시된 행에서 설명(#) 문자가 제거되었습니다.

```
# This lsm_fixed_volume file must be found in the
# $ACS_HOME/data/external/fixed_volume
# directory. This is a sample lsm_fixed_volume file that may be
# edited your particular configuration.
# Comments may appear anywhere in this file, but must include a
# pound sign in the first column.
# Blanks lines are also allowed throughout the file for
# readability, and
# will be ignored.
# For all the LSM identifiers found in this file, a ”best” attempt
```

```
# will be
# made to return the volume to its home location at dismount.
# A valid LSM identifier consists of the ACS number, separated by a comma,
# and followed by the LSM number. Leading or trailing blanks are
# ignored.
#     ACS,LSM
      0,0
      0,1
# ACS 1, LSM 0 through 3
#     ACS,LSM
#     1,0
#     1,1
#     1,2
#     1,3
```

혼합 매체 라이브러리 관리

혼합 매체를 사용하는 ACSLS에서는 동일한 라이브러리에 있는 테이프 드라이브 및 매체 (카트리지) 유형의 혼합을 지원합니다. ACSLS 혼합 매체 지원에서는 로봇이 테이프 드라이브에서 호환되지 않는 매체 유형을 마운트하지 못합니다. 예를 들어, SL8500에서 로봇은 T10000 테이프 드라이브에 LTO 카트리지를 마운트하지 않습니다.

ACSLS 혼합 매체를 지원하려면 카트리지에서 매체 문자를 포함하는 매체 ID 레이블이 필요합니다.

주:

드라이브 유형, 매체 유형 및 지원되는 드라이브와 매체 호환성의 최신 목록은 ACSLS Product Information Guide를 참조하십시오.

ACSLS에는 카트리지에서 사용되는 기록 형식의 정보가 제한되어 있습니다. ACSLS에서는 테이프 드라이브의 데이터 경로에 액세스하지 못하므로 ACSLS에서 기록 형식 비호환성을 발견하여 방지할 수 없습니다. 그러나 최신 T9840, T9940 및 T10000 테이프 드라이브에서 마운트 해제 시 매체 기록 형식을 보고합니다. ACSLS에서 데이터베이스에 이 정보를 저장하므로 다음과 같은 명령으로 정보를 표시할 수 있습니다.

```
display volume [vol_id(s)] –f recording_format_family recording_format_model
```

자세한 내용은 “display volume ”의 내용을 참조하십시오.

주:

9310 및 더 이상 사용되지 않는 기타 라이브러리의 경우 venter 명령을 사용하여 레이블이 지정되지 않은 카트리지를 라이브러리에 입력할 수 있습니다. venter 명령은 입력할 카트리지의 매체 유형을 지정하는 옵션을 제공하지 않습니다. 혼합 매체 환경에서 ACSLS는 가상으로 입력된 카트리지의 테이프 드라이브/매체 비호환성을 방지할 수 없습니다. 그러나 SL8500 및 SL3000 라이브러리는 volser 레이블이 없는 매체 레이블을 읽을 수 없으므로 venter에서는 이 라이브러리를 지원하지 않으며, ACSLS에서는 이러한 라이브러리에 카트리지를 입력하기 위해 매체 레이블이 필요합니다.

ACSLS 혼합 매체 설정 표시

drives_media.sh 유틸리티는 현재 ACSLS에서 지원되는 드라이브 유형, 매체 유형 및 드라이브와 매체 호환성 설정을 표시합니다. 새 드라이브와 매체에 대한 지원이 추가되므로 해당 드라이브와 매체가 표시됩니다.

ACSLS 혼합 매체 설정을 표시하려면 다음을 입력하십시오.

drives_media.sh

정보를 화면에 씁니다(표준 출력).

/tmp 디렉토리의 파일에 ACSLS 혼합 매체 설정을 출력하려면 다음을 입력하십시오.

drives_media.sh -f

세 개의 파일에 정보를 씁니다. (파일이 이미 있는 경우 파일을 덮어씁니다.)

```
/tmp/drive_types.txt
/tmp/media_types.txt
/tmp/media_compatibility.txt
```

SCSI 연결 LSM의 혼합 매체 제한사항

SCSI 연결 LSM의 다음과 같은 제한사항에 유의하십시오.

- 일부 테이프 드라이브에서는 제어 경로를 통한 동적 쓰기 보호 설정을 지원하지 않으므로 '읽기 전용' 옵션을 사용하여 카트리지를 마운트할 때 주의해야 합니다. 모든 StorageTek 드라이브에서 이 기능을 지원합니다. 비 StorageTek 드라이브의 경우 동적 쓰기 보호가 지원되는지 확인하는 것이 좋습니다. 특히 LTO 드라이브와 일부 초기 DLT 드라이브에서는 "읽기 전용" 옵션을 지원하지 않습니다.

  드라이브에서 이 기능을 지원하지 않는 경우, 읽기 전용 마운트의 쓰기 사용 안함 보호를 사용하여 카트리지를 마운트하면 데이터가 유실될 위험이 있습니다.
- ACSLS에서는 DLT 테이프 드라이브가 있는 SCSI 연결 라이브러리에 대해 *venter* 명령을 지원하지 않습니다.
- SCSI 연결 라이브러리에서 DLT 테이프 드라이브에 대한 가상 마운트 및 마운트 해제를 수행할 수 없습니다. 그러므로 ACSLS에서 이러한 카트리지를 관리할 수 있으려면 컴팩트 테이프 카트리지에 외부 레이블이 있어야 합니다.
- SCSI 연결 라이브러리의 자동 정리 작업은 ACSLS가 아니라 라이브러리 마이크로 코드에서 처리합니다. 라이브러리 제어판은 사용자가 자동 정리 작업의 라이브러리 제어를 수행할 수 있는 메뉴를 제공합니다. 자세한 내용은 해당 라이브러리 설명서를 참조하십시오.
- 라이브러리를 통해 정상 로드 또는 빠른 로드 옵션을 선택할 수 있습니다. 그러나 일부 테이프 관리 시스템에서는 빠른 로드 옵션을 지원하지 않습니다.

스크래치 환경 설정 지정

scratch mount 요청에서 사용할 매체 유형을 명시적으로 지정하거나 ACSLS를 통해 매체 유형을 선택할 수 있습니다.

ACSLS에서 매체 유형을 선택하도록 하려면 드라이브 유형마다 우선 순위가 지정된 호환 가능한 매체 유형 목록을 사전 정의해야 합니다. 이 목록은 "스크래치 환경 설정"이라고 합니다.

- 전체 서버의 환경 설정 세트는 하나가 있습니다. 환경 설정은 클라이언트에서 정의하지 않습니다.
- 드라이브의 호환 가능 매체 유형이 나열되지 않은 경우 매체가 선택되지 않습니다.

다음 절에서는 ACSLS에서 스크래치 환경 설정을 판별하는 데 사용하는 사용자 및 시스템 정의 파일에 대해 설명합니다.

사용자 정의 혼합 매체 파일

다음은 *$ACS_HOME/data/external/mixed_media/:*에 있는 사용자 정의 혼합 매체 파일입니다.

- *scratch_preferences.dat*

  사용자 정의 환경 설정 파일입니다. 환경 설정 정의의 기본 소스입니다.
- *scratch_preferences.SAMPLE*

  *scratch_preferences.dat file*을 작성하기 위해 샘플 환경 설정 파일을 복사할 수 있습니다.

시스템 정의 혼합 매체 파일

*drives_media.sh -f*를 사용하여 ACSLS 혼합 매체 설정 표시

다음에 저장된 매체 호환성 설정을 검토하십시오.

- */tmp/media_compatibility.txt*

  이 파일은 시스템 정의 호환성 설정 파일입니다. 사용자 정의 환경 설정 파일이 없거나 드라이브 유형이 누락된 경우에만 사용됩니다.
- */tmp/drive_types.txt*

  시스템에서 정의한 지원 드라이브 유형 목록
- */tmp/media_types.txt*

  시스템에서 정의한 지원 매체 유형 목록

ACSLS에서 혼합 매체 파일을 사용하는 방법

다음 표는 ACSLS에서 혼합 매체 파일을 사용하여 스크래치 마운트 요청의 매체 유형을 선택하는 방법을 설명합니다.

표 8.3. ACSLS에서 혼합 매체 파일을 사용하는 방법

***scratch_preferences.dat file*이 다음과 같은 경우**	**ACSLS에서 다음을 수행합니다.**
존재하지 않습니다.	*media_compatibility file* 시스템에서 정의 사용

***scratch_preferences.dat file*이 다음과 같은 경우**	**ACSLS에서 다음을 수행합니다.**
드라이브의 매체 유형을 두 개 이상 나열합니다.	나열된 순서대로 매체 유형 선택
특정 드라이브의 매체 유형을 나열하지 않습니다.	*media_compatibility file* 시스템에서 데이터 사용
특정 드라이브 유형을 나열하지 않습니다.	*media_compatibility file* 시스템에서 데이터 사용

스크래치 환경 설정 파일 정의

이 절차를 사용하여 *scratch_preferences.dat* 파일을 정의합니다. 이 파일에는 지정된 드라이브 유형에 대해 선택할 스크래치 카트리지 유형의 정렬된 목록이 포함되어 있습니다. ACSLS에서는 매체 유형이 명시적으로 지정되지 않은 *mount ** 명령에 이 파일을 사용합니다.

다음 예제는 *scratch_preferences.SAMPLE*의 컨텐츠를 보여줍니다.

```
Drive Type Name   Media Type Preference Name
4480                   3480
SD3                   DD3A
SD3                   DD3B
SD3                   DD3C
```

스크래치 환경 설정 파일을 정의하려면 다음을 수행하십시오.

1. *acsss*로 로그인하십시오.
2. 외부 혼합 매체 디렉토리로 변경:

   cd $ACS_HOME/data/external/mixed_media
3. 샘플 스크래치 환경 설정 파일을 복사하여 사용자 정의 파일 작성:

   cp scratch_preferences.SAMPLE scratch_preferences.dat
4. *vi*와 같은 텍스트 편집기를 사용하여 *scratch_preferences.dat* 파일에서 환경 설정 목록 수정:
   - 파일의 맨 위에 있는 설명의 지침을 따릅니다.
   - 드라이브 유형의 매체 유형을 두 개 이상 사용하려는 경우 개별 행에 각 매체 유형을 입력합니다. 환경 설정의 순서는 하향식입니다.
5. 파일을 저장합니다.
6. *cmd_proc*에서 ACSLS를 다시 시작합니다.

   start

이중 TCP/IP 연결 관리

이중 TCP/IP는 SL8500 및 SL3000용으로 구입할 수 있는 옵션입니다. 라이브러리에 대한 TCP/IP 연결을 두 개 제공합니다. 그러나 두 연결 중 하나만 작동하는 상태로 라이브러리를 계속 사용할 수 있습니다.

이중 TCP/IP의 용도는 실패한 통신 경로를 자동으로 인식하여 피하기 위한 것입니다. 이 작업은 자동으로 수행되므로 작동하지 않는 연결을 수동으로 전환할 필요가 없습니다. 자세한 내용은 “이중 TCP/IP 지원 ”의 내용을 참조하십시오.

이중 LMU 구성 관리

ACSLS는 다음으로 구성된 이중 LMU 구성을 지원합니다.

- ACS를 관리하는 **활성** LMU
- 활성 LMU가 실패하는 경우 ACS를 관리하는 **활성** 역할로 자동 전환하는 *standby* LMU입니다.

  두 LMU 모두 LSM에 연결된 LAN에 연결됩니다. 첫번째로 전원이 켜진 LMU는 처음에 활성 상태인 반면 두 번째로 전원이 켜진 LMU는 처음에는 대기 상태입니다. LMU가 주기적으로 서로의 상태를 확인하므로, 활성이 실패하면 대기가 활성 역할을 이어 받습니다.

  주:

  ACSLS에서는 호스트/LMU 마이크로 코드 호환성 레벨 12(이상)가 로드된 9330 및 L5530 LMU에 대해서만 이중 LMU 구성을 지원합니다. 두 LMU에 동일한 마이크로코드 레벨을 로드해야 합니다. ACSLS는 직렬 연결 또는 TCP/IP를 통해 이러한 LMU와 통신합니다. TCP/IP 연결 LMU마다 이더넷 연결은 하나뿐입니다.

  제한사항: ACSLS에서는 자동으로 LMU 전환을 시작하지 **않습니다**. ACSLS와 활성 LMU 사이의 통신을 유실한 경우에도 ACSLS에서 LMU 전환을 시작하지 않습니다. ACSLS가 기존 활성 LMU와 계속 통신하려고 시도합니다. Redundant Electronics(중복 전자 부품)이 있는 SL8500 또는 SL3000library와 관련된 ACSLS-HA 구성 외에는 ACSLS에서 LMU 전환을 자동으로 시작하지 않습니다.

ACSLS 이중 LMU 지원에는 다음이 포함됩니다.

- ACSLS가 활성 및 대기 LMU 모두에 연결됩니다. ACSLS에서 두 LMU 모두에 대한 연결을 계속 모니터링합니다. 통신이 유실되면 ACSLS가 이 조건을 보고합니다.
- 활성 LMU가 실패하면 대기 LMU가 새 활성 LMU의 기능을 자동으로 수행합니다. 이 경우, ACSLS가 자동 전환을 인식하여 새 활성 LMU에 요청을 보냅니다. ACSLS에서 진행중인 트랜잭션도 복구합니다(감사 제외).
- *switch lmu* 명령을 실행하여 활성 LMU에서 대기 LMU로 ACS 관리를 수동으로 전환할 수 있습니다. ACSLS에서는 대기 LMU에 ”활성으로 강제 스위치오버” 전송을 보냅니다. 대기가 새 활성 LMU 기능을 인계 받습니다. 수동으로 스위치오버한 후, ACSLS가 미해결 트랜잭션을 복구합니다(감사의 경우 제외).

직렬 이중 LMU 구성의 중복성을 향상시키려면 이중 직렬 케이블을 사용하여 ACSLS 서버를 각 LMU에 연결하는 것이 좋습니다.

LMU는 ACSLS 서버를 중지하지 않고도 IPL을 수행할 수 있으므로 ACSLS 이중 LMU를 지원하면 단일 LMU 구성도 향상됩니다.

query lmu 명령은 단일 LMU와 이중 LMU ACS 구성 모두의 포트 상태 및 LMU를 표시합니다. 자세한 내용은 “query lmu ”의 내용을 참조하십시오.

switch lmu 명령을 사용하여 ACS 활성 LMU에서 대기 LMU로 ACS 관리를 수동으로 전환할 수도 있습니다. 자세한 내용은 “switch lmu ”의 내용을 참조하십시오. 이중 LMU에 연결하는 LAN 케이블을 교체하는 등의 하드웨어 유지 관리를 위해 수동으로 스위치오버할 수 있습니다.

예를 들어, LMU A가 활성 역할을 수행하고 LMU B가 대기 역할을 수행한다고 가정합니다. LAN 케이블을 교체해야 하는 경우 다음을 수행할 수 있습니다.

1. LMU B로 전환합니다.
2. 오프라인 상태의 LMU A로 포트를 전환(*Vary*)합니다.
3. LAN 케이블을 교체합니다.
4. 다시 온라인 상태의 LMU A로 포트를 전환(*Vary*)합니다.
5. LMU A로 다시 전환합니다.

   LMU 스위치오버 후에(자동 또는 수동), ACSLS가 모든 미해결(활성 및 보류) 요청을 복구합니다. 스위치오버 중에 각 미해결 요청을 완료하는 시간은 LMU 사이를 전환하고 이전 미해결 요청을 복구하는 데 필요한 시간만큼 증가합니다. 그러므로 요청을 복구하는 데는 3 - 5분이 걸릴 수 있습니다.

이중 LAN 클라이언트 구성 관리

ACSLS 5.2 이상에서는 기본 LAN과 보조(백업) LAN으로 구성된 이중 LAN 클라이언트 구성을 지원합니다. 기본 LAN이 실패하면 클라이언트가 보조 LAN으로 전환됩니다. ACSLS는 해당 클라이언트에 대한 모든 미해결 메시지를 제거하고 보조 LAN을 사용하여 통신을 시작합니다. ACSLS에서 보조 LAN으로 전환하기 전에 미해결 메시지를 모두 제거하므로, 요청이 성공적으로 완료되어도 클라이언트에서 성공 메시지를 받지 못합니다.

예를 들어, 클라이언트에서 10개의 카트리지를 꺼내도록 ACSLS에 요청합니다. ACSLS가 카트리지 꺼내기를 시작하므로 ACSLS와 기본 클라이언트 LAN 사이의 통신에 실패합니다. ACSLS는 해당 클라이언트에 대한 모든 미해결 메시지를 제거하고 보조 LAN을 통해 통신을 시작합니다. ACSLS가 10개의 카트리지 꺼내기를 성공적으로 모두 완료했지만 클라이언트에 성공 메시지를 보내지 않습니다. 클라이언트에서 요청이 성공적으로 완료되었음을 검증해야 합니다. 이 예제에서 클라이언트가 꺼낸 볼륨의 ID에 대해 볼륨 쿼리 요청을 발행하면 ACSLS에서 *volume not found* 오류 메시지를 반환합니다. 이 메시지는 ACSLS가 카트리지를 꺼냈음을 확인합니다.

LAN 통신 스위치오버가 발생하는 경우 ACSLS에서 일시적인 요청이 성공적으로 완료되는지 확인합니다. 그러나 스위치오버 후에는 클라이언트가 기본 LAN에서 원래 제출된 영구적 미해결 요청(예: CAP 작업)을 취소하고 보조 LAN의 통신을 통해 이러한 요청을 다시 제출해야 합니다. 기본 LAN 통신을 통해 지정된 리소스(예: CAP, 잠금, 드라이브 등)는 보조 LAN으로 스위치오버된 후에 지정된 상태로 남아 있습니다.

다음 절에서는 다음을 수행하여 이중 LAN 클라이언트 작업을 위해 ACSLS를 구성하는 방법에 대해 설명합니다.

- *csc_ip_switch.dat* 파일을 작성하여 기본 및 보조 LAN의 IP 주소 등록

- 다중 홈 ACSLS 서버의 두번째 이더넷 포트 설치
- TCP/IP 연결 시간 초과 간격을 설정하여 시스템이 백업 LAN으로 스위치오버하는 시간을 줄입니다.

기본 및 보조 LAN의 IP 주소 등록

ACSLS에 기본 및 보조 LAN의 IP 주소를 등록하려면 *$ACSSS_HOME/data/internal/client_config/* 디렉토리에 *csc_ip_switch.dat* 파일을 작성합니다. 다음은 *csc_ip_switch.dat* 파일의 예제입니다.

```
#The following entry is System Zed's primary and secondary LAN IP addresses.
129.80.30.40  129.80.30.50
```

위의 예제에 표시된 대로 설명 앞에 # 기호가 옵니다. 항목의 왼쪽 열에는 클라이언트 시스템의 LAN IP 주소 다음에 하나 이상의 공백이 오고 오른쪽 열에는 클라이언트 시스템의 보조 LAN IP 주소가 있습니다. 이 예제에서 System Zed의 기본 LAN IP 주소는 *129.80.30.40*이고 보조 LAN IP 주소는 *129.80.30.50.* 입니다.

csc_ip_switch.dat 파일을 작성하거나 업데이트한 다음, ACSLS가 실행 중이면 ACSLS를 중지하고 다시 시작해야 합니다.

다시 시작 시 ACSLS에서 *csc_ip_switch.dat* 파일을 성공적으로 읽으면, ACSLS가 ACSLS 이벤트 로그에 *2010 I* 이중 경로 옵션이 활성화됨이라는 성공 메시지를 로깅합니다. 그렇지 않으면 이중 LAN 지원이 활성화되지 않습니다.

다중 홈 ACSLS 서버의 두번째 이더넷 포트 설치

이 절에서는 멀티 홈 ACSLS 서버의 두번째 이더넷 포트를 설치하는 절차를 설명합니다. 두번째 이더넷 포트는 백업 LAN을 제어하는 두번째 제어 경로 어댑터에 대한 연결을 제공합니다. 이 절차를 수행하려면 SBUS Buffered Ethernet 카드가 필요합니다. Oracle에서 이더넷 카드를 부품 번호 X1053A로 주문할 수 있습니다.

설치 절차는 다음과 같습니다.

- 하드웨어 설치 및 커널 재구성
- 새 이더넷 포트의 호스트 이름 정의
- */etc/notrouter* 파일 작성

두번째 이더넷 포트를 설치하려면 다음을 수행하십시오.

1. 제조업체 지침에 따라 SBus Buffered Ethernet 카드를 설치합니다.
2. 새 장치의 시스템 커널을 재구성합니다.
   a. 시스템의 전원을 켭니다. 부트하기 시작하면 *[[STOP]]-[[A]]*를 눌러 PROM 모니터를 입력합니다.
   b. *ok* 프롬프트에서 다음과 같이 서버를 부트합니다.

```
boot -r
```

3. 부트가 완료되면 *root*로 로그인합니다.
4. 두번째 이더넷 포트의 호스트 이름을 작성합니다.

   ```
   echo 2nd_host_name > /etc/hostname/hme1
   ```

   여기서 *2nd_host_name*은 두번째 이더넷 포트의 호스트 이름입니다.
5. 다음 명령을 입력합니다.

   ```
   touch /etc/notrouter
   ```

   /etc/notrouter 파일을 작성합니다.
6. 서버 재부트:

   ```
   reboot
   ```

절차를 완료합니다.

TCP/IP 연결 시간 초과 간격 설정

UNIX 시스템 변수 *tcp_ip_abort_cinterval*이 클라이언트와 ACSLS 서버 사이의 TCP/IP 연결 시간 초과 간격을 설정합니다. 이 변수의 기본값(180초)을 변경하면 백업 LAN으로 스위치오버하는 시간이 줄어들 수 있습니다. 그러나 실제 스위치오버 시간은 구성과 실패 유형에 따라 달라집니다.

예를 들어, ACSLS 서버 포트가 실패하고 여러 클라이언트가 이 포트를 통해 통신 중인 경우 ACSLS에서 각 클라이언트와의 통신을 순차적으로 복구합니다. 그러므로 실패하는 포트의 여러 클라이언트를 복구할 때는 실패한 포트와 통신하는 단일 클라이언트를 복구할 때보다 시간이 오래 걸립니다.

백업 LAN으로의 스위치오버 시간을 줄이려면 다음을 수행합니다.

1. ACSLS 서버에서 루트로 로그인합니다.
2. 프롬프트에서 다음 명령을 입력합니다.

   ```
   /usr/sbin/ndd -set /dev/tpc tcp_ip_abort_cinterval 15000
   ```

   이 명령은 TCP/IP 연결 시간 초과 간격을 15초(기본값은 180초)로 변경합니다.

   힌트: 이 명령을 영구적으로 만들려면(모든 서버 재부트 전체에서) "구성 가능 매개변수 설정" 절에서 */etc/rc2.d/S69inet* 파일에 명령을 추가합니다.

이벤트 알림 등록

이벤트 알림을 사용하면 ACSAPI 클라이언트의 테이프 라이브러리에서 발생하는 이벤트를 추적할 수 있습니다. 이 기능은 CSC 툴킷에서 제공합니다. 특히 이벤트 알림을 사용하여 ACSAPI 클라이언트에서 다음을 수행할 수 있습니다.

- 라이브러리 리소스 이벤트 및/또는 볼륨 이벤트 등록
- 이러한 이벤트 등록 취소
- 라이브러리 이벤트의 등록 상태를 확인하고 이벤트 발생 시 알림

등록이 삭제될 때까지 클라이언트 등록 요청 및 이벤트 알림 메시지가 클라이언트에 전달됩니다. 이벤트 알림이 클라이언트가 활성인지 검증하기 위해 클라이언트의 등록 상태를 주기적으로 확인합니다. 그러면 더 이상 활성 상태가 아닌 클라이언트에 응답을 보내지 않고 불필요한 네트워크 리소스 사용을 방지합니다.

다음 유형의 이벤트를 추적할 수 있습니다.

- 볼륨 추가와 삭제 및 ACSLS 데이터베이스에서 카트리지를 정리하기 위해 최대 사용이 초과되는 경우.
- 온라인에서 오프라인, 진단 또는 복구로 변경되는 LSM 또는 드라이브 등의 라이브러리 구성 요소 상태 변경 또는 열렸거나 닫힌 CAP.
- 작동 불가능한 로봇손 등의 하드웨어 장애

원하는 상태로 자동 복구

ACSLS에서 이제 사용자가 원하는 가용성 상태로 테이프 라이브러리와 드라이브를 복원합니다. 이 작업은 ACS, 포트, LSM 및 테이프 드라이브의 현재 상태 및 원하는 상태 모두를 추적하여 수행합니다. 라이브러리 또는 드라이브를 ACSLS에서 더 이상 액세스할 수 없거나 작동하지 않는 경우 ACSLS가 현재 상태를 오프라인으로 변경합니다. 라이브러리나 드라이브가 액세스 가능해지거나 다시 작동 가능하게 되면 ACSLS에서 자동으로 복구하고 원하는 상태가 온라인인 경우 다시 온라인 상태로 만듭니다.

현재 상태 및 원하는 상태

- 원하는 상태가 라이브러리와 테이프 드라이브 가용성을 관리합니다. 원하는 상태는 ACS, 포트 연결, LSM 또는 테이프 드라이브의 원하는 가용성입니다. 모든 라이브러리 구성 요소의 원하는 초기 상태는 온라인입니다. 명시적 vary 명령을 사용하여 원하는 상태를 설정할 수 있습니다. (*cmd_proc*, ACSLS GUI 또는 ACSAPI 클라이언트와 다릅니다.) 라이브러리 상태가 변경되어 ACSLS에서 내부적으로 생성하는 *vary*가 아닙니다. query lmu 및 display 명령을 사용하여 라이브러리 구성 요소의 원하는 상태를 볼 수 있습니다.
- ACS, 포트, LSM 또는 드라이브의 현재 상태("상태"로 지정됨)는 원하는 상태로 제한되는 구성 요소의 현재 가용성입니다. 현재 상태는 라이브러리 구성 요소가 준비되어 통신 중인지 나타내며, 원하는 구성 요소 상태 및 상위 레벨 구성 요소로 제한됩니다.

  예를 들어, LSM의 원하는 상태가 온라인이지만 준비되지 않은 경우 현재 상태는 오프라인입니다. LSM이 다시 준비가 되면 라이브러리에서 ACSLS에 메시지를 보내고 ACSLS가 LSM을 자동으로 복구한 다음 현재 상태를 다시 온라인으로 만듭니다.

  그러나 LSM의 원하는 상태가 오프라인이면 ACSLS에서 현재 상태를 오프라인으로 설정합니다. LSM이 준비되지 않음 상태가 된 후 다시 준비된 상태가 되면 ACSLS에서 LSM의 현재 상태를 오프라인으로 둡니다.

현재 상태는 아래로 캐스케이딩되고 원하는 상태는 그렇지 않음

라이브러리 구성 요소의 현재 상태는 하위 레벨 구성 요소로 캐스케이딩됩니다.

- 특히 ACS에 더 이상 액세스할 수 없으면 ACS의 현재 상태가 오프라인입니다. 모든 LSM 및 드라이브에 액세스할 수 없으므로 이 또한 현재 상태가 오프라인으로 설정됩니다.
- *vary* 명령을 사용하여 ACS의 원하는 상태를 오프라인으로 변경하면 ACS의 현재 상태가 오프라인으로 설정되고 ACS에 있는 모든 드라이브와 LSM의 상태가 오프라인으로 설정됩니다.
- 마찬가지로 LSM의 현재 상태가 오프라인으로 변경되면 LSM에 있는 모든 드라이브의 현재 상태가 오프라인이 됩니다. 드라이브의 원하는 상태는 변경되지 않습니다.

라이브러리 구성 요소의 원하는 상태를 변경해도 낮은 레벨 구성 요소의 원하는 상태에는 영향을 미치지 않습니다.

- ACS의 원하는 상태를 변경해도 ACS의 드라이브와 LSM의 원하는 상태에는 영향을 미치지 않습니다.
- LSM의 원하는 상태를 변경해도 LSM에 있는 드라이브의 원하는 상태에는 영향을 미치지 않습니다.
- 따라서 LSM에서 선택한 드라이브를 액세스할 수 없도록 오프라인 상태로 변경할 수 있습니다. 그런 다음 유지 관리를 위해 LSM을 오프라인으로 전환(*vary*)할 수 있습니다. LSM을 다시 온라인으로 전환(*Vary*)하면 선택된 드라이브가 오프라인 상태로 남는 반면, 다른 드라이브는 다시 온라인 상태가 됩니다.

실제 라이브러리와 드라이브의 가용성은 논리 라이브러리와 드라이브에 영향을 미칩니다.

- 논리적 라이브러리와 논리적 라이브러리 내 테이프 드라이브의 가용성은 기본 물리적 라이브러리와 논리적 라이브러리 모두에 대해 설정하는 원하는 상태에 의해서도 제어됩니다.
- 물리적 라이브러리와 논리적 라이브러리 모두에 대한 원하는 상태가 온라인인 경우 논리적 라이브러리와 논리적 테이프 드라이브의 현재 상태에 기본 물리적 라이브러리 및 드라이브의 현재 상태가 반영됩니다.
- 그러나 실제 ACS 또는 드라이브의 원하는 상태는 온라인이지만 논리 라이브러리나 드라이브의 원하는 상태가 오프라인이면, 논리 라이브러리나 드라이브가 오프라인 상태로 남으며 사용 불가능합니다.

라이브러리가 임시로 사용 불가능한 경우 마운트와 마운트 해제 큐에 넣기 및 재시도

임시 라이브러리 정전이 감지되면 ACSLS에서 마운트 및 마운트 해제 요청을 큐에 넣습니다. 필요한 모든 라이브러리 구성 요소의 원하는 상태가 온라인이면, 모든 소스의 마운트 및 마운트 해제 요청을 자동으로 큐에 넣고 재시도합니다. 즉, 다음 소스로부터의 마운트 및 마운트 해제 요청을 모두 자동으로 큐에 넣고 재시도합니다.

- ACSAPI 클라이언트

- *cmd_proc*
- 논리 라이브러리에 있는 테이프 드라이브의 섬유 연결 클라이언트

라이브러리 하드웨어의 원하는 상태가 **온라인**이지만 현재 상태가 **오프라인**이면 임시 정전이 발생합니다. 임시 정전의 예로는 LSM 도어가 열려 있는 경우, ACSLS에서 라이브러리와의 통신을 유실한 경우 또는 LC 전환 작업 중이 있습니다. 임시 라이브러리 또는 테이프 드라이브 정전 중에 마운트 및 마운트 해제를 큐에 넣고 라이브러리가 사용 가능하게 되면 재시도합니다.

라이브러리 하드웨어의 원하는 상태가 오프라인이면 ACSLS가 마운트 또는 마운트 해제 요청에 실패하고 해당 오류 상태가 표시됩니다.

ACSLS 7.3.1에서, 마운트 및 마운트 해제 큐에 넣기 및 재시도를 관리하기 위해 두 개의 동적 변수, M*OUNT_RETRY_DELAY* 및 *MOUNT_RETRY_TIME_LIMIT*가 도입되었습니다. 각 매개변수는 다음을 수행합니다.

- *MOUNT_RETRY_DELAY*는 큐에 넣은 마운트와 마운트 해제를 재시도하는 빈도 또는 라이브러리와 드라이브의 가용성을 다시 확인하는 빈도를 제어합니다.
- *MOUNT_RETRY_TIME_LIMIT*는 마운트 및 마운트 해제를 큐에 넣고 재시도하는 시간 제한입니다. 이 다음에 요청이 실패합니다.

테이프 드라이버를 이동, 추가 또는 제거할 때 ACSLS 재구성

위치 간에 테이프 드라이브가 스왑되고 라이브러리의 테이프 드라이브를 다른 테이프 드라이브로 바꿀 때마다 ACSLS 데이터베이스의 드라이브 일련 번호와 드라이브 테이프를 업데이트하도록 ACSLS를 재구성해야 합니다. 라이브러리에 삽입하거나 제거할 때 테이프 드라이브를 추가하거나 제거하도록 ACSLS를 재구성해야 합니다. 매체 검증 풀에 테이프 드라이브를 추가하면 ACSLS에서 액세스하지 못하도록 제거되며, 매체 검증 풀에서 테이프 드라이브를 제거하면 ACSLS에서 액세스할 수 있습니다. 이러한 경우에도 ACSLS를 재구성해야 합니다.

테이프 드라이브가 기존 드라이브를 대체하는 경우, ACSLS가 라이브러리에서 테이프 드라이브 상태를 읽을 때까지 드라이브 유형과 일련 번호가 업데이트되지 않습니다. 이 동작은 다음 시점에 발생합니다.

- ACSLS 시작.
- ACS 또는 LSM이 준비되지 않음 상태가 된 다음 준비됨 상태가 되면, ACSLS가 라이브러리를 복구합니다.
- 고객이 ACS, LSM 또는 테이프 드라이브를 오프라인으로 변경한 후 다시 오프라인으로 변경할 때.
- 테이프 드라이브, LSM 또는 ACS를 재구성할 때.

테이프 드라이브를 추가하거나 제거할 때, ACSLS 데이터베이스의 드라이브를 추가하거나 삭제하려면 ACSLS 구성을 업데이트해야 합니다. ACSLS 테이프 드라이브 구성을 업데이트하면 마운트 시 오류가 방지되고 잘못된 테이프 드라이브에 카트리지가 마운트되지 않습니다.

ACSLS에 구성된 테이프 드라이브 업데이트

라이브러리에서 테이프 드라이브를 이동하거나 대체한 경우 동적 구성을 사용하여 드라이브 테이프와 일련 번호를 업데이트합니다. 한 테이프 드라이브가 동일한 위치에 있는 다른 테이프 드라이브를 대체하는 경우 업데이트할 때 고객의 확인이 필요하지 않습니다. 테이프 드라이브를 라이브러리에서 제거하거나 라이브러리에 삽입하는 경우 고객이 구성 변경을 확인해야 합니다.

테이프 드라이브를 업데이트할 때:

- *config* 요청을 발행하기 전에 영향받는 모든 구성 요소가 준비되었는지 확인합니다.
- ACSLS를 사용으로 설정한 경우 동적 구성을 사용하여 ACSLS 데이터베이스 업데이트가 완료됩니다. 동적 구성은 중단되지 않으며, 구성을 업데이트하는 동안 ACSLS가 요청 처리를 계속할 수 있습니다.
- *config lsm* 또는 *config acs*를 발행한 다음 영향받는 LSM 또는 ACS를 감사하는 것이 좋습니다.

Unix 명령 프롬프트에서 동적 구성 유틸리티 명령을 실행하여 테이프 드라이브 구성을 업데이트합니다. *acsss*로 로그인해야 합니다.

- *config drive* <panel_id>

  변경이 단일 패널 또는 SL8500 레일에 있는 테이프 드라이브에만 영향을 미치는 경우 *config drive* <panel_id>를 사용하여 패널에 있는 모든 테이프 드라이브의 드라이브 구성을 업데이트합니다.
- *config lsm* <lsm_id>

  config lsm <lsm_id>를 사용하여 두 개의 드라이브 패널이 있는 SL3000의 모든 테이프 드라이브에 대한 드라이브 구성을 업데이트합니다.

  주:

  config lsm <lsm_id>가 LSM의 CAPs 및 스토리지 용량도 업데이트한 다음 LSM을 감사해야 합니다.
- *config acs* <acs_id>

  config acs <acs_id>를 사용하여 ACS의 모든 구성을 업데이트합니다(예: SL8500 라이브러리 복합기).

  주:

  config acs <acs_id>도 전체 ACS의 CAPs 및 스토리지 용량도 업데이트한 다음 ACS를 감사해야 합니다.

매체 검증

매체 검증을 통해 고객들이 SLConsole 또는 STA(StorageTek Tape Analytics)를 사용하여 모든 T10000 테이프 카트리지 유형을 확인할 수 있습니다. T10000C 및 T10000D 드라

이브의 전용 "매체 검증 풀"이 사용됩니다. 매체 검증 풀의 드라이브는 ACSLS에 사용할 수 없습니다. ACSLS에서 아직 드라이브가 구성되지 않은 경우 ACSLS에서 드라이브에 구성하려고 하면, 라이브러리에서 "드라이브가 설치되지 않음"으로 보고합니다.

매체 검증 풀에 드라이브 추가

드라이브를 ACSLS 제어에서 제거하고 매체 검증 드라이브 풀에 추가하면, 드라이브를 포함하는 LSM(SL8500 레일 또는 SL3000 라이브러리)이 먼저 ACSLS에 대해 준비되지 않음 상태가 된 다음 준비됨 상태가 됩니다. ACSLS가 라이브러리에서 구성이 변경됨 메시지도 받습니다.

ACSLS 호스트가 드라이브를 오프라인 상태로 자동 업데이트합니다. 드라이브가 매체 검증 풀에 남아 있으면 *config drives* <panel_id> 유틸리티를 사용하여 ACSLS 구성에서 드라이브를 제거합니다.

주:

라이브러리가 온라인 상태이고 다른 장치로 마운트 및 마운트 해제가 발생하는 동안 *config drives*를 실행할 수 있습니다.

매체 검증 풀에서 드라이브 제거

매체 검증 풀에서 드라이브를 제거한 후:

- 분할되지 않은 라이브러리의 경우, ACSLS에서 즉시 드라이브 슬롯을 사용할 수 있습니다.
- 분할된 라이브러리의 경우 검증 풀에서 제거된 드라이브 슬롯은 파티션에 지정되지 않습니다. SL 콘솔을 사용하여 드라이브 슬롯을 파티션에 지정합니다.

드라이브가 매체 검증 드라이브 풀에서 제거되어 호스트에 사용 가능하게 되면 라이브러리가 ACSLS에 구성이 변경됨 메시지를 보냅니다.

드라이브가 ACSLS 구성에 있으면 드라이브를 온라인 상태로 변경합니다. 드라이브가 ACSLS 구성에 없으면 *config drives* <panel_id> 유틸리티로 추가합니다.

주:

라이브러리가 온라인 상태이고 다른 장치로 마운트 및 마운트 해제가 발생하는 동안 *config drives*를 실행할 수 있습니다.

9

9장. 카트리지 관리

ACSLS는 정교한 카트리지 관리 기능을 제공합니다. 이러한 기능은 여러 가지 방법으로 제공됩니다.

- 손실된 카트리지의 복구와 같이 자동으로 제공
- 존재하지 않는 카트리지 및 꺼낸 카트리지에 대한 정보 보존과 같이 기본적으로 사용으로 설정
- 카트리지가 감사에 의해 데이터베이스에 추가될 때 또는 카트리지를 CAP를 통해 넣을 때 볼륨 속성 지정과 같이 고객이 정의

적절한 카트리지 관리 기능을 사용하면 ACSLS가 제공하는 가치가 향상됩니다.

카트리지 관리는 다음 항목으로 구성됩니다.

LSM 채우기

카트리지는 라이브러리가 오프라인 상태이거나 CAP를 통해 라이브러리에 넣을 때 수동으로 셀에 배치할 수 있습니다.

라이브러리 및 ACSLS가 제대로 작동하기 위해서는 마운트 해제, 통과 및 꺼내기 작업을 수행할 수 있도록 각 LSM에 사용 가능한 사용 가능한 셀이 몇 개 있어야 합니다. 각 LSM에 설치된 각 테이프 드라이브에 대해 사용 가능한 셀을 하나 이상 보유해야 합니다.

LSM의 사용 가능한 셀 개수를 확인하려면 다음 명령을 실행합니다.

```
query lsm lsm_id
```

SL8500에서 각 레일은 LSM으로 정의됩니다.

CAP 사용

다음 절에서는 CAP 유형, 상태, 모드 및 우선 순위를 검토합니다.

CAP 유형

CAP의 각 유형에는 카트리지와 함께 로드하기 위한 표준 용량 및 방법이 있습니다. LSM에 둘 이상의 CAP 유형이 있을 수 있습니다. 다음 표에는 지원되는 CAP 유형, 식별자 및 용량, 로드 방법이 나와 있습니다.

표 9.1. CAP 유형

CAP 유형	식별자 및 용량	로드 방법
StorageTek VTL	CAP 0: 카트리지 20개 보유	감사를 사용하여 가상 볼륨을 검색합니다. “VTL 동작”을 참조하십시오.
SL3000	CAP 6 및 선택적 CAP 1-5, CAP 7-10: 각각 카트리지 26개씩 보유	CAP에 로드된 이동식 매거진 2개에 카트리지가 13개씩 배치됩니다.
SL8500 회전식	CAP 0 및 선택적 CAP 1: 각각 카트리지 39개 보유	CAP에 로드된 이동식 매거진 3개에 카트리지가 13개씩 배치됩니다.
SL8500 대량	CAP 0 및 CAP 1: 각각 카트리지 33개 또는 36개 보유	CAP에 로드된 3개의 이동식 매거진에 카트리지가 11개 또는 12개씩 배치됩니다. “대량 CAP”를 참조하십시오.
SL500	CAP 0: 카트리지 5-25개 보유	CAP에 로드된 이동식 매거진에 카트리지 5개가 배치됩니다. 기본 모듈의 매거진 1개, CAP를 포함하는 확장 모듈의 매거진 2개입니다.
L180	CAP 0: 카트리지 10개 보유	CAP에 로드된 이동식 매거진 2개에 카트리지가 5개씩 배치됩니다.
L700	CAP 0 및 선택적 CAP 1: 각각 카트리지 20개 보유	CAP에 로드된 이동식 매거진 2개에 카트리지가 4개씩 배치됩니다.
개선됨(4410 및 9310)	CAP 0 및 CAP 1: 각각 카트리지 40개 보유	CAP에 로드된 이동식 매거진에 카트리지가 배치됩니다.
9360	CAP 0: 카트리지 20개 보유, 선택적 CAP 1: 카트리지 30개 보유	CAP에 로드된 이동식 매거진에 카트리지가 배치됩니다.
우선 순위(PCAP)	CAP 2: 카트리지 1개 보유	카트리지를 한 번에 하나씩 CAP에 직접 넣습니다.

CAP 유형	식별자 및 용량	로드 방법
9710 또는 9740 CAP	CAP 0은 카트리지 14개를 보유하거나 카트리지 10개가 있는 매거진을 보유합니다.	카트리지는 CAP에 로드된 이동식 매거진에 배치되거나 CAP 셀에 직접 로드됩니다.
9714, 9730 또는 9738 CAP	CAP 0: 카트리지 1개 보유	카트리지를 단일 셀 CAP에 직접 로드합니다.
레거시 4400	CAP: 카트리지 21개 보유	카트리지를 CAP 셀에 직접 로드합니다.

CAP 상태

CAP 상태는 카트리지를 넣고 꺼낼 수 있는지 여부를 결정합니다. 다음 표에서는 유효한 CAP 상태에 대해 설명합니다. CAP 상태를 확인하는 절차는 “CAP 정보 표시 ”를 참조하십시오. 장치 상태 변경에 대한 자세한 내용은 “query pool ” 명령을 참조하십시오.

주:

SL8500 라이브러리에 대한 자세한 내용은 “SL8500 내부 주소 및 ACSLS 주소 이해 ”를 참조하십시오. SL500 라이브러리에 대한 자세한 내용은 “SL500 CAP 동작 ”을 참조하십시오.

표 9.2. CAP 상태

상태	설명	요청 처리 방법
online	정상 작동 상태입니다.	모든 요청이 수락되고 처리됩니다.
offline	CAP가 논리적으로 사용 안함으로 설정됩니다.	모든 요청이 거부됩니다.
offline-pending	전환 상태입니다. CAP가 온라인에서 오프라인으로 전환될 때 발생합니다.	모든 새 요청이 거부됩니다. 현재 및 보류 중인 요청이 완료 상태로 처리됩니다.
diagnostic	클라이언트 응용 프로그램의 간섭 없이 진단 작업에 CAP를 사용할 수 있습니다.	클라이언트 응용 프로그램의 요청이 거부됩니다. *cmd_proc*의 요청이 처리됩니다.
recovery	전환 상태입니다. CAP가 오프라인에서 온라인으로 전환될 때 발생합니다.	새 요청이 거부됩니다.

CAP 모드

CAP 모드는 CAP가 카트리지 넣기 및 꺼내기에 사용되는 방식을 제어합니다. 다음 표에서는 유효한 CAP 모드에 대해 설명합니다. CAP 모드를 확인하는 절차는 “CAP 정보 표시 ”를 참조하십시오. CAP 모드 변경에 대한 자세한 내용은 “query cap ”을 참조하십시오.

HINT: CAP가 사용 중인 동안 CAP 모드를 변경할 수 없습니다. 즉, 수동 또는 자동 넣기 작업 중 도어가 열려 있는 경우 넣기 작업을 완료할 때까지는 해당 모드를 변경할 수 없습니다.

표 9.3. CAP 모드

모드	설명	넣기/꺼내기에 미치는 영향
수동	사용 중이 아닐 때 CAP는 잠겨 있습니다. 이는 모든 다중 카트리지 CAP의 초기 모드입니다.	명령을 명시적으로 실행한 후에만 카트리지를 넣거나 꺼낼 수 있습니다. 이전에 정의된 CAP 우선 순위에 따라 명령에 `cap_id`를 지정하거나 ACSLS에서 자동으로 CAP를 선택하도록 허용합니다.

모드	설명	넣기/꺼내기에 미치는 영향
		일부 클라이언트 응용 프로그램의 경우 CAP가 수동 모드여야 합니다. 테이프 관리 시스템에 대한 설명서를 참조하십시오.
자동	사용 중이 아닐 때 CAP는 잠금 해제되어 있습니다. 이는 모든 우선 순위 CAP의 초기 모드입니다. 분할된 라이브러리에서 CAP 모드를 자동으로 설정할 수 없습니다. 이에 대한 예외는 자동 모드로 설정할 수 있는 SL3000의 전용 CAP(분할 영역 하나에만 지정됨)입니다. SL8500 액세스 도어가 열리거나 닫혀 있을 때 SL8500은 CAP를 잠긴 상태로 둡니다. CAP가 잠겨 있으면 자동 모드 넣기에 사용할 수 없습니다.	*enter* 명령을 명시적으로 실행하지 않고 카트리지를 넣을 수 있습니다. CAP 도어를 열고, 카트리지를 안에 두고, CAP를 닫으면 넣기가 시작됩니다. *cancel* 명령을 사용하여 진행 중인 자동 넣기 작업을 취소(*cancel*)할 수 없습니다. 진행 중인 자동 넣기를 종료하려면 다음을 수행합니다. *CAP* 도어가 열린 경우 모든 카트리지를 제거하고 도어를 닫습니다. *CAP* 도어가 닫혀 있고 카트리지를 라이브러리로 이동 중인 경우 남아 있는 카트리지를 라이브러리에 넣도록 허용합니다. 그러면 *enter* 작업이 종료됩니다. 카트리지를 꺼내려면 *eject* 명령을 명시적으로 실행해야 합니다. 이전에 정의된 CAP 우선 순위에 따라 명령에 `cap_id`를 지정하거나 ACSLS에서 자동으로 CAP를 선택하도록 허용할 수 있습니다. ACSLS가 CAP를 자동 모드로 표시하지만 CAP가 잠겨 있어 열리지 않고 자동 넣기에 사용할 수 없는 경우 ACSLS 및 SL8500을 동기화한 다음 CAP를 자동 넣기로 되돌립니다. *set cap mode manual* `cap_id` *set cap mode automatic* `cap_id`

CAP 우선 순위

CAP 우선 순위는 CAP 요청이 CAP ID에 별표(*)를 지정할 때 ACSLS가 자동으로 CAP를 선택하는 방식을 지정합니다. 다음 표에서는 CAP 우선 순위 및 해당 영향에 대해 설명합니다. CAP 우선 순위를 확인하는 절차는 “CAP 정보 표시 ”를 참조하십시오. CAP 우선 순위를 변경하는 방법에 대한 자세한 내용은 “query cap ” 명령을 참조하십시오.

표 9.4. CAP 우선 순위

우선 순위	영향
16(가장 높음)	처음에 사용
15(다음으로 가장 높음)	다음에 사용
-	
1(가장 낮음)	마지막에 사용
0	자동 선택되지 않음(모든 CAP의 초기 우선 순위)

CAP 우선 순위 및 자동 CAP 선택은 다음 명령에 적용됩니다.

- *audit*
- *eject*
- *enter*
- *venter*

cap_id의 전체 또는 일부에 대해 이러한 명령을 별표(*)와 함께 입력하면 ACSLS는 요청에서 지정된 각 ACS 또는 LSM에 대한 0이 아닌 가장 높은 우선 순위를 가진 사용 가능한 CAP를 자동으로 선택합니다.

예:

- *audit * server*

  ACSLS는 각 ACS에서 0이 아닌 가장 높은 우선 순위 CAP를 선택합니다.
- *enter 0,1,**

  ACSLS는 LSM 0,1에서 0이 아닌 가장 높은 우선 순위 CAP를 선택합니다.

CAP 정보 표시

다음은 *query cap* 명령을 사용하여 현재 CAP 정보를 표시하는 작업에 대한 몇 가지 지침입니다.

- 선택한 CAP에 대한 정보를 표시하려면 다음을 입력합니다.

  query cap cap_id cap_id ...
- 라이브러리의 모든 CAP에 대한 정보를 표시하려면 다음을 입력합니다.

  query cap all

카트리지 넣기

카트리지를 수동 또는 자동으로 넣도록 선택할 수 있습니다.

- 카트리지를 수동으로 넣으려면 *enter* 명령을 실행해야 합니다. 그러면 카트리지를 넣을 수 있도록 CAP 잠금이 해제됩니다.
- 자동 모드인 CAP를 열면 자동 넣기가 시작됩니다. CAP가 자동 모드이면 enter 명령을 실행할 필요가 없습니다.

다음 단계에서는 넣기 프로세스에 대해 설명합니다.

1. 넣기를 시작하면 CAP가 잠금 해제되고 예약됩니다. 다른 호스트가 해당 CAP를 사용할 수 없습니다.
2. CAP를 연 후 카트리지를 CAP에 넣고 CAP를 닫습니다. 이제 CAP가 잠깁니다.

   ACSLS 라이브러리 로봇이 CAP의 카트리지를 검사/감사합니다. 넣기 중인 모든 카트리지에는 이미 이 ACSLS 서버에서 관리되는 다른 *vol_ids*와 중복되지 않는 유효한 외부 레이블이 있어야 합니다.

   주:

   가상 넣기를 통해 레이블이 없는 카트리지를 일부 라이브러리에 넣을 수 있습니다.
3. ACSLS는 라이브러리의 홈 셀을 유효한 카트리지에 할당하고 지정된 홈 셀 위치로 이동합니다.

중복 카트리지 및 외부 레이블이 없는 카트리지는 CAP에 남아 있으므로 제거해야 합니다.

4. 완료 시 CAP가 잠금 해제되므로 카트리지를 더 넣을 수 있습니다.
   - CAP가 자동 모드인 경우 자동 넣기가 완료되고 CAP가 예약 해제되어 사용 가능한 상태가 됩니다.
   - CAP가 수동 넣기인 경우 CAP는 계속 수동 넣기로 예약되어 있습니다. 수동 넣기를 종료하려면 넣기가 시작된 *cmd_proc*에서 *cancel* 명령 또는 *Ctrl + c*를 사용하여 해당 넣기를 취소합니다.

enter 명령에 대한 자세한 내용은 “enter ”를 참조하십시오.

주:

카트리지 추적이 사용으로 설정된 경우 이벤트 로그가 모든 카트리지 넣기를 기록합니다.

표 9.5. 카트리지 넣기 명령

작업	명령
자동 모드로 카트리지 넣기	*set cap mode automatic* cap_id
수동 모드로 카트리지 넣기	*enter* cap_id
가상 레이블이 있는 카트리지 넣기(venter)	*venter cap_id vol_id vol_id* LSM 도어를 열고 누락되거나 읽을 수 없는 레이블이 있는 카트리지를 스토리지 셀에 놓지 마십시오. ACSLS에서 이러한 카트리지를 관리할 수 없습니다. 감사 중 ACSLS는 스토리지 셀에 누락되거나 읽을 수 없는 레이블이 있는 카트리지가 있으면 해당 카트리지를 꺼냅니다.

넣기 요청 종료

이 절차를 사용하여 현재 또는 보류 중인 수동 넣기나 가상 넣기를 종료하거나 취소할 수 있습니다.

cancel 명령을 사용하여 진행 중인 자동 넣기 작업을 취소할 수 없습니다. 진행 중인 자동 넣기를 종료하려면 다음을 수행합니다.

- CAP 도어가 열린 경우 모든 카트리지를 제거하고 도어를 닫습니다.
- CAP 도어가 닫혀 있고 카트리지를 라이브러리로 이동 중인 경우 남아 있는 카트리지를 라이브러리에 넣도록 허용해야 합니다. 그러면 넣기가 종료됩니다.

수동 넣기를 취소하려면 다음을 수행합니다.

1. 현재 및 보류 중인 모든 라이브러리 작업을 표시합니다.

   query request all
2. 취소할 enter/venter 요청의 *request_id*를 기록해 둡니다.
3. *cmd_proc*에서 다음을 입력합니다.

cancel request_id

여기서 *request_id*는 취소할 요청의 식별자입니다.

4. CAP가 잠금 해제될 때까지 기다린 후, CAP를 열고, 카트리지를 모두 제거합니다.

   *cmd_proc*는 취소 요청이 수신되기 전 라이브러리에 넣은 카트리지 수를 나타내는 메시지를 표시합니다. 이러한 카트리지는 ACSLS 제어에서 유지됩니다.

"enter "를 참조하십시오.

카트리지 꺼내기

라이브러리에서 카트리지를 꺼내려면 *eject* 명령을 실행해야 합니다.

다음 단계에서는 꺼내기 프로세스에 대해 설명합니다.

1. 꺼내기를 시작하면 CAP가 잠깁니다. 다른 호스트가 해당 CAP를 사용할 수 없습니다.
2. 로봇이 지정된 카트리지를 지정된 CAP에 놓으면, ACSLS는 카트리지가 저장된 셀 위치를 다른 카트리지에 사용할 수 있도록 합니다.
3. CAP를 열고, CAP에서 카트리지를 모두 제거하고, CAP를 닫습니다. 그러면 ACSLS가 CAP를 검사하여 비어 있는지 확인합니다. 이제 CAP를 넣기 또는 감사와 같은 다른 작업에 사용할 수 있습니다.

   eject 명령에 카트리지가 가득 찬 CAP를 둘 이상 지정하고 채워진 CAP를 비우고, CAP를 닫으면 ACSLS는 모든 카트리지를 꺼낼 때까지 꺼내기 프로세스를 계속합니다.

eject 명령에 대한 자세한 내용은 "eject "를 참조하십시오. "ejecting.sh "도 참조하십시오.

볼륨 통계 수집이 사용으로 설정된 경우 *acsss_stats.log*가 카트리지 꺼내기를 모두 기록합니다. "일반 제품 동작 변수 설정"을 참조하십시오.

CAP 복구

이 절에서는 CAP 복구에 대해 설명합니다.

일반 CAP 복구 절차

다음은 일반 CAP 복구 절차입니다.

CAP 복구 전에 넣기 및 꺼내기 완료

가능한 경우 넣기 또는 꺼내기를 취소하고 CAP를 복구하는 대신 넣기 또는 꺼내기를 완료합니다. 이렇게 하면 덜 복잡해지고 CAP가 중단될 위험이 적어집니다.

- 카트리지가 가득 찬 CAP 넣기를 완료한 다음 해당 작업을 취소하여 수동 넣기를 종료합니다. 자동 모드의 CAP만 카트리지가 가득 찬 CAP를 한 번에 하나씩 넣습니다.
- 가능한 경우 eject 명령에 지정된 모든 카트리지를 꺼냅니다. 그렇지 않으면 꺼내기 취소를 시도하기 전에 ACSLS가 카트리지가 가득 찬 CAP를 꺼내고 CAP를 비웁니다.

중단된 CAP를 강제로 오프라인으로 전환한 다음 온라인으로 전환하여 복구

CAP를 복구하려면 강제로 오프라인으로 전환해야 합니다. CAP를 강제로 오프라인으로 전환한 다음 다시 온라인으로 변경하면 CAP가 복구될 뿐만 아니라 일반적으로 CAP를 사용 중인 중단된 *enter* 또는 *eject*도 종료됩니다.

1. CAP를 강제로 오프라인으로 전환합니다.

   vary cap cap_id offline force

   현재 로봇 요청만 완료되고 CAP가 즉시 오프라인 상태로 전환됩니다. 보류 중인 요청이 삭제되고 새 요청이 거부됩니다.

   중단된 수동 *enter* 또는 *eject*는 일반적으로 취소됩니다.
2. *enter* 또는 *eject* 요청이 계속 활성 상태인 경우 해당 요청을 취소합니다.

   enter 또는 *eject* 요청이 계속 활성 상태인지 확인하려면 다음을 입력합니다.

   query request all

   enter 또는 *eject*가 계속 활성 상태인 경우 다음 명령을 입력하여 해당 요청을 취소합니다.

   cancel request_id
3. CAP를 다시 온라인 상태로 전환합니다.

   vary cap cap_id online

   이 작업으로 CAP가 복구되고 다른 요청에 사용할 수 있게 됩니다.

액세스 도어를 연 후 CAP 복구

SL8500 또는 SL3000 액세스 도어를 열고 닫았거나 SL8500 또는 SL3000을 다시 초기화한 후 ACSLS는 이제 자동 넣기 모드인 CAP를 잠금 해제합니다.

SL8500 또는 SL3000 라이브러리를 다시 초기화한 후 CAP가 잠겨 있고 CAP를 복구해야 하는 경우 아래의 적절한 절차에 따라 CAP를 복구하십시오.

자동 넣기에 사용된 CAP가 잠금 해제되지 않는 경우

자동 넣기를 위해 잠금 해제되지 않는 CAP를 복구하려면 ACSLS와 라이브러리 간에 CAP 상태를 동기화해야 합니다.

1. CAP 모드를 수동으로 설정하여 자동 넣기 모드를 종료합니다.

   set cap mode manual cap_id
2. CAP를 다시 자동 모드로 설정합니다.

set cap mode automatic *cap_id*

수동 넣기에 사용된 CAP가 잠금 해제되지 않는 경우

수동 넣기를 위해 잠금 해제되지 않는 CAP를 복구하려면 ACSLS와 라이브러리 간에 CAP 상태를 동기화해야 합니다.

1. CAP를 강제로 오프라인으로 전환합니다.

   vary cap cap_id *offline force*
2. CAP를 다시 온라인 상태로 전환합니다.

   vary cap cap_id *online*
3. 수동 넣기를 다시 시작합니다.

   enter cap_id

꺼내기에 사용된 CAP가 잠금 해제되지 않는 경우

꺼내기를 수행 중이었던 CAP를 복구하려면 잠긴 CAP에 남아 있는 카트리지를 모두 제거하고 ACSLS와 라이브러리 간에 CAP 상태를 동기화해야 합니다.

1. CAP에서 카트리지를 모두 제거합니다.
   a. CAP를 강제로 오프라인으로 전환(*Vary*)합니다.

      vary cap cap_id *offline force*
   b. CAP를 다시 온라인으로 전환(*Vary*)합니다.

      vary cap cap_id *online*
2. 다음 작업 중 하나를 선택합니다.

   CAP가 자동 모드인 경우
   a. CAP 모드를 수동으로 설정하여 자동 넣기 모드를 종료합니다.

      set cap mode manual cap_id
   b. CAP를 자동 모드로 설정합니다. 이렇게 하면 CAP가 잠금 해제됩니다.

      set cap mode automatic cap_id
   c. CAP를 열고 CAP에 남아 있는 카트리지를 모두 제거합니다.

   CAP가 자동 모드가 아닌 경우
   a. 수동 *enter*를 시작합니다.

      enter cap_id
   b. CAP에 남아 있는 카트리지를 모두 제거합니다.

c. 넣기를 취소합니다.

넣기를 기다리고 있는 *cmd_proc*에서 *Ctrl + c*를 사용하거나 *enter* 요청 ID를 취소합니다.

3. 꺼내기를 다시 시작합니다.

enter cap_id vol_id | *volrange…*

L1400, L700, L700e 또는 L180 라이브러리에서 CAP를 잠금 해제하는 복구 절차

L1400, L700, L700e 또는 L180 라이브러리에서 넣기 또는 꺼내기 시 사용되는 CAP가 잠금 해제되지 않는 경우 라이브러리에서 IPL을 수행하여 CAP를 복구할 수 있습니다. 아래의 적절한 절차에 따라 CAP를 복구하십시오.

수동 넣기에 사용된 CAP가 잠금 해제되지 않는 경우

수동 넣기를 위해 잠금 해제되지 않는 CAP를 복구하려면 다음을 수행합니다.

1. *enter*를 취소*(Cancel)*합니다.

넣기가 완료될 때까지 기다리고 있는 cmd_proc에서 *Ctrl + c*를 사용하거나 *enter* 요청 ID를 *cancel*합니다.

2. 운영자 패널에서 RESET 버튼을 눌러 라이브러리를 다시 IPL합니다.
3. 라이브러리가 초기화를 마친 후 다른 넣기를 시작합니다.

자동 넣기에 사용된 CAP가 잠금 해제되지 않는 경우

자동 넣기를 위해 잠금 해제되지 않는 CAP를 복구하려면 다음을 수행합니다.

1. CAP 모드를 다시 수동으로 설정하여 자동 넣기 모드를 종료합니다.

set cap mode manual cap_id

2. 운영자 패널에서 RESET 버튼을 눌러 라이브러리를 다시 IPL합니다.
3. 라이브러리가 초기화를 마친 후 CAP를 다시 자동 모드로 설정합니다.

set cap mode automatic cap_id

꺼내기에 사용된 CAP가 잠금 해제되지 않아 비울 수 없는 경우

꺼내기를 위해 잠금 해제되지 않는 CAP를 복구하려면 (CAP가 가득 찼거나 모든 볼륨을 꺼낸 후) 다음을 수행합니다.

1. 라이브러리의 액세스 도어를 열고, CAP에서 카트리지를 모두 제거하고, 액세스 도어를 닫습니다.
2. 운영자 패널에서 RESET 버튼을 눌러 라이브러리를 다시 IPL합니다.

라이브러리를 다시 IPL하면 ACSLS에서 "라이브러리 오류"와 함께 꺼내기를 종료합니다.

3. 또는 라이브러리를 감사합니다.

   라이브러리가 초기화를 마친 후 감사를 실행하는 것은 권장되는 작업이지만 필수는 아닙니다.
4. 일부 카트리지만 꺼낸 경우 다른 꺼내기를 시작합니다.

새 카트리지 및 재활성화된 카트리지에 자동으로 정책 적용

이 절에서는 새 카트리지 및 재활성화된 카트리지에 자동으로 정책을 적용하는 방법에 대해 설명합니다.

청소 카트리지 속성 자동 지정

최신 청소 카트리지에는 청소 카트리지용으로만 예약되는 매체 유형 레이블이 지정됩니다. 예를 들어 T10000 이전 버전과 호환되는 청소 카트리지에는 "CL"의 매체 도메인 및 유형 레이블이 지정되고, LTO 범용 청소 카트리지에는 "CU" 레이블이 지정됩니다.

ACSLS는 이러한 매체 도메인 및 유형의 카트리지만 청소 카트리지가 된다고 인식하고 있으므로 이러한 카트리지가 감사, 넣기 또는 카트리지 복구를 통해 추가될 때 자동으로 청소 카트리지 속성을 설정합니다. 여기에는 해당 카트리지를 청소 카트리지로 식별하고 해당 최대 청소 사용량을 설정하는 작업이 포함됩니다.

watch_vols 정책

watch_vols 유틸리티는 카트리지를 넣거나 다시 넣을 때 데이터베이스에 추가되거나 감사를 통해 재활성화된 카트리지에 자동으로 속성을 지정할 수 있습니다. 정책은 vol_attr.dat 파일에 지정되고 *vol_id* 또는 *vol_range*에 의해 선택됩니다. 이 유틸리티는 다음을 자동으로 수행할 수 있습니다.

- *vol_attr.dat* 정책 테이블에 나열된 특정 볼륨 또는 *vol_id* 범위를 기반으로 하는 볼륨 소유권을 지정합니다.
- 스크래치 풀에 카트리지를 지정합니다.
- 새 카트리지 및 재활성화된 카트리지를 특정 LSM으로 이동합니다.
- 논리적 라이브러리에 카트리지를 지정합니다.

자세한 내용은 “watch_vols ”를 참조하십시오.

청소 카트리지

읽기/쓰기 기록 헤드에서 얼룩진 오염물 및 달라붙어 있는 파편을 제거하려면 정기적으로 테이프 드라이브를 청소해야 합니다. 드라이브 제어 장치는 드라이브를 청소해야 할 때 테이프가 각 드라이브를 통과하고 ACSLS에 메시지를 보내는 정도를 추적합니다.

청소 카트리지에 대한 자세한 내용은 다음 항목을 참조하십시오.

ACSLS에서의 자동 청소

ACSLS는 광 섬유 또는 SCSI 연결 라이브러리(SL150, SL500 및 L700)가 아닌 TCP/IP 또는 직렬(HLI) 연결 라이브러리(SL8500, SL3000 및 9310)에 대한 자동 청소를 수행할 수 있습니다.

자동 청소가 사용으로 설정된 경우 ACSLS는 필요할 때 자동으로 테이프 드라이브에 청소 카트리지를 마운트합니다. *AUTO_CLEAN* 동적 변수를 *TRUE*(기본값)로 설정하면 자동 청소가 사용으로 설정됩니다.

최신 테이프 드라이브는 필요에 따라 청소를 요청합니다. 드라이브가 라이브러리에 알리며 이로 인해 ACSLS에 메시지가 전달됩니다. ACSLS는 드라이브를 청소해야 함을 기록합니다. ACSLS가 드라이브에 대한 다음 마운트 요청을 처리하는 경우 청소 작업을 수행하는 마운트보다 우선합니다. 여기에는 호환 가능한 청소 카트리지 선택, 청소 카트리지 마운트, 청소 카트리지 마운트 해제, 원래 마운트 요청에 지정된 데이터 카트리지 마운트 진행이 포함됩니다.

ACSLS에서 청소 작업 중에 모두 소모된 청소 카트리지 마운트와 같은 복구 가능한 문제가 발생하는 경우 다른 청소 카트리지를 선택하고 청소 작업을 재시도합니다. *AUTO_CLEAN_RETRY_LIMIT* 동적 변수는 재시도 수(기본값: 1회 재시도, 재시도 범위: 0-5회)를 제어합니다. *acsss_config*를 사용하고 General Product Behavior Variables를 선택하여 이 변수를 표시하고 변경합니다.

UNIFORM_CLEAN_USE 동적 변수는 청소 카트리지를 선택하는 데 사용되는 방법을 정의합니다. 옵션은 다음과 같습니다.

- *VOLID_SORT* – `vol_id`별로 정렬합니다. 다음 클리너를 사용하기 전에 클리너 하나를 모두 사용합니다.
- *LEAST_USED* – 사용량별로 정렬합니다. 균등하게 사용량을 분산합니다.
- *MOST_CAPACITY* – 남은 사용량별로 정렬합니다. 모든 클리너를 동시에 완전히 사용합니다.

기본값은 *VOLID_SORT*입니다. *acsss_config*를 사용하고 General Product Behavior Variables를 선택하여 이 변수를 표시하고 변경합니다.

ACSLS에서의 자동 청소에 대한 자세한 내용은 다음 항목을 참조하십시오.

- “청소 카트리지 최대 사용량 ”
- “청소 카트리지 넣기 ”

- "CSI 조정 변수 설정"의 *AUTO_CLEAN*
- "CSI 조정 변수 설정"의 *AUTO_CLEAN_RETRY_LIMIT*

청소 카트리지 최대 사용량

서로 다른 각각의 청소 카트리지 유형에는 드라이브가 모두 사용되었음(만료됨 또는 소모됨)을 보고하기 전 최대 사용 수가 있습니다. 이 최대 사용량은 청소 카트리지의 유형에 따라 다릅니다. ACSLS가 청소 카트리지를 추가할 때 ACSLS 데이터베이스에 카트리지의 최대 사용량이 기록됩니다. ACSLS는 카트리지의 *access_count*(카트리지가 마운트된 횟수 등)가 *max* 사용량보다 적을 때만 자동 청소를 위한 청소 카트리지를 선택합니다. 테이프 드라이브에서 청소 카트리지가 모두 사용되었음(소모됨)을 보고하면 ACSLS는 액세스 수를 *max* 사용량보다 큰 값으로 설정합니다.

ACSLS가 청소 카트리지에 대해 자동으로 설정하는 최대 사용량은 카트리지가 지원하는 실제 청소 사용량보다 큽니다. 이는 일부 응용 프로그램이 청소를 요청한 드라이브 없이 청소 카트리지의 마운트를 예약하기 때문입니다. 드라이브가 청소될 준비가 되지 않은 경우 헤드가 너무 빨리 마모되지 않게 하기 위해 "의사 청소" 작업을 수행할 수도 있습니다. 즉, 실제로 청소 카트리지를 사용하지 않고도 드라이브의 액세스 수가 증분되었다는 의미입니다. 더 높은 *max* 사용 값을 사용하면 이러한 카트리지가 모두 소모되었음을 드라이브에서 보고할 때까지 사용됩니다.

수동으로 청소 카트리지 정의

set clean 명령을 사용하여 청소 카트리지를 정의하고 해당 최대 사용량을 설정할 수 있습니다.

set clean max_usage vol_id | volrange

설명:

- *max_usage*는 ACSLS가 카트리지 청소를 위해 카트리지를 선택하는 작업을 중지하기 전에 청소 카트리지가 사용되는 횟수입니다.
- *vol_id* | *volrange*는 청소 카트리지 또는 카트리지 범위를 지정합니다.

*set clean*을 사용하여 다음을 수행합니다.

- 청소 카트리지의 최대 사용 수를 변경합니다.

  예를 들어 청소할 필요가 없는 드라이브에 청소 카트리지가 수동으로 마운트되었으며 *access_count*가 증분되었지만 "의사 청소"만 수행되었습니다. 청소 카트리지의 전체 사용량을 얻으려면 더 높은 *max_usage*를 설정합니다.

  set clean max_usage vol_id|volrange
- 카트리지의 청소 카트리지 속성을 해제로 설정합니다. 예를 들어 데이터 카트리지를 청소 카트리지로 잘못 정의한 경우 카트리지의 청소 카트리지 속성을 해제로 설정하여 카트리지를 데이터 카트리지로 재정의합니다.

  set clean off vol_id|volrange

청소 카트리지 모니터링

사용한 청소 카트리지를 꺼내 라이브러리에서 청소 카트리지를 모니터해야 합니다. 필요에 따라 새 청소 카트리지를 넣습니다.

- 청소 카트리지를 모두 표시하려면 다음을 수행합니다.

  query clean all
- ACS에서 한 *media_type*의 청소 카트리지를 모두 표시하려면 다음과 같이 display 명령을 사용합니다.

  *display volume * -home acs,*,*,*,* -media media_type*
- 카트리지의 최대 청소 사용량 및 현재 사용량을 표시하려면 다음을 수행합니다.

  *display volume * -home acs,*,*,*,* -media media_type –f vol_id acs lsm media max_use access_count*
- 최대 청소 사용량 및 현재 사용량과 함께 ACS의 청소 카트리지를 모두 표시하려면 다음을 수행합니다.

  display volume CLN -home acs,*,*,*,* -f acs lsm type media max_use access_count*
- 사용된 청소 카트리지(이러한 카트리지는 꺼내서 새 청소 카트리지로 교체해야 함)를 모두 표시하려면 다음을 수행합니다.

  *display volume * -spent_clean*

참조:

- “청소 카트리지 넣기 ”
- “사용된 청소 카트리지 꺼내기 ”

청소 카트리지 넣기

청소 카트리지를 넣을 때 다음 절차를 완료했는지 확인하십시오.

- 매체 유형이 라이브러리의 드라이브 유형과 호환되는 청소 카트리지를 사용합니다. ACSLS가 각 청소 작업에 대해 올바른 유형의 카트리지를 자동으로 선택합니다.

  드라이브 유형과 호환되는 청소 카트리지를 보려면 *ACSLS Product Information* 설명서에서 매체와 드라이브 간 호환성 표를 확인하거나 *drive_media.sh* 유틸리티를 사용합니다.
- 라이브러리의 각 드라이브 유형에 대해 최소한 몇 개의 청소 카트리지만 정의합니다. 대부분의 사이트에서는 4개의 드라이브마다 하나 이상의 청소 카트리지가 있는 것이 좋습니다.

ACSLS에 대한 청소 카트리지를 정의하려면 다음을 수행합니다.

1. CAP를 넣기 준비 상태로 만듭니다.

자세한 내용은 "카트리지 넣기 "를 참조하십시오.

2. 청소 카트리지를 넣습니다.

   *cmd_proc*는 넣은 카트리지의 카트리지 ID를 포함하는 메시지를 표시합니다.

   "청소 카트리지 속성 자동 지정 "에 설명된 대로 ACSLS는 감사, 넣기 또는 카트리지 복구를 통해 청소 카트리지를 넣거나 추가할 때 자동으로 해당 청소 카트리지를 정의합니다. 여기에는 최대 사용량이 포함됩니다.

사용된 청소 카트리지 꺼내기

청소 카트리지가 해당 최대 사용량에 도달했거나 드라이브가 청소 카트리지가 소모되었음을 보고하면 ACSLS는 이벤트 로그에 메시지를 기록합니다. ACSLS는 라이브러리에 카트리지를 남겨 두지만 더 이상 청소에 이 카트리지를 선택하지 않습니다. 사용한 청소 카트리지를 꺼내고 교체품을 넣어야 합니다.

사용한 청소 카트리지를 꺼내려면 다음을 수행합니다.

1. *query clean* 및 *display volume*을 사용하여 최대 사용량을 초과하거나 소모된 청소 카트리지를 식별합니다.

   ```
   query clean all
   ```

   ```
   display volume * -spent_clean
   ```

2. 청소 카트리지를 꺼냅니다.

   eject cap_id vol_id | volrange

   설명:

   *cap_id*는 청소 카트리지를 꺼내는 데 사용되는 CAP를 지정합니다.

   vol_id | *volrange*는 꺼낼 청소 카트리지의 ID를 지정합니다.

3. 소모된 청소 카트리지를 제거합니다.

"청소 카트리지 모니터링 "을 참조하십시오.

수동으로 드라이브 청소

자동 청소가 사용 안함으로 설정되었거나 작동 중이 아닌 경우 이 절차를 사용하여 드라이브를 청소합니다.

수동으로 드라이브를 청소하려면 다음을 수행합니다.

1. 청소할 드라이브와 호환되는 청소 카트리지 유형을 결정합니다.

   각 드라이브 유형에 대한 청소 카트리지 목록은 *Product Information Guide*의 드라이브 및 매체 호환성 표를 참조하십시오.

2. 사용 가능한 청소 카트리지를 표시합니다.

   query clean all

   호환 가능한 모든 청소 카트리지를 드라이브와 동일한 ACS에 표시하려면 *display* 명령을 사용합니다.

   *display volume * -home acs,*,*,*,* -media* media_type

   카트리지의 최대 청소 사용량 및 현재 사용량을 표시하려면 다음을 수행합니다.

   *display volume * -home acs,*,*,*,* -media media_type -f* vol_id acs lsm media max_use *access_count*

   최대 청소 사용량 및 현재 사용량과 함께 ACS의 청소 카트리지를 모두 표시하려면 다음을 수행합니다.

   display volume CLN -home acs,*,*,*,* -f acs lsm type media max_use access_count*
3. 이렇게 나열된 청소 카트리지에서 호환 가능한 청소 카트리지를 선택하고 드라이브에 마운트합니다.

   mount vol_id drive_id
4. 드라이브를 청소하고 청소 카트리지를 언로드한 후 청소 카트리지를 마운트 해제합니다.

   dismount vol_id drive_id

광 섬유 연결 라이브러리의 청소 카트리지

광 섬유 연결 라이브러리의 드라이브에 대해 ACSLS 자동 청소가 지원되지 않습니다. 청소 카트리지를 수동으로 마운트한 ACSLS를 사용하여 이러한 드라이브만 청소할 수 있습니다. 그러나 광 섬유 연결 라이브러리의 라이브러리 GUI를 사용하여 자동 청소를 사용으로 설정할 수 있습니다. 자세한 내용은 해당 라이브러리 설명서를 참조하십시오.

드라이브가 청소되지 않을 때 수행할 작업

다음에는 드라이브가 청소되지 않을 때 시도할 수 있는 문제 해결 팁 몇 가지가 나와 있습니다.

자동 청소가 사용으로 설정되어 있는지 확인

자동 청소가 사용 안함으로 설정된 경우 드라이브를 청소해야 할 때 ACSLS는 메시지를 이벤트 로그에 기록하고 *cmd_proc*로 청소 메시지를 표시합니다. 수동으로 청소 카트리지를 마운트해야 합니다.

*acsss_config*를 사용하여 자동 청소를 사용으로 설정하거나 사용 안함으로 설정합니다. 또한 *acsss_config*를 사용하여 선택 및 질의를 위해 청소 카트리지를 정렬하는 방법을 지정할 수 있습니다.

AUTO_CLEAN 동적 변수를 기본 설정인 **TRUE**(설정)로 설정하면 자동 청소가 사용으로 설정됩니다. *AUTO-CLEAN*을 보려면 다음을 입력합니다.

```
dv_config -e AUTO_CLEAN
```

ACSLS는 광 섬유 연결 라이브러리에 대한 자동 청소를 수행하지 않습니다.

드라이브에 대한 청소 카트리지가 있는지 확인

모든 청소 카트리지가 만료되거나(*max_usage* 값이 초과됨) 드라이브가 소모된 것으로 보고한 경우 ACSLS는 드라이브를 청소하지 않고 원래 마운트 요청을 수행합니다. 해당 마운트 및 청소하지 않은 드라이브에 대한 각 후속 마운트의 경우 ACSLS는 메시지 376 N "*Drive drive_id: No Cleaning cartridge available*"을 이벤트 로그에 게시합니다. “수동으로 청소 카트리지 정의 ”에 설명된 대로 드라이브 유형과 호환되는 청소 카트리지를 더 추가합니다.

드라이브가 청소되지 않는 경우 라이브러리에 드라이브에 대한 청소 카트리지가 있으며 해당 카트리지에 사용량이 남아 있는지 확인하십시오.

*cmd_proc*에서 display 명령을 사용하여 다음을 확인할 수 있습니다.

- 모든 청소 카트리지 및 해당 사용량

  ```
  display volume * -clean -f media access_count max_use
  ```
- 특정 매체 유형의 모든 볼륨

  예를 들어 LTO 청소 카트리지를 모두 표시하려면 다음을 수행합니다.

  ```
  display volume * -media LTO-CLNU -f access_count max_use
  ```
- 모두 사용된(소모된) 모든 청소 카트리지 및 해당 사용량

  ```
  display volume * -spent_clean -f media access_count max_use
  ```

SL8500 또는 SL3000 라이브러리에 대한 SLConsole을 사용하여 자동 청소가 사용 안함으로 설정되어 있는지 확인

SL8500 또는 SL3000에 대한 자동 청소가 작동하지 **않는** 문제가 발생한 경우 SLConsole을 사용하여 라이브러리에 대해 자동 청소가 사용으로 설정되지 않았는지 확인하십시오.

ACSLS를 사용하여 자동 청소가 사용으로 설정된 경우, 마운트 해제 후 라이브러리로부터 "drive needs cleaning" 메시지를 수신하면 다음 마운트 이전에 자동으로 청소 카트리지를 마운트합니다.

SLConsole을 사용하여 라이브러리 레벨에서 자동 청소가 사용으로 설정된 경우 라이브러리가 자동 청소를 수행합니다. 라이브러리 자동 청소가 사용으로 설정된 경우 라이브러리는 "drive needs cleaning" 메시지를 ACSLS에 보내지 않습니다. ACSLS는 드라이브를 청소해야 함을 알지 못합니다. 라이브러리는 ACSLS에 마운트 해제 응답을 보내기 전에 해당 시스템 셀 중 하나의 청소 카트리지를 마운트하여 드라이브를 청소하려고 시도합니다.

따라서 라이브러리가 자동 청소를 수행하려고 시도하지만 시스템 셀에 청소 카트리지가 없는 혼동이 발생할 수 있습니다. ACSLS는 일반 스토리지 셀에서 청소 카트리지를 관리할 수 있지만 ACSLS가 "drive needs cleaning" 메시지를 수신하지 않습니다. 따라서 드라이브가 청소되지 않습니다.

이 문제를 해결하려면 다음을 수행합니다.

- ACSLS 자동 청소가 사용으로 설정되었지만 드라이브가 청소되지 않은 경우 라이브러리에도 자동 청소가 사용으로 설정되어 있는지 확인합니다.
- 라이브러리에서 자동 청소가 사용으로 설정된 경우 SLConsole을 사용하여 사용 안함으로 설정합니다.

  SLConsole 또는 라이브러리 운영자 패널을 사용합니다.
  a. System Detail 탭을 선택합니다.
  b. Library를 선택합니다.
  c. Auto Clean 탭을 선택합니다.
  d. Configure 탭을 선택합니다.
  e. 이 분할 영역(아니면 "Partition 1 또는 None")에 대해 자동 청소가 사용으로 설정되어 있는지 확인합니다.
  f. 자동 청소가 사용으로 설정된 경우 사용 안함으로 설정합니다.

청소 카트리지가 불완전으로 표시되었는지 확인

오류가 있는 청소 카트리지를 반복 선택하지 않도록 자동 청소는 불완전한 카트리지를 선택하지 않습니다. 카트리지에 읽을 수 없는 레이블이 있음을 라이브러리가 보고하는 경우 카트리지가 불완전으로 표시됩니다.

display 명령을 사용하여 불완전으로 표시된 청소 카트리지를 식별할 수 있습니다. 이 명령은 청소 카트리지의 ACS, LSM, 유형, *max_use* 및 *access_count*를 표시하기도 합니다.

```
display volume CLN* -f media_status acs lsm media_status type max_use
access_count
```

불완전 상태를 지우려면 다음을 수행합니다.

- 카트리지를 꺼내고 검사하여 상태가 좋은 경우 해당 카트리지를 라이브러리에 다시 넣습니다.
- 카트리지를 넣으면 불완전 상태가 지워집니다.

스크래치 카트리지 관리

스크래치 카트리지에는 데이터가 포함되어 있지 않거나 겹쳐쓸 수 있는 데이터가 포함되어 있습니다. 사용자 또는 응용 프로그램은 스크래치 카트리지를 마운트하여 해당 카트리지에 새 데이터를 씁니다.

스크래치 상태를 지정하려면 다음을 수행합니다.

- 카트리지는 *set scratch* 명령으로 스크래치 카트리지로 정의되고 스크래치 풀에 지정될 수 있습니다.
- *watch_vols* 유틸리티는 카트리지의 *vol_id* 또는 *volrange*에 따라 자동으로 스크래치 풀에 카트리지를 지정할 수 있습니다. “watch_vols ”를 참조하십시오.

볼륨 스크래치 상태를 지웁니다.

- 마운트 스크래치 또는 일반 마운트 요청을 통해 카트리지가 성공적으로 마운트되는 경우 카트리지의 스크래치 상태가 지워집니다.

> **주:**
>
> *set scratch* 명령은 스크래치 상태를 지우는 데 사용할 수 있습니다. 볼륨을 마운트할 때 볼륨의 스크래치 상태가 지워져도 *pool id*는 지워지지 않습니다. 따라서 풀에 데이터 볼륨이 지정됩니다.

set scratch 명령은 다음을 사용하여 스크래치 풀에 데이터 볼륨을 지정하는 데 사용될 수도 있습니다.

set scratch off pool_id vol_id | volrange

스크래치 마운트 요청을 충족하려면 라이브러리에 사용 가능한 스크래치 카트리지가 충분히 있는지 확인해야 합니다. 자세한 내용은 다음 항목을 참조하십시오.

- “라이브러리에 스크래치 카트리지 추가 ”
- “스크래치 풀 균형 조정 ”

다음 절에서는 스크래치 카트리지 및 스크래치 풀 관리에 대한 추가 정보를 제공합니다.

- “스크래치 풀 및 스크래치 카트리지 정보 표시 ”
- “스크래치 풀 삭제 ”
- “스크래치 카트리지 마운트 ”
- “카트리지 스크래치 해제 ”

스크래치 풀 및 스크래치 카트리지 정보 표시

스크래치 풀 정보를 표시하려면 다음 ACSLS 기능을 사용합니다.

- *query pool*

  스크래치 풀 속성을 표시합니다. “query pool ”을 참조하십시오.
- *query scratch*

  스크래치 카트리지 정보를 표시합니다. “query scratch ”를 참조하십시오.
- *query mount **

  지정된 스크래치 풀 및 선택적으로 풀 내 특정 카트리지 매체 유형에 대한 매체 호환 카트리지의 상태를 표시합니다. “query mount * ”를 참조하십시오.

- 사용자 정의된 볼륨 보고서

  선택한 스크래치 볼륨 정보를 보고합니다. “로깅 볼륨 통계 보고서 작성”을 참조하십시오.

라이브러리에 스크래치 카트리지 추가

이 절차를 사용하여 라이브러리에 스크래치 카트리지를 추가합니다.

라이브러리에 스크래치 카트리지를 추가하려면 다음을 수행합니다.

1. 필요한 경우 새 스크래치 풀을 만듭니다.

   자세한 내용은 “query scratch ”를 참조하십시오.
2. 라이브러리에 스크래치 카트리지를 넣습니다.

   자세한 내용은 “카트리지 넣기 ”를 참조하십시오.
3. 2단계에서 넣은 카트리지를 스크래치 카트리지로 정의하고 해당 카트리지를 스크래치 풀에 지정합니다.

   이는 *watch_vols* 유틸리티의 *vol_attr.dat*에 정의된 정책을 사용하거나 *set scratch*를 사용하여 수행될 수 있습니다.

스크래치 풀 균형 조정

이 절차를 사용하여 하나의 풀에서 다른 풀로 스크래치 카트리지를 이동하여 스크래치 풀의 균형을 조정합니다.

스크래치 풀의 균형을 조정하려면 다음을 수행합니다.

1. 모든 스크래치 풀의 속성을 표시합니다.

   ```
   query pool all
   ```

   자세한 내용은 “query pool ”을 참조하십시오.
2. *query scratch* 명령을 사용하여 균형을 조정할 풀의 스크래치 카트리지 ID를 표시합니다.

   자세한 내용은 “query scratch ”를 참조하십시오.
3. *set scratch* 명령을 사용하여 하나의 풀에서 다른 풀로 스크래치 카트리지를 이동합니다.

   예를 들어 YUMA20-YUMA80(현재 풀 5-풀 10에 있음) 카트리지를 다시 지정하려면 다음을 입력합니다.

   ```
   set scratch 10 YUMA20-YUMA80
   ```

   자세한 내용은 “set scratch ”를 참조하십시오.

스크래치 풀 삭제

스크래치 풀을 관리하기 위해 더 이상 스크래치 카트리지를 포함하지 않는 모든 스크래치 풀을 삭제할 수도 있습니다. 공통 풀(풀 0)은 삭제할 수 없습니다. 빈 스크래치 풀만 삭제할 수 있습니다. 데이터 또는 스크래치 카트리지를 포함하는 스크래치 풀은 삭제할 수 없습니다. 그러나 “비어 있는 모든 풀 삭제 ”를 사용하여 빈 풀을 모두 삭제할 수 있습니다. ACSLS는 스크래치 또는 데이터 카트리지를 포함하는 풀을 삭제하지 않습니다.

스크래치 풀 비우기

이 절차를 사용하여 스크래치 풀을 삭제하기 전에 스크래치 풀을 비웁니다.

스크래치 풀을 비우려면 다음을 수행합니다.

1. 데이터 카트리지를 풀 외부로 이동하려면 다음을 입력합니다.

   ```
   set scratch off 0 vol_id volrange ...
   ```

   여기서 *vol_id* 또는 *volrange*는 공통 풀(풀 0)로 이동할 데이터 카트리지를 지정합니다. 자세한 내용은 “set scratch ”를 참조하십시오.
2. 스크래치 카트리지를 풀 외부로 이동하려면 다음 중 하나를 수행합니다.
   - 카트리지를 다른 풀로 이동합니다.
   - “카트리지 꺼내기 ”를 참조하십시오. 그러나 스크래치 카트리지를 꺼내는 경우 ACSLS는 더 이상 이러한 카트리지를 관리하지 않습니다. 나중에 이러한 카트리지를 사용하려는 경우 해당 카트리지를 다시 넣고 스크래치 풀에 지정해야 합니다.

단일 풀 삭제

단일 풀을 삭제하려면 다음을 수행합니다.

```
delete pool pool_id
```

비어 있는 모든 풀 삭제

delete pool all 명령은 스크래치 또는 데이터 카트리지를 포함하는 풀이 아닌 빈 스크래치 풀만 삭제합니다.

빈 풀을 모두 삭제하려면 다음을 수행합니다.

```
delete pool all
```

스크래치 카트리지 마운트

스크래치 마운트(*cmd_proc*를 사용하는 *mount **) 명령은 지정된 드라이브와 호환되고 가능한 가까운 스크래치 카트리지를 선택하여 해당 드라이브에 마운트합니다. 풀이 지정된 경우 해당 풀에 스크래치 카트리지를 지정해야 합니다.

스크래치 카트리지가 해당 홈 셀에서 검색되지 않거나 다른 복구 가능한 오류가 발생하여 스크래치 카트리지의 마운트가 실패하는 경우 ACSLS는 자동으로 다른 스크래치 카트리지를 선택하고 마운트를 재시도합니다.

스크래치 카트리지가 마운트될 때마다 해당 카트리지를 마운트한 ACSAPI 사용자가 자동으로 소유하도록 하는 볼륨 액세스 제어 정책을 설정할 수 있습니다. “볼륨 소유권 설정 ”을 참조하십시오.

다음 절차를 사용하여 단일 매체 및 혼합 매체 환경에서 스크래치 카트리지를 마운트합니다.

단일 매체 환경

- 지정된 풀에서 카트리지를 마운트하려면 다음을 수행합니다.

  mount * drive_id pool_id

  지정된 풀에서 사용 가능한 카트리지가 없으며 풀이 "overflow"에 대해 설정된 경우 ACSLS는 공통 풀(풀 0)에서 카트리지를 선택합니다.
- 공통 풀에서 카트리지를 마운트하려면 다음을 수행합니다.

  mount * drive_id

혼합 매체 환경

- 지정된 풀에서 지정된 매체 유형의 스크래치 카트리지를 마운트하려면 다음을 수행합니다.

  *mount * drive_id pool_id media media_type*

  지정된 풀에서 사용 가능한 카트리지가 없으며 풀이 *overflow*에 대해 설정된 경우 ACSLS는 공통 풀(풀 0)에서 지정된 매체 유형의 카트리지를 선택합니다.
- 지정된 풀에서 스크래치 환경 설정에 따라 결정된 매체 유형의 스크래치 카트리지를 마운트하려면 다음을 수행합니다.

  *mount * drive_id pool_id media **

  지정된 풀에서 사용 가능한 카트리지가 없으며 풀이 *overflow*에 대해 설정된 경우 ACSLS는 정의된 스크래치 환경 설정에 따라 공통 풀(풀 0)에서 카트리지를 선택합니다.
- 공통 풀에서 지정된 매체 유형의 카트리지를 마운트하려면 다음을 수행합니다.

  *mount * drive_id media media_type*
- 공통 풀에서 스크래치 환경 설정에 따라 결정된 매체 유형의 카트리지를 마운트하려면 다음을 수행합니다.

  *mount * drive_id media **

카트리지 스크래치 해제

스크래치 카트리지는 마운트될 때 자동으로 데이터 카트리지 상태에 다시 지정됩니다.

이 절차를 사용하여 잘못 스크래치된 카트리지를 "스크래치 해제"(카트리지를 데이터 카트리지 상태로 되돌림)합니다.

카트리지를 스크래치 해제하려면 다음을 수행합니다.

1. *query pool* 및 *query scratch* 명령을 사용하여 스크래치 해제할 카트리지 및 해당 카트리지의 풀 ID를 표시합니다.

   자세한 내용은 “query pool ” 및 “query scratch ”를 참조하십시오.
2. 선택한 카트리지를 스크래치 해제하려면 다음을 입력합니다.

   set scratch off 0 vol_id volrange ...

   여기서 *vol_id* 또는 volrange는 스크래치 모드에서 변경하고 공통 풀(풀 0)로 이동할 카트리지를 지정합니다. 자세한 내용은 “set scratch ”를 참조하십시오.

부재 카트리지 및 꺼낸 카트리지 지원 사용

ACSLS의 부재 카트리지 지원을 사용하면 라이브러리에서 찾을 수 없는 카트리지를 삭제하는 대신 부재 카트리지로 표시할 수 있습니다. 이러한 카트리지가 나중에 라이브러리에서 검색되는 경우 ACSLS는 이러한 카트리지를 데이터베이스에 다시 추가하는 대신 활성 상태로 변경합니다. 재활성화는 액세스 개수와 풀, 볼륨 액세스 제어 소유권 및 잠금과 같은 액세스 설정을 보존합니다.

마찬가지로, 꺼낸 카트리지 지원은 카트리지를 꺼냈을 때 카트리지 정보를 보존합니다. 카트리지는 다시 넣을 때 재활성화됩니다.

*ABSENT_VOLUME_RETENTION_PERIOD*가 0이 아닌 일 수로 설정되어 있을 때 부재 볼륨 및 꺼낸 볼륨 지원이 사용으로 설정됩니다. 기본값은 5일입니다.

부재 카트리지 및 꺼낸 카트리지 지원의 추가적인 측면은 다음과 같습니다.

- *-d* 옵션이 지정되어 있는 경우가 아니면 수동 볼륨 삭제(*del_vol*) 유틸리티가 볼륨을 부재 볼륨으로 보존합니다. 이 옵션이 지정된 경우 볼륨이 부재 상태 또는 꺼낸 상태의 만료를 기다리지 않고 삭제됩니다.
- ACSLS는 SL3000 및 SL8500 라이브러리에서 손실된 카트리지의 위치를 질의합니다.
- ACSLS는 라이브러리의 예상 위치에서 검색되지 않은 볼륨을 검색함으로써 볼륨 복구 기능을 개선합니다. ACSLS는 자동으로 볼륨을 삭제하는 대신 기록된 위치를 모두 검색합니다.
- 클라이언트는 *ENABLE_STATUS_VOLUME_ABSENT* 및 *ENABLE_STATUS_VOLUME_MISSING* 구성 설정을 통해 부재 상태, 꺼낸 상태 또는 누락된 상태를 ACSAPI를 통해 보고할지 여부를 지정할 수 있습니다.
- *-i* 옵션과 함께 *volrpt* 유틸리티를 사용하면 부재 상태 또는 꺼낸 상태의 볼륨 레코드가 보고됩니다. 기본적으로 volrpt는 부재 볼륨 또는 꺼낸 볼륨을 보고하지 않습니다.

부재 카트리지, 꺼낸 카트리지 및 누락된 카트리지

ACSLS는 3가지 카트리지(볼륨) 상태를 보고합니다.

- *missing*

  카트리지를 라이브러리에서 찾을 수 없으며, LSM이 오프라인이거나 드라이브가 통신 중이 아니므로 카트리지에 대해 기록된 위치를 하나 이상 검색할 수 없습니다. 카트리지에 대한 정보가 보존되었습니다.
- *absent*

  카트리지를 라이브러리에서 찾을 수 없습니다. 카트리지에 대해 기록된 위치를 모두 검색했는데 해당 위치 중 어디에도 카트리지가 없습니다. 카트리지에 대한 정보가 보존됩니다. 보존 기간이 만료되기 전에 카트리지가 발견되거나 라이브러리에 카트리지를 다시 넣으면 해당 카트리지가 재활성화됩니다.
- *ejected*

  카트리지를 꺼냈습니다. 카트리지에 대한 정보가 보존되며, 보존 기간이 만료되기 전에 카트리지가 발견되거나 라이브러리에 카트리지를 다시 넣으면 해당 카트리지가 재활성화됩니다.

카트리지(볼륨) 상태 보고

ACSLS는 ACSAPI 요청에 대한 응답이 아닌 ACSLS 명령에 대한 응답으로 상태가 "missing", "absent" 또는 "ejected"로 서로 다른 카트리지(볼륨)를 보고합니다.

ACSLS 명령에 대한 응답으로 표시되는 정보는 카트리지를 "missing", "absent" 또는 "ejected"로 식별합니다.

그러나 ACSLS가 ACSAPI 요청에 대한 응답으로 표시하는 카트리지 상태 정보는 다음 ACSLS 동적 변수로 제어됩니다.

1. *missing*
   - ACSLS 동적 변수 *ENABLE_STATUS_VOLUME_MISSING*이 TRUE인 경우 ACSLS는 *STATUS_VOLUME_MISSING*을 보고합니다.
   - ACSLS 동적 변수 *ENABLE_STATUS_VOLUME_MISSING*이 FALSE인 경우 ACSLS는 *STATUS_VOLUME_IN_TRANSIT*를 보고합니다.
2. *absent*
   - ACSLS 동적 변수 *ENABLE_STATUS_VOLUME_ABSENT*가 TRUE인 경우 ACSLS는 *STATUS_VOLUME_ABSENT*를 보고합니다.
   - ACSLS 동적 변수 *ENABLE_STATUS_VOLUME_ABSENT*가 FALSE인 경우 ACSLS는 볼륨을 ACSLS 데이터베이스에서 삭제된 것으로 처리하고 *STATUS_VOLUME_NOT_IN_LIBRARY*를 보고합니다.
3. *ejected*
   - ACSLS 동적 변수 *ENABLE_STATUS_VOLUME_EJECTED*가 TRUE인 경우 ACSLS는 *STATUS_VOLUME_EJECTED*를 보고합니다.
   - ACSLS 동적 변수 *ENABLE_STATUS_VOLUME_EJECTED*가 FALSE인 경우 ACSLS는 볼륨을 ACSLS 데이터베이스에서 삭제된 것으로 처리하고 *STATUS_VOLUME_NOT_IN_LIBRARY*를 보고합니다.

ABSENT_VOLUME_RETENTION_PERIOD 동적 변수

ABSENT_VOLUME_RETENTION_PERIOD 동적 변수는 부재 볼륨 및 꺼낸 볼륨이 ACSLS 데이터베이스에 보존되는 기간을 제어하고 이러한 볼륨이 보존되는 일 수를 지정합니다. 다음과 같이 두 개의 특수 값이 있습니다.

- 값 0일은 볼륨이 삭제되어 부재 볼륨 또는 꺼낸 볼륨으로 표시되지 않음을 지정합니다. 이는 ACSLS 6.1 이전 ACSLS 릴리스의 동작입니다.
- 값 999일은 부재 볼륨 및 꺼낸 볼륨이 데이터베이스에 영구적으로 보존됨을 지정합니다.

카트리지 복구

카트리지 복구(*acscr*)는 스토리지 셀 또는 테이프 드라이브의 실제 컨텐츠가 ACSLS 데이터베이스에 저장된 정보와 일치하지 않을 때마다 불일치를 해결하기 위해 호출되는 ACSLS 내부 프로세스입니다. 이는 다음을 통해 수행됩니다.

- 라이브러리에서 볼륨의 홈 셀 및 드라이브(가능한 경우)를 검사하도록 합니다. 그런 다음 해당 결과로 ACSLS 데이터베이스를 업데이트합니다.
- ACSLS(SL3000 및 SL8500 라이브러리 사용)에서 라이브러리에 카트리지의 위치를 묻고 라이브러리의 응답을 기반으로 ACSLS 데이터베이스를 업데이트하여 카트리지를 복구할 수 있도록 합니다.

카트리지 복구에서 다른 위치에 기록된 카트리지와 같은 불일치를 찾는 경우 다른 복구 요청을 만들고 해당 요청을 요청 대기열에 추가합니다. 이를 "계단식"이라고 합니다.

다른 프로세스가 라이브러리의 실제 컨텐츠와 ACSLS 데이터베이스 간의 불일치를 발견할 때 복구 요청을 카트리지 복구에 전달합니다. 이 경우 카트리지 복구는 카트리지가 누락됨으로 표시되고, 부재로 변경되고, 재활성화되는 중앙 위치입니다. 따라서 다른 여러 ACSLS 명령 및 유틸리티의 동작처럼 보이는 것은 실제로 데이터베이스를 라이브러리에서 보고하는 정보와 일치하도록 업데이트할 때 카트리지 복구에서 수행됩니다.

다른 프로세스가 복구 요청을 카트리지 복구에 전달할 때 다음을 수행할 수 있습니다.

1. 계속하고 카트리지 복구에서 비동기식으로 계속하도록 합니다. 카트리지 복구가 독립적으로 진행됩니다.
2. 손실된 특정 카트리지가 필요한 경우 카트리지 복구에서 이 복구 요청 처리를 마치고 검색한 것을 보고할 때까지 기다립니다.

누락된 카트리지

다음 경우에 카트리지가 누락된 것으로 표시됩니다.

- 카트리지 복구가 라이브러리에서 카트리지를 찾을 수 없는 경우
- 홈 셀 및 드라이브(카트리지에 기록된 드라이브 위치가 있는 경우) 등 카트리지에 대해 기록된 위치 중 일부를 검사할 수 없는 경우

예를 들어 카트리지 복구가 오프라인 LSM 또는 오프라인 드라이브에서 홈 셀을 검사할 수 없는 경우 및 다른 위치에서 카트리지를 찾을 수 없는 경우 카트리지를 누락된 것으로 표시합니다.

카트리지 복구는 카트리지의 홈 셀을 검사하고 홈 셀에서 다른 카트리지를 찾는 경우가 아니면 카트리지의 홈 위치를 보존합니다. 이 상황에서는 카트리지를 "homeless"로 표시합니다(*home_lsm* 필드에 *-1*이 표시됨).

카트리지 복구에서 누락된 카트리지를 찾을 때 누락된 카트리지를 찾은 위치에 따라 데이터베이스에서 해당 카트리지의 상태를 "*home*" 또는 "*in drive*"로 변경합니다.

- 카트리지가 기록된 홈 셀이 아닌 셀에서 검색되는 경우 카트리지 복구는 카트리지의 홈 셀에서 중복된 카트리지를 검색했는지 여부를 확인합니다.
- 카트리지가 기록된 홈 셀에 없는 경우 카트리지 복구는 카트리지가 검색된 셀을 해당 새 홈 셀로 기록합니다.
- 새 카트리지가 중복 카트리지인 경우 카트리지 복구가 이를 이벤트 로그에 보고합니다. 중복된 카트리지를 꺼내지 않습니다.
- 카트리지 복구가 드라이브에서 "homeless" 카트리지를 찾는 경우 새 홈 셀을 지정하지 않습니다. 카트리지가 마운트 해제되면 마운트 해제 프로세스가 새 홈 셀을 지정합니다.

부재 카트리지 및 꺼낸 카트리지

이 절에서는 부재 카트리지 및 꺼낸 카트리지에 대해 설명합니다.

카트리지를 찾을 수 없음

카트리지 복구에서 기록된 위치를 모두 검사할 수 있고 카트리지를 찾을 수 없는 경우 다음을 수행합니다.

1. *ABSENT_CARTRIDGE_RETENTION_PERIOD*가 0인 경우 카트리지 복구는 다음을 수행합니다.
   - 데이터베이스에서 카트리지 레코드를 삭제합니다.
   - 카트리지의 홈 셀이었던 셀에 대한 데이터베이스의 셀 레코드를 "empty"로 표시합니다.
2. *ABSENT_CARTRIDGE_RETENTION_PERIOD*가 0보다 큰 경우 카트리지 복구는 다음을 수행합니다.
   - 카트리지가 아직 부재 카트리지 또는 꺼낸 카트리지로 표시되지 않은 경우 데이터베이스에 있는 카트리지 레코드의 상태를 "absent"로 변경합니다.
   - 카트리지를 "homeless"로 기록합니다(*home_lsm* 필드에 -1 표시).
   - 카트리지의 이전 홈 셀의 데이터베이스에 있는 셀 레코드를 "empty"로 표시합니다.

카트리지를 찾음

카트리지 복구가 꺼낸 카트리지 또는 부재 카트리지를 찾는 경우 카트리지를 재활성화합니다.

스토리지 셀에서 꺼낸 카트리지 또는 부재 카트리지가 검색되는 경우 해당 카트리지는 새 홈 셀이 되고 카트리지 복구는 데이터베이스에서 카트리지의 상태를 "home"으로 변경합니다.

드라이브에서 카트리지가 검색되는 경우 카트리지가 마운트 해제될 때 ACSLS가 새 홈 셀을 지정합니다.

수동 볼륨 삭제 유틸리티 사용

수동 볼륨 삭제 유틸리티 *del_vol*을 통해 오프라인이며 사용할 수 없는 LSM에 있는 볼륨에 액세스할 수 있습니다. LSM에서 카트리지를 수동으로 제거하고 해당 카트리지를 다른 LSM에 다시 넣으려는 경우 ACSLS는 *duplicate volume* 메시지를 표시하고 카트리지를 넣지 않습니다. *del_vol* 유틸리티를 사용하면 먼저 데이터베이스에서 볼륨을 삭제한 다음 오프라인 LSM에서 수동으로 제거하고 해당 볼륨을 온라인 LSM에 성공적으로 다시 넣을 수 있습니다.

del_vol 유틸리티는 이제 볼륨 삭제 옵션과 함께 볼륨을 부재 볼륨으로 보존합니다. 부재 상태 또는 꺼낸 상태의 만료를 기다리지 않고 볼륨을 삭제할 수 있습니다.

주:

온라인 LSM에서 카트리지를 제거하려면 해당 카트리지에 대해 *eject* 명령을 실행합니다. 카트리지가 실제로 LSM에 없는 경우 *-f*(force 옵션)로 *del_vol*을 실행할 수 있습니다. 이 유틸리티를 사용하려면 ACSLS 및 데이터베이스가 작동되고 실행 중이어야 합니다. 시스템이 복구 중일 때는 *del_vol*을 실행하지 마십시오. 예측할 수 없는 결과가 발생할 수 있습니다.

이 유틸리티에 대한 자세한 내용은 “del_vol ”을 참조하십시오.

del_vol 유틸리티를 사용하여 카트리지를 삭제하려면 다음을 수행합니다.

1. *acsss*로 로그인하십시오.
2. 카트리지를 삭제합니다.

```
del_vol vol_id
```

만료된 카트리지 식별

테이프 카트리지가 해당 설계 수명을 넘으면 매체가 약해지고 카트리지 게이트와 같은 기계 부품이 마모될 수 있습니다. 카트리지의 설계 수명이 끝나면 해당 카트리지의 데이터를 새 카트리지로 마이그레이션하고 기존 카트리지를 폐기해야 합니다. 이렇게 하면 기계적 카트리지 구성 요소에 오류가 발생하거나 데이터가 읽을 수 없게 될 위험을 피할 수 있습니다.

카트리지의 날짜상 수명과 사용량은 서로 다릅니다. 약 9,840개의 카트리지가 10년 동안 사용되었는데 사용 패턴은 서로 다릅니다. 심층 아카이브용으로 사용되어 거의 액세스하지 않은 카트리지가 있는 반면 매일 사용된 카트리지도 있습니다. 설계 수명을 넘은 카트리지를 식별하는 것이 중요합니다.

폐기되어야 할 카트리지를 식별하려면 사용량을 확인해야 합니다. 카트리지 사용량은 카트리지의 디렉토리에 기록되고 카트리지가 마운트 해제되기 전 드라이브가 해당 디렉토리를 업데이트합니다.

ACSLS에서 제어하는 라이브러리에 있는 카트리지의 경우:

- ACSLS가 관리하는 일부 라이브러리의 경우 카트리지 사용량이 "보증 수명" 및 "수명 종료"의 백분율로 표시됩니다.
- 이전 ACSLS 릴리스 및 라이브러리의 경우 ACSLS *access_count*가 *display* 명령 및 *volrpt* 유틸리티로 표시될 수 있습니다.

카트리지 수명 종료 백분율

최신 펌웨어를 실행 중인 최신 라이브러리 및 최신 펌웨어를 실행 중인 StorageTek 드라이브가 있을 경우 테이프 드라이브는 카트리지가 마운트 해제될 때 카트리지의 "보증 수명 종료" 및 "수명 종료 백분율"을 라이브러리에 보고합니다. 그런 다음 라이브러리가 이를 ACSLS에 보고합니다. ACSLS가 이 정보를 해당 데이터베이스에 저장하며 사용자는 ACSLS *display volume* 명령을 실행하여 이 정보를 볼 수 있습니다. “display 명령 옵션 사용 ”을 참조하십시오.

예: ACS, LSM, 매체 및 *end_of_life* 정보와 함께 end_of_life별로 정렬하여 T9840 카트리지를 모두 표시하려면 다음을 수행합니다.

```
display volume * -media STK1R -f acs lsm media end_of_life warranty
_life -s end_of_life
```

특히 이 정보는 다음 라이브러리 및 드라이브에 대해 ACSLS에 보고됩니다.

라이브러리:

- SL3000
- SL8500(4.10 펌웨어 사용)

테이프 드라이브:

- 모든 T10000 테이프 드라이브 - 1.38 펌웨어 사용
- T9840A, T9840C 및 T9840D(T9840B를 제외한 모든 T9840 테이프 드라이브) – 1.42 펌웨어 사용
- T9940A 및 T9940B 테이프 드라이브 – 1.42 펌웨어 사용

액세스 수

많은 경우 카트리지 수명 종료 보고를 사용할 수 없습니다. 이러한 경우 ACSLS *access_count*가 사용 가능한 최상의 정보입니다. ACSLS 데이터베이스는 볼륨이 선택되거나 액세스된 횟수를 기록합니다. 이는 카트리지가 ACS 내 연결된 라이브러리 그룹에 있었을 때 마운트된 횟수를 계산하는 데 사용될 수 있습니다.

이 정보는 라이브러리 유형에 상관없이 수집되므로 9310s, 4410s 및 9360s, SL8500s 및 SL3000s에 대해 유지 관리됩니다. *ACSLS*는 이 정보를 수십 년간 저장해 왔으므로 아직도 하위 레벨의 릴리스를 사용 중인 사용자라도 이 정보를 이용할 수 있습니다. 그러나 이 데이터에는 제한이 있습니다. 가장 큰 제한은 라이브러리에 카트리지를 넣었을 때 이 수가 0으로 설정된다는 점입니다.

정보는 설정된 보존 기간 동안 볼륨에 대해 보존되므로 해당 보존 기간 내에 ACS에서 카트리지를 꺼내 동일하거나 다른 ACS에 다시 넣어도 이 수가 보존됩니다. 기본 보존 기간은 5일입니다. 그러나 라이브러리에서 볼륨을 꺼낸 후 해당 볼륨이 볼륨 정보에 대한 보존 기간보다 더 오래 오프사이트에 남아 있는 경우 해당 볼륨에 대한 정보가 ACSLS 데이터베이스에서 삭제됩니다.

단일 라이브러리에 남아 있는 카트리지의 경우 이러한 ACSLS 액세스 수가 매우 유용합니다. T9840 카트리지의 경우 ACSLS *access_count*가 11,000을 초과하는 경우 해당 카트리지의 수명이 끝나가고 있거나 이미 끝난 것입니다. T10000 카트리지의 수명 종료 값은 마운트 16,000회입니다.

새 릴리스의 ACSLS를 설치할 때 데이터베이스 정보를 보존하고 마이그레이션할 수 있도록 ACSLS가 도구를 제공하므로 이 정보가 10년 뒤로 거슬러 올라갈 수 있습니다. 카트리지의 데이터가 없는 경우 이것이 유일한 옵션입니다.

ACSLS 카트리지 마운트 수 세부정보

ACSLS 필드를 *access_count*라고 합니다. 다음을 계산합니다.

- 마운트(마운트 해제는 계산되지 않음)
- 넣기 및 꺼내기(넣기 및 꺼내기는 매우 드묾)
- 이동(*cmd_proc*를 사용하는 *move* 명령이 거의 사용되지 않고 ACSAPI 클라이언트가 사용할 수 없음)

*access_count*는 주로 카트리지가 마운트된 횟수를 계산한 것입니다. ACSLS는 꺼낸 볼륨을 *ABSENT_VOLUME_RETENTION_PERIOD*(기본값: 5일) 동안 기억합니다. 카트리지를 ACS 간에 이동하고, 오프사이트로 전송하고, 온사이트로 다시 전환할 때 ACSLS는 *access_count*를 기억할 수 있습니다.

다음을 모두 사용하여 ACSLS *access_count*를 확인할 수 있습니다.

- ACSLS *display* 명령

  액세스 수별로 정렬되고 매체 유형 ACS 및 LSM도 표시하는 9840 데이터 카트리지를 모두 보려면 다음을 수행합니다.

  ```
  display volume * -media STK1R -s access_count -f media access_count acs lsm
  ```

- *volrpt* 유틸리티

  *volrpt*는 사용(*access_count*)별로 정렬할 수 있고 선택한 필드만 포함할 수도 있습니다. 예를 들어 *vol_id, media type, access_count* 및 *location*을 포함하는 사용자 정의 volrpt가 스크립트를 통한 추가 처리를 위해 플랫 파일로 출력될 수 있습니다.

카트리지 보증 및 수명 종료 임계값

보증 및 수명 종료 임계값은 아래 표에 나와 있습니다.

표 9.6. 수명 종료 임계값

임계값	마운트
9x40(T9840 및 T9940) 보증	10,000
9x40 수명 종료	11,000
T10000 보증	15,000
T10000 수명 종료	16,000

활성 LSM에서 가장 오래 전에 액세스한 카트리지 이동

ACSLS가 테이프 드라이브에서 카트리지를 마운트 해제할 때 다른 LSM에서 가져온 카트리지를 테이프 드라이브와 동일한 LSM의 새 홈 셀로 "Float"하는 방식의 전달을 방지하려고 합니다.

예를 들어 SL8500 라이브러리 3, 레일 2(LSM 9)에서 라이브러리 1, 레일 4(LSM 3)의 드라이브로 카트리지를 마운트하는 경우 두 번의 수평 전달 및 한 번의 엘리베이터 전달이 필요합니다. ACSLS가 카트리지를 마운트 해제할 때 LSM 4에서 새 홈 셀을 찾고 마운트 해제 시 전달을 피하려고 합니다.

문제:

LSM에 사용 가능한(지정되지 않음) 스토리지 셀이 없는 경우 카트리지를 LSM에 Float할 수 없습니다. 테이프 드라이브의 LSM에 사용 가능한 셀이 없는 경우 ACSLS는 계속 카트리지를 드라이브에 대해 가장 가까운 LSM으로 마운트 해제하려고 하는데 이 경우 한 번 이상 전달해야 합니다.

해결 방법:

오랜 시간 동안 액세스하지 않은 카트리지를 식별하고 해당 카트리지를 전체 LSM의 외부로 이동하여 카트리지가 마운트 해제 시 Float할 수 있는 사용 가능한 셀을 제공합니다.

ACSLS가 카트리지에 대해 기록하는 정보에서 카트리지를 마운트 또는 마운트 해제하거나, 넣거나, 이동할 때마다 *access_date*가 업데이트됩니다. *access_date*는 활성 상태가 아닌 이러한 카트리지를 식별하는 데 사용될 수 있습니다.

볼륨을 동일한 ACS 내에서만 이동하고 카트리지 꺼내기 또는 해당 상태 변경과 같이 이러한 카트리지의 향후 마운트를 방지할 작업을 수행하지 않으므로 이 전체 프로세스는 안전합니다.

가장 오래 전에 사용한 카트리지를 식별하고 이동하려면 아래 절차를 수행합니다.

- 사용 가능한 셀이 너무 적은 LSM 및 셀이 비어 있는 LSM을 식별합니다.
- 가장 이른 액세스 날짜별로 LSM의 카트리지를 선택합니다.
- 가득 찬 LSM에서 셀이 비어 있는 LSM으로 카트리지를 이동합니다.

사용 가능한 셀이 너무 적은 LSM 및 셀이 비어 있는 LSM 식별

cmd_proc 사용:

query lsm all

Free Cell Count 열을 통해 비활성 카트리지를 이동할 수 있는 사용 가능한 셀이 있는 LSM과 사용 가능한 셀이 거의 없거나 전혀 없는 LSM을 모두 식별할 수 있습니다.

예:

```
ACSSA> query lsm all
2011-08-29 18:15:45        LSM Status
Identifier  State  Free Cell  Audit  Mount  Dismount   Enter   Eject
                   Count      C/P    C/P     C/P       C/P     C/P
1,0        online   1         0/0    3/0     3/0       0/0     0/0
1,1        online   1         0/0    4/0     5/0       0/0     0/0
1,2        online   1         0/0    3/0     3/0       0/0     0/0
1,3        online   0         0/0    4/0     5/0       0/0     0/0
1,4        online   388       0/0   11/0     1/0       0/0     0/0
1,5        online   162       0/0    4/0     5/0       0/0     0/0
1,6        online   552       0,0    7/0     2/0       0/0     0/0
1,7        online   601       0/0    5/0     3/0       0/0     0/0
```

액세스 날짜별로 LSM의 카트리지 검토

이제 ACS의 다른 LSM으로 이동할 수 있는 비활성 카트리지를 식별해야 합니다.

간단한 정렬을 위해 액세스 날짜가 보고되는지 확인

정렬에 유용한 방식으로 액세스 날짜가 보고되는지 확인합니다. 날짜가 보고되는 형식은 TIME_FORMAT 동적 변수에 의해 제어됩니다.

- 카트리지를 액세스 날짜별로 쉽게 정렬할 수 있도록 기본 형식 *TIME_FORMAT=%Y-%m-%d %H:%M:%S*를 사용합니다. UNIX 명령 프롬프트에 다음을 입력합니다.

  ```
  dv_config -p TIME_FORMAT
  ```

  변수 프롬프트에 ?를 입력하여 도움말을 표시합니다.
- 변경사항을 적용한 경우 공유 메모리에서 동적 변수를 업데이트합니다.

  ```
  dv_config -u
  ```

LSM의 카트리지에 대한 마지막 액세스 날짜의 분포 확인

사용 가능한 셀이 충분히 확보되어 있지 않은 각 LSM에 대해 마지막 액세스 날짜별로 정렬된 카트리지를 나열합니다. *VOLID* 및 *access_date*만 선택하는 사용자 정의 *volrpt*가 필요합니다.

자세한 내용은 다음의 주석 헤더를 참조하십시오.

```
$ACS_HOME/data/external/volrpt/owner_id.volrpt
```

행의 필드는 *field_name*, *field_length* 및 *delimiter_length* (필드 후 공백)입니다.

다음 예에는 두 개의 활성 행이 있습니다. 6자리 *VOLUME_IDs*가 있습니다. *ACCESS_DATE*에서는 시간이 아닌 날짜 부분만 필요합니다.

```
VOLUME_ID        6     2
ACCESS_DATE     10     2
```

보고서를 만들려면 다음을 수행합니다.

1. *$cd ACS_HOME/data/external/volrpt*를 입력합니다.
2. *owner_id.volrpt*를 복사하고 *access_date.volrpt*와 같이 파일에 저장합니다.

   "로깅 볼륨 통계 보고서 작성"을 참조하십시오.
3. 텍스트 편집기를 사용하여 *ACCESS_DATE*를 편집합니다.
4. LSM에 대한 정렬된 카트리지 목록을 만듭니다.

   volrpt -l <lsm_id> -d -f access_date.volrpt | sort -k 2,2 -0 vols_sorted_lsm_##

   여기서 access_date.volrpt는 사용자 정의 보고서의 이름이며 ##은 LSM 번호입니다.

   vols_sorted_lsm_## 파일에서 각 LSM의 마지막 액세스 날짜에 대한 분포를 확인합니다.

가득 찬 LSM에서 셀이 비어 있는 LSM으로 카트리지 이동

이제 이동할 카트리지 목록을 만들고 카트리지를 사용 가능한 공간이 있는 LSM으로 이동해야 합니다.

이동할 카트리지 목록 만들기

1. 액세스 날짜별로 정렬된 카트리지 목록이 있는 파일을 가져와 액세스 날짜를 제거하여 카트리지 목록만 남깁니다.

   cat vols_sorted_LSM_## | cut –d" " –f1 > vols_LSM_##_tmp
2. 각 *vols_LSM_##* 파일을 가져와 이동할 카트리지 처음 100개(또는 지정하는 개수)를 선택합니다.

   head -100 vols_LSM_##_tmp > vols_LSM_##

   위 작업을 모두 조합할 수 있습니다.

   cat vols_sorted_LSM_## | cut –d" " –f1 | head -100 > vols_LSM_##

사용 가능한 공간이 있는 LSM으로 카트리지 이동

이동할 카트리지가 있는 소스 LSM 각각에 대해 해당 카트리지에 사용 가능한 공간이 있는 대상 LSM을 선택합니다.

1. moving.sh 유틸리티를 사용하여 카트리지를 *–t <lsm_id>*(예: –t 0,8)에 지정된 새 LSM으로 이동합니다.

   ```
   moving.sh -f vols_LSM_## -t <lsm_id>
   ```

2. 각 LSM에 대해 별도의 *moving.sh*를 실행합니다.

   라이브러리가 사용 중인 경우 한 번에 한 개 또는 두 개의 *moving.sh* 유틸리티만 실행할 수도 있습니다.

사용 안함으로 설정된 LSM의 드라이브에 수동으로 카트리지 로드

LSM에 오류가 발생하여 해당 LSM을 오프라인으로 전환하는 경우 데이터 경로를 여전히 사용할 수 있으면 카트리지를 라이브러리 드라이브에 계속 수동으로 로드할 수 있습니다.

사용 안함으로 설정된 LSM의 드라이브에 수동으로 카트리지를 로드하려면 다음을 수행합니다.

1. LSM 도어를 엽니다.
2. 이미 드라이브에 있는 카트리지의 카트리지 레이블을 기록해 두고 카트리지를 제거합니다. 이 절차의 끝에서 이 카트리지를 교체해야 합니다.
3. 읽거나 쓸 카트리지가 있는 드라이브를 로드합니다.

   LSM이 복구될 때까지 필요한 만큼 이 단계를 반복한 다음 4단계를 계속합니다.

   주의:

   이 단계에서는 라이브러리 셀에서 카트리지를 제거하고 해당 카트리지를 드라이브에 로드할 수 있습니다. 이러한 카트리지의 셀 위치를 기록해 두고 4단계에서 카트리지를 이 위치로 복귀시킵니다.

4. LSM이 복구된 후 드라이브에서 카트리지를 모두 제거하고 해당 카트리지를 2단계에서 기록해 둔 원래 카트리지로 교체합니다.
5. LSM 도어를 닫고, LSM을 다시 온라인으로 *vary*하고, 정상적인 작업을 재개합니다.

10

10장. 데이터베이스 관리

데이터베이스에는 라이브러리 구성 및 모든 라이브러리 카트리지 위치에 대한 모든 정보가 포함되어 있습니다.

백업 및 복구되는 ACSLS 제어 파일에는 *data/external* 아래의 *$ ACS_home*에 있는 고객이 구성 가능한 파일과 *data/internal/client_config*에 있는 일부 파일이 포함됩니다.

이 장에서는 데이터베이스 가져오기 및 내보내기, 가져온 데이터베이스 및 라이브러리 구성 확인, 데이터베이스 백업, 데이터베이스 복원 및 복구에 대해 설명합니다.

- 데이터베이스 내보내기 및 가져오기는 다음을 포함합니다.
  - 디스크 파일 또는 로컬 테이프 장치로 데이터베이스 내보내기
  - 디스크 파일 또는 로컬 테이프 장치에서 데이터베이스 가져오기
  - ACSLS 제어 구성 파일 가져오기
  - 사용자 정의된 동적 변수 병합
  - 가져온 데이터베이스 및 라이브러리 구성 확인
- 데이터베이스 및 ACSLS 제어 파일 백업은 다음을 포함합니다.
  - 자동 데이터베이스 백업
  - 로컬 테이프 장치 또는 디스크에 수동 백업 수행
  - UNIX 파일에 백업
  - 다른 서버에 복원 가능한 백업 만들기
- 데이터베이스 및 ACSLS 제어 파일 복구 및 복원은 다음을 포함합니다.
  - 가장 최근 백업으로 데이터베이스 복원
  - 특정 파일에서 복구
  - ACSLS 제어 파일 복원
  - 다른 서버에서 만든 백업 복원
  - 데이터베이스 다시 시작

사용되는 유틸리티

다음과 같은 유틸리티를 사용합니다.

- *bdb.acsss* 유틸리티:
  - 지정된 UNIX 파일에 백업
  - 테이프 장치에 백업

    - 기본 파일 및 위치에 백업
- *rdb.acsss* 유틸리티:
    - 데이터베이스 손상 복구
    - 의도하지 않은 결과를 생성하는 변경 복구
    - 서버 오류 복구
- *db_export.sh* 및 *db_import.sh* 유틸리티는 ACSLS 버전 간에 마이그레이션하는 데 사용됩니다. 여기에는 이후 릴리스로의 전환 또는 이전 릴리스로의 전환이 포함됩니다.

ACSLS를 설치하면 데이터베이스 관리 소프트웨어도 자동으로 설치됩니다. 다음과 같은 경우 ACSLS를 설치한 후 ACSLS 데이터베이스가 초기화됩니다.

- *acsss_config*를 사용하여 라이브러리 하드웨어를 구성하는 경우
- *db_import.sh*를 사용하여 이전에 내보낸 데이터베이스를 가져오는 경우
- *rbd.acsss.sh*를 사용하여 다른 서버에서 만든 데이터베이스 백업을 복구하는 경우

데이터베이스 내보내기

이 절에서는 ACSLS 데이터베이스 및 연관된 ACSLS 제어 파일을 이전 ACSLS 버전 또는 같은 ACSLS 릴리스 레벨에서 마이그레이션하거나 이전 릴리스로 되돌리는 방법에 대해 설명합니다.

주:

ACSLS가 실행 중인 동안에는 *db_export.sh*를 실행할 수 없습니다. 이로 인해 일관적인 데이터베이스 복사본을 얻을 수 있습니다.

db_export.sh 유틸리티는 테이프에 있는 데이터베이스 또는 디스크에 있는 지정된 파일의 ASCII 표현을 만들며, ACSLS 제어 파일도 수집합니다. 이 유틸리티는 두 가지 방법으로 사용할 수 있습니다.

- 이 유틸리티를 옵션 없이 실행하면 내보낸 파일이 기본 테이프 장치(*/dev/rmt/0n*)에 복사됩니다.

  ```
  db_export.sh
  ```

- 다른 테이프 장치를 사용하려면 *-f* 옵션 뒤에 원하는 테이프 장치를 지정합니다.

  ```
  db_export.sh -f /dev/rmt/3n
  ```

- 동일한 시스템의 로컬 파일에 내보내려면 *-f* 옵션을 사용하여 파일 경로 이름을 지정합니다.

  ```
  db_export.sh -f /export/save/acsls_export.03_Dec_2014
  ```

파일에 저장할 경우 2개의 개별 파일이 생성됩니다. 데이터베이스 테이블은 지정된 파일 이름으로 저장됩니다. 기타 제어 파일의 경로 이름은 *.misc* 확장자와 동일합니다.

그런 다음 *db_export.sh*에 의해 생성된 파일이 업그레이드 또는 복구 시 *db_import.sh* 유틸리티에 대한 입력으로 사용됩니다.

주:

이것이 모든 이전 ACSLS 버전을 최신 버전으로 마이그레이션하기 위한 더 좋은 방법입니다.

db_export.sh 유틸리티를 *-f* 옵션과 함께 또는 옵션 없이 실행할 경우 내보낼 ACSLS의 버전을 선택하라는 메시지가 표시됩니다.

```
$ db_export.sh
```

*/dev/tape*로 데이터베이스 내보내기

아래 옵션을 통해 선택하여 내보낼 릴리스를 선택합니다.

테이프에 내보낼 경우 no-rewind 장치가 필요합니다.

```
1: ACSLS 5.3.2 or 5.4
2: ACSLS 6.0 or 6.0.1
3: ACSLS 6.0.1 with L700e
4: ACSLS 6.0.1 with PUT0201
5: ACSLS 6.1, 7.0, or 7.1/7.1.1 before PUT0701
6: ACSLS 7.1/7.1.1 with PUT0701 or ACSLS 7.2 (any)
7: ACSLS 7.3 (any)
8: ACSLS 8.0, 8.01, 8.02, and 8.1
9: ACSLS 8.2 or later
E: Exit
```

이전 릴리스로 내보내기 전에 지원되지 않는 테이프 라이브러리, 드라이브 및 카트리지 제거

일부 테이프 라이브러리, 테이프 드라이브 또는 카트리지 매체 유형을 지원하지 않는 이전 ACSLS 릴리스로 데이터베이스를 내보낼 경우 데이터베이스를 내보내기 전에 지원되지 않는 테이프 라이브러리를 구성에서 제거하고 테이프 드라이브 및 카트리지를 라이브러리에서 제거합니다. 그렇게 하지 않으면 다음과 같은 상황이 발생할 수 있습니다.

- 라이브러리를 지원하지 않는 이전 ACSLS 릴리스를 선택할 경우 데이터베이스를 내보내기 전에 구성에서 라이브러리를 제거하라는 메시지가 표시됩니다.
- 이전 ACSLS 릴리스로 테이프 드라이브를 내보냈는데 이 릴리스에서 테이프 드라이브를 지원하지 않을 경우 해당 드라이브가 "알 수 없음"으로 보고되고 드라이브를 사용할 수 없습니다.
- 이전 ACSLS 릴리스로 카트리지를 내보냈는데 이 릴리스에서 해당 매체 유형을 지원하지 않을 경우 카트리지가 존재하지 않는 것으로 표시되며, 라이브러리에서 카트리지를 수동으로 제거해야 합니다.

Linux로 내보내기 전에 논리적 라이브러리 제거

Linux에서 실행되는 ACSLS는 광 섬유 대상을 사용하여 액세스되는 논리적 라이브러리를 지원하지 않습니다. Linux에서 실행되는 ACSLS로 데이터베이스를 내보낼 경우 논리적 라

이브러리를 모두 제거하십시오. 그렇게 하지 않으면 Linux에서 실행되는 어떠한 논리적 라이브러리도 사용할 수 없게 됩니다.

디스크 파일로 내보내기

다음 절차에 설명된 대로 ACSLS 데이터베이스 및 ACSLS 제어 파일을 디스크 파일로 내보낼 수 있습니다.

1. *acsss*로 로그인하십시오.
2. ACSLS를 사용 안함으로 설정합니다.

   acsss disable(UNIX 명령 프롬프트)
3. *db_export.sh* 유틸리티를 시작합니다.

   db_export.sh -f /path/db_file
4. 마이그레이션할 버전에 대해 원하는 옵션을 선택합니다.
   - 이 유틸리티를 실행하면 성공적인 테이블 데이터 내보내기를 나타내는 출력이 표시됩니다.
   - 내보내기가 완료되면 성공적으로 내보냈다는 메시지가 표시됩니다.
   - *db_export.sh* 유틸리티는 -f 옵션을 사용하여 지정한 위치에 *db_file* 및 *db_file.misc*라는 두 파일을 만듭니다.
5. 이러한 파일은 제거되지 않도록 안전한 위치에 저장하거나 안전한 위치로 이동해야 합니다.

   다음 디렉토리는 ACSLS 유지 관리를 설치할 때 제거 또는 삭제될 수 있으므로 이러한 파일을 해당 디렉토리에 저장하지 **마십시오**.
   - *$ACS_HOME*
     (ACSSS 홈 디렉토리)
   - *$ACSDB_BACKUP_DIR*
     (예: /export/backup)(ACSLS 백업이 저장되는 디렉토리)
   - */tmp*

     주:

     운영체제의 새 릴리스를 설치하려면 내보낸 파일을 ACSLS 서버에 저장하지 **마십시오**.
6. ACSLS 및 데이터베이스를 시작하려면 다음 명령을 입력합니다.

   acsss enable

테이프로 내보내기

다음 절차에 설명된 대로 ACSLS 데이터베이스 및 ACSLS 제어 파일을 테이프로 내보낼 수 있습니다.

데이터베이스 및 ACSLS 제어 파일을 테이프로 내보내려면 다음을 수행하십시오.

1. *acsss*로 로그인하십시오.

2. ACSLS를 사용 안함으로 설정합니다.

   acsss disable(UNIX 명령 프롬프트)
3. 기본 테이프 장치에 빈 테이프를 넣습니다.
4. *db_export.sh* 유틸리티를 시작합니다.

   db_export.sh -f tape_device

   예: *dbexport.sh -f /dev/rmt/0mn*
5. 내보내기에 원하는 옵션을 선택합니다.

   이 유틸리티를 실행하면 성공적인 테이블 데이터 내보내기와 성공적인 ACSLS 파일 백업을 나타내는 출력이 표시됩니다. 내보내기가 완료되면 메시지가 표시됩니다.
6. 프로그램이 완료되고 메시지가 다시 나타난 경우에만 드라이브에서 카트리지를 제거합니다.

   주의:

   프로그램에서 내보내기를 완료하기 전에 카트리지를 제거하면 파일이 손실됩니다. "카트리지 보호"라고 쓰고 컨텐츠가 내보낸 데이터베이스라고 명확히 표시하십시오.

   카트리지를 라이브러리에 남겨 두지 마십시오.
7. ACSLS 및 데이터베이스를 시작하려면 다음 명령을 입력합니다.

   acsss enable

데이터베이스 가져오기

db_import.sh 유틸리티를 사용하면 다음 속성이 새 데이터베이스로 가져와집니다.

- 볼륨: 이러한 데이터베이스 테이블에 다음과 같이 라이브러리의 각 볼륨과 연관된 모든 정보가 포함됩니다.
  - 볼륨의 위치
  - 카트리지 유형(예: 데이터, 스크래치, 청소)
  - 마지막으로 연결된 스크래치 풀
  - 카트리지의 현재 상태(홈, 마운트됨 등)
  - 항목 날짜 및 마지막 액세스 날짜
  - 항목 날짜 이후의 마운트 수
  - 최대 사용(청소 카트리지)
  - 연관된 잠금 ID 및 사용자 ID(카트리지가 잠겨 있는 경우)
- ACS 및 라이브러리: 데이터베이스 테이블에 ACS 및 라이브러리 구성 요소(예: LSM, 드라이브, 패널, 셀)가 포함됩니다.
- ACSLS 제어 파일에 다음을 비롯하여 초기 설치 이후의 모든 구성 업데이트가 포함됩니다.
  - 액세스 제어 정보

  - 고정 볼륨 환경 설정
  - 스크래치 매체 환경 설정
  - 사용자 정의 volrpt 템플리트
- 동적 및 정적 변수: 이전 릴리스에서 사용자 정의된 동적 변수를 가져올 수 있습니다.

이 절에서는 *db_import.sh* 유틸리티를 사용하여 다음을 수행하는 방법에 대해 설명합니다.

- ACSLS 데이터베이스 다시 만들기
- 중요 ACSLS 제어 파일 복구
- *db_export.sh* 유틸리티를 사용하여 내보낸 데이터에서 사용자 정의 동적 변수 복구

디스크 파일에서 가져오기

다음 절차에 설명된 대로 ACSLS 데이터베이스 및 ACSLS 제어 파일을 디스크 파일에서 가져올 수 있습니다.

ACSLS 데이터베이스, ACSLS 제어 파일 또는 사용자 정의 동적 변수를 디스크 파일에서 가져오려면 다음을 수행하십시오.

1. *acsss*로 로그인하십시오.
2. ACSLS를 사용 안함으로 설정합니다.

   acsss disable(UNIX 명령 프롬프트)
3. *db_import.sh* 유틸리티를 시작합니다.

```
db_import.sh -f db_file

ACSLS Import Utility

   If importing from tape, a no-rewind device is required.

    What would you like to do:

    1) Import data, control files, and dynamic variables from
       from a DIFFERENT release or platform version of ACSLS (upgrade)

    2) Import data, control files, and dynamic variables from the SAME
       release (version and PUT level) and platform of ACSLS(Disaster Recovery)

    3) Import database tables only (any level of ACSLS)

    4) Import control files only (any level of ACSLS)

    5. Merge customized dynamic variables only (any level of ACSLS)
```

```
E) Exit

Please select one of the above:
```

- 옵션 1 - 다른 릴리스 또는 플랫폼 버전에서 데이터, 제어 파일 및 동적 변수 가져오기

  다른 릴리스로 이동하거나 ACSLS를 업그레이드할 때 이 옵션을 사용하여 데이터베이스 파일, 제어 파일 및 동적 변수를 가져옵니다.

  주의:

  기존 데이터베이스, 제어 테이블 및 동적 변수 설정은 삭제되고 재구성된 다음 내보내기를 통해 제공된 데이터로 채워집니다. 결과는 최종적이며, 복구하려면 데이터베이스를 재구축해야 합니다. 기존 테이블의 정보를 유지하려면 *db_export.sh*를 사용하여 테이블 데이터를 내보낸 다음 계속하십시오.

  이 옵션은 이전 환경의 사용자 정의 동적 변수도 복구합니다. 따라서 이전 사용자 정의 동적 변수를 기록할 필요 없이 ACSLS 버전을 업그레이드할 때 유용합니다. 액세스 제어 파일을 비롯하여 *data/external* 아래의 *acs.home* 디렉토리에 있는 모든 파일이 복구됩니다. 액세스 제어를 구성한 경우 *data/internal/client_config*도 복구됩니다.
- 옵션 2 - 동일한 릴리스 또는 플랫폼 버전에서 데이터, 제어 파일 및 동적 변수 가져오기

  데이터베이스와 제어 파일을 모두 포함하여 ACSLS 환경을 다시 만들려면 이 옵션을 사용합니다. 이 옵션은 다음과 같은 경우에 사용됩니다.
  - 하드웨어 업그레이드 중이나 하드웨어 오류에서 복구할 때
  - 데이터를 내보낸 ACSLS 서버와 동일하게 ACSLS 서버를 재구축해야 함
- 옵션 3 - 임의의 ACSLS 릴리스 레벨에서 데이터베이스 테이블만 가져오기

  ACSLS 릴리스 레벨에서 데이터베이스 파일만 가져오려면 이 옵션을 사용합니다.

  이 옵션은 기존 데이터베이스 테이블과 제어 파일을 삭제하고 재구성한 다음 내보낸 데이터베이스를 통해 제공된 데이터로 채웁니다. 기존 테이블의 정보를 유지하려면 *db_export.sh*를 사용하여 데이터를 내보낸 다음 계속하십시오.
- 옵션 4 - 임의의 ACSLS 릴리스 레벨에서 ACSLS 제어 파일 가져오기

  임의의 ACSLS 버전에서 ACSLS 제어 파일만 가져오려면 이 옵션을 사용합니다. 이 옵션은 액세스 제어 파일을 비롯하여 *data/external* 아래의 *acs.home* 디렉토리에 있는 모든 파일을 가져옵니다. 액세스 제어를 구성한 경우 *data/internal/client_config*도 가져와집니다.

  이 옵션은 동일한 버전에서 ACSLS 데이터베이스 파일, 제어 파일 및 동적 변수를 복구합니다. 이 옵션은 액세스 제어 파일을 비롯하여 *data/external* 아래의 *acs.home* 디렉토리에 있는 모든 파일을 복구합니다.

이 옵션은 이전 환경의 사용자 정의 동적 변수를 복구합니다. 따라서 이전 사용자 정의 동적 변수를 기록할 필요 없이 ACSLS 버전을 업그레이드할 때 매우 유용한 옵션입니다.

이 옵션을 선택하면 데이터베이스 내보내기를 통해 설정이 수집된 다음 새 변수 설정으로 공유 메모리가 재구성됩니다.

- 옵션 5 - 사용자 정의 동적 변수만 병합

  따라서 이전에 사용자 정의한 동적 변수를 기록할 필요 없이 ACSLS 버전을 업그레이드할 때 매우 유용한 옵션입니다. 이 옵션을 선택하면 데이터베이스 내보내기를 통해 설정이 수집되고 새 변수 설정으로 공유 메모리가 재구성됩니다.

  경고:

  ACSLS 7.2.0에서 가져올 때 이 옵션을 실행하기 전에 ACSLS를 시작하면 특정 데이터가 손실될 수 있습니다. 이전 버전에서 ACSLS를 업그레이드하는데 사용자 정의 동적 변수가 있는 경우 ACSLS를 시작하기 전에 사용자 정의 변수를 가져와야 합니다.

4. "가져온 데이터베이스 및 라이브러리 구성 확인 "에 설명된 대로 설치를 확인합니다.
5. ACSLS를 시작하려면 다음 명령을 입력합니다.

   acsss enable

테이프에서 가져오기

ACSLS 데이터베이스를 가져오고, ACSLS 제어 파일을 복구하고, 테이프에서 사용자 정의 동적 변수를 재구성하려면 다음 절차를 따르십시오.

1. *acsss*로 로그인하십시오.
2. ACSLS를 사용 안함으로 설정합니다.

   acsss disable(UNIX 명령 프롬프트)
3. *db_export.sh* 명령을 사용하여 내보낸 데이터베이스 테이프를 테이프 드라이브에 넣습니다.
4. UNIX 명령 프롬프트에 다음을 입력하여 데이터베이스 가져오기 유틸리티를 실행합니다.

   db_import.sh

   "디스크 파일에서 가져오기 "에 설명된 대로 *db_import.sh* 유틸리티의 기본 메뉴가 표시되고 자세한 정보가 제공됩니다.

   주:

   한 터미널에서 테이프를 되감고 있는 동안 다른 터미널에서 *db_import* 유틸리티를 실행하면 "실패" 메시지가 표시됩니다.
5. 메뉴 옵션은 3 단계를 참조하십시오.
6. "가져온 데이터베이스 및 라이브러리 구성 확인 "에 설명된 대로 설치를 확인합니다.

7. 기본 테이프 장치(no-rewind)가 아닌 테이프에서 가져옵니다.
8. ACSLS를 시작하려면 다음 명령을 입력합니다.

   ```
   acsss enable
   ```

새 플랫폼으로 광 섬유 mchanger 마이그레이션

SCSI 매체 교환기(mchanger)는 ACSLS와 광 섬유 연결 라이브러리 간에 통신을 진행하는 광 섬유 연결 라이브러리 장치 드라이버입니다. ACSLS에 연결되는 각 광 섬유 연결 라이브러리에 대해 하나의 mchanger를 만들어야 합니다.

문제를 발생시킬 수 있는 다른 플랫폼 및/또는 릴리스로 ACSLS를 가져올 경우 */dev/mchanger#* 장치 드라이버 링크의 숫자가 변경될 수 있습니다. 예를 들어, 이전 ACSLS 서버에서 */dev/mchanger3*을 통해 연결된 SL500 또는 SL150 라이브러리의 경우 새 ACSLS 서버에서 */dev/mchanger4*를 통해 연결될 수 있습니다.

Linux에서 mchanger 이름 형식이 다르기 때문에 한 ACSLS Linux 서버에서 새 Linux 서버로 이동할 경우 이는 문제가 되지 않습니다. Linux 서버에서는 mchanger 이름에 숫자 대신 라이브러리 일련 번호가 포함됩니다.

광 섬유 연결 라이브러리에 대해 매체 교환기 드라이버를 구성하고 새 ACSLS 릴리스 또는 서버 플랫폼으로 마이그레이션할 때 다음 절차를 따르면 문제를 방지할 수 있습니다.

1. Solaris 또는 AIX ACSLS 서버에서 이전 ACSLS 서버의 각 광 섬유 연결 라이브러리와 연관된 mchanger 번호를 기록합니다. "이전 ACSLS 서버의 광 섬유 연결 라이브러리에 대한 세부정보 기록 "을 참조하십시오.
2. 라이브러리의 새 mchanger 이름으로 구성을 업데이트합니다. "광 섬유 연결 라이브러리의 mchanger 이름을 변경하기 위해 ACSLS 재구성 "을 참조하십시오.

이전 ACSLS 서버의 광 섬유 연결 라이브러리에 대한 세부정보 기록

Solaris 또는 AIX ACSLS 서버에서 이전 ACSLS 서버의 데이터베이스를 내보내기 전에 이전 ACSLS 서버의 각 광 섬유 연결 라이브러리와 연관된 mchanger 번호를 기록합니다. 각 광 섬유 연결 ACS와 연관된 mchanger와 라이브러리의 일련 번호를 표시하는 *cmd_proc* 및 *showDevs.sh* 유틸리티의 출력을 저장합니다.

cmd_proc:

- *query lmu all*

  ACSLS에 의해 제어되는 모든 ACS와 해당 포트 연결을 표시합니다. Solaris 및 AIX 시스템에서 광 섬유 연결 라이브러리의 포트 이름은 */dev/mchanger#*입니다. 여기서 *#*은 숫자입니다.

- *display lsm * -f type serial_num*

  ACSLS에 의해 관리되는 모든 LSM의 라이브러리 유형과 일련 번호를 표시합니다. 라이브러리 유형(예: SL500 또는 SL150)을 사용하여 광 섬유 연결 라이브러리를 식별합니다. 일련 번호를 사용하여 특정 라이브러리를 식별합니다.

유틸리티:

```
showDevs.sh -s
```

showDevs.sh 유틸리티를 –s 옵션과 함께 사용하면 광 섬유 연결 라이브러리를 식별하는 mchanger 장치 링크, 라이브러리 유형, 라이브러리 일련 번호 및 세부정보가 표시됩니다.

광 섬유 연결 라이브러리의 mchanger 이름을 변경하기 위해 ACSLS 재구성

데이터베이스를 가져온 후 Linux로 또는 Linux에서 마이그레이션하거나 Solaris에서 동일한 mchanger 번호를 구성하지 않은 경우 이러한 라이브러리에 대해 새 mchanger 이름으로 구성을 업데이트해야 합니다.

acsss_config 사용:

1. *acsss*로 로그인하십시오.
2. *showDevs.sh*를 사용하여 모든 광 섬유 연결 라이브러리를 표시합니다.

   showDevs.sh 유틸리티를 *–s* 옵션과 함께 사용하면 광 섬유 연결 라이브러리를 식별하는 mchanger 장치 링크, 라이브러리 유형, 라이브러리 일련 번호 및 세부정보가 표시됩니다.
3. *acsss_config*에서 프롬프트로 복사해서 붙여 넣을 수 있도록 *showDevs.sh*의 출력을 파일로 저장합니다.
4. *showDevs.sh -s*의 출력을 한 터미널 창에 표시한 상태에서 두번째 터미널 창을 열고 *acsss*로 로그인합니다.
5. 두번째 터미널 창에서 *acsss_config*를 실행합니다.
6. Option 8: Define or Change Library Configuration을 선택합니다.
7. Configure library communications? (y/n)에 **y**로 응답합니다.
8. Library server data base exists and will be overwritten, continue (y or n)?에 **y**로 응답합니다.
9. *query lmu all*의 저장된 출력을 참조하여 이전 ACSLS 서버에 구성된 모든 ACS를 재구성합니다.
   a. 이전 ACSLS 서버에서와 동일한 ACS 번호와 동일한 순서로 모든 ACS를 구성합니다.
   b. 이전 ACSLS 서버에서와 동일한 포트 연결을 사용하여 광 섬유가 아닌 연결 라이브러리를 분할되거나 분할되지 않은 것으로 구성합니다.
10. 광 섬유 연결 라이브러리에 대한 포트 연결을 구성할 경우 새 ACSLS 서버에서 사용되는 새 mchanger 링크 이름을 지정합니다. Linux에서 mchanger 링크 이름을 지정하는 쉬운 방법은 *showDevs.sh* 출력에서 mchanger 링크 이름을 복사하여 *acsss_config* 프롬프트 뒤에 붙여 넣는 것입니다.
11. ACSLS 라이브러리 하드웨어 재구성을 완료합니다.

가져온 데이터베이스 및 라이브러리 구성 확인

카트리지를 마운트 또는 마운트 해제하여 ACSLS를 확인하려면 다음 절차를 따르십시오.

1. *acsss*로 로그인했는지 확인합니다.
2. ACSLS가 실행되고 있지 않은 경우 다음 명령을 입력하여 ACSLS를 시작합니다.

   ```
   acsss enable
   ```
3. 다음 명령을 입력하여 *cmd_proc*에서 서버를 질의합니다.

   ```
   query server
   ```

   서버가 복구 모드에 있음을 나타내는 메시지가 표시되는 경우 서버가 실행 중임을 나타내는 메시지가 표시될 때까지 기다립니다.
4. 다음 항목이 온라인 상태인지 확인합니다. 온라인 상태가 아닌 경우 *vary* 명령을 사용하여 온라인 상태로 전환합니다.

   ```
   query port all

   query acs all

   query lsm all

   query drive all
   ```
5. LSM에 카트리지가 하나 이상 있습니까?
   - 예 - 절차를 계속합니다.
   - 아니오 - LSM에 카트리지를 넣습니다.
6. 다음 명령을 입력하여 카트리지를 마운트합니다.

   ```
   mount vol_id drive_id
   ```

   사용 가능한 드라이브의 ID를 가져오려면 *query drive* 명령을 사용하고, 라이브러리 카트리지의 ID를 가져오려면 *query volume* 명령을 사용합니다.
7. 성공적인 마운트를 나타내는 메시지가 표시되었습니까?

   성공적인 마운트 메시지는 다음과 같습니다.

   ```
   Mount: vol_id mounted on drive_id
   ```
   - 예 - 절차가 완료되었습니다.
   - 아니오 - 오류 메시지가 나타나는 경우 이 확인 절차를 다시 실행하여 사용 가능한 유효 드라이브 및 라이브러리 카트리지를 지정했는지 확인합니다. 마운트 또는 마운트 해제가 계속 실패하는 경우 고객 지원 센터에 문의하십시오.
8. 다음 명령을 입력하여 카트리지를 마운트 해제합니다.

   ```
   dismount vol_id drive_id force
   ```

여기서 *vol_id*는 볼륨이고 *drive_id*는 6단계에서 지정한 드라이브입니다.

자동 데이터베이스 백업

ACSLS에서는 매일 자정이나 *acsss_config*의 백업 옵션에 지정된 시간과 요일에 디스크에 데이터베이스 백업 파일을 자동으로 만듭니다.

테이프에 수동 백업 수행

ACSLS에서 만드는 자동 데이터베이스 백업 이외에 *bdb.acsss* 유틸리티를 주기적으로 실행하여 테이프 백업을 수동으로 만들어야 합니다. 이 테이프 백업을 오프사이트에 저장했다가 필요 시 데이터베이스 재해 복구에 사용할 수 있습니다.

ACSLS 서버에 재해가 발생할 경우 오프사이트 장치로 전송된 일반 백업을 사용하여 빠르게 복원할 수 있습니다.

다음 작업 후 *bdb.acsss*를 사용하여 데이터베이스를 테이프에 수동으로 백업합니다.

- *acsss_config* 실행
- 데이터베이스 가져오기
- 전체 라이브러리 감사
- 모든 데이터베이스 복구

ACSLS 서버에 연결된 지정된 테이프 장치에 백업

ACSLS 서버에 연결된 지정된 테이프 장치에 ACSLS 데이터베이스를 백업하려면 다음을 수행하십시오.

1. *acsss*로 로그인하십시오.
2. 테이프 장치에 빈 테이프를 넣습니다.
3. 터미널 창에서 다음 명령을 입력합니다.

   bdb.acsss -f tape_device

   여기서 *tape_device*는 ACSLS 서버에 연결된 테이프 장치를 지정합니다.
4. 백업의 진행 상황을 보고하는 메시지가 나타납니다.

   다음 메시지가 나타날 때까지 기다립니다.

   Check tape device (/dev/rmt/0mn) to make sure you have a tape in the tape drive.

   [Hit RETURN to continue or Ctrl-C to exit]

   Press RETURN.

5. 다음 메시지가 나타날 때까지 기다립니다.

   ```
   ACSLS database backup successfully completed.
   ```

   예: ACSLS 데이터베이스를 */dev/rmt/0mn* 테이프 장치에 백업하려면 다음 명령을 입력합니다.

   ```
   bdb.acsss -f /dev/rmt/0mn
   ```

UNIX 파일에 백업

재해 복구를 위해 UNIX 파일이 원격 디스크에 있지 않은 한 UNIX 파일에 백업하지 않는 것이 좋습니다. “bdb.acsss ”를 참조하십시오.

UNIX 파일에 ACSLS 데이터베이스를 백업하려면 다음을 수행하십시오.

1. *acsss*로 로그인하십시오.
2. 터미널 창에서 다음 명령을 입력합니다.

   ```
   bdb.acsss -f db_file
   ```

   여기서 *db_file*은 ACSLS 데이터베이스를 포함할 UNIX 파일을 지정합니다. 파일에 대한 쓰기 권한이 있어야 합니다.
3. 다음 메시지가 나타날 때까지 기다립니다.

   ```
   ACSLS database backup successfully completed.
   ```

복구 및 복원

이 절에서는 다음 복원/복구 절차에 대해 설명합니다.

- 가장 최근 백업으로 손상되거나 손실된 데이터베이스 복원
- 지정된 날짜 및 시간으로 손상되거나 손실된 데이터베이스 복원
- 디스크 장애에서 복구
- 고장난 서버에 대한 재해 복구
- 특정 백업 파일에서 복구
- 비데이터베이스 ACSLS 제어 파일 복원

대부분의 절차에서는 *rdb.acsss* 유틸리티를 사용합니다. 이 유틸리티에서는 가장 최근 백업이나 지정한 날짜 및 시간에서 데이터베이스를 복원하고, *bdb.acsss*로 만든 백업을 사용하여 재해 복구를 진행하고, *bdb.acsss*로 만든 ACSLS 제어 파일을 복원하는 옵션을 제공합니다. 이러한 옵션에 대한 자세한 내용은 “rdb.acsss ”를 참조하십시오.

주:

카트리지의 홈 셀이 백업 후 마지막 위치와 다르게 변경된 경우 복원된 데이터베이스가 최신 버전이 아닙니다. 마운트 해제 시 카트리지 이동을 방지하려면 각 LSM이 ACS 내의 유일한 LSM이거나(대부분의 SCSI 라이브러리에 해당) 전달 포트를 통해 다른 LSM에 연결되는 모든 LSM에 대해 Extended Store Feature를 사용으로 설정해야 합니다.

자세한 내용은 “Extended Store 기능 사용 ”을 참조하십시오. 모든 연결된 LSM에 대해 Extended Store Feature를 사용으로 설정하지 않았거나 카트리지를 넣거나 꺼낸 경우 복원 후 라이브러리를 감사하여 데이터베이스를 최신 상태로 만들고 전달 포트를 통해 다른 LSM에 연결되는 모든 LSM을 사용으로 설정하십시오.

주:

-f 옵션을 *rdb.acsss* 유틸리티에 대한 일반 옵션으로 지정하지 마십시오. 외부 네트워크 파일 또는 대체 테이프 장치에 데이터베이스를 백업한 경우 *rdb.acsss*를 입력한 후에만 *-f* 옵션을 사용하십시오. 세번째 복구 옵션을 선택합니다. 메시지가 표시되면 *-f*와 외부 네트워크 파일 또는 대체 테이프 장치에 대한 경로 이름을 입력합니다. 자세한 내용은 6를 참조하십시오.

가장 최근 백업으로 데이터베이스 복원

이 절차에서는 자동 백업을 통해 로컬 디스크에 생성된 가장 최근 백업으로 데이터베이스를 복원합니다. ACSLS 제어 파일도 복원됩니다.

가장 최근 백업으로 손상되거나 손실된 데이터베이스를 복원하려면 다음 단계를 완료하십시오.

1. *acsss*로 로그인하십시오.
2. ACSLS를 사용 안함으로 설정합니다.

   ```
   acsss disable
   ```
3. 다음 명령을 입력합니다.

   ```
   rdb.acsss
   ```
4. 옵션 1을 선택합니다.

   ```
   1. Restore from a current local disk backup
   ```
5. 절차는 “rdb.acsss ”를 참조하십시오.
6. ACSLS를 시작하려면 다음 명령을 입력합니다.

   ```
   acsss enable
   ```

고장난 서버에서 복구

기본 디스크와 보조 디스크가 모두 손실되거나 손상된 경우 이 절차를 사용하여 재해 복구를 진행할 수 있습니다.

고장난 서버에서 복구하려면 다음을 완료하십시오.

1. 운영체제를 설치합니다.

2. ACSLS를 설치합니다.

   주의:

   디스크 장애가 발생하기 전에 사용한 것과 동일한 디렉토리에 ACSLS를 설치해야 합니다.

3. *acsss*로 로그인하십시오.
4. ACSLS를 사용 안함으로 설정합니다.

   ```
   acsss disable
   ```

5. 다음 명령을 입력합니다.

   ```
   rdb.acsss
   ```

6. 옵션 2를 선택합니다.

   ```
   2. Restore from a previous tape or network file backup
   ```

7. 절차는 “rdb.acsss ”를 참조하십시오.
8. ACSLS를 시작하려면 다음 명령을 입력합니다.

   ```
   acsss enable
   ```

9. 기본 설정을 수락하지 않으려면 *acsss_config*를 실행하여 자동 백업 날짜 및 시간과 보존 기간을 다시 지정해야 합니다.

ACSLS 제어 파일 복원

이 절차에서는 ACSLS 제어 파일을 복원합니다. 이러한 파일은 *data/external* 디렉토리에 있는 모든 파일(예: 액세스 제어 파일, 고정 볼륨 파일, 스크래치 환경 설정 파일, 사용자 정의 *volrpt* 파일)을 포함하는 비데이터베이스 파일입니다. 이러한 파일은 *bdb.acsss* 테이프 백업 또는 외부 네트워크 파일에서 복원됩니다.

ACSLS 제어 파일을 복원하려면 다음을 완료하십시오.

1. *acsss*로 로그인하십시오.
2. ACSLS를 사용 안함으로 설정합니다.

   ```
   acsss disable
   ```

3. 다음 명령을 입력합니다.

   ```
   rdb.acsss
   ```

4. 옵션 4를 선택합니다.

   ```
   Restore only ACSLS non-database control files
   ```

5. 절차는 “rdb.acsss ”를 참조하십시오.
6. ACSLS 및 데이터베이스를 시작하려면 다음 명령을 입력합니다.

   ```
   acsss enable
   ```

11

11장. 보고 및 로깅

이 장에서는 다음 작업을 수행하는 방법을 설명합니다.

- 사용자 정의 볼륨 보고서 작성
- 볼륨 이동 통계 보고서 작성

사용자 정의 볼륨 보고서 작성

volrpt 유틸리티를 사용하여 볼륨 보고서를 작성할 수 있습니다. 자세한 내용은 “volrpt ”의 내용을 참조하십시오. *$ACS_HOME/data/external/volrpt/owner_id.volrpt*는 사용자 정의된 볼륨 보고서를 작성하는 템플리트로 사용하거나 실행할 수 있는 샘플 입력 파일입니다. *$ACS_HOME/data/external/volrpt* 디렉토리에 사용자 정의된 볼륨 보고서를 저장할 수도 있습니다. 사용자 정의 볼륨 보고서를 작성하려면 이 절차를 사용하여 *volrpt* 유틸리티에 대한 입력 파일을 작성합니다.

사용자 정의 볼륨 보고서를 작성하려면 다음을 완료합니다.

1. *acsss*로 로그인하십시오.
2. UNIX 명령 도구를 엽니다.
3. 다음과 같이 사용자 정의 볼륨 보고서 디렉토리로 변경합니다.

   ```
   cd /home/ACSSS/data/external/volrpt
   ```
4. 샘플 볼륨 보고서 파일을 새 사용자 정의 파일로 복사합니다.

   ```
   cp owner_id.volrpt my.volrpt
   ```

   *my.volrpt*는 새 파일에 지정할 이름입니다.
5. 사용자 정의 보고서에 표시할 필드와 형식을 지정하려면 vi와 같은 텍스트 편집기를 사용하여 *my.volrpt* 파일을 편집합니다.
   - 샘플 파일에 나열된 필드를 지정할 수 있습니다.
   - 각 항목의 형식은 *field_name field_length delimiter_length*입니다.
   - 원하는 크기로 필드 길이와 분리자를 만들 수 있습니다. 지정하는 모든 필드가 보고서를 인쇄할 때 하나의 행에 들어가기만 하면 됩니다.
   - 샘플 파일에 자세한 편집 지침이 제공됩니다.
6. 파일 편집이 완료되면 파일을 저장합니다.

사용자 정의 볼륨 보고서

다음 예제에 표시된 파일과 같은 입력 파일에 필드, 필드 길이 및 구분자 길이를 지정하여 사용자 정의된 보고서를 만듭니다.

```
####################################################################
#
# FIle name: owner_id.volrpt
#
# This file describes the report layout for volrpt invoked with   # the -f option. #
 volrpt -f <filename>
#
# The format of a line is:
# field_namefield_lengthdelimiter_length
#
# The field length is the number of characters which will be printed for
# the field. The delimiter length is the number of spaces that will be
# printed after the field. If you leave out the lengths, or specify a
# value of -1, the default values will be used. Default delimiters are
# always 2. ere are the fields and their default lengths.
#
# ACCESS_COUNT       5      2
# ACCESS_DATE       15      2
# CELL_ID           14      2
# DRIVE_ID          10      2
# ENTRY_DATE        15      2
# LABEL_ATTR         5      2
# LOCK_ID            5      2
# LOCK_TIME         15      2
# MAX_USE            5      2
# MEDIA_TYPE         7      2
# OWNER_ID          20      2
# POOL_ID            5      2
# VOLUME_ID          6      2
# VOL_STATUS        17      2
# VOLUME_TYPE        4      2
#
# Revision History:
# xx/xx/xx Name     Changes
#
####################################################################
VOLUME_ID            6      2
MEDIA_TYPE           7      2
DRIVE_ID            12      2
CELL_ID             14      2
OWNER ID            -1      0
```

표시된 사용자 정의 보고서를 이 입력 파일을 사용하여 작성합니다. 무엇보다도 사용자 정의 보고서는 다음 예제에 표시된 대로 볼륨의 소유자를 보고하는 데 사용할 수 있습니다.

입력 파일을 사용하는 사용자 정의 볼륨 보고서의 예:

```
2014-06-30 13:22:07

TOTAL    VOLUMES:2   SEQUENCE:   sort by volume identifier

Volume   Media                     Home                   Owner
```

```
Label    Type       Drive ID       Location              ID

RB1400   3480       Not-in-drv     0, 1, 1, 0, 0         cray
RB1401   DD3A       0, 0, 1, 0     0, 1, 2, 0, 0         cray
```

사용자 정의 볼륨 보고서 예제

다음 *volrpt*는 ACS 0 및 1, 비어 있고 꺼낸 볼륨을 포함하며, 볼륨의 상태를 표시합니다.

입력 *volrpt* 옵션:

```
volrpt -f my.volrpt -a 0 1 -i
```

이러한 제어문은 출력을 선택하고 포맷하는 데 사용합니다.

사용자 정의 볼륨 보고서 제어문의 예제:

```
CELL_ID          14           2
VOLUME_ID         6           2
VOL_STATUS       17           0
POOL_ID           5           2
ACCESS_COUNT      5           1

LOCK_ID           5           1
OWNER_ID         20           0
```

비어 있거나 꺼낸 볼륨을 표시하는 사용자 정의 *volrpt*의 예제:

```
VOLUME REPORT UTILITY

2014-06-03 15:27:48

TOTAL VOLUMES: 61      SEQUENCE: sort by volume identifier

Home            Volume   Volume         Pool  Times  Lock Owner
Location---     Label   Status          ID    Mount  ID   ID------------

1, 0, 0, 0, 0   ABC001  VOLUME_HOME      0    2     0     presc
0,-1, 0, 0, 0   ABC002  VOLUME_ABSENT    0    0     0
0,-1, 0, 0, 0   ABC003  VOLUME_ABSENT    0    0     0
1, 3, 0, 0, 3   ABC004  VOLUME_MISSING   0    0     0
1, 3, 0, 0, 4   ABC005  VOLUME_MISSING   4    0     28001 tom
1, 3, 0, 0, 5   ABC006  VOLUME_MISSING   0    0     0
0,-1, 0, 0, 0   ABC007  VOLUME_ABSENT    0    0     0
1, 0, 0, 0, 7   ABC008  VOLUME_HOME      0    0     0
0,-1, 0, 0, 0   ABC009  VOLUME_ABSENT    0    0     0
0,-1, 0, 0, 0   ABC010  VOLUME_ABSENT    0    0     0     presc
1, 0, 0, 0,10   ABC011  VOLUME_HOME      0    0     0
1, 0, 0, 0,12   ABC012  VOLUME_HOME      0    0     2371  abc012
1, 0, 0, 0,13   ABC013  VOLUME_HOME      0    0     28001
1, 0, 0, 0,14   ABC014  VOLUME_HOME      0    0     28001
0,-1, 0, 0, 0   ABC015  VOLUME_ABSENT    1    0     29712
0,-1, 0, 0, 0   ABC016  VOLUME_EJECTED   1    0     29712
0,-1, 0, 0, 0   ABC017  VOLUME_ABSENT    1    0     29712
0,-1, 0, 0, 0   ABC018  VOLUME_ABSENT    1    0     29712
1, 0, 0, 0,19   ABC019  VOLUME_HOME      1    0     0
1, 0, 0, 0,20   ABC020  VOLUME_HOME      1    0     0
```

```
0,-1, 0, 0, 0   ABC021  VOLUME_ABSENT    0    0     0
0,-1, 0, 0, 0   ABC022  VOLUME_ABSENT    4    0     0
```

로깅 볼륨 통계 보고서 작성

볼륨 통계 로그 파일(*acsss_stats.log*)을 사용하여 볼륨 이동 통계를 로깅할 수 있습니다. 이러한 통계는 ACSLS에서 볼륨의 위치가 변경되었음을 감지하는 모든 경우에 대한 항목으로 구성됩니다. ACSLS는 입력, 꺼내기, 마운트 및 마운트 해제에 대한 항목 및 감사에서 볼륨의 위치가 변경되었음을(일반적으로 수동으로 이동하여) 감지하는 모든 경우를 로깅합니다.

acsss_config 구성 프로그램을 사용하여 다음을 수행합니다.

- *LIB_VOL_STATS* 변수를 사용하여 볼륨 통계 로깅을 사용하거나 사용 안함으로 설정합니다.
- 볼륨 통계 로그 파일의 최대 크기를 지정합니다.
- 볼륨 통계 로그 파일의 롤오버 파일 수를 지정합니다.

stats_report 유틸리티에서는 *acsss_stats.log*를 사용하여 모든 마운트 및 테이프 드라이브 사용을 보고합니다.

볼륨 통계 로그 파일에는 볼륨 통계 로깅의 사용 여부를 표시하는 수집 모드 항목과 볼륨 통계 항목이 포함됩니다.

볼륨 통계 로그 파일 항목의 예제:

```
2014-06-30 08:53:00 CONFIG
Library volume statistics on.

2014-06-30 09:23:08 EJECT
U01120 Home 0,0,1,3,5 Cap 1,0,0 Client Host Id 129.81.15.25

2014-06-30 10:36:05 ENTER
PB0444 Home 0,0,4,3,5 Cap 0,0,0 Client Host Id 129.81.15.25

2014-06-30 10:42:48 MOUNT
PB0478 Home 0,0,1,35,1 Drive 0,0,1,0 Client Host Id Local

2014-06-30 10:43:19 DISMOUNT
PB0478 Home 0,0,1,35,1 Drive 0,0,1,0 Client Host Id Local

2014-06-30 10:43:19 AUDIT
RB0478 0,0,1,35,1  STATUS_VOLUME_NOT_FOUND Client Host Id JBHUTTO

2014-06-30  10:43:19  AUDIT
PB0444  0,0,1,32,1 STATUS_VOLUME_FOUND Client Host Id JBHUTTO

2014-06-30 10:45:00 CONFIG
Library volume statistics off.

2015-01-16 09:51:07 ACSCR
0A1235 Home 0,0,5,14,14 STATUS_VOLUME_NOT_FOUND Client Host Id Local

2015-01-16 09:40:13 ACSCR
```

```
0A123A Home 0,0,5,14,15 STATUS_VOLUME_FOUND Client Host Id Local
```

위의 예에서 수집 모드 항목은 통계 수집이 2014년 6월 30일 오전 8:53에 시작되어 같은 날 오전 10:45에 종료되었음을 보여줍니다. 이러한 수집 시작 및 종료 시간에는 이 수집 기간 동안의 볼륨 통계 항목이 포함됩니다.

볼륨 통계 항목의 형식은 다음과 같습니다.

- *yyyy-mm-dd hh:mm:ss command*

  vol_id home_loc function_loc client_host_ID

설명:

- *yyyy-mm-dd*는 항목의 연, 월, 일입니다. 4자리 숫자의 연도 형식이 지원됩니다. *acsss_config*를 사용하여 날짜 형식을 지정합니다.
- *hh:mm:ss*는 항목의 시간, 분 및 초입니다.
- *command*는 볼륨을 이동했거나 (감사가) 볼륨이 이동되었음을 감지한 클라이언트 요청 또는 ACSLS 명령입니다.
  - *MOUNT*는 마운트 요청입니다.
  - *DISMOUNT*는 마운트 해제 요청입니다.
  - *ENTER*는 수동 또는 자동 모드 입력 요청입니다.
  - *EJECT*는 꺼내기 요청입니다.
  - *AUDIT*는 감사 요청입니다.
  - *ACSMV*는 이동 요청입니다.
  - *ACSCR*은 볼륨 복구 활동입니다. 이 활동은 ACSLS 처리를 통해 자동으로 작성됩니다.
- *vol_id*는 볼륨 식별자입니다.
- *home_loc*는 볼륨의 홈(스토리지 셀) 위치입니다.
- *function_loc*는 다음과 같이 볼륨을 사용한 요청의 볼륨 위치입니다.
  - *mount* 또는 *dismount* 요청

    위치는 전송 ID입니다.
  - *enter* 또는 *eject* 요청

    위치는 CAP ID입니다.
- *audit* 요청은 감사를 통해 다음 오류 중 하나를 감지했음을 지정합니다.
  - *STATUS_VOLUME_FOUND*

    감사를 통해 데이터베이스에 지정된 위치와 일치하지 않는 위치에서 볼륨을 찾았습니다.
  - *STATUS_VOLUME_NOT_FOUND*

    감사를 통해 데이터베이스에 지정된 위치에서 볼륨을 찾지 못했습니다.
  - *Volume Recovery activity*

위치는 셀 ID 또는 전송 ID일 수 있으며, 볼륨 복구에서 다음 상황 중 하나를 감지했음을 나타냅니다.

› *STATUS_VOLUME_FOUND*

데이터베이스에 기록되지 않은 볼륨을 찾았으므로 추가됩니다.

› *STATUS_VOLUME_NOT_FOUND*

데이터베이스의 볼륨이 기록된 위치에 없으므로 삭제됩니다.

- *client_host_ID*는 다음 중 하나입니다.
  - 클라이언트 응용 프로그램 요청의 경우 호스트 IP 주소입니다.
  - *cmd_proc* 명령의 경우 *cmd_proc*를 시작한 셸 환경에 환경 변수 *LIBVOLSTATS_CMD_PROC_ID*가 설정되어 있으면(ASCII 문자만), 항목은 환경 변수 값의 처음 12자입니다.
  - *cmd_proc* 명령의 경우 환경 변수 *LIBVOLSTATS_CMD_PROC_ID*가 설정되지 않았거나 비ASCII 문자를 포함하면 항목은 로컬입니다.

12

12장. 유틸리티 참조

이 장에서는 다음 ACSLS 유틸리티에 대해 설명합니다.

- “acs_renumber.sh ”를 사용하면 연결된 라이브러리를 재구성하지 않고 라이브러리 컴플렉스에서 지정된 ACS의 식별자를 변경할 수 있습니다.
- “acsss 매크로 ”는 ACSLS를 시작하고 중지할 뿐 아니라 ACSLS의 유지 관리 및 문제 해결을 제어하고 모니터링합니다.
- “bdb.acsss ”는 ACSLS 데이터베이스 및 ACSLS 제어 파일을 백업합니다.
- “동적 구성(config) 유틸리티 ”는 ACSLS가 온라인 상태로 실행되는 동안 ACSLS 라이브러리(및 구성 요소)에 대한 구성 변경을 동적으로 구현합니다. 이러한 구성 유틸리티는 다음과 같습니다.
- “config acs ”는 동적으로 ACS를 추가하거나 기존 ACS 및 그 구성 요소를 재구성합니다.
- *“config drives ”* - 기존 드라이브 패널에서 동적으로 드라이브를 추가하고 드라이브 유형을 변경하며 드라이브를 삭제합니다.
- “config lsm ”은 동적으로 기존 LSM과 해당하는 모든 구성 요소를 재구성합니다. 이러한 구성 요소에는 CAP, 패널 및 드라이브가 포함됩니다.
- “config ports ”는 동적으로 ACS에 대한 포트 연결을 재구성합니다.
- *“db_export.sh ”*는 ACSLS의 업그레이드 설치 또는 다시 설치를 준비하기 위해 ACSLS 데이터베이스 정보 및 ACSLS 제어 파일을 내보냅니다.
- *“db_import.sh ”*는 db_export.sh 유틸리티를 사용했을 때 내보낸 ACSLS 데이터베이스 정보 및 ACSLS 제어 파일을 가져옵니다.
- *“del_vol ”*은 오프라인 LSM에서 볼륨을 삭제합니다.
- “drives_media.sh ”는 현재 ACSLS 릴리스에서 지원되는 모든 드라이브 유형, 매체 유형 및 드라이브와 매체 간 호환성을 표시합니다.
- “ejecting.sh ”는 대규모 꺼내기 작업을 빠르고 효율적으로 수행합니다.
- “free_cells.sh ”를 사용하면 ACSLS에서 제어되는 라이브러리의 사용 가능한 셀을 모니터링 및 관리할 수 있습니다.
- “getHba.sh ”는 광 섬유 채널 HBA 포트를 관리합니다.
- “get_license_info ”는 사용 권한 라이센스를 소프트웨어적으로 강제하는 기능이 ACSLS에서 더는 사용되지 않으므로 릴리스 ACSLS 7.3.1 및 8.0.1부터 제거되었습니다. ACSLS 제어 라이브러리에서 사용 가능한 셀 수를 표시하고 관리하려면 “free_cells.sh ”를 사용하십시오.
- “greplog ”는 *acsss_event* 로그를 필터링하여 특정 키워드를 포함하는 메시지를 포함하거나 제외합니다.

- “install_scsi_Linux.sh ”는 ACSLS에 대한 라이브러리를 구성할 때 사용할 수 있는 */dev/mchanger* 링크를 만듭니다.
- “lib_type.sh ”는 지정된 ACS ID에 연결된 LSM의 LSM 유형을 반환합니다.
- “licensekey.sh ”는 라이센스 키 검증이 더는 사용되지 않으므로 릴리스 ACSLS 7.3.1 및 8.0.1부터 제거되었습니다.
- “moving.sh ”는 여러 카트리지를 하나 이상의 LSM으로 이동합니다.
- “probeFibre.sh ”는 Emulex(LP10000) 또는 QLogic(QLA2300) 광 섬유 채널 HBA 뒤에 연결된 각 장치의 모델 번호, 개정 수준 및 대상 LUN 주소를 표시합니다.
- *“rdb.acsss ”*는 ACSLS 데이터베이스와 ACSLS 제어 파일을 복원합니다.
- *“showDevs.sh ”*는 Solaris에 구성된 각각의 mchanger 장치에 대한 세부정보를 표시합니다.
- “showDrives.sh ”는 ACSLS에 연결되어 구성된 모든 드라이브 목록을 나타냅니다.
- *“stats_report ”*는 라이브러리 볼륨 통계 정보를 수집합니다.
- “userAdmin.sh ”는 ACSLS GUI 사용자 암호를 관리합니다. 사용자 추가, 사용자 제거, 사용자 나열, 사용자 암호 변경을 수행할 수 있습니다.
- *“volrpt ”*는 볼륨 보고서를 만듭니다.
- *“watch_vols ”*는 CAP를 통해 입력되므로 볼륨에 대한 소유권 및 풀 연결을 자동으로 지정합니다.

개요

ACSLS 유틸리티를 사용하려면 다음 일반 지침을 따르십시오.

- 일반적으로 이 장에 설명된 유틸리티는 사용자 *acsss*에 의해 실행됩니다. 유틸리티 실행에 필요한 권한 및 환경 종속성을 상속하려면 사용자 *acsss*로 로그인해야 합니다.

  *su*를 사용하려는 경우 *su - acsss*를 사용해야 합니다.
- 다음 작업 후에 수동으로 데이터베이스를 테이프에 백업하려면 *bdb.acsss*를 사용하는 것이 좋습니다.
  - 라이브러리 하드웨어 구성
  - 데이터베이스 가져오기. 새 ACSLS 버전으로 업그레이드한 후에는 이전 버전으로 만든 데이터베이스 백업을 사용하지 마십시오. 업그레이드한 후에 바로 새 백업을 만드십시오.
  - 모든 데이터베이스 복구
- 정확하고 일관된 데이터베이스를 복구하려면 항상 최신 데이터베이스 백업을 사용하십시오.

유틸리티가 실패하는 경우 모든 이벤트 로그를 보존하십시오. 이러한 로그는 문제 해결을 지원하는 데 도움이 됩니다.

레거시 시작/중지 스크립트

ACSLS 7.x에 사용된 시작 및 중지 스크립트는 ACSLS 8.x에서 지원되지 않습니다.

ACSLS 8.x는 라이브러리 관리 응용 프로그램을 시작 및 중지하기 위해 Solaris SMF(서비스 관리 기능)와 통합된 새 메커니즘을 제공했습니다. 이는 ACSLS에 사용된 *rc.accsss* 및 *kill.acsss*를 대체합니다. 이 메커니즘은 또한 응용 프로그램 상태를 모니터링하는 기능도 제공합니다.

acsss 명령으로 ACSLS 8.x를 시작 및 중지할 수 있습니다. 단일 명령 *acsss*는 ACSLS 시작, 종료 및 모니터링 기능을 제공합니다. 해당 유틸리티는 *$ACS_HOME* 디렉토리에 상주하고 모든 사용자가 액세스할 수 있습니다.

유틸리티 명령

다음 절에서는 ACSLS 유틸리티에 대해 설명합니다.

acs_renumber.sh

연결된 라이브러리를 재구성하지 않고 라이브러리 컴플렉스에서 지정된 ACS의 식별자를 변경할 수 있는 간단한 도구입니다. 라이브러리의 모든 LSM, CAP, 드라이브 및 볼륨이 ACS와 관련하여 식별되므로 사용자가 새로 지정하는 ACS ID와 각 라이브러리 리소스가 일치하도록 이 유틸리티는 다양한 데이터베이스 테이블을 모두 업데이트합니다.

새 논리 라이브러리는 현재 활성 패턴을 사용합니다. 예를 들어, ACS 0을 1로 번호를 다시 지정하면 1001과 1002는 원래대로 남아 있지만, ACS 1의 새 논리 라이브러리는 2001이 됩니다. ACS 6을 ACS 0으로 번호를 다시 지정하면 7001은 원래대로 남아 있지만, ACS 0의 새 논리 라이브러리는 1003이 됩니다. 새로 추가된 논리 라이브러리가 ACS를 기반으로 하여 예측할 수 있지만 실제로 더는 관련이 없습니다.

주:

이 유틸리티에서 변경한 사항은 ACSLS 서버에만 적용되고 이러한 리소스를 사용하는 클라이언트 응용 프로그램에는 적용되지 않습니다. 그러므로 서버에서 ACS ID를 변경한 후에 클라이언트 데이터베이스를 재구성해야 할 수 있습니다.

주:

ACSLS는 이 스크립트를 실행하기 전에 사용 안함으로 설정되어야 합니다.

ACS의 지정된 번호를 변경하려면 *acs_renumber.sh*를 실행하십시오. 대화식 세션에서 먼저 변경한 내용이 클라이언트 응용 프로그램에 영향을 미친다는 경고 메시지가 표시되고 계속 작업을 수행할 것인지 묻는 메시지가 표시됩니다.

```
$ acs_renumber.sh

        N O T I C E

  Changes made by this script will
  impact client applications that
  use ACSLS. Specifically, drive
  i.d. mappings and LSM id's will change.

Continue...? (y or n):
```

*y*라고 응답하면 내용을 변경하기 전에 루틴에서 자동으로 기존 데이터베이스가 백업됩니다. 변경한 내용을 원래대로 되돌려야 할 경우 이 작업을 통해 이전 구성으로 복원할 수 있습니다. (또한, *acs_renumber.sh* 루틴을 반복하여 변경을 되돌릴 수도 있습니다.)

루틴에서 현재 구성된 ACS의 목록이 표시되고 각 ACS의 번호를 다시 지정할 것인지를 묻는 메시지가 표시됩니다. 번호를 다시 지정하는 경우 지정할 새 값을 묻습니다.

```
Current ACS list:

   ACS-0 (SL8500)

Do you wish to renumber ACS-0? (y or n):

What is the new value for ACS-0? 5
```

입력(이 예에서는 5라고 응답했음)을 수락하면 루틴에서 보류 중인 변경을 확인할 것인지를 묻는 메시지가 표시됩니다.

```
Change ACS-0 to ACS-5.

   Correct? (y or n):
```

y라고 응답하면 루틴에서 관련 데이터베이스 테이블이 모두 업데이트되기 시작하고 변경한 사항을 체크포인트하기 위해 자동으로 데이터베이스가 백업됩니다.

```
Updating tables:    Changing ACS-0 to ACS-5

  acstable: 1 records
  captable: 4 records
  celltable: 13424 records
  drivetable: 128 records
  handtable: 16 records
  lmutable: 0 records
  lsmtable: 8 records
  paneltable: 280 records
  porttable: 1 records
  ptptable: 16 records
  scr_distr_table: 0 records
  volumetable: 0 records

Complete!
Current ACS list:

   ACS-5 (SL8500)

Now backing up the database changes...
```

acsss 매크로

acsss 매크로는 ACSLS와 관련된 다양한 서비스를 시작 및 작동 중지하는 기본적인 시작, 중지 및 상태 명령입니다. 설치에 따라 ACSLS 응용 프로그램은 Solaris 또는 Linux 시스템에 설치된 최대 7개의 서비스를 한 단위로 구성됩니다.

- *acsdb* - ACSLS 라이브러리 데이터베이스를 관리합니다.
- *acsls* - 라이브러리 작업을 실행하는 라이브러리 제어 소프트웨어입니다.

- *weblogic* - ACSLS GUI용 웹 서버입니다.
- *surrogate* - java 서비스와 acsls 사이의 통신 링크입니다.
- *rmi-registry* - 명명된 java 객체 및 방법에 대한 조회 서비스입니다.
- *smce* - 논리 라이브러리의 SCSI 매체 교환기 에뮬레이션입니다.
- *stmf* - 논리적 라이브러리의 대상 모드 프레임워크입니다.

acsls 및 *acsdb* 서비스는 모든 설치에 공통으로 적용됩니다. *weblogic*, *surrogate* 및 *rmi-registry* 서비스는 ACSLS GUI 지원이 구성된 위치에 있습니다. *smce* 및 *stmf* 서비스는 논리 라이브러리가 (Solaris에) 구성된 경우에만 적용됩니다.

모든 서비스는 다른 구성 요소 간의 종속성에서 요구하는 정의된 순서대로 이러한 서비스를 시작 및 중지하는 단일 매크로 *acsss*를 사용하여 ACSLS 사용자가 조작합니다. 이 매크로는 Solaris의 SMF(서비스 관리 기능) 및 Linux의 *init.d* 서비스 유틸리티에 명령을 실행하여 실제 작업을 수행합니다.

형식

acsss <명령>

명령 없이 *acsss*를 입력하면 옵션 목록이 표시됩니다.

옵션

명령	기능
enable	ACSLS와 관련된 모든 서비스를 시작하는 기본 방법입니다. 사용으로 설정하면 다양한 서비스가 사용 상태를 유지하고 시스템 재부트 후에도 자동으로 다시 사용으로 설정됩니다.
temp-enable	*acsss enable*과 동일하지만 시스템 재부트 후 서비스가 다시 시작되지 않습니다.
maint-enable	ACSLS 데이터베이스와 관련되지 않은 일반적인 유지 관리 작업에 사용됩니다. 이 옵션은 GUI 기반구조를 사용으로 설정해 주므로, GUI 사용자는 ACSLS가 사용 안함으로 설정되어 있는 동안 로그인 상태를 유지할 수 있습니다. 이 방법은 소프트웨어 부분 패치 설치 컨텍스트에 사용됩니다. acsls 또는 smce 서비스 모두 사용으로 설정되지 않습니다.
db	*db_export*, *db_import* 및 *acsss_config*를 포함하는 데이터베이스 유지 관리 작업에 사용되는 선호 제어 모드입니다. 이를 통해 ACSLS 데이터베이스 엔진이 사용으로 설정되고 ACSLS GUI를 포함하는 모든 기타 ACSLS 서비스가 사용 안함으로 설정됩니다.
disable	ACSLS 작업을 중지하는 데 사용되는 기본 방법입니다. 완전히 종료되는 것은 아니며 acsls 및 smce 서비스가 사용 안함으로 설정된 후 데이터베이스 및 GUI 로그인 세션이 유지 관리 작업에 대해 활성 상태로 유지되도록 합니다. 결과 상태는 *acsss* maint-enable의 상태와 동일합니다. 이 방법은 서비스가 사용 안함으로 설정되기 전에 ACSLS 및 라이브러리가 유휴 상태로 놓이기 때문에 가장 안전하게 서버를 중지할 수 있습니다.
force-disable	*acsss disable*과 동일하지만 작업이 acsls 및 smce를 사용 안함으로 설정하기 전에 유휴 상태가 될 때까지 기다리지 않습니다.
shutdown	모든 ACSLS 서비스를 완전히 종료합니다. 소프트웨어 설치/제거 컨텍스트 및 종료할 데이터베이스(acsdb) 또는 종료할 GUI 기반구조(rmi-registry 및 surrogate)가 필요한 기타 유지 관리 컨텍스트에 사용됩니다.

명령	기능
status	다양한 ACSLS 서비스에 대한 빠른 상태 보고서를 제공합니다.
a-status	*acsdb* 서비스의 작업 상태를 반환합니다.
d-status	*acsls* 서비스의 작업 상태를 반환합니다.
g-status	ACSLS GUI의 상태를 표시합니다.
l-status	다양한 ACSLS 서비스의 상세 정보 상태 요약을 제공하며 문제 해결 컨텍스트에서 추가 분석에 필요한 데이터를 로깅하기 위한 포인터가 포함됩니다. 포인터가 가리키는 로그는 서비스 시작 또는 종료를 실패한 경우와 같은 컨텍스트에서 유용합니다.
p-status	*acsss* 상태와 비슷하며 이 보고서에는 각각의 서비스 계약에서 모니터링하는 다양한 프로세스 ID 목록이 포함됩니다.
w-status	WebLogic 서비스의 상태를 표시합니다.
timeout	Solaris의 *acsls* 서비스에 대한 SMF 시작 *timeout*을 보고합니다.
legal	ACSLS 법적 통지를 영어 또는 프랑스어로 표시합니다.

대부분의 경우 맨 위 *enable*, *disable* 및 *status* 세 개의 명령만 사용합니다. 나머지 명령은 소프트웨어 서비스 제공 컨텍스트에서 편의에 따라 사용됩니다.

bdb.acsss

bdb.acsss 유틸리티는 ACSLS 환경을 재구성해야 하는 ACSLS 데이터베이스 내용 및 ACSLS 제어 파일을 백업합니다. 백업은 사용자가 이름을 지정한 tar 파일 또는 테이프 장치에 배치되거나 기본값으로 정의된 디렉토리에 배치됩니다.

이 유틸리티는 ACSLS를 종료하지 않고 ACSLS 데이터베이스 백업을 수행합니다(핫 백업).

-f 옵션이 없는 */export/backup/<time_stamp>.tar* 파일이 만들어집니다. *time_stamp*는 *bdb.acsss* 명령이 실행된 시간입니다. *bdb.acsss*를 실행한 후에 테이프의 내용을 확인하려면 특정 테이프 장치에 대한 다음 예를 수정하십시오.

Solaris의 경우:

```
tar tvbf 2048 /dev/rmt/0mn
```

tar tvbf 명령을 실행한 후에 테이프가 다음 블록으로 진행합니다. *tar tvbf* 명령 실행 후에 *rdb.acsss*를 실행하려는 경우 테이프를 되감았거나 다시 배치했는지 확인하십시오.

형식

bdb.acsss [*-f backup_file* | *tape_device*]

옵션 없이 *bdb.acsss*를 입력하면 기본 백업이 수행되어 이 백업이 수행된 시점으로 되돌리는 데이터베이스 복원 기능을 제공합니다.

옵션

- *-f backup_file*

ACSLS 데이터베이스 백업을 포함할 UNIX 파일을 지정합니다. 파일 및 디렉토리에 대한 쓰기 권한이 있어야 합니다.

- *-f tape_device*

  ACSLS 서버에 연결되고 구성된 모든 테이프 장치를 지정합니다.

사용법

필요한 경우 데이터베이스 복구에 사용할 수 있는 백업을 만들려면 *bdb.acsss* 유틸리티를 사용하여 ACSLS 데이터베이스를 테이프 또는 외부 네트워크 파일에 백업합니다.

다음 작업 후에 수동으로 데이터베이스를 백업하려면 bdb.acsss를 사용하는 것이 좋습니다.

- *acsss_config* 실행
- 데이터베이스 가져오기. 새 ACSLS 버전으로 업그레이드 후에 이전 버전으로 만든 데이터베이스 백업을 사용하지 마십시오.
- 전체 라이브러리 감사
- 모든 데이터베이스 복구

예 1:

```
$ bdb.acsss -f /export/backup/my_backup
```

이 예에서는 *my_backup*이라는 파일이 */export/backup* 디렉토리에 만들어졌습니다. 이제 파일 보관 여부, 파일 저장 위치 또는 파일을 다른 파일 시스템, 다른 서버 또는 쓰기 가능 CD 장치로 이동할 것인지를 선택할 수 있습니다.

그러면 이 파일을 사용하여 백업이 수행된 시점의 상태로 데이터베이스를 복원할 수 있습니다. 예를 들어, 백업이 금요일 오후 1시에 수행되었고 복원은 월요일 오전 6시에 수행되는 경우 데이터베이스가 금요일 오후 1시의 상태로 되돌려집니다.

이와 동일한 *-f* 옵션 내에서 파일 이름 대신 테이프 장치를 지정할 수 있으면 백업이 명명된 테이프 장치로 이동합니다.

예 2:

```
$bdb.acsss -f /dev/rmt/0mn
```

이 예에서는 테이프 장치 */dev/rmt/0mn*에 테이프 아카이브가 만들어졌습니다. 이 아카이브는 저장하여 오프사이트 위치에서 나중에 사용할 수 있습니다.

참조:

- “rdb.acsss ”
- 10장. 데이터베이스 관리

동적 구성(config) 유틸리티

동적 구성(*config*) 유틸리티를 사용하면 ACSLS가 온라인 상태로 실행되는 동안 ACSLS 라이브러리(및 구성 요소)에 대한 구성 변경을 구현할 수 있습니다. 이러한 구성 변경은 *acsss_config.log* 파일에 기록됩니다.

다음과 같은 동적 구성 유틸리티가 지원됩니다.

- *config acs*
- *config drives*
- *config lsm*
- *config ports*

config 유틸리티를 사용하면 다음과 같은 이점이 있습니다.

- ACSLS가 계속 실행되므로 영향을 받지 않는 라이브러리 구성 요소에 대한 마운트 요청을 수행할 수 있습니다.
- 다른 모든 구성 정보가 변경되지 않은 상태에서 지정된 라이브러리 구성 요소를 다시 구성할 수 있습니다. 예를 들어, 다음과 같습니다.
  - 특정 ACS를 지정할 때는 다른 ACS의 구성 요소가 영향을 받지 않습니다.
  - 특정 LSM을 지정할 때는 다른 LSM의 구성 요소가 영향을 받지 않습니다.
  - 모든 기존 드라이브에 대한 드라이브 패널(패널에 있는 드라이브) 마운트 및 마운트 해제가 영향을 받지 않습니다.

다음 중요 사항을 알아두어야 합니다.

- 동적 *config* 유틸리티를 사용하려면 ACSLS가 실행 중이어야 합니다.
- 초기 ACSLS 구성을 만들려면 *acsss_config*를 사용해야 합니다. 6장. *ACSLS* 동작을 제어하는 변수 설정을 참조하십시오.
- 이벤트 알림에서 모든 동적 구성 변경을 보고합니다.
- 동적 구성을 실행하기 전에 추가 또는 재구성 중인 모든 구성 요소가 준비되어 있어야 합니다.
- *acsss_config.log* 파일은 표시된 메시지에 관한 세부정보를 제공합니다.
- 구성 변경을 확인하지 않은 경우 *[[CTRL]]+C*로 작업을 취소할 수 있습니다.
- 동적 구성은 구성 변경 전후에 자동 백업을 수행합니다.
- 구성 변경을 확인한 후에는 취소할 수 없습니다. 구성 변경을 되돌리려면 ACSLS를 종료하고 구성 변경 수행 바로 전에 만든 백업을 복원하십시오.

  10분 후에 구성 변경 확인이 시간 초과됩니다.
- 유일한 (또는 마지막) ACS를 제거할 수 없습니다.
- ACS의 마지막 CAP 또는 ACSLS에 정의된 마지막 드라이브를 제거하지 마십시오.

동적 구성 제한 사항

동적 구성 유틸리티에는 두 가지 중요한 제한 사항이 있습니다.

- ACS를 삭제하거나 라이브러리에 대한 포트(연결)를 삭제하거나 변경할 수 없습니다.
- 기존 SCSI/광 섬유 연결 라이브러리에서는 *config acs* 및 *config lsm* 유틸리티의 드라이브 구성**만 업데이트**할 수 있습니다. 패널 또는 CAP 구성은 업데이트되지 않습니다. *config drives* 및 *config acs new*는 SCSI/광 섬유 연결 라이브러리와 제한 없이 작동합니다. c*onfig ports* 유틸리티는 SCSI/광 섬유 연결 라이브러리에서 지원되지 않습니다.

해결 방법:

동적 구성을 통해 지원되지 않는 이러한 구성 변경의 경우 ACSLS를 작동 중지하고 *acsss_config*를 사용하십시오.

수행 금지 사항

- 상태 정보는 라이브러리에 대한 광범위한 I/O를 포함하므로 라이브러리와 그 구성 요소에 대한 상태 정보를 표시하는 데 동적 구성을 사용하지 **마십시오**.

  대신 *query* 또는 *display* 명령을 사용하십시오.
- 한 번에 두 개 이상의 구성 작업을 수행하려고 하지 마십시오.

  한 번에 동적 구성 작업을 하나만 수행할 수 있습니다. 이렇게 하면:
  - ACSLS와 구성 중인 라이브러리 간의 I/O로 인한 수행 문제를 최소화합니다.
  - 여러 구성 작업 간의 복잡한 상호 작용을 피합니다.

config acs

config acs 유틸리티를 사용하면 다음을 수행할 수 있습니다.

- ACS를 추가하거나 기존 ACS 및 그 구성 요소를 재구성합니다.
- 모든 ACS 번호를 순서대로 지정하지 않고 라이브러리를 구성하거나 재구성할 수 있습니다.

  예: 9310 라이브러리에서 SL8500으로 마이그레이션한 다음, 9310을 제거하려고 합니다. 9310은 번호가 ACS 0으로 지정되어 있고 SL8500은 번호가 ACS 1로 지정되어 있습니다. *config acs*를 사용하면 이제 모든 카트리지와 드라이브를 SL8500으로 마이그레이션할 수 있고 나중에 SL8500의 번호를 다시 지정하지 않고 9310을 제거할 수 있습니다.
  - ACSLS를 종료하지 않고 *config acs* acs_id new로 SL8500 라이브러리를 추가합니다.
  - 드라이브를 9310에서 SL8500으로 이동하고 *config acs* acs_id로 두 개의 ACS에서 모두 드라이브 구성을 업데이트합니다.
  - 카트리지를 9310에서 제거하고 이를 SL8500에 넣습니다.
  - 끝으로, 정전을 예약하여 ACSLS를 종료하고 *acsss_config*를 사용하여 9310을 구성에서 제거합니다. 9310을 구성하지 마십시오. SL8500은 ACS 1로 지정해야 합니다(기본값 0이 아님).
- 분할된 ACS를 구성합니다.

- *config lsm* 유틸리티를 사용하면 기존 LSM만 재구성할 수 있으므로 LSM을 추가 또는 제거합니다.

각 ACS에 하나 이상의 CAP가 있어야 합니다. 다른 분할 영역과 공유되는 CAP일 수 있습니다. 전체 ACSLS 시스템에 대해 하나 이상의 드라이브가 구성되어 있어야 합니다.

예를 들어, ACSLS에서 4개의 라이브러리를 지원하는 경우 이 중 3개의 라이브러리에는 드라이브가 없을 수 있습니다. 하지만 네번째 라이브러리에는 하나 이상의 드라이브가 포함되어야 합니다.

형식

- 새 ACS를 추가하려면 다음 명령을 입력하십시오.

  config acs new
- 기존 ACS를 재구성하려면 다음 명령을 입력하십시오.

  config acs acs_id

새 ACS 추가

새 ACS를 추가하려면:

1. 다음 명령을 입력합니다.

   config acs new
2. 새 ACS에 대한 ACS 번호를 지정합니다.

   ACSLS에서는 모든 ACS 번호를 순서대로 지정하지 않고 라이브러리를 구성하거나 재구성할 수 있습니다.

   이미 사용된 ACS 번호와 처음 다섯 개의 사용 가능한 ACS 번호가 표시됩니다.

   새 ACS에 대한 ACS 번호를 입력합니다.
3. 분할된 SL8500 또는 SL3000 중에 ACS가 배치될 위치를 선택합니다.
   - *y*라고 입력하면 *ACS*에 대한 분할 영역 *ID*를 묻는 메시지가 표시됩니다. 이 분할 영역 ID는 SLConsole의 분할 영역 ID와 일치해야 합니다.
   - 분할된 라이브러리가 아니거나 SCSI/광 섬유로 연결된 경우 **n**을 입력하십시오.

   ACSLS는 분할된 SCSI/광 섬유 연결 라이브러리(예: SL500)를 지원하지 않습니다. 또한 분할된 SCSI/광 섬유 연결 라이브러리에는 분할 영역 ID가 없습니다.
4. 장치 또는 호스트 ID 다음에 나오는 ACS에 대한 연결 수를 입력합니다.

   최대 15개의 연결이 있을 수 있습니다.

   주:

   모든 포트가 동일한 ACS에 연결되어 있는지 확인하십시오.

새 ACS 구성이 표시됩니다.

5. 새 ACS의 추가 항목을 확인합니다.

   확인 후에 구성 정보가 표시되고 데이터베이스가 업데이트됩니다.

기존 ACS 재구성

ACS를 재구성할 때 가능하면 ACS가 온라인 상태이거나 진단 모드에 있어야 합니다.

ACS를 재구성하려면:

1. 다음 명령을 입력합니다.

   ```
   config acs acs_id
   ```

   이전 구성과 새 구성이 표시됩니다.

2. 새 구성을 확인합니다.

   확인 후에 데이터베이스가 업데이트됩니다.

   구성이 변경되지 않은 경우 확인을 요청하지 않고 구성을 표시한 다음, 유틸리티가 종료됩니다. 예를 들어, 다음과 같습니다.

   - 드라이브 유형 및/또는 일련 번호 변경만
   - LSM 일련 번호 변경
   - 4410과 9310 간의 LSM 유형 변경만
   - 손 개수(예: SL8500 로봇) 변경

   하지만, 이러한 변경이 확인을 요구하는 다른 변경과 함께 발생하면 새 구성을 확인하십시오. 그런 다음 데이터베이스가 업데이트됩니다.

   ACS와 그 구성 요소가 데이터베이스에서 제거됩니다.

config acs 제한 사항

- SCSI 연결 라이브러리의 경우 *config acs*는 드라이브 구성만 업데이트합니다. SCSI 연결 라이브러리는 드라이브 추가, 제거 또는 변경을 인식하는 IPL 상태여야 합니다. 라이브러리가 IPL 상태가 되면 모든 드라이브가 준비된 것입니다.
- SCSI 연결 라이브러리의 경우 *config acs*는 패널 또는 CAP 구성을 업데이트하지 않습니다. 이 구성을 업데이트하려면 ACSLS를 작동 중지하고 *acsss_config*를 사용해야 합니다.

config drives

config drives 유틸리티를 사용하면 기존 드라이브 패널의 모든 드라이브를 재구성할 수 있습니다. 여기에는 드라이브 추가, 기존 드라이브의 드라이브 유형 및 일련 번호 업데이트, 데이터베이스에서 제거된 드라이브 삭제가 포함됩니다.

드라이브 구성에 대한 동적 변경을 위해 *config drives* 유틸리티를 사용합니다. 이 변경 작업에는 기존 드라이브 패널의 드라이브 설치, 대체 또는 제거가 포함됩니다. 예를 들어, 스토리지 셀의 개수 및/또는 위치, CAP의 개수 또는 크기, 스토리지 셀 패널을 드라이브 패널로 대체 등의 테이프 라이브러리 하드웨어 구성에 대한 기타 변경사항은 *config lsm* 또는 *config acs* 유틸리티를 사용하여 수행되어야 합니다.

주:

- 변경된 드라이브 구성이 있는 패널을 포함하는 LSM은 온라인 상태이거나 진단 모드에 있어야 합니다.
- 재구성 중인 드라이브 패널의 모든 드라이브가 준비되어 있어야 합니다.
- SCSI 연결 라이브러리는 드라이브 추가, 제거 또는 변경을 인식하는 IPL 상태여야 합니다. 라이브러리가 IPL 상태가 되면 모든 드라이브가 준비된 것입니다.
- 새 드라이브가 기존 드라이브를 대체하고, LSM을 전환하며, 드라이브 유형이 온라인 상태이거나 실행 중인 경우 *config drives*는 자동으로 드라이브 유형 및 드라이브 일련 번호를 업데이트합니다.

형식

config drive panel_id 또는 *config drives* panel_id

사용법

기존 드라이브 패널의 모든 드라이브를 재구성하려면:

1. 다음 명령을 입력합니다.

   config drive panel_id 또는 *config drives* panel_id

   해당 패널에 대한 이전 드라이브 구성과 새 드라이브 구성이 표시됩니다.
2. 구성 변경을 확인합니다.

   확인 후에 데이터베이스가 업데이트됩니다.
   - 구성이 변경되지 않은 경우 확인을 요청하지 않고 구성을 표시한 다음, 유틸리티가 종료됩니다.
   - 드라이브 유형 또는 일련 번호만 변경된 경우 확인을 요청하지 않고 ACSLS 데이터베이스가 업데이트됩니다.

config lsm

이 유틸리티를 사용하면 기존 LSM과 해당하는 모든 구성 요소를 재구성할 수 있습니다. 이러한 구성 요소에는 CAP, 패널 및 드라이브가 포함됩니다.

ACS에서 LSM을 추가하거나 삭제하려면 *config acs* 유틸리티를 사용해야 합니다.

패널이 변경되는 경우의 절차:

- 제거 또는 변경 중이고 카트리지가 빈 패널이 있는 경우 LSM이 온라인 상태를 유지할 수 있습니다.
- 제거 또는 변경 중이고 카트리지가 포함된 패널이 있는 경우 LSM을 재구성하고 영향받는 패널을 감사할 때까지 영향받는 LSM을 진단으로 전환(*vary*)하는 것이 좋습니다. 이렇게 하지 않을 경우 마운트 및 마운트 해제가 실패할 수 있습니다.
- 패널을 추가하고 이러한 패널에 카트리지를 수동으로 배치한 경우에는 데이터베이스를 조정하도록 감사를 실행하십시오.

형식

config lsm lsm_id

사용법

LSM을 재구성하려면:

1. 다음 명령을 입력합니다.

   config lsm lsm_id

   이전 구성과 새 구성이 표시됩니다.

   패널 옆의 "*y*"는 패널 유형이 변경되었음을 알려줍니다. 자세한 내용은 *acsss_config.log* 파일을 참고하십시오.
2. 새 구성을 확인합니다.

   확인 후에 데이터베이스가 업데이트됩니다.

   구성이 변경되지 않은 경우 확인을 요청하지 않고 구성을 표시한 다음, 유틸리티가 종료됩니다.

   사소한 변경사항은 확인 없이 자동으로 수행됩니다. 예를 들어, 다음과 같습니다.
   - 드라이브 유형 및/또는 일련 번호 변경만
   - LSM 일련 번호 변경
   - 4410과 9310 간의 LSM 유형 변경만
   - 손 개수(예: SL8500 로봇) 변경

   하지만, 이러한 변경이 확인을 요구하는 다른 변경과 함께 발생하면 새 구성을 확인하십시오. 그런 다음 데이터베이스가 업데이트됩니다.

config lsm 제한 사항

- SCSI 연결 라이브러리의 경우 *config lsm*은 드라이브 구성만 업데이트합니다. SCSI 연결 라이브러리는 드라이브 추가, 제거 또는 변경을 인식하는 IPL 상태여야 합니다. 라이브러리가 IPL 상태가 되면 모든 드라이브가 준비된 것입니다.
- 패널 또는 CAP 구성을 업데이트하지 않습니다. 이 구성을 업데이트하려면 ACSLS를 작동 중지하고 *acsss_config*를 사용하십시오.

config ports

config ports 유틸리티를 사용하면 동적으로 ACS에 포트 연결을 추가할 수 있습니다.

주:

모든 새 포트는 기존 포트와 같은 ACS에 연결되어야 합니다.

config acs acs_id를 실행한 다음, *config ports* acs_id를 실행하십시오.

ACS를 다른 ACS로 대체하거나 포트 연결 주소를 변경하려면 ACSLS를 작동 중지하고 *acsss_config*를 사용하십시오.

형식

config ports acs_id 또는 config port acs_id

사용법

포트를 추가하려면:

1. 다음 명령을 입력합니다.

   config port acs_id 또는 config ports acs_id

   지정된 ACS에 대한 현재 포트 연결이 표시됩니다.
2. 지정된 ACS에 대한 포트 연결 수를 입력합니다.

   최대 15개의 연결이 있을 수 있습니다.
3. 장치 또는 호스트 ID를 지정합니다.

   주:

   새 포트가 기존 포트와 같은 ACS에 연결되어 있는지 확인하십시오.

   이전 구성과 새 구성이 표시됩니다.

   포트 순서를 변경하는 것은 구성 변경이 아닙니다. 현재 데이터베이스에 기록된 순서대로 연결이 표시됩니다.
4. 새 구성을 확인합니다.

   확인 후에 데이터베이스가 업데이트됩니다.

   구성이 변경되지 않은 경우 구성을 표시하고 유틸리티가 종료됩니다.

config ports 제한 사항

config ports acs_id 유틸리티는 SCSI/광 섬유 연결 라이브러리에서 지원되지 않습니다.

config ports 유틸리티는 라이브러리에 대한 포트(연결)를 삭제하거나 변경합니다. ACSLS를 작동 중지하고 *acsss_config*를 사용해야 합니다.

db_export.sh

db_export.sh 유틸리티는 ACSLS의 업그레이드 설치 또는 다시 설치를 준비하기 위해 ACSLS 데이터베이스 테이블 데이터 및 ACSLS 제어 파일을 내보냅니다.

주:

db_export.sh 는 ACSLS가 실행 중인 경우 실행할 수 없습니다. *db_export.sh*를 실행하기 전에 *acsss disable*을 실행하십시오.

형식

db_export.sh -f [db_file | tape_device *]*

옵션

- *-f* db_file

  ACSLS 데이터베이스의 백업을 포함할 UNIX 파일을 지정합니다. 파일 및 디렉토리 모두에 대한 쓰기 권한.

주:

- 데이터베이스를 파일로 내보내는 경우 해당 파일은 비휘발성 디렉토리에 상주해야 합니다. ACSLS를 다시 설치하려고 다시 설치를 수행하면 *$ACS_HOME* 또는 *$ACSDB_BKUP (such as /export/backup)* 디렉토리가 삭제됩니다. 내보낸 파일을 파일 시스템의 다른 위치에 배치하십시오.
- 경로를 지정하지 않고 파일 이름을 지정하면 *db_export*는 현재 작업 중인 디렉토리의 해당 파일 이름 아래에 데이터베이스 파일을 저장합니다. ACSLS 제어 파일은 같은 디렉토리의 <*filename*>*.misc*라는 파일에 저장됩니다.
- 테이프 라이브러리, 테이프 드라이브 또는 카트리지 매체 형식 중 일부를 지원하지 않는 이전 ACSLS 릴리스로 데이터베이스를 내보낼 경우, 데이터베이스를 내보내기 전에 지원되지 않는 테이프 라이브러리를 구성에서 제거하고, 테이프 드라이브와 카트리지를 사용 중인 라이브러리에서 제거하십시오.
- *-f* tape_device

  ACSLS 서버에 연결되고 구성된 모든 테이프 장치를 지정합니다.

옵션을 지정하지 않으면 테이프 장치가 기본값으로 지정됩니다.

사용법

db_export.sh 유틸리티를 사용하여 ACSLS의 다시 설치 또는 ACSLS의 업그레이드 설치를 준비하십시오.

참조:

- “db_import.sh ”
- “rdb.acsss ”
- “데이터베이스 내보내기 ”
- “새 플랫폼으로 광 섬유 mchanger 마이그레이션 ”

db_import.sh

db_import.sh 유틸리티는 *db_export.sh* 유틸리티를 사용했을 때 내보낸 ACSLS 데이터베이스 테이블 데이터 및 ACSLS 제어 파일을 가져옵니다. ACSLS 제어 파일은 *$ACS_HOME/data/external*에 있고 ACSLS에 대해 사용자 정의 가능한 변수 및 구성으로 구성됩니다. 이러한 제어 파일은 액세스 제어 설정, 스크래치 환경 설정, Extended Store LSM, 사용자 정의 *volrpt* 설정, 볼륨 속성(watch_vols 유틸리티용) 등을 지정합니다. *db_import.sh* 유틸리티는 또한 다른 운영체제로 이동하거나 이전 릴리스에서 이동할 때 재해 복구 기능을 제공하고 사용자 정의 동적 변수를 유지합니다.

형식

db_import.sh -f [db_file | tape_device *]*

옵션

- *-f* db_file

  *db_export.sh*로 만든 UNIX 파일을 지정합니다.
- *-f* tape_device

  ACSLS 서버에 연결되고 구성된 모든 테이프 장치를 지정합니다.

옵션을 지정하지 않으면 테이프 장치가 기본값으로 지정됩니다.

사용법

db_import.sh 유틸리티를 사용하여 *db_export.sh* 유틸리티를 통해 내보낸 ACSLS 데이터베이스를 가져오십시오.

주:

db_import 유틸리티는 ACSLS가 실행 중인 경우 실행되지 않습니다. *db_import.sh*를 실행하기 전에 *acsss disable*을 실행하십시오.

참조:

- “db_export.sh ”
- “rdb.acsss ”
- “데이터베이스 내보내기 ”
- “새 플랫폼으로 광 섬유 mchanger 마이그레이션 ”

del_vol

del_vol 유틸리티는 라이브러리에서 볼륨을 찾습니다. 볼륨을 찾을 수 없는 경우 *del_vol*은 요청에 따라 볼륨을 누락 또는 없음으로 표시하거나 데이터베이스에서 해당 볼륨을 삭제합니다.

볼륨에 대해 참조된 모든 위치를 확인할 수 없고(예: LSM이 오프라인 상태이거나 드라이브가 준비되지 않았음) *-n*(*no_confirm_flag*)가 켜져 있지 않으면 볼륨을 없음으로 표시하거나 삭제를 확인하라는 메시지가 나타납니다. 볼륨이 삭제되면 관련된 볼륨과 정보(예: 스크래치 풀 구성원과 현재 및 보류 중인 lock)가 데이터베이스에서 제거됩니다.

-q(*quiet_flag*)를 입력하지 않으면 볼륨 관련 정보가 표시됩니다. 여러 옵션을 사용하는 경우 구분된 옵션 또는 연속된 문자열로 옵션의 형식을 지정할 수 있습니다.

볼륨을 없음으로 표시하거나 삭제하기 전에 *del_vol*은 *-n* 옵션이 지정되지 않으면 확인을 요청합니다.

- 볼륨이 라이브러리에 있으면 데이터베이스에서 활성 볼륨 상태가 됩니다.
- 볼륨을 라이브러리에서 찾을 수 없고 볼륨을 삭제할 것으로 지정하지 않으면 볼륨이 없음으로 표시됩니다.
- 볼륨을 찾을 수 있는 셀 또는 드라이브에 액세스할 수 없고(라이브러리 또는 드라이브가 오프라인이거나 작동하지 않기 때문에) 볼륨을 삭제할 것으로 지정하지 않으면 볼륨이 누락으로 표시됩니다.

주:

- 이 유틸리티는 라이브러리에서 찾은 볼륨을 삭제하지 않습니다.
- 없음 또는 꺼냄 상태가 만료될 때까지 기다리지 않고 *del_vol* 유틸리티를 사용하여 볼륨 레코드를 제거할 수 있습니다.
- *del_vol* 유틸리티를 지원하려면 ACSLS가 실행 중이어야(사용으로 설정되어야) 합니다.

형식

del_vol [-n] [-d] [-f] [-q] vol_id

옵션

- *-n*

  확인 안함 모드. 사용자에게 확인을 요청하지 않고 찾을 수 없는 볼륨을 없음으로 표시하거나 삭제합니다.

- *-q*

  자동 모드. 데이터베이스에서 추출된 모든 정보를 출력하지 않습니다.

- *-d*

  없음 또는 꺼냄 상태의 만료까지 기다리지 않고 지정된 볼륨을 삭제합니다.

- *-f*

테이프 드라이브에 끼어 있는 카트리지를 다시 넣을 수 있습니다. *-f* 옵션으로 볼륨이 실제로 라이브러리에 있는지 확인하지 않고 볼륨을 삭제하거나 없음으로 표시할 수 있습니다. 이렇게 하면 결함이 있는 테이프 드라이브에 위치한 볼륨을 데이터베이스에서 삭제할 수 있습니다. 이 옵션을 사용하여 드라이브에서 볼륨을 제거하고 데이터베이스에서 삭제한 다음, 드라이브를 복구하는 동안 정상 라이브러리에 사용할 볼륨을 다시 넣을 수 있습니다.

- *vol_id*

  삭제할 볼륨 일련 번호입니다.

  주:

  볼륨 일련 번호에 달러($) 기호가 포함되어 있을 경우 작은따옴표로 묶어주십시오. 예: del_vol 'AB $001'

사용법

*del_vol*을 사용하여 오프라인 LSM에서 카트리지를 제거한 다음, 자동으로 마운트될 수 있도록 이 카트리지를 온라인 LSM에 다시 넣을 수 있습니다.

- 카트리지를 오프라인 LSM에서 제거합니다.
- *del_vol*을 사용하여 카트리지를 없음으로 표시합니다.
- 카트리지를 온라인 LSM에 넣습니다.

이 유틸리티를 사용하려면 ACSLS와 데이터베이스가 실행 중(유휴 아님)이어야 합니다.

주:

실수로 카트리지를 데이터베이스에서 삭제하는 경우 삭제된 카트리지를 데이터베이스에 다시 넣으려면 이 카트리지의 홈 셀을 포함하는 하위 패널을 감사해야 합니다. 시스템이 복구 중인 동안 *del_vol*을 실행하면 예측 불가능한 결과가 발생할 수 있습니다. 또한, LSM 온라인 전환(*vary LSM online*) 중에는 복구 시퀀스가 발생합니다.

예

카트리지 정보를 출력하지 않고 카트리지 *U01102*를 삭제하려면:

```
del_vol -q U01102
```

drives_media.sh

이 루틴은 현재 ACSLS 릴리스에서 지원되는 모든 드라이브 유형, 매체 유형 및 드라이브와 매체 간 호환성을 표시합니다. 정보는 일반적으로 표준 출력에 표시됩니다.

지원되는 현재 라이브러리 목록, 드라이브 유형, 매체 유형 및 드라이브와 매체 간 호환성에 대한 자세한 내용은 *ACSLS Product Information Guide*를 참조하십시오.

형식

```
drives_media.sh [ -f, -h ]
```

옵션

- *-f*

  정보는 다음 세 가지 파일에 작성됩니다.

  - */tmp/drive_types.txt*
  - */tmp/media_types.txt*
  - */tmp/media_compatibility.txt.*

- *-h*

  구문 메시지를 표시합니다.

ejecting.sh

ejecting.sh 유틸리티를 사용하면 대량 꺼내기 원격 저장 작업을 손쉽게 수행할 수 있습니다. 이 유틸리티는 지정된 CAP 및 볼륨 목록을 통해 전체 작업이 완료될 때까지 여러 꺼내기 작업을 병렬로 실행합니다. 사용자가 정렬 순서대로 볼륨을 꺼내도록 요청하지 않으면 이 유틸리티는 가장 가까운 지정된 CAP로 각 볼륨을 꺼냅니다.

작업에서 운영자의 불필요한 CAP 조작을 줄일 수 있는 경우 가능하다면 가장 가까운 CAP 대신 빈 공간이 있는 근처 CAP가 사용됩니다. 일반적인 접근 방법은 카트리지 이동을 줄이고, 불필요한 LSM 전달 마이그레이션을 없애며, 운영자의 전반적인 작업량을 줄입니다.

모든 꺼내기 작업은 작업이 시작된 단일 셸 창에서 모니터링되고 요약됩니다. 여러 XTERM 창과 함께 사용하려면 *-x* 옵션을 참조하십시오. 특정 CAP가 가득 차고 운영자가 *CAP*에서 카트리지를 제거할 준비가 될 때마다 사용자에게 알려줍니다. 운영자는 전체 작업이 완료되면 알림을 받습니다.

지난 10일 동안의 모든 꺼내기 작업 결과를 보여주는 로그 세트는 *$ACS_HOME/log/ejectingLogs* 디렉토리에 보관됩니다. 각 개별 로그는 작업이 완료되었을 때 받은 시간 기록으로 식별됩니다.

형식

표준: *ejecting.sh [-dmox] -c <CAP list> -v <volume list file>*

정책 사양: *ejecting.sh [dmox] -p <policy file>*

레거시 형식: *ejecting.sh <CAP ID> <volume list file>*

옵션

- *-c <CAP list>*

  CAP ID 목록은 명령줄에서 공백으로 구분됩니다. CAP 목록의 모든 CAP는 같은 ACS에 있어야 합니다.

  예: *-c 0,1,0 0,1,1 0,5,0 0,5,1 0,9,0*

별표(*)를 사용하는 와일드카드 표현식은 LSM에서 모든 LSM 또는 모든 CAP, 아니면 두 가지를 모두 지정하는 데 유효합니다.

예:

- *-c 0,1,** (LSM-1에서 모든 CAP)
- *-c 0,*,0* (모든 LSM에서 CAP-0)
- *-c 0,*,** (모든 LSM에서 모든 CAP)

온라인 상태의 사용 가능한 CAP만 옵션으로 선택됩니다. 와일드카드 표현식이 지정되면 우선 순위가 0이 아닌 CAP만 선택됩니다.

ACS는 숫자 표현식이어야 하고 와일드카드에서 참조할 수 없습니다.

더 큰 라이브러리 컨텍스트에서 CAP 선택에 따라 원격 저장 작업의 속도 및 효율성과 관련하여 큰 차이가 나타날 수 있습니다. CAP를 너무 많이 지정하면 운영자가 부분적으로 채워진 다중 CAP를 제공하는 불필요한 작업이 추가될 수 있습니다. 주어진 작업 부하에 비해 너무 적은 CAP를 지정하면 병목 현상을 초래할 수 있고 이로 인해 로봇 기술 대기 시간이 늘어날 수 있습니다. 대규모 볼륨 목록에 대한 일반적인 경험 규칙에 따르면 선택한 CAP는 여러 레일 및 여러 라이브러리 모듈의 라이브러리 컴플렉스 전반에 분산되어야 합니다. 대량 볼륨에 대해 적은 수의 CAP를 선택하는 경우 라이브러리를 지역으로 분할하고 각 지역 중심에 위치한 CAP를 선택하십시오.

- *-v <volume file>*

  단순 텍스트 파일에 대한 전체 또는 상대 경로 이름을 사용하는 파일 사양입니다. 파일에는 꺼낼 볼륨을 나타내는 VOL-ID(VOLSERS)의 목록을 포함해야 합니다. 지정된 CAP와 같은 ACS에 포함된 볼륨만 영향을 받습니다. 마운트된 볼륨은 배출되지 않습니다.

- *-p <policy file>*

  정책 파일은 전체 또는 상대 경로 이름을 사용하는 사양입니다. 이 텍스트 파일에는 CAP 및 볼륨에 대해 정의된 정책이 포함됩니다. 파일 형식에는 작업에 사용할 CAP 목록 다음에 ”caps:” 단어를 포함하고, 볼륨 목록 파일의 전체 경로 이름 다음에 "vols:" 단어를 포함합니다.

  예:

  ```
  caps: 0,1,0 0,1,1 0,5,0 0,5,1 0,9,0 0,9,1

  vols: /export/backup/volumes_to_eject.txt
  ```

- *-d*

  display 옵션은 *eject* 작업 실행 전에 *ejecting.sh*가 볼륨과 CAP 간 지정 사항을 표시하도록 지시합니다. 볼륨과 볼륨이 이동하는 CAP의 전체 목록을 표시하거나 각 LSM에서 각 CAP로 마이그레이션하는 볼륨 수를 보여주는 요약 정보를 간단히 표시하도록 선택할 수 있습니다.

  디스플레이를 표시하기 전에 작업을 계속하거나 중단하도록 선택할 수 있습니다.

- *-m*

  "04"와 "99" 사이의 운영자 메시지 코드가 있는 작업에 레이블을 지정합니다. 지원되는 라이브러리에서 이 숫자 코드는 운영자 콘솔에 표시됩니다.

- *-o*

  원격 저장 작업에서는 볼륨이 정렬된 순서대로 누적되어야 할 시간이 있습니다. 이 옵션을 사용하면 루틴이 볼륨 목록에서 찾은 순서 및 지정된 CAP 순서에 따라 지정된 볼륨을 나열된 CAP로 이동합니다. 볼륨을 모두 꺼낼 때까지 CAP 순서는 첫번째 CAP에서 마지막까지 반복됩니다.

  주:

  CAP 순서에 대한 볼륨이 우선하므로 이 옵션은 LSM 전달 루트를 제한하여 볼륨 이동을 최적화하려고 시도하지 않습니다.

- -x

  각 별도의 CAP *eject* 페이로드에 대한 전용 XTERM 창을 사용하십시오. 이 옵션은 전체 mass-eject 작업 도중에 개별 꺼내기 작업을 추적하는 데 유용할 수 있습니다. XTERM 세션은 각 CAP *eject*가 시작될 때 팝업되고, 해당 *eject*가 완료되면 사라집니다.

  Windows 터미널에서 작동할 때 X11 지원 소프트웨어가 설치되어 있는지 확인하십시오. X11은 Solaris 또는 Linux의 표준입니다. 로컬 시스템에서 ACSLS 서버에 대한 디스플레이 액세스 컨트롤을 열어야 합니다.

  예: *xhost + <acsls_server_hostname>*

  이 유틸리티는 DISPLAY를 보낼 위치를 결정하기 위해 로그인 ID(*who am i*)를 확인합니다. 디스플레이를 보려면 로컬 콘솔 또는 데스크탑 시스템에서 ACSLS 서버로 직접 로그인해야 합니다.

레거시 형식

ejecting.sh <CAP ID> <volume file>

이 유틸리티의 레거시 양식이 보존되어 있습니다. 이 양식은 단순한 볼륨 목록을 포함하는 파일에 대해 단일 CAP ID와 경로 이름을 사용합니다. 또한, CAP의 크기에 최적화된 일련의 *eject* 명령을 구성하고 표준 출력에 대한 결과 명령을 표시합니다. 디스플레이는 지정된 CAP를 통해 전체 목록을 꺼내는 데 필요한 수만큼의 *eject* 명령을 포함합니다.

운영자는 작업을 실행하기 위해 레거시 *ejecting.sh*의 출력을 직접 *cmd_proc*에 파이프할 수 있습니다.

예:

ejecting.sh 0,1,0 /export/backup/myVolumeList | cmd_proc -lq

또는, *cmd_proc*로 나중에 재지정할 수 있는 파일로 출력을 재지정할 수 있습니다.

예:

ejecting.sh 0,1,0 /export/backup/myVolList > /tmp/eject.dat cmd_proc -lq < /tmp/eject.dat

ejecting.sh Logs

ejecting.sh의 모든 인스턴스가 *$ACS_HOME/log/ejectingLogs/* 디렉토리에 로깅됩니다. 각 로그 파일은 날짜/시간 기록으로 명명됩니다. 예:

```
ejecting.log.14-Oct_13:13:10
```

각 *ejecting.log*는 운영자 셸에서 표시될 때 전체 꺼내기 작업을 요약합니다. 발생한 모든 오류가 여기에 표시됩니다.

*ejecting.log*에는 다음 이유로 유틸리티에서 무시한 볼륨 목록을 포함할 수 있습니다.

- 볼륨 ID가 유효하지 않습니다.
- 볼륨이 지정된 ACS에 포함되어 있지 않습니다.
- 볼륨이 사용 중입니다.

ejectingLogs/ 디렉토리의 누적 파일은 10일 후에 제거됩니다. 10일 보다 오래된 로그는 *ejecting.sh*가 각각 새로 호출되어 제거됩니다.

free_cells.sh

free_cells.sh 유틸리티를 사용하면 ACSLS에서 관리되는 라이브러리의 사용 가능한 셀을 모니터링 및 관리할 수 있습니다. 이 유틸리티는 LSM, ACS 및 ACSLS 서버에 대한 사용 가능한 셀 수를 보고합니다.

이 유틸리티는 *$ACS_HOME/utils* 디렉토리에 있습니다.

형식

free_cells.sh

옵션

- *-a*

  각 ACS 및 LSM의 사용 가능한 셀, 할당된 셀 및 총 셀 수와 ACSLS 서버에서 관리되는 사용 가능한 셀, 할당된 셀 및 총 셀 수를 표시합니다.

예

각 LSM의 사용 가능한 셀

- LSM은 L5500이 아님(총 사용 가능한 셀 수만 나열)
  - 선택한 옵션 없음

LSM 1,3
총 사용 가능한 셀 수 = 2,345

- *-a* 옵션(또한, 할당된 셀 및 총 셀 수 나열)

LSM 1,3
총 사용 가능한 셀 수 = 3,345
할당된 셀 = 3,155
총 셀 수 = 6,500

- LSM이 L5500임(LTO 및 비LTO 사용 가능한 셀을 별도로 나열)
  - 선택한 옵션 없음

LSM 0,2
LTO 사용 가능한 셀 = 573
비LTO 사용 가능한 셀 = 467
총 사용 가능한 셀 수 = 1,040

  - *-a* 옵션(또한, 할당된 셀 및 총 셀 수 나열)

LSM 0,2
LTO 사용 가능한 셀 = 573
비LTO 사용 가능한 셀 = 467
총 사용 가능한 셀 수 = 1,040
할당된 셀 = 4,460
총 셀 수 = 5,500

각 ACS의 사용 가능한 셀

ACS에 LSM이 하나만 있는 경우 ACS만 다음 예에 표시된 대로 나열됩니다.

- ACS가 L5500을 포함하지 않음
  - 선택한 옵션 없음(총 사용 가능한 셀 수만 표시)

ACS 1
총 사용 가능한 셀 수 = 5,342

  - *-a* 옵션(또한, 할당된 셀 및 총 셀 수 나열)

ACS 1
총 사용 가능한 셀 = 5,342
할당된 셀 = 5,658
총 셀 수 = 11,000

- ACS가 L5500을 포함(LTO 및 비LTO 사용 가능한 셀을 별도로 나열)
  - 선택한 옵션 없음

ACS 0
LTO 사용 가능한 셀 = 1,573
비LTO 사용 가능한 셀 = 968

총 사용 가능한 셀 수 = 2,541

- *-a* 옵션(또한, 할당된 셀 및 총 셀 수 나열)

  ACS 0
  LTO 사용 가능한 셀 = 1, 573
  비LTO 사용 가능한 셀 = 968
  총 사용 가능한 셀 수 = 2,541
  할당된 셀 = 2,959
  총 셀 수 = 5,500

ACSLS 서버에서 관리되는 사용 가능한 셀

- ACS가 L5500을 포함하지 않음

  총 사용 가능한 셀 수만 표시되는 경우

  - 선택한 옵션 없음

    ACSLS 서버
    총 사용 가능한 셀 수 = 7,883

  - *-a* 옵션

    ACSLS 서버
    총 사용 가능한 셀 수 = 7,883
    할당된 셀 =14,117
    총 셀 수 =22,000

- ACS가 L5500을 포함

  L5500 라이브러리가 관리되고 있는 경우 LTO 사용 가능한 셀 및 총 사용 가능한 셀 수만 표시됩니다. L5500 라이브러리가 비LTO 셀에서 LTO를 분리하는 유일한 라이브러리이므로 비LTO 사용 가능한 셀이 표시되지 않습니다.

  - 선택한 옵션 없음

    ACSLS 서버
    LTO 사용 가능한 셀 = 1,573
    총 사용 가능한 셀 수 = 7,883

  - *-a* 옵션

    ACSLS 서버
    LTO 사용 가능한 셀 = 1,573
    총 사용 가능한 셀 수 = 7,883
    할당된 셀 =14,117
    총 셀 수 =22,000

getHba.sh

getHba.sh 유틸리티는 광 섬유 채널 HBA 포트를 관리합니다.

형식

getHba.sh

getHba.sh 유틸리티는 설치 시 실행되며, *install.sh*에서 호출하는 *install_acsss.sh*에서 호출합니다. 이 유틸리티는 새 HBA가 시스템에 추가되었거나 HBA 포트가 재배열될 때마다 바로 실행됩니다. 이 유틸리티는 ACSLS 논리 라이브러리에 대한 클라이언트 액세스 지점을 나타내기 위해 개시자에서 대상 모드로 변경할 적절한 HBA 포트를 식별합니다.

이 유틸리티를 사용하는 가장 좋은 방법은 *getHba.sh*가 실행되기 전에 FC 연결을 설정하는 것입니다. 이렇게 하면 *getHba.sh*를 통해 기존 연결에 대한 유용한 정보를 표시할 수 있습니다.

유틸리티에서는 먼저 대상 모드 어댑터가 이미 구성되어 있는지 평가합니다. 대상 포트가 없으면 유틸리티 플로우는 아래 설명된 대로 계속됩니다. 유틸리티가 기존 대상 포트를 감지하면 다음 선택 항목 메뉴를 표시합니다.

원하는 작업을 선택하십시오.

1. 기존 HBA 포트 구성을 유지합니다.
2. 추가 대상 모드 포트를 구성합니다.
3. 기존 대상 포트를 개시자 모드로 복원합니다.

옵션	설명
1	이 유틸리티를 종료합니다.
2	유틸리티는 현재 개시자 모드에서 작업 중인 포트를 나열합니다. 포트에 "원격 HBA에 연결되었습니다."라고 표시되면 다른 끝에 개시자가 있으며, 이 개시자는 ACSLS 대상 포트가 될 잠재 후보가 됨을 의미합니다. 포트에 "대상 장치에 연결되었습니다."라고 표시되면 테이프 라이브러리 또는 디스크가 연결되어 있을 수 있으며, 포트는 대상 모드 작업에 대해 잘못된 선택이 됩니다.
3	대상 모드 작업에 대해 구성된 각 포트를 식별하고 해당 포트를 개시자 모드로 복원할 것인지 확인하는 프롬프트가 나타납니다.

옵션 2의 예

대상 모드로 변경할 로컬 HBA 포트를 선택합니다. 다음 목록에서 선택하십시오.

1. HBA 포트 WWN 2100001b32055d85가 연결되어 있지 않습니다.
2. HBA 포트 WWN 2101001b32255d85가 원격 HBA에 연결되었습니다.
3. HBA 포트 WWN 2102001b32055d85가 대상 장치에 연결되었습니다.
4. 이 항목에 없음.

주:

논리 라이브러리 기능을 사용하지 않으려는 경우 "이 항목에 없음"을 선택하십시오.

선택 후에 확인을 요청합니다.

```
2
HBA Port WWN 2101001b32055d85 /pci@0,0/pci10de,377@f/pci1077,143@0
Is this correct? (y or n):
```

원하는 작업을 변경할 기회가 있습니다. "n"이라고 응답하면 사용 가능한 포트 목록이 선택 프롬프트와 함께 다시 표시됩니다. ”y”라고 응답하면 재구성할 추가 포트가 있는지 물어봅니다.

유틸리티는 대상 그룹과 대상 그룹 멤버 추가 작업을 진행하고, 대상 모드 변경사항이 적용되도록 서버를 재부트하도록 지시합니다.

대상 그룹 만드는 중: 2101001b32255d85

옵션 3의 예

이 옵션을 사용하면 기존 대상의 구성을 해제하고 고유 모드로 HBA를 개시자로서 복원할 수 있습니다.

```
# cd $ACS_HOME/install
# ./getHba.sh
A Target-mode port has already been configured:
      Target: wwn.2100001B32050A28
          Connected to ...
               Initiator: wwn.210100E08BA61A29
      Please select a desired action:
      1) Keep the HBA port configuration as it is.
      2) Configure an additional target-mode port.
      3) Restore a target port to initiator mode.
      3
      Target: wwn.2100001b32050a28
Do you wish to restore this port to initiator mode? (y or n): y
Removing 'qlt' binding in /etc/driver_aliases
Are there additional ports you wish to reconfigure? (y or n): n
A reboot will be necessary for these changes to take effect.
```

get_license_info

이 유틸리티는 사용 권한 라이센스를 소프트웨어적으로 강제하는 기능이 ACSLS에서 더는 사용되지 않으므로 릴리스 ACSLS 7.3.1 및 8.0.1부터 제거되었습니다. ACSLS 제어 라이브러리에서 사용 가능한 셀 수를 표시하고 관리하려면 “free_cells.sh ”를 사용하십시오.

greplog

greplog 유틸리티를 사용하여 특정 키워드가 포함된 메시지를 포함하거나 제외하도록 *acsss_event* 로그를 필터링할 수 있습니다. 이 루틴의 구문은 UNIX '*grep*' 기능과 유사합니다. *greplog*는 *acsss_event.log*와 함께 사용하도록 특별히 설계되었지만, 레코드가 빈 행으로 구분된 메시지 파일 유형과 함께 작동할 수 있습니다.

형식

greplog -[v|i] <keyword> <logfile>

옵션

- *-v*

  선택사항입니다. 키워드가 포함된 메시지를 제외한 모든 메시지를 로그에 표시합니다.
- *-i*

  선택사항입니다. 지정된 키워드의 대소문자를 무시합니다.
- *-keyword*

  키워드를 포함하는 여러 행의 전체 메시지를 반환합니다.
- *-logfile*

  로그 파일 목록입니다.

사용법

유틸리티가 로그 파일에 대해 특별히 설계되었으므로 *greplog*는 키워드를 포함하는 단일 행이 아닌 해당 단어가 포함된 여러 행의 전체 메시지를 반환합니다. *-i* 옵션을 사용하면 *greplog*는 지정된 키워드의 대소문자를 무시합니다. *-v* 옵션을 사용하면 *greplog*는 키워드를 포함하는 메시지를 제외한 모든 메시지를 로그에 표시합니다. *greplog*는 *acsss_event.log*와 함께 사용하도록 특별히 설계되었지만, 레코드가 빈 행으로 구분된 메시지 파일 유형과 함께 작동할 수 있습니다.

install_scsi_Linux.sh

i*nstall_scsi_Linux.sh* 유틸리티는 ACSLS에 대한 라이브러리를 구성할 때 사용할 수 있는 */dev/mchanger** 링크를 만듭니다. 그러한 mchanger 이름은 이제 라이브러리에서 보고된 일련 번호를 사용하여 구성되며, SAN 패브릭 또는 서버 재부트(모두 다 라이브러리에 대한 기본 장치 경로를 변경할 수 있음)의 변경사항을 유지하는 안정적인 식별자를 ACSLS에 제공합니다.

결과 */dev/mchanger* 링크 및 관련 라이브러리에 대한 정보가 *showDevs.sh* 유틸리티를 사용하는 스크립트 출력의 일부로 표시됩니다. 이 유틸리티는 또한 라이브러리 정보를 표시하기 위해 (mchanger 링크를 만든 후) 독립형 작업으로 실행될 수 있습니다.

형식

i*nstall_scsi_Linux.sh*

샘플 출력:

```
================================================================================
# install/install_scsi_Linux.sh
[root@acslsdevx1 install]# ./install_scsi_Linux.sh
Installing SCSI device(s) for Oracle StorageTek ACSLS.
Adding ACSLS rules for udev ...
Starting udev:                                             [  OK  ]
Successfully built the following...
   /dev/mchanger-3500104f00079f9d2: STK SL500 V-1485 336-cells 10-drives
```

```
   /dev/mchanger-3500104f0007a8532: STK SL500 V-1485 205-cells 6-drives
   /dev/mchanger-3500104f000cc6a67: STK SL150 V-0182 59-cells 4-drives
Installation of SCSI device(s) successfully completed.
#
================================================================================
# utils/showDevs.sh
   /dev/mchanger-3500104f00079f9d2: STK SL500 V-1485 336-cells 10-drives
   /dev/mchanger-3500104f0007a8532: STK SL500 V-1485 205-cells 6-drives
     /dev/mchanger-3500104f000cc6a67: STK SL150 V-0182 59-cells 4-drive
#
================================================================================
```

lib_type.sh

이 루틴은 지정된 ACS ID에 연결된 LSM의 LSM 유형을 반환합니다. 일반 유형의 여러 LSM이 구성에 있으면 여러 LSM에 대해 단일 유형만 반환됩니다.

형식

lib_type.sh <ACS ID>

licensekey.sh

라이센스 키 검증이 더는 사용되지 않으므로 릴리스 ACSLS 7.3.1 및 8.0.1부터 제거되었습니다.

moving.sh

*moving.sh utility*는 여러 카트리지를 하나 이상의 LSM으로 이동합니다. 이 유틸리티는 이동할 카트리지를 나열하는 파일을 읽습니다. 다음 카트리지일 수 있습니다.

- 하나 이상의 LSM에 있는 카트리지
  - 동일한 LSM 또는 다른 LSM의 다른 패널로 이동 중인 패널의 카트리지
  - 선택한 카트리지 그룹

*moving.sh*의 제한 사항은 다음과 같습니다.

- *vol_list_file*의 모든 대상 LSM과 카트리지가 동일한 ACS에 있어야 합니다.
- 대상 LSM이 오프라인이거나 사용 가능한 셀이 포함되지 않은 경우 카트리지가 해당 LSM으로 이동하지 않습니다.

주:

- ACSLS가 실행 중인 경우에만 *moving.sh* 유틸리티가 실행됩니다.
- 내부적으로 *moving.sh*는 라이브러리 성능(마운트 및 마운트 해제)에 영향을 주지 않도록 한 번에 하나의 카트리지만 이동합니다.
- 별도의 볼륨 목록을 만든 후 병렬로 여러 이동 유틸리티를 실행할 수 있습니다. 다음을 확인합니다.
  - 대상 LSM이 동일합니다. LSM에 사용 가능한 셀이 모든 카트리지를 수용하기에 충분한지 확인하십시오.

◦ 하나의 SL8500 라이브러리 내에서 이동하는지 확인합니다. 엘리베이터가 두 대만 있으므로 한 번에 두 개보다 많은 이동 유틸리티를 실행하면 성능이 저하됩니다.

형식

moving.sh -f vol_list_file -t lsm_id or list of lsm_ids

설명:

- *-f vol_list_file*

  이동할 볼륨 목록을 포함하는 파일의 이름입니다.

  주:

  볼륨 ID는 다음 규칙을 따라야 합니다. 행별 하나의 카트리지 ID. vol_ids는 유효한 ACSLS 볼륨 ID여야 합니다. vol_ids에 선행 또는 후행 공백이 포함되는 경우 작은따옴표 또는 큰따옴표로 묶어야 합니다.

- *-t lsm_id* s

  카트리지를 이동할 하나 이상의 LSM ID를 지정합니다. 각 LSM ID는 공백으로 구분되어야 하고 동일한 ACS에 속해야 합니다.

사용법

moving.sh 유틸리티를 사용하여 카트리지 목록을 다른 LSM으로 이동하거나 동일한 LSM에서 패널을 다른 패널로 이동하십시오.

사용자 정의 볼륨 보고서 또는 *display volume* 명령을 사용하여 LSM에서 이동할 볼륨 목록을 포함하는 파일을 만들 수 있습니다.

다음과 같은 경우에 *moving.sh* 유틸리티를 사용합니다.

- SL8500은 초기에 분할되거나 다시 분할되고, 하나 이상의 레일(LSM)이 기존 분할 영역(ACS)에서 제거되는 경우 *moving.sh*는 분할 영역에서 제거 중인 LSM에서 분할 영역에 남아 있는 LSM으로 카트리지를 이동할 수 있습니다.
- LSM이 ACS에서 제거되면 *moving.sh*는 ACS에 남아 있는 LSM으로 카트리지를 이동할 수 있습니다.

  예를 들어, SL8500이 라이브러리 컴플렉스(ACS)에서 제거되면, *moving.sh*는 제거 중인 SL8500에서 라이브러리에 남아 있게 될 LSM으로 카트리지를 이동합니다. 이 작업은 또한 9310 LSM이 9310의 ACS에서 제거되면 적용됩니다.
- 스토리지 확장 모듈이 SL8500에서 제거되면, 제거 중인 패널에서 라이브러리에 남아 있는 패널로 카트리지를 이동할 수 있습니다.
- 라이브러리 성능을 최적화하려면 카트리지 아카이브에 사용되는 드라이브가 없거나 적은 LSM으로 비활성 카트리지를 이동하십시오. 이렇게 하여 활성인 새 카트리지용 드라이브가 있는 LSM의 공간을 확보합니다.

볼륨 목록 파일 만들기

시작하기 전에 LSM에서 이동할 볼륨 목록을 포함하는 파일을 만들어야 합니다. `volrpt`(사용자 정의 볼륨 보고서) 또는 *`display volume`* 명령을 사용할 수 있습니다.

- *`vol_list_file`* 만들기

  `volrpt -d -f custom_volrpt_input -l  lsm_id > vol_list_file`

  `custom_volrpt_input` 파일의 위치:

  `VOLUME_ID  6`

  샘플 출력:

```
$ volrpt -d -f my_custom -l 0,2 > my_file_list
$ cat my_file_list
ABC744
ABC748
ABC756
ACS151
EN0823
O00373
```

- *`display volume`* 명령을 사용하여 `vol_list_file`을 만들기
  a. 볼륨 목록을 표시합니다.

     예:

     `display volume * -home acs,lsm,panel,*,* -f` `vol_id`

     이 예에서는 *`-home`* 매개변수로 식별된 패널의 모든 볼륨을 선택합니다. 행과 열이 와일드카드로 지정됩니다. `vol_id`만 출력됩니다.

     샘플 출력:

```
ACSSA> display volume * -home 0,3,5,*,* -f vol_id
2007-02-12 15:31:45              Display Volume
Vol_id
PG0350
PG0353
PG0356
PG0358
PQ0616
```

  b. *`vol_list_file`*을 만들고 이름을 지정합니다.
  c. 볼륨 목록(display 명령으로 만들어짐)을 잘라내어 이 파일에 붙여넣습니다.

d. 출력을 편집합니다.

*vol_list_file*에는 빈 행과 선행 공백을 포함할 수 없습니다. 다음 vi 명령을 사용하여 이를 제거하십시오.

```
:%s/^[ ]*//g
```

이 작업을 수행하지 않으면 다음 예와 같은 오류 메시지가 표시됩니다.

```
$ moving.sh -f my_file_list -t 0,2
  Error in file my_file_list.
  Invalid entry
 ABC748
 ABC756
ACS151
EN0823
```

이 오류 메시지는 볼륨 ABC748과 756 앞에 추가 공백이 있었기 때문에 생성되었습니다.

카트리지 그룹 이동을 위한 절차

다음 절차에서는 아래 작업 방법에 대해 설명합니다.

- ACS에서 LSM을 제거하기 전에 카트리지 이동
- 패널을 변경하거나 제거하기 전에 카트리지 이동

ACS에서 LSM을 제거하기 전에 카트리지 이동

라이브러리를 재구성하거나 다시 분할한 후 LSM을 ACS에서 제거하면 해당 LSM의 모든 카트리지에 액세스할 수 없게 됩니다. 그러므로 LSM을 제거하기 전에 해당하는 모든 카트리지를 ACS에 남아 있을 LSM으로 이동해야 합니다. 다음 절차를 따르십시오.

- 레일(LSM)이 분할된 SL8500의 레거시 분할 영역에서 제거될 때.
- LSM이 ACS에서 제거될 때. ACS가 9310 또는 SL8500 라이브러리를 포함할 수 있습니다.

1. 새 구성을 계획합니다.
   - 성능을 위해 카트리지 및 드라이브를 구성합니다.
   - 라이브러리 구성을 변경하기 바로 전에 LSM을 비웁니다.
   - 비우고 있는 LSM에 포함할 카트리지 수, 카트리지를 이동하고 있는 LSM의 사용 가능한 셀 수를 결정합니다.

     *free_cells.sh -a*를 사용하여 이러한 LSM(할당된 셀) 및 사용 가능한 셀의 카트리지 수를 알아봅니다.
2. 이동 및 재구성을 예약합니다.

- 시스템에 미치는 영향을 최소화하도록 이동을 예약합니다.

  카트리지 이동 작업에는 시간이 걸리고 라이브러리를 재구성하거나 SL8500을 다시 분할하는 작업은 중단되기 쉽습니다.
- 카트리지를 이동할 대상 LSM에 사용 가능한 셀이 충분한지 확인하십시오. 필요한 경우 공간을 확보하기 위해 카트리지를 꺼내십시오.

3. 제거 중인 LSM의 모든 드라이브를 오프라인으로 전환(*Vary*)합니다.

   이를 통해 다음을 방지합니다.
   - LSM에서 로봇에 대한 경합.
   - 이 LSM에 마운트.

     그렇지 않으면, 이 LSM에 마운트된 카트리지가 LSM의 새 홈 셀에 Float할 수 있으므로 비우려는 LSM을 채울 수 있습니다.
4. 비우는 중인 LSM을 진단 모드로 전환(*Vary*)하여 다음 명령을 통해 *cmd_proc*에 대한 액세스만 제한합니다.

   ```
   vary lsm lsm_id diagnostic
   ```

   예: *vary lsm 0,1 diagnostic*
5. 사용자 정의 *volrpt*를 실행하여 다음 명령을 통해 비우는 중인 LSM의 모든 카트리지를 파일로 출력합니다.

   ```
   volrpt -f custom_volrpt_input -l from_lsm_id > move_vols_list
   ```

   *custom-volrpt_input*의 위치:

   ```
   VOLUME_ID   6
   ```

   예: *volrpt -f volrpt_input -l 0,1 > move_vols_list*

   자세한 내용은 “로깅 볼륨 통계 보고서 작성”을 참조하십시오.
6. 다음 명령을 사용하여 카트리지를 비우는 중인 LSM 외부로 이동합니다.

   ```
   moving.sh -f move_vols_list -t dest_lsm_id(s)
   ```
7. 카트리지를 LSM에 넣었거나 카트리지가 LSM에 "Float"될 수 있으므로 *volrpt*를 사용하여 LSM이 비었는지 확인합니다.

   ```
   volrpt -l from_lsm_id
   ```

   비어 있지 않은 경우 사용자 정의 *volrpt*를 다시 실행하여 현재 LSM에 있는 볼륨을 선택합니다. 그런 다음 *moving.sh*를 다시 실행합니다(3단계 및 4단계).

   주:

   원래 볼륨 목록으로 *moving.sh*를 다시 실행**하지 마십시오**.
8. 비우는 중인 LSM을 오프라인으로 전환(*vary*)하여 볼륨이 이동하지 않도록 합니다.

```
vary lsm lsm_id offline
```

주:

분할 영역 및/또는 ACS에서 LSM을 제거하십시오.

9. `config acs acs_id` 또는 acsss_config를 사용하여 ACS를 재구성합니다.

분할 영역에서 셀을 제거하기 전에 카트리지 이동

주:

SL3000은 드라이브와 셀 레벨로 분할할 수 있고 SL8500은 고급 분할을 사용하여 드라이브와 셀 어레이 레벨로 분할할 수 있습니다. 셀이 하나의 분할 영역에서 다른 분할 영역으로 다시 지정된 경우, 이러한 셀에 있는 카트리지가 고립되며, 이전에 있던 분할 영역에서 더 이상 액세스할 수 없게 됩니다. 다른 분할 영역을 관리하는 호스트가 이 카트리지에 있는 데이터를 겹쳐 쓸 수 있습니다.

분할 영역 경계 변경 시 카트리지가 고아가 되지 않게 하려면: 라이브러리를 다시 분할하기 전에 분할 영역에 남게 될 셀로 카트리지를 이동합니다. SL3000은 단일 LSM이므로 기존 ACSLS move 명령이 작동하지 않습니다. 라이브러리 이외의 다른 곳에 카트리지를 이동합니다. 분할 영역에서 제거될 다른 셀에 카트리지를 이동할 수도 있습니다.

다음 방법 중 하나를 사용하여 카트리지를 이동합니다.

- SLConsole(StorageTek Library Console)을 사용합니다.

  라이브러리를 감사하여 볼륨의 위치를 감사합니다.

  자세한 정보와 절차를 보려면 *SL8500* 또는 *SL3000* 사용 설명서를 참조하십시오.
- 다음 ACSLS 절차를 따릅니다.

1. “volrpt ” 또는 “display 명령 옵션 사용 ”을 이용하여 볼륨 위치를 표시합니다.
2. 다음 display 명령을 사용하여 특정 패널에서 사용 가능한 (빈) 셀의 목록을 표시합니다.

   ```
   display cell a,l,p,*,* -status empty -f status
   ```

   자세한 내용은 “display 명령 옵션 사용 ”을 참조하십시오.
3. LSM ID 대신 사용 가능한 셀을 지정하여 카트리지를 특정 셀로 이동합니다. 셀 이동을 위해 move 명령을 사용합니다.

   ```
   move AAAAAA a,l,p,r,c
   ```

패널을 변경하거나 제거하기 전에 카트리지 이동

9310에서 셀 패널을 드라이브 패널로 변경하거나 SL8500에서 스토리지 확장 모듈을 제거하기 전에 카트리지를 이동해야 합니다.

ACS에서 LSM을 제거하기 전에 카트리지 이동 과 같은 1~4단계 절차를 따릅니다.

5단계: 비우는 중인 패널에서 카트리지를 선택하고 파일로 출력합니다.

a. 사용자 정의 *volrpt*를 실행하여 비우는 중인 LSM의 모든 카트리지를 파일로 출력합니다. 패널 번호(홈 셀 ID)를 포함합니다.

   volrpt -f custom_volrpt_input *-l from_lsm_id > move_vols_list_1*

   *custom-volrpt_input*의 위치:

   ```
   VOLUME_ID  6
   CELL_ID       14
   ```

   비우는 중인 패널에서 볼륨을 선택하고 이러한 vol_ids를 *move_vols_list_2*로 출력합니다.

b. *display volume* 명령을 사용하여 비우는 중인 패널에서 카트리지를 선택합니다.

   *display volume * -home acs,lsm,panel,*,* -f volume* > move_vols_list_2

   여기서는 *-home* 매개변수로 식별된 패널에서 모든 볼륨을 선택합니다. 행과 열이 와일드카드로 지정됩니다. vol_id만 출력되고 출력 내용이 파일에 기록됩니다.

   앞의 공백과 뒤의 빈 행을 제거하여 출력을 편집합니다.

   주:

   대상 즉, "끝" LSM이 소스 즉, "시작" LSM과 같고, 둘 이상의 패널을 비우는 중인 경우 일부 볼륨이 비우는 중인 패널로 다시 이동합니다. 해당 패널을 지우려면 패널에서 멀리 있는 볼륨을 선택하고 볼륨을 반복해서 이동해야 합니다.

ACS에서 LSM을 제거하기 전에 카트리지 이동 과 같은 6~9단계 절차를 따릅니다.

10단계. *config lsm* lsm_id 또는 acsss_config를 사용하여 LSM을 재구성합니다.

예

- 카트리지를 LSM 0,4에서 LSM 0,0 및 0,1로 이동

  카트리지를 LSM 0,4에서 LSM 0,0 및 0,1로 이동하려면 volrpt를 사용하여 LSM 0,4의 카트리지 목록을 포함하는 파일을 만든 다음, 아래와 같이 *moving.sh* 유틸리티를 실행합니다.

  샘플 출력:

  ```
  $ moving.sh -f vol_list.txt -t 0,0 0,1
  Number of free cells in LSM 0,0 : 308
  Number of free cells in LSM 0,1 : 362
  ----------------------------------------
  Total number of free cells : 670
  Total number of volumes to move : 7
  ```

```
Cartridge CAB001 moved to 0,0,3,0,0
Cartridge CAB002 moved to 0,0,4,0,0
Cartridge CAB003 moved to 0,0,5,0,0
Cartridge CAB004 moved to 0,0,6,0,0
Cartridge CAB005 moved to 0,0,7,0,0
Cartridge CAB006 moved to 0,0,8,0,0
Cartridge CAB007 moved to 0,0,9,0,0

Summary
=======
Number of free cells remaining in LSM 0,0 : 301
Number of free cells remaining in LSM 0,1 : 362
------------------------------------------------
Total number of free cells remaining : 663
Number of cartridges moved : 7
Number of cartridges not moved : 0
```

- LSMs 0,4 0,5 0,6 및 0,7에서 LSMs 0,0 0,1 0,2 및 0,3으로 카트리지 이동

  각 LSM을 인접한 LSM으로 이동하여 성능을 최적화하려면:
  - *volrpt*를 사용하여 LSM 0,4 0,5 0,6 및 0,7의 카트리지 목록을 포함하는 파일을 준비합니다.
  - 네 개의 *moving.sh* 유틸리티를 별도의 UNIX 명령 터미널에서 동시에 실행합니다.

    사용된 소스 및 대상 LSM과 전달 포트가 모든 면에서 다르므로 *moving.sh*의 개별 인스턴스 간의 경합이 없습니다.

    샘플 출력:

```
moving.sh -f vol_list_0-4.txt -t 0,0
moving.sh -f vol_list_0-5.txt -t 0,1
moving.sh -f vol_list_0-6.txt -t 0,2
moving.sh -f vol_list_0-7.txt -t 0,3
```

성능을 위한 카트리지 관리

moving.sh 유틸리티는 비활성 카트리지를 아카이브 LSM으로 이동하는 데 사용할 수 있습니다. 아카이브 LSM은 마운트될 가능성이 적은 카트리지를 저장하는 드라이브가 적거나 없는 LSM입니다. SL8500의 상단 레일은 CAP에 대한 직접적인 액세스가 없으므로 아카이브 LSM에 적합합니다.

라이브러리에 있지 않아도 되는 비활성 카트리지는 배출할 수 있지만, 자동 마운트에 사용해야 할 비활성 카트리지는 아카이브 LSM으로 이동해야 합니다.

비활성 카트리지를 아카이브 LSM으로 이동하려면 다음 절차를 완료합니다.

1. 비활성 카트리지를 식별합니다. 예를 들어, 지난 3개월 동안 액세스한 적이 없는 카트리지를 선택하려면:
2. 사용자 정의 volrpt를 실행하여 검사 중인 LSM의 모든 카트리지를 출력하고 결과를 파일로 출력합니다.

   ```
   volrpt -f custom_volrpt_input -l from_lsm_id > move_vols_list_1
   ```

   custom-volrpt_input의 위치:

   ```
   VOLUME_ID 6

   ACCESS_DATE 15
   ```
3. *access_date*가 3개월보다 이전인 카트리지를 선택하고 이동할 볼륨 목록과 함께 이러한 vol_ids를 파일로 출력합니다.
4. 비활성 카트리지를 아카이브 LSM으로 이동합니다.

   ```
   moving.sh move_vols_list_2 archival_lsm_id
   ```

참조:

- “display 명령 옵션 사용 ”
- “volrpt ”

probeFibre.sh

이 유틸리티는 현재 광 섬유 채널 HBA 뒤에 직접 연결 또는 SAN 연결된 모든 라이브러리를 표시합니다.

probeFibre.sh 유틸리티는 각 광 섬유 연결 라이브러리의 모델 번호, LUN ID 및 WWPN(World Wide Port Name)을 표시합니다. *probeFibre.sh* 유틸리티는 mchanger 장치가 각 라이브러리에 대해 만들어지기 전에도 실행할 수 있습니다.

이 유틸리티에는 루트 액세스가 필요합니다.

형식

```
probeFibre.sh [-v] [-p]
```

옵션

인수 없음.

각 라이브러리 장치에 대한 공급업체, 모델, LUN ID 및 WWPN을 표시합니다.

- *-v*

  각 포트에서 감지된 라이브러리 장치(WWNN 포함)와 함께 HBA(호스트 버스 어댑터)의 모델 번호 및 각 개시자 포트의 WWPN을 포함한 구조화된 출력을 생성합니다.

- *-p*

  각 필드가 콜론으로 구분된 *vendor:model:version:driver:target:lun:wwpn*을 포함하는 출력을 생성합니다.

rdb.acsss

rdb.acsss 유틸리티는 자동 백업 기능 또는 *bdb.acsss* 유틸리티로 만든 백업을 사용하여 ACSLS 데이터베이스 및 ACSLS 제어 파일을 복원합니다. ACSLS 제어 파일은 *$ACS_HOME/data*에 있으며 ACSLS에 대해 서로 다른 여러 환경 변수를 정의합니다. 이러한 파일은 액세스 제어 설정, 스크래치 환경 설정, Extended Store LSM, 사용자 정의 *volrpt* 설정, 볼륨 속성(*watch_vols* 유틸리티의 경우) 등을 지정합니다.

테이프 백업에서 복원 중인 경우 ACSLS 데이터베이스 및 제어 파일을 테이프에서 복원하기 전에 테이프 장치를 되감거나 배치해야 합니다. *rdb.acsss*를 실행하기 전에 다음 명령 중 하나를 사용하여 테이프를 되감거나 백업 파일이 상주한 정확한 위치에 테이프를 배치하십시오.

```
mt -f /dev/rmt/0mn rewind
mt -f /dev/rmt/0mn nbsf 1
```

형식

rdb.acsss

메뉴 옵션

*rdb.acsss*를 실행하면 메뉴에 아래의 예와 같이 네 개의 옵션이 표시됩니다.

```
Please enter the number followed by Return for your choice from
the following menu.
Press? followed by the Return key for help.
   1: Restore from a list of current local disk backup files
   2: Restore from a previous tape or file backup
   3: Restore database only (do not include ACSLS control files)
   4: Restore only ACSLS non-database control files
   E: Exit
```

1. 현재 로컬 디스크 백업에서 복원

   현재 로컬 디스크에 있는 모든 ACSLS 백업 파일이 나열됩니다.

   설명: 데이터베이스가 백업으로 복원됩니다. ACSLS 제어 파일은 백업에서만 복원됩니다. 백업이 기본 백업 디렉토리(*$ACSDB_BKUP*)에 저장됩니다. 데이터베이스가 나열 및 선택된 데이터베이스 백업으로 복원됩니다. 일반적으로 8개의 다른 날짜가 나열되어 있지만, *csss_config*에 설정된 데이터베이스 보존 기간에 따라 달라집니다.

   사용법: 이 옵션을 사용하여 손상된 데이터베이스를 복원하십시오. 이 옵션을 통해 모든 백업이 표시되고 표시된 데이터베이스 백업으로 복원할 수 있습니다.

   예:

```
Menu choice: 1
rcvr_previous.sh 2642:  ACSLS database recovery started.
You have taken backups on the following days. Please enter the corresponding date
 and time to the backup that you wish to recover from. ACSLS database and control
 files will be restored to that time.
2011-10-02 04:38:48
2011-10-03 00:00:01
2011-10-04 00:00:01
2011-10-05 00:00:01
2011-10-05 11:49:06
Please enter the recovery date and time (YYYY-MM-DD HH:MM:SS):
HINT: You may copy and paste to enter the date and time.
```

관련 백업에서 원하는 날짜 및 시간을 입력해야만 해당 데이터베이스가 원하는 시점으로 복원됩니다.

2. 이전 테이프 또는 파일 백업에서 복원

설명: 이 옵션을 선택하여 다른 파일 시스템(예: NFS) 또는 백업 장치(예: 테이프)에 복사된 데이터베이스를 복구합니다. ACSLS 컨트롤 파일이 복원됩니다.

사용법: 데이터베이스를 서버 또는 전혀 다른 서버에라도 복원해야 하는 심각한 이벤트(예: 하드웨어 장애)에 사용하십시오. 플랫폼(OS 버전/업데이트 및 ACSLS 릴리스/PUT 레벨)은 동일해야 합니다.

```
Option 2:
Menu choice: 2.
rcvr_manual.sh 2635: ACSLS recovery started
To recover the ACSLS environment either:
- Mount a ACSLS backup tape in a tape device and
  specify this tape device with '-f tape_device', or
- Specify a file name containing a ACSLS backup with '-f   backup_file'.
```

ACSLS 데이터베이스가 지정된 파일에서 복구됩니다.

-f [backup_file | tape_device]를 입력하십시오.

예 1: *-f* backup_file을 사용하여 파일 지정

테이프*(*사용된 경우*)*를 마운트하고 백업 소스를 입력하십시오*. -f /export/backup/*my_backup.bak

이렇게 하면 *my_backup.bak*라는 백업을 복원합니다. 데이터베이스와 ACSLS 제어 파일이 복원되고 ACSLS는 해당 백업이 실행된 시점의 상태로 되돌아갑니다.

예 2: 테이프 장치에 만든 백업 복원

테이프 장치에 만든 백업을 복원하는 경우 동일한 옵션을 사용하지만 조금 다르게 작동합니다. 백업이 테이프 장치에 만들어지면 tar 아카이브가 만들어지지만 이름은 지정되지 않습니다. 테이프에서 백업을 복원할 때 테이프 장치만 지정됩니다.

힌트: 테이프 장치를 **되감지 않음**을 사용해야 합니다.

테이프*(*사용된 경우*)*를 마운트하고 백업 소스를 입력하십시오*. -f /dev/rmt/0mn*

이 명령을 통해 */dev/rmt/0mn* 장치로 이동하고 유효한 데이터베이스 백업이 있는지 확인합니다. 데이터베이스 백업이 있고 유효하면 이 백업이 복원됩니다.

테이프 되감기 절차:

*rdb.acsss*를 시도하기 전에 테이프를 되감거나 백업 파일이 상주하는 올바른 위치에 테이프가 배치되어 있어야 합니다.

주:

tar tvbf 명령을 실행한 후에 테이프가 다음 블록으로 진행합니다. *tar tvbf* 명령 실행 후에 *rdb.acsss*를 실행하려는 경우 테이프를 되감았거나 다시 배치했는지 확인하십시오.

a. 다음 명령을 사용하여 테이프를 되감거나 배치할 수 있습니다.

   mt -f /dev/rmt/0mn rewind or mt -f /dev/rmt/0mn nbsf 1 ---> SOLARIS

b. *bdb.acsss* 후에 테이프의 내용을 확인하려면 다음 명령을 사용합니다.

   tar tvbf 2048 /dev/rmt/0mn ---> SOLARIS

3. 데이터베이스만 복원*(ACSLS* 제어 파일을 포함하지 않음*)*

설명: 이 옵션은 데이터만 복원하는 기능을 제공합니다. 일부 환경에서는 데이터를 포함하는 ACSLS 데이터베이스를 복원해야 할 수 있지만, ACSLS 데이터베이스가 아닌 제어 파일을 복원하지 않아도 됩니다.

```
Option 3:
Menu choice: 3
To recover the ACSLS database data only, either:
- Mount an ACSLS backup tape in a tape device and specify this tape device with '-
f tape_device', or
- Specify a file name containing an ACSLS backup with '-f backup_file'.
The ACSLS database data will be recovered from the file specified.
   ****This option does not include the ACSLS control files****
Please enter -f [ backup_file | tape_device ]:
```

4. *ACSLS* 데이터베이스가 아닌 제어 파일만 복원

설명: ACSLS 제어 파일만 복원합니다. *$ACS_HOME/data/internal* 디렉토리에 있는 파일을 복원하기 전에 백업이 기존 파일로 만들어지고 끝에 "*.bak*" 확장자가 추가됩니다.

```
$ACS_HOME/data/internal/dynamic_variables/dv_config.dat.bak
$ACS_HOME/data/internal/dynamic_variables/dv_trace.dat
$ACS_HOME/data/internal/release.vars.bak
```

*$ACS_HOME/data/external*에 있는 파일의 경우에는 해당하지 않습니다. 복구 전에 ACSLS 제어 파일의 백업이 수행되지 않습니다.

```
Option 4:
Menu choice: 4
To recover the ACSLS non-database control files either:
- Mount an ACSLS backup tape in a tape device and
 specify this tape device with '-f tape_device', or
- Specify a file name containing an ACSLS backup with '-f backup_file'.
ACSLS non-database control files will be recovered from the file specified.
Please enter -f [ backup_file | tape_device ]:
```

예:

```
Please enter -f [ backup_file | tape_device ]: -f $ACSDB_BKUP/my_file.bak
```

- *-f $ACSDB_BKUP/my_file.bak*은 지정된 파일에서 ACSLS 제어 파일을 복구합니다.
- *-f /dev/rmt/0mn*은 지정된 테이프 장치에서 ACSLS 제어 파일을 복구합니다.

5. 종료

   rdb.acsss 유틸리티를 종료하면 기본 디렉토리 *$ACSDB_BKUP*에 대한 백업이 시작됩니다.

참조:

다음에 대한 복구 절차:

- “데이터베이스 내보내기 ”
- “데이터베이스 내보내기 ”
- “bdb.acsss ”

showDevs.sh

showDevs.sh 유틸리티는 */dev* 디렉토리의 각 mchanger 인스턴스와 연관된 중요한 장치 속성을 표시합니다. 중요한 속성에는 라이브러리 모델 번호와 개정 수준, 셀 용량과 연결된 장치 수가 포함됩니다. 아래 옵션을 사용하여 추가 속성을 표시할 수 있습니다.

형식

showDevs.sh [-w][-s]

옵션

이 유틸리티는 몇 가지 옵션과 함께 실행할 수 있습니다.

인수 없음.

이 옵션은 각 mchanger, 라이브러리 모델 및 코드 레벨과 셀 및 장치 수를 표시합니다.

- *-w*

  World Wide Name - 기본 정보와 함께 연결된 라이브러리의 WWPN을 표시합니다.
- *-s*

  일련 번호 - 기본 정보와 함께 라이브러리 일련 번호를 표시합니다.

  주:

  서버 측 HBA 정보(HBA 포트의 WWPN 포함) 및 연결된 모든 라이브러리의 WWPN을 표시하려면 사용자 루트로서 *probeFibre.sh* 유틸리티를 사용하십시오.

showDrives.sh

이 유틸리티는 ACSLS에 연결되고 구성된 모든 드라이브 목록을 나타냅니다. 간단한 드라이브 위치 목록이 드라이브 유형별로 정렬됩니다. 상세 정보 표시(*-v*) 옵션이 사용되면 유틸리티에서 드라이브 상태 및 각 드라이브의 지정된 논리 상태가 요약되어 표시됩니다.

형식

showDrives.sh [-v]

stats_report

stats_report 유틸리티는 라이브러리 볼륨 통계 보고서를 생성합니다. 이 유틸리티를 실행하려면 *acsss*로 로그인해야 합니다.

형식

*stats_report [*vol_statsX.log ...*]*

설명:

vol_statsX.log -

1. 이 선택적 인수를 사용하여 하나 이상의 아카이브된 볼륨 통계 로그 파일 이름을 지정할 수 있습니다.

   (아카이브된 파일에 *vol_statsX.log*(0 <= X <= 8인 경우) 형식이 있습니다.)

다음과 같이 입력된 하나의 아카이브된 파일 사용

```
$stats_report  vol_stats0.log
```

시간 중심 보고서와 드라이브 중심 보고서가 생성되고 사용자 입력 파일 이름이 보고서 파일 이름으로 보류됩니다(앞에서 표시됨).

예를 들어, *vol_stats0.log*를 지정하면 보고서가 *$ACS_HOME/log* 디렉토리에 다음과 같이 생성됩니다.

vol_stats0_drive_centric.txt 및 *vol_stats0_time_centric.txt*

2. 한 번에 모든 아카이브된 볼륨 통계 파일에 대한 보고서를 생성하려면 아래 절차를 따릅니다.
   a. 개별 파일에서 전체 로그를 생성합니다.

```
$cd $ACS_HOME/log
$cat vol_stats8.log .... vol_stats0.log
acsss_stats.log  >  vol_statsXXXX.log
   where vol_statsXXXX.log
```

   (문자열 *vol_stats*가 필요하지만, XXXX는 FULL 등과 같은 것이 될 수 있음) 모든 vol_statsX.log(0 <= X <= 8인 경우) 및 acsss_stats.log의 역순으로 연결된 파일입니다.

   b. *stats_report*를 실행합니다.

```
$stats_report vol_statsXXXX.log
```

   보고서가 *vol_statsXXXX_drive_centric.txt* 및 *vol_statsXXXX_time_centric.txt*로 생성됩니다.

   파일 이름을 인수로 지정하지 않으면 시간 중심 보고서와 드라이브 중심 보고서가 *$ACS_HOME/log/acsss_stats.log*에서 생성됩니다.

사용법

- *stats_report*는 현재 *acsss_stats.log*를 사용하여 볼륨 통계의 두 가지 보고서를 준비합니다. 변수 *LIB_VOL_STATS*를 설정하여 수집하는 라이브러리 볼륨 통계를 사용으로 설정하십시오. 이 작업은 *acsss_config*(옵션 3) 프로세스 또는 명령줄 명령 *dv_config -p LIB_VOL_STATS*를 통해 수행될 수 있습니다. 그러면 로그가 기본 크기 500KB에 도달할 때 *ACSLS*에서는 자동으로 9개의 *acsss_stats.log* 파일을 롤링 및 유지 관리합니다.
- 보존할 로그 파일 크기 및 파일 수는 변수 *LIB_STATS_FILE_NUM* 및 *VOL_STATS_FILE_SIZE*를 통해 제어됩니다. 이러한 변수는 위에서 설명된 *LIB_VOL_STATS*와 같은 방법을 사용하여 설정됩니다.
- 두 가지 보고서 유형은 다음과 같습니다.

◦ *drive_centric.txt*

이 보고서는 순서대로 정렬된 보고서 목록을 포함합니다. 각 드라이브 레코드에는 드라이브에 마운트된 모든 카트리지, 요청자, 요청 시간 및 마운트 기간이 포함됩니다.

◦ *time_centric.txt*

주:

이 보고서는 시간별 기간에 나열된 드라이브 리소스의 사용을 포함합니다. 하나의 기간의 각 레코드에는 요청자, 특정 드라이브, 해당 드라이브 기간 중 마운트 수 및 해당 시간 중 드라이브 사용 기간이 포함됩니다.

드라이브 사용이 하나의 기간에 대해 60분을 초과하는 경우 마운트가 두 기간에 걸쳐있었음을 의미하므로, 두번째 기간에 나열되지 않습니다. *stats_report*에서 만든 첫번째 보고서는 드라이브 관점입니다.

주:

- 로그에 *DISMOUNT* 레코드가 있지만, 관련된 *MOUNT* 레코드가 없는 경우 다음이 원인일 수 있습니다.
  - 로그가 롤오버되었습니다.
  - 작업이 알 수 없는 일부 로깅 문제로 인해 로깅되었습니다.

  이런 경우 해당 레코드는 생성된 보고서에서 누락됩니다.
- 로그에 *MOUNT* 레코드가 있지만, 관련된 *DISMOUNT* 레코드가 없는 경우 다음이 원인일 수 있습니다.
  - *DISMOUNT*가 아직 발생하지 않았습니다.
  - 작업이 일부 알 수 없는 로깅 문제로 인해 로깅되지 않았습니다.

  이런 경우 마운트 기간을 -1로 설정하면, 위에서 언급된 사례가 나타납니다. 이러한 레코드는 시간 중심 보고서에서 총 마운트 기간 계산 시 누락됩니다.
- 일광절약시간제에서 표준 시간으로 설정하는 경우 계산된 마운트 기간이 음수인 시나리오가 발생할 수 있습니다. 이를 방지하려면 마운트 기간의 절대값을 취합니다.

userAdmin.sh

userAdmin.sh 메뉴 방식 유틸리티는 ACSLS GUI 사용자 암호를 관리합니다. 이 유틸리티는 $*ACS_HOME*/*install* 디렉토리에 있습니다. 사용자 추가, 사용자 제거, 사용자 나열, 사용자 암호 변경을 수행할 수 있습니다. 이 유틸리티를 사용하려면 WebLogic이 실행 중이어야 합니다. 작동 중이 아닌 경우 이 유틸리티가 WebLogic을 시작하고 메뉴를 표시하기 전에 온라인인지 확인합니다.

이 유틸리티는 *root*로 실행되며, *acsls_admin* 인증이 필요합니다. *acsls_admin* 사용자 계정은 ACSLS 8.4 설치 중에 구성됩니다.

사용자를 추가하거나 사용자의 암호를 변경하는 경우 사용자 이름 및 암호 지정에 대한 프롬프트가 표시됩니다. 암호는 크기 및 올바른 문자의 WebLogic 기준에 대해 검증됩니다.

사용자가 제거되는 경우 해당 계정에 활성 GUI 세션이 여전히 있을 수 있습니다. 사용자가 로그아웃하거나 세션을 종료하면 사용자가 다시 로그인할 수 없게 됩니다. 세션을 강제로 즉시 종료하는 유일한 방법은 GUI를 다시 시작하는 것입니다. ACSLS GUI를 다시 시작하는 (모든 세션을 종료하는) 옵션이 제공됩니다.

이 유틸리티를 사용하여 *acsls_admin* 사용자에 대한 암호를 변경할 수 없습니다. *acsls_admin*에 대한 암호를 변경하거나 재설정해야 할 경우 다음을 수행해야 합니다.

1. wlinstall.*sh* 유틸리티를 실행합니다.

```
$installDir/wlinstall/wlinstall.sh
```

2. *userAdmin.sh*를 실행하여 나머지 사용자 계정을 다시 설정합니다.

형식

```
userAdmin.sh
```

예

```
# ./userAdmin.sh
     ACSLS GUI User Administration
     Weblogic is online.
Please enter the acsls_admin password:
Authenticating.........Connected!
Menu:
1) Add a user account.
2) Remove a user account.
3) Change a user password.
4) List users.
5) Restart ACSLS GUI.
6) Exit.
Please select by number: 1
--- Add a User ---
Please enter the id of the user you wish to add: acsss
Do you wish to add a GUI account for user 'acsss'? (y/n) y
Please assign a password for 'acsss'.
     Passwd: Please confirm password:
     Passwd:
Connecting..........
User accounts has been added.
Please select by number: 2
--- Remove a User ---
Please enter the name of the user you wish to remove: accounts
 Do you wish to remove the ACSLS GUI account for user 'accounts'? (y/n) y
Connecting..........
The account for accounts has been removed for future logins.>
To disable any current login session for accounts, you
 must restart the ACSLS GUI.
Please select by number: 3
--- Change Password ---
Enter the user name: acsss
Passwd: Please confirm password:
Passwd:
Connecting..........
Password changed for acsss!
Please select by number: 4
```

```
--- List Users ---
Connecting.........

Configured WebLogic users:
      OracleSystemUser
      acsls_admin
      acsss
Please select by number: 5
Do you wish to restart the ACSLS GUI (affects all users)? (y/n) y
 Restarting:
    Disabling WebLogic ........................
    Enabling WebLogic ..............................
Please select by number: e
#
```

volrpt

volrpt 유틸리티는 볼륨 보고서를 만듭니다.

형식

volrpt [-s vol|*loc*|*use] [-d] [-f* filename*] [-z] [-a*|*-l*|*-v* identifier_list*] [-i]*

옵션

- *-s*

  정렬 순서를 지정합니다. 이 옵션을 지정하지 않으면 기본값은 볼륨 ID별 정렬로 지정됩니다. 이 옵션을 지정하면 다음 값 중 하나를 지정해야 합니다.

  - *vol*

    볼륨 ID별로 정렬합니다.

  - *loc*

    볼륨 홈 위치별로 정렬합니다.

  - *use*

    볼륨 사용(마운트 수)별로 정렬합니다.

- *-d*

  출력에 페이지 나눔 또는 헤더 정보가 포함되지 않도록 지정합니다. 이 출력은 *pr*과 같은 다른 프로그램에 입력으로 사용될 수 있습니다.

- *-f filename*

  파일 이름은 사용자 정의 volrpt 템플리트를 지정합니다.

- *-z*

  0으로 식별자 필드를 채웁니다.

- *-a*

보고서를 지정된 ACS로 제한합니다. 여러 ACS를 지정할 수 있습니다(공백을 사용하여 *acs_ids* 구분).

- *-l*

  보고서를 지정된 LSM으로 제한합니다. 여러 LSM을 지정할 수 있습니다(공백을 사용하여 *lsm_ids* 구분).

- *-v*

  보고서를 지정된 볼륨(또는 볼륨 범위)으로 제한합니다. 단일 *vol_id*, 공백으로 구분된 vol_ids의 목록 또는 *vol_id-vol_id*에 의해 표시되는 볼륨 범위를 지정할 수 있습니다.

- *identifier_list*

  -v, -a 및 -l 옵션에서 설명되었습니다. 이는 ACS, LSM 및 볼륨(또는 볼륨 범위)의 목록입니다.

- *-i*

  모든 볼륨을 포함하며, 없는 카트리지 및 꺼낸 카트리지를 포함합니다.

  이 옵션을 지정하지 않으면 없는 카트리지와 꺼낸 카트리지가 보고되지 않습니다.

사용법

volrpt 유틸리티를 사용하여 라이브러리 카트리지 보고서를 만들고 카트리지의 물리적 위치, 기록, 속성 및 사용을 포함합니다. 또한, volrpt를 사용하여 데이터베이스 복원 후에 데이터베이스를 검증할 수 있습니다. *-a, -l* 또는 *-v* 옵션을 사용하여 보고서를 위한 ACS, LSM 또는 카트리지를 지정할 수 있습니다. 이러한 옵션을 하나도 지정하지 않으면 *volrpt*는 ACS 0만 보고합니다.

주:

선행 및 후행 공백을 특히 주의해야 합니다. 선행 및 후행 공백을 포함하는 볼륨에 대해 인수를 지정할 경우 작은따옴표로 인수를 묶어주어야 합니다. 작은따옴표가 셸 구성 요소 사이에서 전달되는지 확인하려면 따옴표가 이스케이프 문자로 태그가 지정되어야 합니다. UNIX에서 표준 이스케이프 문자는 백슬래시(**)입니다.

예:

선행 공백이 있는 볼륨 ID를 참조할 로컬 시스템에서 *volrpt* 명령을 나타내려면 다음과 같이 명령을 제출합니다.

```
volrpt -v /'0000/'-/'9999/'
```

원격 셸*(rsh)*을 통해 동일한 명령을 제출하려면 전체 인수를 큰따옴표로 묶어줍니다.

```
rsh <acsls_hostname> -l acsss bin/volrpt -v "/' 0000/'-/' 9999/'"
```

다음 예에서는 표준 볼륨 보고서를 보여줍니다. 여기에는 볼륨 ID, 위치, 레이블 유형, 매체 유형 및 사용 기록에 대한 필드가 포함됩니다.

```
VOLUME REPORT UTILITY
```

```
2002-06-30 14:01:21
TOTAL VOLUMES: 400 SEQUENCE: sort by volume identifier
Volume Home    LabelVolume    Times|---Entered---||--Last Used--|
 Label Location AttrType/Media MountedDateTime      DateTime
CLN000 0,0,1,0,3 ExtC/STK1U    108/22/0109:30     10/04/01 14:26
RB0000 0,1,2,1,10Ext.D/STK1R    310/01/0108:16     10/01/01 08:18
RB1400 0,0,10,1,3Ext.S/STK1R    24310/01/0109:30     10/06/01 11:04
RB1401 0,0,10,3,5Virt.D/STK1R    1210/01/0103:29     10/05/01 23:11
  ”      "      " "  "  ""
  "      "      "  "  "  ""
  "      "      "  "  "  ""
TB1440 0,1,3,1,9 Ext.D/STK2P    4308/12/0109:1109/28/0117:52
  "      "      "  "  "  ""
  "      "      "  "  "  ""
  "      "      "  "  "  ""
```

Volume Type/Media 열에서: C는 청소 카트리지를 나타냅니다. D는 데이터 카트리지를 나타냅니다. P는 테이프 드라이브에서 소비한(모두 사용한) 것으로 보고한 청소 카트리지를 나타냅니다. S는 스크래치 카트리지를 나타냅니다.

-f filename 옵션을 사용하여 사용자 정의된 보고서를 만듭니다. 자세한 내용은 “로깅 볼륨 통계 보고서 작성”를 참조하십시오.

$ACS_HOME/data/external/volrpt/owner_id.volrpt*는 사용자 정의된 볼륨 보고서를 만들기 위해 템플리트로 실행 또는 사용할 수 있는 샘플 입력 파일입니다. *$ACS_HOME/data/external/volrpt 디렉토리에 사용자 정의된 볼륨 보고서를 저장할 수도 있습니다.

표준 UNIX 재지정을 사용하여 볼륨 보고서를 파일로 재지정할 수 있습니다.

```
volrpt > file
```

예

기본적으로 volrpt는 목록의 첫번째 ACS만 보고합니다. ACS 0과 ACS 1 모두에 있는 카트리지를 보고하려면 다음 명령을 입력하십시오.

```
volrpt -a 0 1
```

홈 셀 위치로 정렬된 LSMs 0,1 및 2,1의 카트리지를 보고하려면 다음 명령을 입력하십시오.

```
volrpt -s loc -l 0,1 2,1
```

주:

- *volrpt*가 성공적으로 완료되면 지정된 볼륨 보고서를 표시합니다. *-f* 옵션을 지정하고 *volrpt*가 지정된 파일을 찾을 수 없거나 둘 이상의 입력 파일을 지정할 경우 *volrpt*는 메시지를 *stderr*에 인쇄하고 종료합니다. 입력 파일 내의 필드 오류의 경우 *volrpt*는 메시지를 *stderr*에 인쇄하고 오류가 있는 행을 무시하지만 종료되지 않습니다.
- 카트리지가 지정된 볼륨 ID 목록, 범위 또는 라이브러리 구성 요소에 없는 경우 *volrpt*는 볼륨이 없습니다. 메시지를 반환합니다.

- 매개변수가 지정되지 않은 경우 기본값으로 ACS 0을 사용합니다.
- 라이브러리 구성 요소가 *-a, -l* 또는 *-v* 옵션을 통해 지정되었지만 볼륨이 없는 경우 다음과 같은 메시지가 표시됩니다.
  - *-a* 옵션(ACS)

    메시지:

    단일 *acs_id*가 제공되지만 볼륨이 없는 경우 다음 오류 표시: *No Volumes found for ACS: (*<acsid>*)*

    예:

    ```
    $ volrpt -a 2
    No Volumes found for ACS: (2)
    ```

    여러 *acs_ids*가 제공되지만 그중에 볼륨이 있는 acs_id가 없는 경우 다음 오류 표시:

    No Volumes found for ACS: (<acsid1>*)(*<acsid2>*)*

    예:

    ```
    $ volrpt -a 0 1
    No Volumes found for LSM: (0) (1)
    ```

  - *-l* 옵션(LSM)

    메시지:

    단일 *lsm_id*가 제공되지만 볼륨이 없는 경우 다음 오류 표시: *No Volumes found for LSM: (*<lsmid>*)*

    예:

    ```
    $ volrpt -l 1,1
    No Volumes found for LSM: (1,1)
    ```

    여러 lsm_ids가 제공되지만 그중에 볼륨이 있는 lsm_id가 없는 경우 다음 오류 표시: *No Volumes found for LSM: (*<lsmid1>*)(*<lsmid2>*)*

    예:

    ```
    $ volrpt -l 1,1 1,2
    No Volumes found for LSM: (1,1) (1,2)
    ```

  - *-v* 옵션(VOLUME)

메시지:

단일 *volid*가 제공되지만 볼륨이 없는 경우 다음 오류 표시: *Volume(s) not: (*<volid>*)*

예:

```
$ volrpt -v BBB112
No Volumes found: (BBB112)
```

여러 volids가 제공되지만 그중에 볼륨이 있는 volid가 없는 경우 다음 오류 표시: *Volume(s) not found: (*<volid1>*)(*<volid2>*)*

예:

```
$ volrpt -v BBB112 BBB114
No Volumes found: (BBB112) (BBB114)
```

-v 옵션은 볼륨 범위에 사용할 수도 있고 볼륨이 없는 경우 유사한 메시지를 생성합니다.

단일 볼륨 범위가 제공되지만 볼륨이 없는 경우 다음 오류 표시: *Volume(s) not: (*<volrange>*).*

예:

```
$ volrpt -v BBB112-BBB116
No Volumes found: (BBB112-BBB116)
```

다중 볼륨 범위가 제공되지만 볼륨이 없는 경우 다음 오류 메시지 표시: *Volume(s) not: (*<volrange1>*) (*<volrange2>*)*

예:

```
$ volrpt -v BBB112-BBB116 BBB220-BBB224
No Volumes found: (BBB112-BBB116) (BBB220-BBB224)
```

ACS 또는 LSM이 구성되지 않은 경우

*volrpt*가 존재하지 않는 acs_id 또는 lsm_id와 함께 사용되는 경우 해당 식별자에 따라 메시지를 표시합니다.

- *-a* (ACS)

ACS 식별자(<acsid>)가 구성되지 않음

- *-l* (LSM)

  LSM 식별자(<lsmid>)가 구성되지 않음

"로깅 볼륨 통계 보고서 작성"를 참조하십시오.

watch_vols

이 유틸리티는 다음 볼륨에 대해 미리 정의된 정책을 적용합니다.

- 새로 넣음
- 감사 또는 카트리지 복구로 발견됨
- 감사, 카트리지 복구 또는 넣기로 다시 활성화

이러한 정책은 다음 파일에 정의됩니다.

$ACS_HOME/data/external/vol_attr.dat

이 파일은 사용자 정의 볼륨 ID 또는 볼륨 범위의 목록 및 기록된 각 볼륨에 대한 사용자 정의 정책을 포함합니다. 해당 파일에 나열된 각 볼륨 또는 볼륨 범위의 경우 볼륨을 넣으면 볼륨 소유권, 풀 연결, 선호하는 LSM 위치 및/또는 논리 라이브러리 지정을 정의할 수 있습니다. 정책 정의를 위한 특정 지침은 *vol_attr.dat* 파일에 자세히 설명되어 있습니다.

watch_vols 유틸리티는 *acsss_stats.log*를 사용하여 새로 넣은 볼륨이나 감사 중에 또는 카트리지 복구를 통해 발견되거나 다시 활성화된 볼륨이 존재하는지 식별합니다. 이 기능을 사용으로 설정하려면 *acsss_config*(옵션 3)로 볼륨 통계를 사용으로 설정해야 합니다. 볼륨 통계가 사용으로 설정되면 *watch_vols*는 *acsss_stats.log*의 끝 부분을 모니터링하고 *vol_attr.dat*에 정의된 항목과 일치하는 볼륨을 찾습니다. 일치 항목을 찾으면 해당 볼륨에 대해 정의된 정책이 자동으로 적용됩니다.

볼륨 ID는 다음 규칙을 따라야 합니다.

- 행별 하나의 *vol_id* 또는 볼륨 범위.
- vol_id는 유효한 ACSLS 볼륨 ID여야 합니다.
- vol_id에 선행 또는 후행 공백이 포함되는 경우 공백을 밑줄(_)로 나타내야 합니다. 예: _V234_.

형식

watch_vols [start|stop]

사용법

매개변수 없이 *watch_vols*를 호출하여 실행 중인 유틸리티의 상태를 확인할 수 있습니다. *watch_vols*의 상태(실행 중 또는 중지됨)가 확실하지 않은 경우 *watch_vols* 명령을 인수 없이 사용하면 현재 상태가 표시됩니다.

*watch_vols*에 대해 *start* 및 *stop* 두 가지 옵션이 있습니다.

- *watch_vols start*

  시작 매개변수가 호출되면 *watch_vols*는 *vol_attr.dat*에 정의된 정책을 검토합니다. 형식이나 구문에 오류가 있으면 *watch_vols*에서는 오류를 표시하고 *vol_attr.dat*에 필요한 수정 사항을 수행하라는 프롬프트가 표시됩니다. 정의된 정책이 watch_vols에서 허용되면 유틸리티가 데몬을 호출하여 백그라운드로 실행합니다. ACSLS가 실행 중인 경우 데몬이 계속 실행됩니다. ACSLS가 다시 시작할 때마다 데몬이 자동으로 시작됩니다.

  *vol_attr.dat*의 정책 테이블은 언제든지 업데이트할 수 있습니다. 정책을 업데이트하기 위해 *watch_vols*를 중지하지 않아도 됩니다. *watch_vols start*를 실행하여 업데이트를 실행 중인 프로그램에 적용하십시오.
- *watch_vols stop*

  이 명령은 지정된 볼륨에 대한 추가 정책 적용을 중지합니다.

모든 *watch_vols* 활동에 대한 로그가 다음 로그 파일에서 유지 관리됩니다.

$ACS_HOME/log/watch_vols_event.log

볼륨 소유권, *pool_id* 또는 LSM 홈 위치 등 각 변경사항이 이 파일에 로깅됩니다.

예

넣기 작업을 수행 중이며 볼륨을 넣을 때 특정 볼륨을 대상 LSM으로 이동하려고 합니다.

1. *watch_vols disabled*로 대상 LSM을 감사합니다.
2. 대상 LSM의 감사가 완료되면 *watch_vols*를 시작합니다.
3. *vol_attr.dat*에 정의된 정책이 있는 볼륨을 넣습니다.

   그러면 볼륨을 넣은 후에 watch_vols는 지정된 볼륨을 대상 LSM으로 이동합니다.

13

13장. 명령 참조

이 장에서는 일반 명령 구문 및 참조 정보를 포함하여 ACSLS 명령을 사용하는 방법을 배웁니다.

- *“audit ”*

  라이브러리 구성 요소에서 볼륨의 데이터베이스 인벤토리를 만들거나 업데이트합니다.
- *“cancel ”*

  현재 또는 보류 중인 요청을 취소합니다.
- *“clear lock ”*

  드라이브 또는 카트리지의 모든 활성 및 보류 중인 잠금을 제거합니다.
- *“define pool ”*

  스크래치 풀을 만들거나 수정합니다.
- *“delete pool ”*

  빈 스크래치 풀을 삭제합니다.
- *“dismount ”*

  카트리지를 마운트 해제합니다.
- *“eject ”*

  하나 이상의 카트리지를 ACS에서 꺼냅니다.
- *“enter ”*

  CAP를 넣기 모드로 설정합니다.
- “idle ”

  ACSLS가 새 요청 처리를 중지합니다.
- *“lock ”*

  카트리지 또는 드라이브를 사용자 전용으로 잠급니다.
- *“logoff ”*

  cmd_proc를 종료합니다.

- *“move ”*

  지정된 카트리지를 지정된 LSM의 사용 가능한 스토리지 셀로 이동합니다.
- *“mount ”*

  데이터 또는 스크래치 카트리지를 마운트합니다.
- *“query 명령 ”*

  라이브러리 구성 요소의 상태를 표시합니다.
- “set 명령 ”

  여러 라이브러리 구성 요소의 다양한 속성을 설정합니다.
- *“show ”*

  잠금 ID 또는 사용자 ID를 표시합니다.
- *“start ”*

  ACSLS 요청 처리를 시작합니다.
- *“switch lmu ”*

  ACS 관리를 ACS의 활성 LMU에서 대기 LMU로 수동으로 전환합니다.
- *“unlock ”*

  카트리지 또는 드라이브의 활성 잠금을 제거합니다.
- *“vary ”*

  ACS, LSM, CAP, 드라이브 또는 포트의 상태를 변경합니다.
- *“venter ”*

  레이블이 누락되거나 레이블을 읽을 수 없는 하나 이상의 카트리지를 ACS에 넣습니다.

  주:

  L5500, SL500 및 SL8500 라이브러리는 레이블이 없는 카트리지(venter)를 지원하지 않습니다.

일반 명령 구문

이 절에서는 ACSLS 명령의 일반 구문에 대해 설명합니다. 다음 절에서는 각 명령과 해당 구문에 대해 자세히 설명합니다.

ACSLS 명령은 다음과 같은 일반 구문을 사용합니다.

command type identifier state options

설명:

type identifier

ACS 구성 요소와 해당 식별자입니다. 자세한 내용은 “구성 요소 유형 및 식별자 ”를 참조하십시오.

- *state*

  vary 명령에만 해당하는 장치 상태입니다.
- *options*

  명령 옵션입니다. 실행할 명령에 대한 설명을 참조하십시오.

  다음 구문 규칙에 유의하십시오.
- 위에 표시된 순서(명령 이름, 구성 요소와 해당 식별자, 상태 및 옵션)로 명령을 입력합니다.
- 이 장에서는 명령을 소문자로 표시하지만 소문자와 대문자를 조합하여 명령을 입력할 수 있습니다.
- 밑줄은 명령 및 키워드를 최소로 줄인 약어를 표시합니다. 예를 들어 **query server**와 *q ser* 둘 다 *query server* 명령의 유효한 형식입니다.
- 줄임표(...)는 식별자가 반복될 수 있음을 나타냅니다.
- 대괄호 []는 선택사항인 옵션을 묶는 데 사용됩니다.

구성 요소 유형 및 식별자

다음 표에서는 각 ACS 구성 요소 식별자의 유효한 값 범위에 대해 설명합니다. 각 유형에 대해 최대 42개의 식별자를 지정할 수 있습니다. 각 명령에 유효한 구성 요소 유형은 해당 명령 설명을 참조하십시오.

주:

다음 표에 지정된 식별자는 소프트웨어에서 지원하는 유효한 값 범위를 나타냅니다. LSM 유형 및 라이브러리 구성에 따라 특정 사이트에 유효한 식별자 값이 결정됩니다.

표 13.1. ACSLS 구성 요소 유형 및 식별자

구성 요소	유형	식별자	유효한 값
전체 라이브러리	server	없음	없음
ACS	acs	*acs_id*	acs(*0-31*), lsm(*0-99*)
LSM	lsm	*lsm_id*	acs(*0-31*), lsm(*0-99*)
LSM 패널	panel	*panel_id*	acs(*0-31*), lsm(*0-99*), panel(*0-50*),
LSM 하위 패널[1]	subpanel	*subpanel_id*	acs(*0-31*), lsm(*0-99*),

구성 요소	유형	식별자	유효한 값
			panel(*0-50*), startrow(*0-51*), startcolumn(*0-23*), endrow(*0-51*), endcolumn(*0-23*)
LSM 스토리지 셀	subpane*l*	*cell_id*	acs(*0-31*), lsm(*0-99*), panel(*0-50*), row(*0-51*), column(*0-23*)
CAP	cap	*cap_id*	acs(*0-31*), lsm(*0-99*), cap(*0-11*) *cap_id*의 별표(*)는 다음을 수행합니다. acs,lsm,* - ACSLS가 LSM에서 가장 높은 우선 순위로 사용 가능한 CAP를 선택하도록 합니다. acs,* - ACSLS가 ACS에서 가장 높은 우선 순위로 사용 가능한 CAP를 선택하도록 합니다. * - ACSLS가 사용 가능한 셀이 가장 많은 LSM에서 CAP를 선택하도록 하는 넣기 요청에 사용됩니다. * - ACSLS가 꺼내기로 지정된 카트리지가 있는 각 ACS에서 우선 순위가 가장 높은 CAP를 선택하도록 하는 꺼내기 요청에 사용됩니다.
드라이브	drive	*drive_id*	acs(*0-31*), lsm(*0-99*), panel(*0-50*), drive(*0-31*)
드라이브 유형	drive	*drive_type*	최대 10자로 된 드라이브 유형 식별자이며, 숫자(*0-9*) 또는 문자(*A-Z*)를 조합할 수 있습니다. 공백은 허용되지 않습니다.
포트	port	*port_id*	acs(*0-31*), port(*0-3*)
데이터 카트리지, 스크래치 카트리지 또는 청소 카트리지의 볼륨 일련 번호	volume, scratch, clean	*vol_id*	숫자(*0-9*), 문자(*A-Z*), 달러 기호(*$*), 파운드 기호(*#*) 및 공백을 조합하여 구성된 6자 식별자입니다. 작은 따옴표 또는 큰 따옴표를 사용하여 선행 또는 후행 공백이 있는 volser를 묶습니다.

구성 요소	유형	식별자	유효한 값
			중간에 공백이 포함된 volser를 지정하지 마십시오.
볼륨 일련 번호의 범위	volume	*volrange*	대시(-)로 구분된 오름차순 볼륨 범위를 지정합니다. ALPHANUM_VOL_RANGES 동적 변수가 FALSE(기본값)로 설정된 경우: ACSLS가 전체 vol_id(예: AAA000-AAA999)로 지정된 범위의 시작 및 끝 vol_id를 지원합니다. volser의 맨 오른쪽 숫자 부분만 범위로 지정하십시오. 앞에 있는 모든 문자는 동일해야 합니다. *** ALPHANUM_VOL_RANGES 동적 변수가 TRUE로 설정된 경우: ACSLS가 숫자(0-9), 문자(A-Z), 달러 기호($), 파운드 기호(#) 및 공백(선행 및 후행)을 조합하여 구성된 볼륨 범위를 지원합니다. *** *** 작은 따옴표 또는 큰 따옴표를 사용하여 범위에 선행 또는 후행 공백이 있는 volser를 묶습니다. ACSII 조합 시퀀스에 따라 범위는 오름차순 시퀀스여야 합니다.
볼륨 매체 유형	media	*media_type*	최대 10자로 된 매체 유형 식별자이며, 숫자(*0-9*), 문자(*A-Z*) 및 대시(-)를 조합할 수 있습니다. 공백은 허용되지 않습니다.
볼륨 소유자	owner	*owner_id*	
스크래치 풀	pool	*pool_id*	10진수(*0-65534*). pool_id에 별표(*)를 지정하면 볼륨에 현재 pool_id가 다시 지정됩니다.
ACSLS 요청	request	*request_id*	ACSLS가 지정한 고유 10진수(*0-65535*)입니다.
드라이브 또는 볼륨 잠금	lock	*lock_id*	10진수(*0-32767*)
ACSLS 요청	request	*request_id*	ACSLS가 지정한 고유 숫자(*0-65535*) 요청 식별자입니다.

[1]종료 행 및 열은 시작 행 및 열보다 크거나 같아야 합니다. 시작 매트릭스 및 종료 매트릭스 내에 있는 셀만 감사됩니다. 매트릭스는 시작 행 및 시작 열로 시작하여 종료 행 및 종료 열로 확장됩니다.

일반 명령 오류 메시지

구문 오류, 잘못된 식별자, 유형, 옵션, 프로세스 실패, 데이터베이스 오류 등으로 인해 ACSLS가 명령을 거부하면 일반 오류 메시지가 나타납니다. 일반 오류 메시지에 대한 자세한 내용은 *ACSLS Messages*를 참조하십시오.

사용할 수 없는 명령을 입력하면 다음 메시지가 나타납니다.

```
Command access denied.
```

액세스할 수 없는 볼륨을 지정하면 다음 메시지가 나타납니다.

```
Volume access denied.
```

명령

다음 절에서는 ACSLS 명령에 대해 설명합니다.

audit

audit 명령은 라이브러리 카트리지의 실제 인벤토리와 일치하도록 ACSLS 데이터베이스를 업데이트합니다.

다음과 같은 경우에는 항상 감사를 수행해야 합니다.

- 새 라이브러리인 경우
- ACSLS 데이터베이스를 라이브러리 컨텐츠와 재동기화하려는 경우
- 하나 이상의 모듈이 추가, 제거 또는 스왑된 경우
- 카트리지가 도어를 통해 수동으로 추가 또는 제거된 경우
- 라이브러리 구성 설정이 변경된 다음
- ACSLS가 라이브러리에 있는 하나 이상의 카트리지 위치를 모르는 경우
- 라이브러리를 확장했으며 라이브러리를 처음 감사하는 경우

  SL8500을 확장한 경우 라이브러리를 처음 감사하려면 “SL8500 내부 주소 및 ACSLS 주소 이해 ”에 설명된 절차를 참조하십시오.

형식

audit cap_id type identifier...

- *cap_id*

  ACSLS가 에런트 볼륨을 꺼내기 위해 사용하는 CAP를 지정합니다. 특정 CAP를 지정하거나 별표(*)를 입력할 수 있습니다. 별표를 입력하면 ACSLS는 우선 순위가 가장 높은 CAP를 선택합니다.

  각 ACS에 CAP 우선 순위가 설정된 경우에만 서버 또는 여러 개의 ACS를 지정할 수 있습니다. 각 ACS에서 CAP 자동 선택을 허용하려면 *cap_id*에 별표를 지정해야 합니다.

  단일 LSM *audit*는 ACSLS가 데이터베이스를 업데이트할 때까지 기다린 다음 CAP를 예약합니다(에런트 볼륨을 꺼내는 데 필요한 경우).
- *type identifier*

  라이브러리 구성 요소를 지정합니다. 다음 표에는 감사(*audit*)할 수 있는 구성 요소가 나와 있습니다.

표 13.2. Audit에 유효한 구성 요소

라이브러리 구성 요소	유형	식별자
모든 라이브러리	*server*	없음
ACS	*acs*	*acs_id*
LSM	*lsm*	*lsm_id*
LSM 패널	*panel*	*panel_id*

라이브러리 구성 요소	유형	식별자
LSM 하위 패널	*subpanel*	*subpanel_id*

단일 *audit* 요청에 여러 개의 ACS, LSM, 패널 또는 하위 패널을 지정할 수 있습니다. 겹치는 하위 패널은 지정할 수 없습니다.

각 ACS에 CAP 우선 순위가 설정된 경우에만 서버 또는 여러 개의 ACS를 지정할 수 있습니다. 각 ACS에서 CAP 자동 선택을 허용하려면 *cap_id*에 별표를 지정해야 합니다. 이렇게 하지 않으면 감사에서 찾은 잘못된 카트리지를 꺼낼 수 없습니다.

서버 또는 ACS를 감사할 때 상위 감사 프로세스가 각 LSM에 대해 별도의 감사 프로세스를 만듭니다. 이러한 LSM 감사는 병렬로 실행되며 오름차순 *panel_id* 순서로 패널을 하나씩 처리합니다. ACS 또는 모든 라이브러리의 모든 LSM을 감사하려면 서버 또는 ACS만 감사하면 됩니다. 이 경우 별도의 LSM 감사를 여러 번 실행할 필요가 없습니다.

하지만 감사에서는 LSM 내의 여러 구성 요소가 지정된 순서에 관계없이 *acs_id*, `lsm_id` `panel_id`, *subpanel_id*를 기준으로 오름차순으로 구성 요소를 처리합니다. 예를 들어 사용자가 하위 패널을 먼저 지정했더라도 감사에서는 하위 패널 0,0,10,1,7보다 패널 0,0,9를 먼저 처리합니다.

사용법

*audit*는 라이브러리 카트리지의 실제 인벤토리와 일치하도록 ACSLS 데이터베이스를 업데이트합니다. *audit*를 실행하는 목적은 다음과 같습니다.

- CAP를 통해 카트리지를 넣지 않은 경우 데이터베이스에 볼륨 정보를 만듭니다. 예를 들어 라이브러리에 LSM을 추가하고 LSM 도어를 연 다음 LSM에 수동으로 카트리지를 추가하는 경우 *audit*를 실행합니다.
- 라이브러리와 데이터베이스 간의 불일치를 해결합니다. 예를 들어 CAP를 통해 카트리지를 꺼내는 대신 LSM 도어를 열고 수동으로 카트리지를 제거하는 경우 *audit*를 실행합니다. *audit*는 제거된 카트리지에 대한 정보를 데이터베이스에서 삭제합니다.
- *audit*에 지정된 셀의 컨텐츠를 확인합니다(LSM 로봇의 비전 시스템에 디스플레이 모니터를 연결해야 함).

주:

감사는 청소 카트리지의 특정 모델을 인식하고 데이터베이스에 청소 카트리지로 기록합니다. 또한 *audit* 명령은 발견하는 각 청소 카트리지 유형에 적절하게 새 청소 카트리지에 대한 *max_uses*를 설정합니다.

주:

*audit*가 기록된 위치(셀 및 드라이브)에서 카트리지를 찾을 수 없는 경우 *audit*는 카트리지를 존재하지 않는 것으로 표시하거나(존재하지 않는 볼륨 보존이 사용으로 설정된 경우) 볼륨을 삭제합니다. *audit*는 볼륨을 삭제할 때 볼륨에 대한 모든 정보(볼륨 정보, 액세스 제어에 대한 고객 제공 정보, 스크래치 상태, 잠금 ID 및 풀 ID 포함)를 데이터베이스에서 제거합니다. *audit*가 나중에 다른 위치에서 카트리지를 찾을 경우 볼륨 정보를 다시 추가하지만 고객 제공 정보는 손실됩니다.

*audit*는 에런트 볼륨을 꺼내고 해당 정보를 데이터베이스에서 삭제합니다. 에런트 볼륨에는 다음이 포함되어 있습니다.

- 이미 스캔된 레이블과 중복되는 외부 레이블
- 누락되거나 읽을 수 없는 외부 레이블, 가상 레이블 없음
- 잘못된 매체 유형

감사는 LSM 스토리지 셀에만 적용되고 드라이브나 CAP에는 적용되지 않습니다. ACSLS는 감사 중 *cmd_proc* 메시지를 표시하고 *audit*의 모든 데이터베이스 변경사항을 이벤트 로그에 기록합니다. 볼륨 통계 로깅이 사용으로 설정된 경우 ACSLS는 *acsss_stats.log*에 추가 볼륨 발견 및 볼륨을 찾을 수 없음 메시지를 기록합니다. 자세한 내용은 "로깅 볼륨 통계 보고서 작성"을 참조하십시오.

힌트: *audit* 실행 시 다음과 같은 지침을 사용하십시오.

- 감사 중인 ACS 또는 LSM이 온라인 또는 진단 상태여야 합니다. *audit* 중에도 정상적인 라이브러리 처리(마운트 및 마운트 해제 포함)가 발생할 수 있지만 라이브러리 처리로 인해 감사 속도가 느려집니다.

  온라인 상태와 진단 상태의 차이는 다음과 같습니다. 진단 상태의 *audit*는 *cmd_proc*를 통해서만 수행할 수 있습니다. LSM이 진단 상태인 동안에는 클라이언트 요청이 거부됩니다. 이 상태에서는 *audit*가 클라이언트로부터 제출될 수 있는 *mount/dismount* 요청과 경합하지 않고 라이브러리를 배타적으로 실행하기 때문에 더 빠릅니다.
- 전체 LSM에서 *audit*를 시작한 후에는 동일한 LSM에서 다른 *audit*를 시작(*start*)할 수 없습니다. 감사를 취소(*cancel*)하고 다시 실행해야 합니다.
- 모든 감사 요청을 취소(*cancel*)할 수 있지만 ACSLS는 항상 현재 패널 또는 하위 패널에 대한 감사를 마칩니다. 감사를 취소(*cancel*)하면 꺼내도록 표시된 카트리지의 일부 또는 모두를 꺼내지 못할 수 있습니다. 감사를 취소(*cancel*)하면 이미 꺼낸 카트리지는 다시 넣지 않습니다.

주의:

감사를 취소(*cancel*)하거나 감사 중 라이브러리 또는 ACSLS 하드웨어/소프트웨어 오류가 발생한 경우 동일한 감사를 다시 실행해야 합니다. 꺼내도록 표시되었지만 첫번째 감사 중 실제로 꺼내지 않은 카트리지는 더 이상 데이터베이스에 없으며 ACSLS의 제어를 받지 않습니다.

예

- 전체 라이브러리를 감사하고 각 ACS에서 우선 순위가 가장 높은 CAP를 꺼내도록 지정하려는 경우

  ```
  audit * server
  ```
- LSM 0,1을 감사(*audit*)하고 CAP 0,1,1을 꺼내도록 지정하려는 경우

  ```
  audit 0,1,1 lsm 0,x
  ```
- LSM 0,1의 패널 10을 감사(*audit*)하고 LSM 0,1의 우선 순위가 가장 높은 CAP를 꺼내도록 지정하려는 경우

  ```
  audit 0,1,* panel 0,1,10
  ```

주:

다른 프로세스에 예약된 셀은 감사할 수 없습니다. 셀이 예약된 경우 ACSLS는 셀을 사용할 수 있게 될 때까지 데이터베이스를 최대 60회까지 다시 확인합니다. 그래도 셀을 사용할 수 없으면 *audit*는 셀을 건너뛰고 이벤트 로그에 메시지를 기록합니다.

참조:

찾을 정보	참조 항목
라이브러리 감사 지침	"ACS 번호 지정 "
명령 취소	"cancel "
CAP 상태 표시	"query cap "
ACSLS 및 라이브러리 상태 표시	"query server "
ACS 상태 표시	"query acs "
LSM 상태 표시	"query lsm "
CAP 선택 우선 순위 설정	"set cap priority "
CAP 모드(수동 또는 자동) 설정	"set cap mode "
라이브러리 구성 요소 상태 변경	"vary "

명령 영역 메시지

다음 절에서는 감사 메시지에 대해 설명합니다.

성공 메시지

- *audit*가 성공적으로 완료되면 다음 메시지가 나타납니다.

  Audit: Audit completed, Success.
- 또한 감사된 구성 요소를 확인하기 위해 다음 메시지 중 하나가 표시됩니다.

  Audit: Audit of storage server, valid

  Audit: Audit of ACS, acs_id, status valid

  Audit: Audit of LSM, lsm_id, panel_id, valid

  Audit: Audit of panel, panel_id, valid

  Audit: Audit of subpanel, subpanel_id, valid

힌트: 잘못된 셀 위치(드라이브 바로 위 또는 아래에 있거나 추가 열에 있음)를 감사(*audit*)할 경우 ACSLS는 PTP에서 감사 작동을 허용하는 성공 메시지를 반환합니다.

중간 메시지

표시되는 중간 메시지는 두 라인으로 구성되며 첫번째 라인은 다음과 같습니다.

Audit: Intermediate response: Audit activity.

두번째 라인에는 다음 메시지 중 하나가 나타납니다.

- *Audit: Volume ejected, unreadable label.*

  설명: ACSLS가 다음과 같은 카트리지를 꺼냈습니다.

  외부 레이블이 없음

  가상 레이블이 없음 또는

  레이블을 읽을 수 없음
- *Audit: Volume vol_id will be ejected, duplicate label.*

  설명: ACSLS가 감사 중인 셀 범위 내에 중복 외부 레이블이 있는 카트리지를 꺼냈습니다.

  변수: *vol_id*는 중복 레이블이 있는 볼륨입니다.
- *Audit: Volume vol_id found.*

  설명: 감사 중 ACS에서 발견한 볼륨이 ACSLS 데이터베이스에 없습니다. 감사에서 이 볼륨을 데이터베이스에 추가했습니다.

  변수: *vol_id*는 데이터베이스에 추가된 볼륨입니다.
- *Audit: Volume vol_id not found.*

  설명: ACSLS 데이터베이스에 나열된 볼륨이 ACS에 없습니다. 볼륨이 데이터베이스에서 삭제되었습니다.

  변수: *vol_id*는 데이터베이스에서 삭제된 볼륨입니다.
- *Audit: Volume will be ejected, invalid media type*

  설명: ACSLS가 매체 유형이 잘못된 볼륨을 꺼냈습니다.

오류 메시지

- *Audit in progress.*

  설명: 동일한 LSM에 대해 다른 감사가 진행 중이어서 ACSLS가 *audit*를 시작하지 않았습니다.
- *CAP cap_id in use.*

  설명: 감사하도록 지정된 CAP가 사용 중입니다.

  변수: *cap_id*는 사용 중인 CAP입니다.
- *Multiple ACS audit.*

  설명: *cap_id*에 *를 지정하지 않고 *audit* 명령에 ACS를 여러 개 지정해서 감사가 실패했습니다.
- *Not in same ACS.*

설명: 지정된 *cap_id*와 *identifier*가 동일한 ACS에 있지 않아서 `audit`가 실패했습니다.

변수:

*cap_id*는 감사하도록 지정된 CAP입니다.

*identifier*는 감사하도록 지정된 라이브러리 구성 요소입니다.

표시 영역 메시지

- *cap_id* Remove cartridges from CAP.

  설명: 감사에서 CAP를 꺼낸 카트리지로 채웠습니다. CAP를 비운 다음 닫고 감사를 계속하십시오.

  변수: *cap_id*는 꺼낸 카트리지가 포함된 CAP입니다.
- `CAP` *cap_id*`: Place magazines in CAP.`

  설명: 감사를 수행하려면 CAP에 매거진이 필요합니다. CAP를 열고 매거진을 안에 넣은 다음 CAP를 닫으십시오.

  변수: *cap_id*는 매거진이 필요한 CAP입니다.
- `CAP` *cap_id*`: No CAP available, waiting...`

  설명: 카트리지를 꺼내는 데 사용할 수 있는 CAP가 없습니다.

  변수: *cap_id*는 audit 명령에 지정된 대로 나타납니다.
  - `acs, lsm, cap`: `audit` 명령에 CAP가 명시적으로 지정된 경우
  - *acs*, *lsm*, *: `audit` 명령에 CAP가 *acs*, *lsm*, *로 지정된 경우
  - *acs*, *, *: `audit` 명령에 CAP가 *acs*, * 또는 *로 지정된 경우

cancel

`cancel` 명령은 현재 또는 보류 중인 요청을 취소합니다.

형식

```
cancel request_id
```

옵션

- `request_id`

  취소할 요청의 식별자를 지정합니다.

  `cancel` 명령을 사용하면 `audit, define pool, delete pool, eject, enter, lock, query, set` 또는 `venter` 명령이나 클라이언트 응용 프로그램에서 실행한 현재

또는 보류 중인 요청을 취소할 수 있습니다. query request 명령을 사용하면 취소할 요청의 ID를 표시할 수 있습니다.

서버, ACS 또는 LSM에 대한 감사를 취소(*cancel*)할 수 있습니다. ACSLS는 내부적으로 서버, ACS 또는 LSM 감사를 일련의 패널 감사로 변환하므로 ACSLS는 감사의 나머지 부분을 취소하기 전에 현재 패널에 대한 *audit*를 완료합니다. 패널 또는 하위 패널에 대한 감사는 취소(*cancel*)할 수 없습니다. 감사를 취소(*cancel*)하면 이미 꺼낸 카트리지는 다시 넣지 않습니다.

주의:

감사를 취소하거나 감사 중 라이브러리 또는 ACSLS 하드웨어/소프트웨어 오류가 발생한 경우 동일한 감사를 다시 실행해야 합니다. 꺼내도록 표시되었지만 첫번째 감사 중 실제로 꺼내지 않은 카트리지는 더 이상 데이터베이스에 없으며 ACSLS의 제어를 받지 않습니다.

힌트: 취소할 요청을 실행한 *cmd_proc*가 아닌 다른 *cmd_proc*에서 *cancel* 명령을 입력하십시오.

cancel 명령은 보류 중인 모든 요청을 즉시 취소하고 다음과 같이 현재 요청을 처리합니다.

- *audit*

  ACSLS는 내부적으로 서버, ACS 또는 LSM 감사를 일련의 패널 감사로 변환하므로 ACSLS는 감사의 나머지 부분을 취소하기 전에 현재 패널에 대한 감사를 완료합니다.

  주:

  감사를 취소(*cancel*)하거나 감사 중 라이브러리 또는 ACSLS 하드웨어/소프트웨어 오류가 발생한 경우 동일한 감사를 다시 실행해야 합니다. 꺼내도록 표시되었지만 첫번째 감사 중 실제로 꺼내지 않은 카트리지는 더 이상 데이터베이스에 없으며 ACSLS의 제어를 받지 않습니다.

- *define pool*

  ACSLS가 스크래치 풀 정의를 중지하지만 이미 정의된 스크래치 풀은 삭제하지 않습니다.

- *delete pool*

  ACSLS가 스크래치 풀 삭제를 중지하지만 이미 삭제된 스크래치 풀은 재정의하지 않습니다.

- *eject*

  ACSLS가 꺼내기를 중지하고 *cmd_proc*에 이미 꺼냈지만 다시 넣지 않은 모든 카트리지를 제거하라는 메시지가 표시됩니다.

  카트리지가 제거되고 CAP가 닫히고 ACSLS가 CAP 비어 있음을 확인할 때까지 꺼내기가 종료되지 않습니다.

- *enter*

  ACSLS가 *enter*를 중지합니다. CAP에 카트리지가 남아 있으면 *cmd_proc*에 해당 카트리지를 제거하라는 메시지가 표시됩니다. 이미 LSM에 넣은 카트리지는 꺼내지 않습니다.

카트리지가 제거되고 CAP가 닫히고 ACSLS가 CAP 비어 있음을 확인할 때까지 *enter*가 종료되지 않습니다.

주:

Automatic enter. *cancel* 명령을 사용하여 진행 중인 자동 넣기 작업을 취소(*cancel*)할 수 없습니다. 진행 중인 자동 넣기를 종료하려면 다음을 수행합니다.

- CAP 도어가 열린 경우 모든 카트리지를 제거하고 도어를 닫습니다.
- CAP 도어가 닫혀 있고 카트리지를 라이브러리로 이동 중인 경우 남아 있는 카트리지를 라이브러리에 넣도록 허용합니다. 그러면 *enter* 작업이 종료됩니다.

- *lock*

  지정된 요청에 의한 리소스 잠금이 중지되었습니다. 요청에서 지정된 모든 리소스를 아직 확보하지 못한 경우 어떤 리소스도 잠기지 않습니다.

- *query*

  ACSLS가 질의를 취소합니다.

- *set*

  *set cap*의 경우 ACSLS가 CAP 속성 설정을 중지하지만 이미 설정된 속성은 변경하지 않습니다.

  set scratch 또는 *set clean* 요청의 경우 ACSLS가 스크래치 카트리지 또는 청소 카트리지 속성 설정을 중지하지만 이미 설정된 속성은 변경하지 않습니다.

- *venter*

  ACSLS가 넣기를 중지(*stop*)합니다. CAP에 카트리지가 남아 있으면 *cmd_proc*에 해당 카트리지를 제거하라는 메시지가 표시됩니다. 이미 LSM에 넣은 카트리지는 꺼내지 않습니다.

예

- 현재 및 보류 중인 모든 요청의 요청 ID를 표시하려는 경우

  ```
  query request all
  ```

  *query request all*의 출력 예제

  ```
  Identifier      Command      Status

  13              enter        Current

  15              query        Pending
  ```

- 위의 예에서 요청 13(현재 넣기 요청)을 취소하려는 경우

  ```
  cancel 13
  ```

참조:

찾을 정보	참조 항목
라이브러리 카트리지의 실제 인벤토리와 일치하도록 ACSLS 데이터베이스 업데이트	“audit ”
스크래치 풀 만들기 또는 수정	“define pool ”
빈 스크래치 풀 삭제	“delete pool ”
라이브러리에서 카트리지 꺼내기	“eject ”
CAP(수동 모드)가 레이블이 지정된 카트리지를 라이브러리에 넣을 수 있도록 준비	“enter ”
현재 잠금 ID로 드라이브 및 카트리지 잠금(전용으로 지정)	“lock ”
라이브러리 구성 요소의 상태 표시	“query 명령 ”
여러 라이브러리 구성 요소의 다양한 속성 설정	“set 명령 ”
CAP가 레이블이 없는 카트리지를 라이브러리에 넣을 수 있도록 준비	“venter ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

- *`Request request_id canceled.`*

  설명: ACSLS가 요청된 명령을 취소했습니다.

  변수: *`request_id`*는 취소된 명령의 요청 식별자입니다.

중간 메시지

없음

오류 메시지

- *`Request request_id can not be canceled: status.`*

  설명: ACSLS가 지정된 명령을 취소할 수 없습니다.

  변수:
  - *`request_id`*는 ACSLS가 취소할 수 없는 명령의 요청 식별자입니다.
  - *`status`*는 다음 중 하나입니다.
- *`Request identifier request_id invalid.`*

  `cancel` 명령에 잘못된 요청 식별자가 지정되었습니다.
- *`Request identifier request_id not found.`*

  `cancel` 명령에 현재 또는 보류 중이 아닌 요청에 대한 요청 식별자가 지정되었습니다.

표시 영역 메시지

없음

clear lock

clear lock 명령은 지정된 드라이브 또는 카트리지의 모든 활성 및 보류 중인 잠금을 제거합니다.

형식

clear lock type identifier

옵션

type identifier

라이브러리 구성 요소를 지정합니다. 다음 표에는 리소스 잠금을 지울 수 있는 구성 요소가 나와 있습니다.

표 13.3. Clear Lock에 유효한 구성 요소

라이브러리 구성 요소	유형	식별자
드라이브	드라이브	drive_id
볼륨	볼륨	vol_id

사용법

clear lock 명령을 사용하면 지정된 드라이브 또는 카트리지의 모든 활성 및 보류 중인 잠금을 제거할 수 있습니다. 현재 잠금 ID가 0이거나 드라이버 또는 카트리지의 잠금 ID와 일치해야 합니다.

unlock 명령은 드라이브 또는 카트리지의 활성 잠금만 제거합니다. 하지만 *unlock* 명령을 사용하여 모든 드라이브 또는 모든 카트리지의 활성 잠금을 제거할 수 있습니다.

주:

clear lock 명령은 항상 잠금 ID를 0으로 재설정합니다.

예

- 드라이브 1,1,5,2에 대한 모든 잠금을 지우려는 경우

  clear lock drive 1,1,5,2
- 볼륨 NN0108에 대한 모든 잠금을 지우려는 경우

  clear lock volume NN0108

주:

clear lock 명령은 취소(*cancel*)할 수 없습니다.

참조:

찾을 정보	참조 항목
드라이브 및 카트리지 잠금	“lock ”
드라이브 또는 카트리지의 잠금 상태 표시	“query lock ”
잠금 ID 설정	“set lock ”
잠금 또는 사용자 ID 표시	“show ”
드라이브 또는 카트리지의 활성 잠금 제거	“unlock ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

- Clear 요청이 성공하면 다음 메시지가 나타납니다.

  Clear: Completed, Success.
- 또한 요청의 각 식별자에 대해 라이브러리 구성 요소(유형)에 따라 다음 메시지 중 하나가 나타납니다.
  - *Clear: Drive drive_id all locks cleared.*
  - *Clear: Volume vol_id all locks cleared.*

중간 메시지

없음

오류 메시지

- *Clear: Clear lock of drive drive_id failed,*

  Drive identifier drive_id available.

  설명: 지정된 드라이브가 잠겨 있지 않아서 ACSLS가 잠금을 지울 수 없습니다.

  변수: *drive_id*는 지정된 드라이브의 식별자입니다.
- *Clear: Clear lock of volume vol_id failed,*

  Volume identifier vol_id available.

  설명: 지정된 볼륨이 잠겨 있지 않아서 ACSLS가 잠금을 지울 수 없습니다.

  변수: *vol_id*는 지정된 볼륨의 식별자입니다.

표시 영역 메시지

없음

define pool

define pool 명령은 스크래치 풀을 만들거나 수정합니다.

형식

define pool low_water_mark high_water_mark pool_id...[overflow]

옵션

- *low_water_mark*

  볼륨 경고 하위 임계값입니다. 스크래치 카트리지 수가 이 임계값 아래로 떨어지면 ACSLS가 이벤트 로그에 경고 메시지를 기록합니다. 유효한 값은 0 ~ 2^{31}-1입니다. 기본값은 0입니다.
- *high_water_mark*

  볼륨 경고 상위 임계값입니다. 스크래치 카트리지 수가 이 임계값에 도달하거나 초과하면 ACSLS가 이벤트 로그에 경고 메시지를 기록합니다. 이 값은 *low_water_mark* 값보다 커야 합니다.
- *pool_id*

  풀 식별자를 지정합니다. 풀 0은 공통 스크래치 풀이며 항상 존재합니다. 공통 스크래치 풀 속성을 수정할 수 있습니다.
- *overflow*

  이 풀이 *mount scratch* 요청을 충족할 수 없는 경우 지정합니다. ACSLS는 공통 풀(풀 0)에서 카트리지를 선택합니다.

사용법

define pool 명령을 사용하면 스크래치 풀을 만들거나 수정할 수 있습니다.

예

- 새 풀 1에 대해 하위 임계값 0, 상위 임계값 600 및 오버플로우를 정의하려는 경우

  ```
  define pool 0 600 1 overflow
  ```
- 기존 풀 5에 대해 하위 임계값 0, 상위 임계값 600을 정의하고 오버플로우를 정의하지 않으려는 경우

  ```
  define pool 0 600 5
  ```

주:

스크래치 풀은 한 클라이언트 응용 프로그램 또는 사용자 ID가 소유하지 않습니다. 하지만 볼륨 액세스 제어를 사용하여 특정 스크래치 카트리지에 대한 액세스를 제한할 수 있습니다.

참조:

찾을 정보	참조 항목
명령 취소	“cancel ”
빈 스크래치 풀 삭제	“delete pool ”
스크래치 카트리지 관리 지침 및 절차	“LSM 채우기 ”
드라이브에 스크래치 카트리지 마운트	“mount * ”
스크래치 풀 속성 표시	“query pool ”
카트리지 스크래치 속성 설정 또는 지우기	“set scratch ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

- *`define pool`* 명령이 성공하면 다음 메시지가 나타납니다.

  `Define: Define completed, Success.`
- 만들거나 수정된 각 풀에 대해 다음 메시지가 나타납니다.

  `Define: Pool pool_id created.`

중간 메시지

없음

오류 메시지

없음

표시 영역 메시지

- *`Pool pool_id: low water mark warning.`*

  설명: 지정된 스크래치 풀의 볼륨 수가 볼륨 하위 임계값보다 작거나 같습니다.

  변수: *`low_water_mark`*는 지정된 스크래치 풀의 볼륨 하위 임계값입니다.
- *`Pool pool_id: high water mark warning.`*

  설명: 지정된 스크래치 풀의 카트리지 수가 볼륨 상위 임계값보다 크거나 같습니다.

  변수: *`high_water_mark`*는 지정된 스크래치 풀의 볼륨 상위 임계값입니다.

delete pool

delete pool 명령은 빈 스크래치 풀을 삭제합니다.

형식

delete pool pool_id...|all

옵션

- *pool_id*

  풀 ID를 지정합니다.

  풀 0은 공통 풀이며 삭제할 수 없습니다.
- *all*

  빈 스크래치 풀을 모두 지정합니다.

사용법

delete pool 명령을 사용하면 빈 스크래치 풀을 삭제할 수 있습니다. 풀에 스크래치 카트리지가 포함된 경우 첫번째 풀을 삭제하기 전에 해당 카트리지를 다른 풀에 다시 지정해야 합니다. 스크래치 카트리지를 마운트하면 데이터 카트리지가 되지만 해당 스크래치 풀에 남아 있습니다. *set scratch off* 명령을 사용하면 데이터 카트리지를 공통 풀에 다시 지정할 수 있습니다.

예

- 모든 빈 스크래치 풀을 삭제하려는 경우

  delete pool all

빈 풀만 삭제되며 카트리지가 지정된 풀은 영향을 받지 않습니다.

- 스크래치 풀 1을 삭제하려면 아래 절차를 따르십시오.
  a. *Query scratch pool 1*:

     query scratch 1

     ```
     1998-0630>09:35:30>Scratch Status
     Scratch Pool>  Identifier>    Homer location>  Status>    Type
     1)    34813>    0,0,1,8,8>     home>     3480            3480
     1)    34815>    0,0,1,8,1>     home>     3480            3480
     ```

     풀 1에 34813과 34815라는 카트리지가 두 개 있습니다.
  b. 풀 1의 카트리지를 풀 5에 다시 지정하여 풀 1을 비웁니다.

```
set scratch 5 348013 348015
```

c. 풀 1을 삭제합니다.

```
delete pool 1
```

참조:

찾을 정보	참조 항목
명령 취소	“cancel ”
스크래치 풀 만들기 또는 수정	“define pool ”
스크래치 카트리지 관리 지침 및 절차	“LSM 채우기 ”
스크래치 풀 속성 표시	“query pool ”
카트리지 위치 및 매체 유형 표시	“query volume ”
카트리지 스크래치 속성 설정 또는 지우기	“set scratch ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

- *delete pool* 명령이 성공적으로 완료되면 다음 메시지가 나타납니다.

  ```
  Delete: Delete completed, Success.
  ```

- 삭제된 각 풀에 대해 다음 메시지가 나타납니다.

  ```
  Delete: Pool pool_id deleted.
  ```

중간 메시지

없음

오류 메시지

- *Delete: Pool pool_id failed, Pool not empty.*

  설명: 스크래치 풀이 비어 있지 않아서 ACSLS가 스크래치 풀을 삭제할 수 없습니다.

  변수: *pool_id*는 요청된 풀의 식별자입니다.

dismount

dismount 명령은 카트리지를 드라이브에서 마운트 해제합니다.

형식

```
dismount vol_id drive_id [force]
```

옵션

- *vol_id*

  카트리지를 지정합니다.
- *drive_id*

  드라이브를 지정합니다.
- *force*

  드라이브의 카트리지 *vol_id*가 지정된 *vol_id*와 일치하지 않아도 지정된 드라이브에 있는 실제 볼륨의 마운트 해제를 강제 실행합니다.

  또한 이 옵션은 드라이브가 언로드되지 않은 경우에도 마운트 해제를 강제 실행합니다.

사용법

dismount 명령을 사용하면 볼륨을 드라이브에서 마운트 해제하고 카트리지를 사용 가능한 스토리지 셀에 넣을 수 있습니다.

dismount

force 옵션 없이 *dismount* 명령을 사용하면 지정된 카트리지를 지정된 드라이브에서 마운트 해제할 수 있습니다. 비강제 마운트 해제를 수행하려면 다음 사항을 충족해야 합니다.

- 드라이브가 온라인 상태여야 합니다.
- 드라이브의 카트리지 *vol_id*가 *dismount* 명령에 지정하는 *vol_id*와 일치해야 합니다.
- 드라이브를 언로드해야 합니다.

주의:

일반 마운트 해제 시에는 SL500 및 SL150 라이브러리가 드라이브에 로드된 카트리지를 자동으로 되감고 언로드합니다. 카트리지를 되감고 언로드하기 위해 dismount force를 실행할 필요가 없습니다. 이 라이브러리에 대해 dismount 명령을 실행하기 전에 ACSLS 클라이언트 응용 프로그램이 테이프 드라이브에서 읽거나 쓰지 않는지 확인하십시오.

Dismount force

dismount 명령을 force 옵션과 함께 사용하면 지정된 드라이브에 마운트된 실제 카트리지에 대해 마운트 해제를 강제 실행할 수 있습니다. 드라이브의 카트리지 vol_id가 dismount 명령에 지정하는 vol_id와 일치할 필요는 없습니다. 또한 카트리지가 마운트 해제할 수 있도록 준비되지 않은 경우 ACSLS는 드라이브가 카트리지를 자동으로 되감고 언로드하고 마운트 해제하도록 강제 실행합니다. 드라이브가 온라인 상태여야 합니다.

*dismount force*는 드라이브가 응답을 받은 적이 없는 마운트 또는 마운트 해제 작업에서 예약한 것으로 표시된 상태여도 계속 진행됩니다. 예약된 드라이브는 *cmd_proc* 또는 ACSAPI 클라이언트에 사용 중인 것으로 보고됩니다.

force 옵션을 사용하면 레이블을 읽을 수 없거나 알 수 없는 레이블 또는 클라이언트 응용 프로그램이 마운트 해제하지 않은 카트리지를 마운트 해제할 수 있습니다. ACSLS는 레이블을 읽을 수 없거나 레이블이 누락된 경우에도 카트리지를 사용 가능한 스토리지 셀에 반환합니다.

주:

라이브러리가 카트리지를 되감고 업로드하기 전에 드라이브와 현재 드라이브를 사용 중인 모든 응용 프로그램 간의 모든 읽기/쓰기 작업을 중지하거나 일시 중지해야 합니다. 읽기/쓰기 작업이 중단 없이 계속되면 라이브러리 대기 시간 초과 후 dismount force가 실패합니다.

이 고려 사항은 데이터 경로를 통해 호스트에서 드라이브로 요청된 매체 검증에도 적용됩니다. 매체 검증이 완료되거나 호스트가 데이터 경로를 통해 매체 검증을 중지할 때까지 ACSLS의 dismount force는 실패합니다.

예

- 카트리지 EDU200을 드라이브 0,1,10,2에서 마운트 해제하려는 경우

```
dismount EDU200 0,1,10,2
```

참조:

찾을 정보	참조 항목
명령 취소	“cancel ”
라이브러리에서 카트리지 꺼내기	“eject ”
CAP(수동 모드)가 레이블이 지정된 카트리지를 라이브러리에 넣을 수 있도록 준비	“enter ”
드라이브에 스크래치 카트리지 마운트	“mount * ”
드라이브에 데이터 카트리지 마운트	“mount ”
CAP 상태 표시	“query cap ”
드라이브 상태 표시	“query drive ”
CAP 모드(수동 또는 자동) 설정	“set cap mode ”
CAP 선택 우선 순위 설정	“set cap priority ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

- *Dismount: vol_id dismounted from drive_id.*

  설명: 강제 마운트 해제가 성공했습니다.

  변수:

  ◦ *vol_id*는 마운트 해제된 카트리지의 식별자입니다.

표시된 *vol_id*는 실제로 마운트 해제된 볼륨이며, dismount force 명령에 지정된 볼륨일 필요는 없습니다.

- *drive_id*는 지정된 드라이브의 식별자입니다.

- *Dismount: Forced dismount of vol_id from drive_id.*

  설명: 강제 마운트 해제가 성공했습니다.

  변수:

  - *vol_id*는 마운트 해제된 카트리지의 식별자입니다.

    표시된 *vol_id*는 실제로 마운트 해제된 볼륨이며, *dismount force* 명령에 지정된 볼륨일 필요는 없습니다.
  - *drive_id*는 지정된 드라이브의 식별자입니다.

중간 메시지

없음

오류 메시지

- *Dismount: Dismount failed, ACS acs_id full.*

  설명: ACS에 빈 스토리지 셀이 없어서 ACSLS가 카트리지를 마운트 해제할 수 없습니다. ACS에서 다른 카트리지를 하나 이상 꺼내면 ACSLS가 드라이브에서 카트리지를 마운트 해제할 수 있습니다.

  변수: *acs_id*는 카트리지가 포함된 ACS의 식별자입니다.

  > **주:**
  >
  > 사용 가능한 셀을 찾는 데 온라인 LSM만 사용됩니다. 따라서 빈 스토리지 셀이 있는 LSM이 오프라인인 경우에도 이 메시지가 발생할 수 있습니다.

- *Dismount: Dismount failed, Audit in progress.*

  설명: 진행 중인 감사로 인해 ACS의 마지막 빈 셀 위치에 대한 액세스가 잠겨서 ACSLS가 카트리지를 마운트 해제할 수 없습니다.

- *Dismount: Dismount failed, Cartridge in drive drive_id, unreadable label*

  설명: 카트리지에 외부 레이블이 없거나 외부 레이블을 읽을 수 없거나 가상 레이블이 없어서 ACSLS가 카트리지를 마운트 해제할 수 없습니다. forced dismount를 사용하여 카트리지를 마운트 해제하십시오.

  변수: *drive_id*는 지정된 드라이브의 식별자입니다.

- *Dismount: Dismount failed, Drive identifier drive_id available.*

  설명: 지정된 드라이브에 카트리지가 마운트되지 않았습니다.

변수: *drive_id*는 지정된 드라이브의 식별자입니다.

- *Dismount: Dismount failed, Drive identifier drive_id in use.*

  설명: 카트리지가 되감기 및 언로드되지 않아서 ACSLS가 카트리지를 마운트 해제할 수 없습니다. 클라이언트 응용 프로그램이 카트리지를 되감고 언로드할 때까지 기다린 다음 *dismount* 명령을 입력하거나 *forced dismount*를 사용하여 카트리지를 마운트 해제하십시오.

  변수: *drive_id*는 지정된 드라이브의 식별자입니다.

- *Dismount: Dismount failed, Misplaced tape.*

  설명: 카트리지의 외부 레이블이 ACSLS 데이터베이스의 카트리지 식별자와 일치하지 않아서 ACSLS가 카트리지를 마운트 해제할 수 없습니다. 카트리지 외부 레이블과 일치하도록 ACSLS 데이터베이스의 카트리지 식별자가 업데이트됩니다. 마운트 해제를 재시도하십시오.

- *Dismount: Dismount failed, Cartridge not in drive.*

  설명: 카트리지의 외부 레이블이 dismount에 지정된 카트리지 식별자와 일치하지 않아서 ACSLS가 카트리지를 마운트 해제할 수 없습니다. 올바른 카트리지 식별자를 사용하여 *dismount* 명령을 다시 입력하십시오.

표시 영역 메시지

없음

eject

eject 명령은 LSM 내부에서 카트리지를 꺼내 운영자가 제거할 수 있는 CAP에 넣도록 로봇에 지시합니다.

lsm_id 옵션을 선택하면 단일 LSM에서 CAP 여러 개를 사용하여 카트리지를 꺼낼 수 있습니다.

형식

eject cap_id|*lsm_id* [*opmsg opmsg_nbr*] *vol_id*|*volrange...*

옵션

- *cap_id*

  카트리지를 꺼내는 데 사용되는 CAP를 지정합니다.

- *lsm_id*

  lsm_id 옵션을 선택하면 단일 LSM에 제공되는 CAP 여러 개를 사용하여 카트리지를 꺼낼 수 있습니다. 요구 사항(예: 수동, 우선 순위가 0이 아님, 사용 가능)을 충족하는 모든

CAP가 카트리지 꺼내기를 위해 잠금 해제됩니다. 어떤 순서로든 선택한 CAP 중 하나 또는 모두를 통해 카트리지를 꺼낼 수 있습니다. *eject*는 처음에는 우선 순위가 높은 CAP부터 카트리지를 채웁니다.

예: 우선 순위가 2인 CAP와 우선 순위가 5인 CAP가 있습니다. 이 경우 *eject*는 우선 순위 상태가 5인 CAP를 먼저 채운 다음 우선 순위 상태가 2인 CAP를 채웁니다. CAP 하나에서 꺼낼 수 있을 정도의 카트리지만 있는 경우에는 우선 순위가 5인 CAP가 채워집니다.

- *opmsg opmsg_nbr*

  *cmd_proc*를 사용하여 입력한 SL8500 대량 CAP eject 명령에 사용자 정의 운영자 패널 메시지 번호를 지정할 수 있습니다. 대량 CAP가 잠금 해제되어 꺼내고 있는 카트리지를 제거할 수 있게 되면 메시지가 표시됩니다.

  유효한 메시지 번호는 4 ~ 99입니다.
  - 현재는 SL8500 라이브러리의 대량 CAP에 대해서만 사용자 정의 *opmsg*가 표시됩니다.
  - *opmsg* 매개변수는 선택사항입니다. 이 매개변수를 지정하지 않으면 카트리지를 제거하라는 기본 메시지가 전송됩니다.
  - 사용자 정의 *opmsg* 번호는 ACSAPI 클라이언트, ACSLS GUI 또는 *lib_cmd eject*에서의 꺼내기에 대해 지정할 수 없습니다. 이 경우 기본 메시지가 표시됩니다.

  opmsg 번호에 대해 표시될 메시지를 만들려면 SLConsole을 사용하고 다음을 선택합니다.

  ```
  Tools
     Configuration
        CAP Usage Message
  ```

  *opmsg*를 보여주는 SLConsole CAP 상태 표시를 보려면 다음을 선택합니다.

  ```
  Tools
     System Detail
        CAP Folder
             Status
  ```

  꺼내는 동안 ACSLS에서 보낸 메시지 번호에 따라 대량 CAP 메시지가 SLConsole에 표시됩니다. 제거하기 위해 꺼내는 중인 카트리지에 대해 CAP가 잠금 해제된 후 *System Details CAP Status* 페이지에 메시지가 표시됩니다.

  예: 대량 CAP 1,2,1을 통해 카트리지를 꺼낼 때 사용자 정의 운영자 패널 메시지 번호 55를 지정하려는 경우

  ```
  eject 1,2,1 opmsg 55 T10001 T10033-T10067
  ```

- *vol_id* | *volrange*

꺼낼 카트리지 또는 카트리지 범위의 외부 또는 가상 레이블 유형을 지정합니다.

사용법

eject 명령을 사용하면 라이브러리에서 카트리지를 꺼낼 수 있습니다. 이 경우 카트리지가 ACSLS 제어에서 제거됩니다. 로봇이 지정된 카트리지를 지정된 CAP에 넣습니다. 그런 다음 ACSLS가 카트리지 보관 셀 위치를 비웁니다. 카트리지 정보는 *ABSENT_VOLUME_RETENTION_PERIOD* 변수의 값이 0이 아니면 유지되고 *ABSENT_VOLUME_RETENTION_PERIOD*가 0으로 설정되면 삭제됩니다. *eject* 명령에 카트리지로 가득 찬 CAP를 두 개 이상 지정한 경우, CAP가 차면 CAP를 비우고 CAP를 닫은 다음 모든 카트리지를 꺼낼 때까지 꺼내기를 계속하십시오.

단일 *eject* 명령에 카트리지 ID 여러 개를 공백으로 구분하여 지정하면 범위에 없는 여러 개의 카트리지를 꺼낼 수 있습니다.

라이브러리에 연결된 LSM이 두 개 있고 PTP가 작동 중지된 경우 카트리지를 성공적으로 꺼내려면 다음 중 하나를 수행합니다.

- 카트리지가 보관된 LSM에 CAP를 지정합니다. 예를 들어 카트리지 NN0100이 LSM 0,0에 보관된 경우 CAP 0,0,0을 통해 이 카트리지를 꺼내려면 다음과 같이 합니다.

  ```
  eject 0,0,0 NN0101
  ```

- *cap_id*의 경우 카트리지가 보관되는 LSM을 지정하되 CAP 번호에 와일드카드(*)를 사용합니다. ACSLS는 LSM에서 우선 순위가 가장 높은 CAP를 선택합니다. 예를 들어 카트리지 NN0114가 LSM 0,0에 보관된 경우 LSM 0,0의 우선 순위가 가장 높은 CAP를 통해 이 카트리지를 꺼내려면 다음과 같이 합니다.

  ```
  eject 0,0,* NN0114
  ```

예

- CAP 0,0,0을 통해 카트리지 NN0101을 꺼내려는 경우

  ```
  eject 0,0,0 NN0101
  ```

- ACS 0의 우선 순위가 가장 높은 CAP를 통해 카트리지 범위 NN0101-NN0109를 꺼내려는 경우

  ```
  eject 0,* NN0101-NN0109
  ```

- ACS 0의 우선 순위가 가장 높은 CAP를 통해 카트리지 NN0101, NN0103, NN0105 및 NN0107을 꺼내려는 경우

  ```
  eject 0,* NN0101 NN0103 NN0105 NN0107
  ```

- LSM 1,2의 카트리지를 꺼낼 때 우선 순위가 0이 아닌 CAP를 여러 개 사용하려는 경우

  ```
  eject 1,2 RB1000-RB2000
  ```

주:

CAP에서 꺼낸 모든 카트리지를 제거해야 합니다. 다른 작업(예: 넣기 또는 감사)에 CAP를 사용하려면 먼저 꺼낸 카트리지를 모두 언로드한 다음 CAP 도어를 닫아 꺼내기를 완료해야 합니다.

참조:

찾을 정보	참조 항목
명령 취소	“cancel ”
드라이브에서 카트리지 마운트 해제	“dismount ”
카트리지 꺼내기 지침 및 절차	“LSM 채우기 ”
CAP(수동 모드)가 레이블이 지정된 카트리지를 라이브러리에 넣을 수 있도록 준비	“enter ”
CAP 상태 표시	“query cap ”
드라이브 상태 표시	“query drive ”
카트리지 위치 및 매체 유형 표시	“query volume ”
CAP 모드(수동 또는 자동) 설정	“set cap mode ”
CAP 선택 우선 순위 설정	“set cap priority ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

- *`Eject: Eject complete, a cartridges ejected`*
- *`Eject: vol_id ejected from cap_id`*

  설명: ACSLS가 지정된 카트리지를 꺼냈습니다.

  변수:
  - *`nn`*은 꺼낸 카트리지의 수입니다.
  - *`vol_id`*는 꺼낸 카트리지의 카트리지 식별자입니다.
  - *`cap_id`*는 꺼낸 카트리지가 포함된 CAP입니다.

중간 메시지

없음

오류 메시지

- *`Eject: vol_id Eject failed, CAP cap_id full.`*

  설명: CAP가 꽉 차 있어서 카트리지를 꺼내지 못했습니다.

  변수:

- *vol_id*는 꺼내지 않은 카트리지의 카트리지 식별자입니다.
- *cap_id*는 꺼내도록 지정된 CAP입니다.

- *Eject: vol_id Eject failed, CAP cap_id in use.*

  설명: CAP가 사용 중이어서 카트리지를 꺼내지 못했습니다.

  변수:
  - *vol_id*는 꺼내지 않은 카트리지의 카트리지 식별자입니다.
  - *cap_id*는 꺼내도록 지정된 CAP입니다.
- *Eject: vol_id Eject failed, Misplaced tape.*

  설명: 카트리지의 외부 레이블이 스토리지 셀에 대한 데이터베이스의 카트리지 식별자와 일치하지 않아서 ACSLS가 카트리지를 꺼낼 수 없습니다. 데이터베이스가 스토리지 셀의 카트리지에 대한 카트리지 식별자, 매체 유형 및 청소 카트리지 속성을 업데이트합니다.

  변수: *vol_id*는 꺼내지 않은 카트리지의 카트리지 식별자입니다.
- *Eject: vol_id Eject failed, Not in same ACS.*

  설명: 카트리지가 *cap_id*에 지정된 ACS에 없어서 꺼내지 못했습니다.

  변수: *vol_id*는 꺼내지 않은 카트리지의 카트리지 식별자입니다.
- *Eject: vol_id Eject failed, Volume identifier vol_id not found.*

  설명: 카트리지가 다음과 같은 상태여서 꺼내지 못했습니다.
  - 데이터베이스에 지정된 스토리지 셀에 없음
  - 이동 중이 아님 또는
  - 드라이브에 없음
- *ACSLS deletes the volume entry from the database.*

  변수: *vol_id*는 데이터베이스에서 삭제된 카트리지 식별자입니다.
- *Eject: vol_id Eject failed, Cartridge in drive.*

  설명: 카트리지가 드라이브에 마운트되어서 꺼내지 못했습니다.

  변수: *vol_id*는 꺼내지 않은 카트리지의 카트리지 식별자입니다.
- *Eject: vol_id Eject failed, Volume vol_id in use.*

  설명: 카트리지가 다른 요청에서 사용하도록 예약되어 있어서 꺼내지 못했습니다.

  변수: *vol_id*는 꺼내지 않은 카트리지의 카트리지 식별자입니다.

표시 영역 메시지

- *CAP cap_id Remove cartridges from CAP.*

  설명: CAP가 꽉 찼거나 요청된 모든 카트리지가 CAP에 있습니다. CAP를 비우십시오.

변수: *cap_id*는 꺼낸 카트리지가 포함된 CAP입니다.

- *CAP cap_id Place magazines in CAP.*

  설명: 꺼내려면 CAP에 매거진이 필요합니다. CAP를 열고 매거진을 안에 넣은 다음 CAP를 닫으십시오.

  변수: *cap_id*는 매거진이 필요한 CAP입니다.

enter

이 명령을 사용하면 CAP가 수동 또는 자동 모드에서 작동하도록 설정할 수 있습니다. CAP를 자동 모드 또는 수동 모드로 설정하는 절차는 "카트리지 넣기 " 절을 참조하십시오.

- 자동 모드

  CAP가 자동 모드에 있으면 enter 명령을 실행하지 않고도 넣기 작업을 시작할 수 있습니다. CAP 도어를 열고 하나 이상의 카트리지를 안에 넣은 다음 CAP를 닫으면 됩니다. 넣기가 진행되는 동안 CAP가 잠기고 넣기 작업이 완료되면 CAP가 잠금 해제됩니다.
- 수동 모드

  수동 모드에서는 CAP가 잠기며 CAP를 열고 카트리지를 넣기 전에 다음 명령을 실행해야만 사용할 수 있습니다.

  enter cap_id [*opmsg opmsg_nbr*]

옵션

- *cap_id*

  CAP를 지정합니다. 별표(*)로 와일드카드를 지정하여 LSM에서 우선 순위가 0이 아닌 가장 높은 CAP를 선택할 수 있습니다(예: 1,1,* 또는 ACS, 0,*,*).
- *opmsg opmsg_nbr*

  *cmd_proc*를 사용하여 입력한 SL8500 대량 CAP enter 명령에 사용자 정의 운영자 패널 메시지 번호를 지정할 수 있습니다. 넣고 있는 카트리지에 대해 대량 CAP가 잠금 해제되면 메시지가 표시됩니다.

  유효한 메시지 번호는 4 ~ 99입니다.
  - 현재는 SL8500 라이브러리의 대량 CAP에 대해서만 사용자 정의 *opmsg*가 표시됩니다.
  - *opmsg* 매개변수는 선택사항입니다. 이 매개변수를 지정하지 않으면 카트리지를 넣으라는 기본 메시지가 전송됩니다.
  - 넣기가 시작되기 전에 카트리지가 CAP에 남아 있거나 읽을 수 없는 카트리지 또는 중복 카트리지를 넣을 수 없는 경우 해당 카트리지를 제거하라는 메시지가 표시됩니다. 넣기에 대해 *opmsg* 매개변수가 지정된 경우에도 이 메시지가 발생할 수 있습니다.

◦ ACSAPI 클라이언트 또는 ACSLS GUI의 넣기에 사용자 정의 *opmsg* 번호를 지정할 수 없습니다. 이 경우 기본 메시지가 표시됩니다.

opmsg 번호에 대해 표시될 메시지를 만들려면 SLConsole을 사용하고 다음을 선택합니다.

```
Tools
   Configuration
      CAP Usage Message
```

*opmsg*를 보여주는 SLConsole CAP 상태 표시를 보려면 다음을 선택합니다.

```
Tools
   System Detail
      CAP Folder
           Status
```

넣는 동안 ACSLS에서 보낸 메시지 번호에 따라 대량 CAP 메시지가 SLConsole에 표시됩니다. 넣고 있는 카트리지에 대해 CAP가 잠금 해제된 후 *System Details CAP Status* 페이지에 메시지가 표시됩니다.

예: 대량 CAP 1,3.0을 통해 카트리지를 넣을 때 사용자 정의 운영자 패널 메시지 번호 66을 지정하려는 경우

```
enter 1,3,0 opmsg 66
```

- *lsm_id*

  lsm_id 옵션을 선택하면 단일 LSM에 제공되는 CAP 여러 개를 사용하여 카트리지를 넣을 수 있습니다. 요구 사항(예: 수동, 우선 순위가 0이 아님, 사용 가능)을 충족하는 모든 CAP가 카트리지 넣기를 위해 잠금 해제됩니다. 어떤 순서로든 선택한 CAP 중 하나 또는 모두를 통해 카트리지를 넣을 수 있습니다. venter는 유효하지 않으며 CAP가 잘못됨 오류를 반환합니다.

사용법

enter 명령을 사용하면 수동 모드 CAP가 레이블이 지정된 카트리지를 넣을 수 있도록 준비할 수 있습니다.

예

- CAP 0,0,2가 카트리지를 넣을 수 있도록 준비하려는 경우

  enter 0,0,2
- LSM 0,0의 우선 순위가 0이 아닌 가장 높은 CAP가 카트리지를 넣을 수 있도록 준비하려는 경우

```
enter 0,0,*
```

- LSM 1,2에 카트리지를 넣기 위해 CAP를 여러 개 사용하려는 경우

```
enter 1,2
```

> **주:**
>
> LSM에 넣는 카트리지가 해당 LSM과 호환되어야 합니다. 예를 들어 L5500 LSM에는 T9840, T9940 및 LTO 카트리지만 넣을 수 있습니다.

참조:

찾을 정보	참조 항목
명령 취소	“cancel ”
스크래치 풀 만들기 또는 수정	“define pool ”
라이브러리에서 카트리지 꺼내기	“eject ”
카트리지 넣기 지침 및 절차	“카트리지 넣기 ”
드라이브에 스크래치 카트리지 마운트	“mount * ”
드라이브에 데이터 카트리지 마운트	“mount ”
CAP 상태 표시	“query cap ”
CAP 모드(수동 또는 자동) 설정	“set cap mode ”
CAP 선택 우선 순위 설정	“set cap priority ”
CAP가 레이블이 없는 카트리지를 라이브러리에 넣을 수 있도록 준비	“venter ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

레이블이 지정된 모든 카트리지를 넣은 후 다음 메시지가 나타납니다. 메시지의 두번째 라인은 카트리지를 성공적으로 넣을 때마다 반복됩니다.

- *Enter: Enter complete, nn volumes entered*
- *Enter: vol_id Entered through cap_id*
  - *nn*은 넣은 총 카트리지 수입니다.
  - *vol_id*는 넣은 카트리지의 카트리지 식별자입니다.
  - *cap_id*는 카트리지를 넣는 데 사용된 CAP입니다.

중간 메시지

없음

오류 메시지

- *Enter: vol_id Enter failed, ACS acs_id full.*

  설명: ACS에 빈 스토리지 셀이 없어서 카트리지를 넣지 못했습니다. ACSLS가 카트리지를 넣을 수 있도록 ACS에서 하나 이상의 카트리지를 꺼내야(*eject*) 합니다.

  변수:

  - *vol_id*는 넣지 않은 카트리지의 외부 레이블입니다.
  - *acs_id*는 빈 스토리지 셀이 없는 ACS의 식별자입니다.

  주:

  ACSLS는 온라인 LSM에서만 사용 가능한 셀을 검색합니다. 오프라인 상태의 LSMS에 사용 가능한 셀이 있는 경우에도 이 메시지가 발생할 수 있습니다.

- *Enter: vol_id Enter failed, Audit in progress.*

  설명: *audit*로 인해 넣기에 필요한 셀 위치에 대한 액세스가 잠겨서 카트리지를 넣지 못했습니다.

  변수: *vol_id*는 넣지 않은 카트리지의 외부 레이블입니다.

- *Enter: vol_id Enter failed, CAP cap_id in use.*

  설명: 지정된 CAP가 감사, 카트리지 꺼내기 또는 다른 넣기 프로세스에 사용 중이어서 카트리지를 넣지 못했습니다.

  변수:

  - *vol_id*는 넣지 않은 카트리지의 외부 레이블입니다.
  - *cap_id*는 사용 중인 CAP입니다.

- *Enter: vol_id Enter failed, Duplicate label.*

  설명: CAP에 있는 카트리지의 카트리지 식별자가 ACSLS 데이터베이스에 이미 있어서 카트리지를 넣지 못했습니다.

  변수: *vol_id*는 넣지 않은 카트리지의 외부 레이블입니다.

- *Enter: Enter failed, Unreadable label.*

  설명: 카트리지에 외부 레이블이 없거나 외부 레이블을 읽을 수 없어서 카트리지를 넣지 못했습니다.

- *Enter: vol_id Enter failed, Unknown media type label.*

  설명: 카트리지의 외부 레이블에 매체 식별자가 없어서 카트리지를 넣지 못했습니다.

  변수: *vol_id*는 넣지 않은 카트리지의 외부 레이블입니다.

표시 영역 메시지

- *CAP cap_id: Place cartridges in CAP.*

  설명: CAP가 카트리지를 넣을 준비가 되었습니다. CAP를 열고 카트리지 넣기(*enter*)를 수행하십시오.

  변수: *cap_id*는 카트리지를 넣는 데 사용된 CAP입니다.
- *CAP cap_id: Remove cartridges from CAP.*

  설명: 하나 이상의 카트리지를 넣을 수 없습니다.

  변수: *cap_id*는 카트리지를 넣는 데 사용된 CAP입니다. CAP를 열고 카트리지를 제거(*remove*)하십시오.
- *CAP cap_id: CAP cap_id Place magazines in CAP.*

  설명: CAP가 카트리지 넣기에 매거진을 사용합니다. 올바른 매거진에 카트리지를 로드하고 CAP를 연 다음 매거진을 삽입하십시오.

  변수: *cap_id*는 카트리지를 넣는 데 사용된 CAP입니다.

idle

idle 명령은 ACSLS가 새 요청을 처리하는 것을 중지합니다.

형식

- *idle [force]*

  idle 명령의 전체 명령 이름을 입력합니다. ACSLS는 명령의 다른 모든 형식(예: *i*, *id* 또는 *idl*)을 거부합니다.

옵션

- *force*

  새 요청 처리를 강제로 종료합니다.

사용법

idle 명령을 사용하면 ACSLS가 새 요청을 처리하는 것을 중지할 수 있습니다. 예를 들어 유지 관리 작업을 수행하기 전에 또는 ACSLS를 종료하기 전에 ACSLS를 유휴 상태로 지정(*idle*)할 수 있습니다.

> **주:**
>
> *start* 명령을 사용하면 요청 처리를 다시 시작할 수 있습니다.

- *idle*

force 옵션 없이 *idle* 명령을 입력하면 ACSLS가 유휴-보류 중 상태가 됩니다. ACSLS가 현재 및 보류 중인 요청(보류 중인 잠금 요청은 제외되며, 이러한 요청은 취소됨)을 완료하고 "참고"에 나열된 요청을 제외한 새 요청을 거부합니다. 그런 다음 ACSLS가 유휴 상태가 되고 요청 처리가 다시 시작될 때까지 후속 요청을 처리하지 않습니다.

- *Idle force*

  force 옵션과 함께 *idle* 명령을 입력하면 ACSLS가 유휴 상태가 됩니다. ACSLS가 모든 현재 및 보류 중인 요청을 취소하고 "예" 아래의 참고에 나열된 요청을 제외한 새 요청을 거부합니다. 요청 처리가 다시 시작될 때까지 ACSLS가 후속 요청을 처리하지 않습니다. ACSLS가 현재 요청을 완료하지 않았으므로 ACSLS를 강제로 유휴 상태로 전환하면 데이터베이스와 하드웨어가 불일치 상태가 될 수 있습니다. 이 상황을 해결하려면 *audit*를 수행해야 합니다.

힌트: *idle force* 입력 시 현재 요청 처리에 따라 이벤트 로그에 프로세스 실패가 보고될 수 있습니다. 이 경우 영향을 받은 LSM을 오프라인 상태로 전환(*vary*)한 다음 다시 온라인 상태로 전환하십시오. *vary* 명령에 대한 자세한 내용은 “vary ”를 참조하십시오.

예

- ACSLS를 유휴-보류 중 상태로 전환하려는 경우

  ```
  idle
  ```

- ACSLS를 유휴 상태로 강제 전환하려는 경우

  ```
  idle force
  ```

> **주:**
>
> *idle* 또는 *idle-pending* 상태에서는 ACSLS가 *cancel, idle, query, start* 및 *vary* 요청에 대한 새 요청을 수락합니다.

참조:

찾을 정보	참조 항목
라이브러리 구성 요소의 상태 표시	“query 명령 ”
ACSLS 요청 처리 시작	“start ”
라이브러리 구성 요소 상태 변경	“vary ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

ACSLS 요청 처리가 중지되면 다음 메시지가 나타납니다.

```
ACSLM Request Processing Stopped: Success.
```

중간 메시지

없음

오류 메시지

ACSLM Request Processing Stopped: status

설명: ACSLS가 요청 처리를 중지하지 않았습니다.

변수: status는 오류의 원인입니다. 일반 상태 메시지에 대한 자세한 내용은 ACSLS Messages를 참조하십시오.

표시 영역 메시지

변수:

- *Server system idle*

  설명: 요청 처리가 중지되었으며 ACSLS가 유휴 상태입니다.
- *Server system idle is pending*

  설명: ACSLS가 현재 또는 보류 중인 요청을 처리 중입니다. 유휴 상태가 보류 중입니다.

lock

lock 명령은 카트리지 또는 드라이브를 ACSLS가 지정하는 잠금 ID로 잠급니다.

형식

lock type identifier...[wait]

옵션

- *type identifier*

  라이브러리 구성 요소를 지정합니다. 다음 표에는 잠글 수 있는 구성 요소가 나와 있습니다. 단일 *lock* 명령에 드라이브 또는 볼륨 중 하나만 지정할 수 있습니다. 하지만 드라이브와 볼륨에 동일한 *lock ID*를 사용할 수 있습니다.

표 13.4. Lock에 유효한 구성 요소

라이브러리 구성 요소	유형	식별자
드라이브	*drive*	*drive_id*
볼륨	*volume*	*vol_id*

- *wait*

  구성 요소를 사용할 수 없는 경우(잠기거나 사용 중임) 잠금이 보류 중 상태가 되도록 지정합니다. 그런 다음 ACSLS가 구성 요소를 *lock*합니다(사용 가능한 경우). 보류 중인 잠금

요청을 지우거나 취소(*cancel*)할 수 있습니다. ACSLS를 유휴 상태로 전환하면 보류 중인 잠금 요청도 취소됩니다.

사용법

lock 명령을 사용하면 볼륨 또는 드라이브를 ACSLS가 지정하는 잠금 ID로 잠글 수 있습니다. 사용 가능한(잠기거나 사용 중이지 않음) 볼륨 또는 드라이브만 잠글 수 있습니다.

주:

lock 명령을 입력하여 카트리지 또는 드라이브를 잠그면 ACSLS가 볼륨 또는 드라이브에 잠금 ID를 지정한 다음 잠금 ID를 볼륨 또는 드라이브의 잠금 ID로 변경합니다. set lock 명령을 사용하여 잠금 ID를 설정한 다음 lock 명령을 사용하여 set lock으로 설정한 잠금 ID로 볼륨 또는 드라이브를 잠글 수 없습니다.

예

- 드라이브 0,1,10,2를 잠그려는 경우

  ```
  lock drive 0,1,10,2
  ```
- 볼륨 EDU445를 잠그려는 경우

  ```
  lock volume EDU445
  ```

참조:

찾을 정보	참조 항목
지정한 드라이브 또는 카트리지의 모든 활성 또는 보류 중인 잠금 제거	“clear lock ”
드라이브 또는 카트리지의 잠금 상태 표시	“query lock ”
잠금 ID 설정	“set lock ”
잠금 또는 사용자 ID 표시	“show ”
드라이브 또는 카트리지의 활성 잠금 제거	“unlock ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

- *lock* 요청이 성공하면 다음 메시지가 나타납니다.

  ```
  Lock: Lock completed, Success.
  ```
- 요청의 각 식별자에 대해 다음 메시지 중 하나가 나타납니다.
  - *Lock: Drive drive_id locked under lock_id lock_id.*
  - *Lock: Volume vol_id locked under lock_id lock_id.*

    설명:

› *drive_id*는 잠긴 드라이브입니다.
› *vol_id*는 잠긴 카트리지의 카트리지 식별자입니다.
› *lock_id*는 잠금 ID입니다.

중간 메시지

없음

오류 메시지

lock 요청이 실패하면 다음 메시지 중 하나가 나타납니다.

- *Lock: Lock of drive drive_id failed, Drive in use.*

  설명: 지정된 드라이브가 이미 잠겨 있거나 사용 중이어서 ACSLS가 지정된 드라이브를 잠글 수 없습니다.

  변수: *drive_id*는 ACSLS가 잠글 수 없는 드라이브입니다.
- *Lock: Lock of drive drive_id failed, Lock failed.*

  설명: ACSLS가 지정된 드라이브를 잠글 수 없습니다. 올바른 구문 및 드라이브 식별자를 사용하여 *lock* 명령을 다시 입력하십시오.

  변수: *drive_id*는 ACSLS가 잠글 수 없는 드라이브입니다.
- *Lock: Lock of volume vol_id failed, Volume in use.*

  설명: 지정된 카트리지가 이미 잠겨 있거나 사용 중이어서 ACSLS가 지정된 카트리지를 잠글 수 없습니다.

  변수: *vol_id*는 ACSLS가 잠글 수 없는 카트리지입니다.
- *Lock: Lock of volume vol_id failed, Lock failed.*

  설명: ACSLS가 지정된 카트리지를 잠글 수 없습니다. 올바른 구문 및 카트리지 식별자를 사용하여 *lock* 명령을 다시 입력하십시오.

  변수: *vol_id*는 ACSLS가 잠글 수 없는 카트리지입니다.

표시 영역 메시지

없음

logoff

logoff 명령은 *cmd_proc*를 종료합니다.

형식

logoff

옵션

없음

사용법

logoff 명령을 사용하면 *cmd_proc*를 종료할 수 있습니다. 대화식(창)으로 실행할 경우 *cmd_proc*, *logoff*는 *cmd_proc* 창도 종료합니다.

예

- *cmd_proc*를 종료하려는 경우

```
logoff
```

> **주:**
>
> *logoff*는 *cmd_proc*만 종료하고 모든 ACSLS 상태에서 유효하며 ACSLS 작업에 영향을 주지 않습니다.

참조:

찾을 정보	참조 항목
cmd_proc 시작	“cmd_proc 시작 ”
cmd_proc 사용	“cmd_proc 시작 ”

명령 영역 메시지

없음

표시 영역 메시지

없음

mount

mount 명령은 데이터 카트리지를 마운트합니다.

형식

```
mount vol_id drive_id [bypass] [readonly]
```

옵션

- *vol_id*

  카트리지를 지정합니다.
- *drive_id*

드라이브를 지정합니다.

- *bypass*

  bypass 옵션은 다음과 같이 ACSLS가 카트리지를 마운트하기 전에 수행하는 검사를 대체합니다.
  - bypass 옵션은 항상 ACSLS의 외부 레이블 카트리지 ID 확인을 대체합니다.
  - 이 옵션은 ACSLS의 테이프 드라이브와 카트리지 매체 유형 간 호환성 확인을 대체할 수 있습니다.

  라이브러리는 *mount* 요청을 수신하면 카트리지 매체가 테이프 드라이브와 호환되는지 확인합니다. 매체 유형이 호환되지 않거나 알 수 없는 유형이면 라이브러리가 마운트에 실패합니다.
- *readonly*

  카트리지가 쓰기 보호 상태로 마운트되도록 지정합니다.

  주의:

  LTO 드라이브는 쓰기 보호 상태의 마운트를 지원하지 않습니다. LTO 드라이브에 대해 "읽기 전용 마운트"를 시도할 경우 실패하고 이벤트 로그에 "Drive cannot honor write protect" 메시지가 나타납니다.

  일부 초기 97xx SCSI 연결 라이브러리도 쓰기 보호 상태의 마운트를 지원하지 않습니다. *mount* 명령에 *read-only* 옵션을 지정한 경우에도 드라이브가 카트리지에 쓸 수 있습니다. 이러한 드라이브의 카트리지를 쓰기 보호하려면 카트리지의 읽기 전용 보호(예: 손바퀴)를 사용하십시오.

사용법

mount 명령을 사용하면 데이터 카트리지를 마운트할 수 있습니다. *mount* 명령을 입력할 때마다 각 드라이브에 카트리지 하나만 마운트할 수 있습니다.

마운트에 성공하려면 다음 사항을 충족해야 합니다.

- 카트리지와 드라이브가 동일한 ACS에 있어야 합니다.
- 카트리지가 사용 가능한 상태여야 하고 드라이브가 온라인 및 사용 가능한 상태여야 합니다.

예

드라이브 0,0,10,2에 볼륨 EDU010을 마운트하려는 경우

```
mount EDU010 0,0,10,2
```

다음 예에서 YUMA15는 카트리지 레이블에서 7번째 문자가 누락된 DD3C 볼륨입니다. *bypass* 옵션은 매체 호환성 검사를 건너뛰고 드라이브 0,0,4,0 및 SD3 드라이브의 카트리지를 강제로 *mount*합니다.

bypass 옵션을 사용하여 드라이브 0,0,4,0에 YUMA15를 마운트하려는 경우

참조:

찾을 정보	참조 항목
드라이브에서 카트리지 마운트 해제	“dismount ”
CAP(수동 모드)가 레이블이 지정된 카트리지를 라이브러리에 넣을 수 있도록 준비	“enter ”
CAP 상태 표시	“query cap ”
드라이브 상태 표시	“query drive ”
드라이브 또는 카트리지의 잠금 상태 표시	“query lock ”
지정된 데이터 카트리지에 대한 매체 호환 드라이브의 상태 표시	“query mount ”
카트리지 위치 및 매체 유형 표시	“query volume ”
CAP 모드(수동 또는 자동) 설정	“set cap mode ”
CAP 선택 우선 순위 설정	“set cap priority ”
CAP가 레이블이 없는 카트리지를 라이브러리에 넣을 수 있도록 준비	“venter ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

Mount: vol_id mounted on drive_id

설명: ACSLS가 지정된 카트리지를 마운트했습니다.

변수:

- *vol_id*는 ACSLS가 마운트한 카트리지의 카트리지 식별자입니다.
- *drive_id*는 카트리지를 마운트한 드라이브입니다.

중간 메시지

없음

오류 메시지

- *Mount: Mount failed, Audit in progress.*

  설명: *audit*로 인해 지정된 카트리지의 셀 위치에 대한 액세스가 잠겨서 ACSLS가 카트리지를 마운트할 수 없습니다.
- *Mount: Mount failed, In use.*

  설명: 드라이브가 사용 중이거나 요청된 카트리지가 다른 명령에 예약되어서 ACSLS가 카트리지를 마운트할 수 없습니다.
- *Mount: Mount failed, Misplaced tape.*

설명: 카트리지의 외부 레이블이 스토리지 셀에 대한 데이터베이스의 카트리지 식별자와 일치하지 않아서 ACSLS가 카트리지를 마운트할 수 없습니다. 데이터베이스가 스토리지 셀의 카트리지에 대한 카트리지 식별자, 매체 유형 및 청소 카트리지 속성을 업데이트합니다.

- *Mount: Mount failed, Not in same ACS.*

  설명: 지정된 카트리지 및 드라이브가 동일한 ACS에 없어서 ACSLS가 카트리지를 마운트할 수 없습니다.

- *Mount: Mount failed, Cartridge in drive.*

  설명: 지정된 카트리지가 드라이브에 이미 마운트되어 있어서 ACSLS가 카트리지를 마운트할 수 없습니다.

- *Mount: Mount failed, Unreadable label.*

  설명: 카트리지에 레이블이 없거나 레이블을 읽을 수 없거나 가상 레이블이 없어서 ACSLS가 카트리지를 마운트할 수 없습니다.

- *Mount: Mount failed, Invalid media type.*

  설명: 지정된 카트리지의 매체 유형이 잘못되어서 ACSLS가 카트리지를 마운트할 수 없습니다.

- *Mount: Mount failed, Invalid drive type.*

  설명: 지정된 드라이브 유형이 잘못되어서 ACSLS가 카트리지를 마운트할 수 없습니다.

- *Mount: Mount failed, Incompatible media type.*

  설명: 카트리지의 매체 유형이 지정된 드라이브와 호환되지 않아서 ACSLS가 카트리지를 마운트할 수 없습니다.

표시 영역 메시지

없음

mount *

*mount ** 명령은 스크래치 카트리지를 선택하고 마운트합니다.

형식

*mount * drive_id [pool_id] [media media_type | media *]*

옵션

- *drive_id*

  드라이브를 지정합니다.

- *pool_id*

ACSLS가 스크래치 카트리지를 선택하는 풀을 지정합니다. *pool_id*는 선택사항입니다. pool_id를 지정하지 않으면 ACSLS가 공통 풀(풀 0)에서 스크래치 카트리지를 찾으려고 합니다.

*pool_id*를 지정했지만 풀에 스크래치 카트리지(또는 혼합 매체 라이브러리에 대한 올바른 매체 중 하나)가 포함되어 있지 않거나 풀이 overflow에 대해 설정된 경우에는 ACSLS가 공통 풀(풀 0)에서 스크래치 카트리지를 찾으려고 합니다.

- *media media_type | media **

  카트리지 매체 유형을 지정합니다. 매체 유형 지정은 선택사항입니다.

사용법

*mount ** 명령을 사용하면 스크래치 카트리지를 선택하고 마운트할 수 있습니다. 다음 절에서는 ACSLS가 마운트할 스크래치 카트리지를 선택하는 방법 및 스크래치 카트리지에 대한 카트리지 매체 유형을 지정하는 방법에 대해 설명합니다.

ACSLS가 스크래치 카트리지를 선택하는 방법

*mount ** 명령은 다음과 같이 스크래치 카트리지를 선택합니다.

- 지정된 드라이브가 포함된 LSM과의 근접성을 기준으로 ACS에 LSM 목록을 만듭니다.
- 풀 및 매체 유형 조건을 충족하는 스크래치 카트리지를 찾을 때까지 목록에서 각 LSM을 검사합니다.
- 해당 LSM에서 액세스 날짜가 가장 오래된 스크래치 카트리지를 선택합니다.

  그런 다음 선택된 스크래치 카트리지가 드라이브에 마운트됩니다.

  매체 유형 지정

다음 방법 중 하나로 카트리지 매체 유형을 지정할 수 있습니다.

- 다음과 같은 형식의 *mount ** 명령으로 매체 유형을 명시적으로 지정합니다.

  *mount * drive_id [pool_id] media media_type*

  예: 공통 풀(풀 0)에서 9940 스크래치 카트리지를 마운트하려는 경우

  *mount * 0,0,10,2 media STK2P*
- ACSLS가 스크래치 환경 설정에 따라 매체를 선택하도록 매체 유형에 와일드카드(*)를 사용합니다. 자세한 내용은 “Extended Store 기능 사용 ”을 참조하십시오. 스크래치 환경 설정을 사용하려면 다음과 같은 형식의 *mount ** 명령을 입력합니다.

  *mount * drive_id [pool_id] media **

  예: 스크래치 환경 설정을 사용하여 공통 풀(풀 0)에서 스크래치 카트리지를 마운트하려는 경우

```
mount * 0,0,10,2 media *
```

- 다음과 같은 형식의 *mount ** 명령에서 media 옵션을 생략합니다. 이 명령은 ACSLS에 드라이브와 호환되는 매체 유형을 선택하도록 지시합니다.

  ```
  mount * drive_id [pool_id]
  ```

  예: 공통 풀(풀 0)에서 드라이브와 호환되는 스크래치 카트리지를 마운트(*mount*)하려는 경우

  ```
  mount * 0,0,10,2
  ```

예

다음 절에서는 단일 매체 및 혼합 매체 라이브러리에 대한 마운트 스크래치 예를 보여줍니다.

단일 매체 라이브러리

- 풀 5의 스크래치 카트리지를 드라이브 0,0,10,2에 마운트(*mount*)하려는 경우

  ```
  mount * 0,0,10,2 5
  ```

힌트: 풀 5에 사용 가능한 카트리지가 없고 풀이 *overflow*에 대해 설정된 경우 ACSLS는 공통 풀(풀 0)에서 카트리지를 선택합니다.

- 공통 풀(풀 0)의 스크래치 카트리지를 드라이브 0,0,10,0에 마운트(*mount*)하려는 경우

  ```
  mount * 0,0,10,0
  ```

혼합 매체 라이브러리

- 매체 유형이 T10000T2인 풀 5의 스크래치 카트리지를 드라이브 0,0,10,2에 마운트(*mount*)하려는 경우

  ```
  mount * 0,0,10,2 5 media T10000T2
  ```

힌트: 풀 5에 사용 가능한 카트리지가 없고 풀이 *overflow*에 대해 설정된 경우 ACSLS는 공통 풀(풀 0)에서 지정된 매체 유형의 카트리지를 선택합니다.

- 스크래치 환경 설정에 따라 매체 유형이 결정되는 풀 10의 스크래치 카트리지를 드라이브 0,0,2,3에 마운트(*mount*)하려는 경우

  ```
  mount * 0,0,2,3 10 media *
  ```

힌트: 풀 10에 사용 가능한 카트리지가 없고 풀이 overflow에 대해 설정된 경우 ACSLS는 공통 풀(풀 0)에서 지정된 매체 유형의 카트리지를 선택합니다.

- 매체 유형이 *T10000T2*인 공통 풀(풀 0)의 스크래치 카트리지를 드라이브 0,0,10,2에 마운트(*mount*)하려는 경우

  ```
  mount * 0,0,10,2 media T10000T2
  ```

- 스크래치 환경 설정에 따라 매체 유형이 결정되는 공통 풀(풀 0)의 스크래치 카트리지를 드라이브 0,0,2,3에 마운트(*mount*)하려는 경우

  ```
  mount * 0,0,2,3 media *
  ```

- 매체가 드라이브 0,0,2,3과 호환되는 공통 풀(풀 0)의 스크래치 카트리지를 마운트(*mount*)하려는 경우

  ```
  mount * 0,0,2,3
  ```

참조:

찾을 정보	참조 항목
스크래치 풀 만들기 또는 수정	“define pool ”
빈 스크래치 풀 삭제	“delete pool ”
드라이브에서 카트리지 마운트 해제	“dismount ”
스크래치 카트리지 관리	“LSM 채우기 ”
드라이브 상태 표시	“query drive ”
드라이브 또는 카트리지의 잠금 상태 표시	“query lock ”
지정한 스크래치 풀에 대한 매체 호환 드라이브의 상태 표시	“query mount * ”
스크래치 풀 속성 표시	“query pool ”
스크래치 카트리지 상태 표시	“query scratch ”
카트리지 위치 및 매체 유형 표시	“query volume ”
카트리지 스크래치 속성 설정 또는 지우기	“set scratch ”
스크래치 환경 설정 지정	“Extended Store 기능 사용 ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

- *Mount: vol_id mounted on drive_id*

  설명: ACSLS가 지정된 카트리지를 마운트했습니다.

  변수:
  - *vol_id*는 ACSLS가 마운트한 카트리지의 카트리지 식별자입니다.
  - *drive_id*는 카트리지를 마운트한 드라이브입니다.

중간 메시지

없음

오류 메시지

- *Mount: Mount failed, Audit in progress.*

설명: *audit*로 인해 지정된 카트리지의 셀 위치에 대한 액세스가 잠겨서 ACSLS가 카트리지를 마운트할 수 없습니다.

- *Mount: Mount failed, In use.*

  설명: 드라이브가 사용 중이거나 요청된 카트리지가 다른 명령에 예약되어서 ACSLS가 카트리지를 마운트할 수 없습니다.

- *Mount: Mount failed, Misplaced tape.*

  설명: 카트리지의 외부 레이블이 스토리지 셀에 대한 데이터베이스의 카트리지 식별자와 일치하지 않아서 ACSLS가 카트리지를 마운트(*mount*)할 수 없습니다. 데이터베이스가 스토리지 셀의 카트리지에 대한 카트리지 식별자, 매체 유형 및 청소 카트리지 속성을 업데이트합니다.

- *Mount: Mount failed, Not in same ACS.*

  설명: 지정된 카트리지 및 드라이브가 동일한 ACS에 없어서 ACSLS가 카트리지를 마운트(*mount*)할 수 없습니다.

- *Mount: Mount failed, Cartridge in drive.*

  설명: 지정된 카트리지가 드라이브에 이미 마운트되어 있어서 ACSLS가 카트리지를 마운트할 수 없습니다.

- *Mount: Mount failed, Unreadable label.*

  설명: 카트리지에 레이블이 없거나 레이블을 읽을 수 없거나 가상 레이블이 없어서 ACSLS가 카트리지를 마운트(*mount*)할 수 없습니다.

- *Mount: Mount failed, Invalid media type.*

  설명: 지정된 카트리지의 매체 유형이 잘못되어서 ACSLS가 카트리지를 마운트(*mount*)할 수 없습니다.

- *Mount: Mount failed, Invalid drive type.*

  설명: 지정된 드라이브 유형이 잘못되어서 ACSLS가 카트리지를 마운트(*mount*)할 수 없습니다.

- *Mount: Mount failed, Incompatible media type.*

  설명: 카트리지의 매체 유형이 지정된 드라이브와 호환되지 않아서 ACSLS가 카트리지를 마운트(*mount*)할 수 없습니다.

- *Mount: Mount failed, No compatible scratch cartridges in pool.*

  설명: 지정된 드라이브의 ACS에 스크래치 카트리지의 매체 유형과 일치하는 스크래치 카트리지가 없어서 ACSLS가 카트리지를 마운트(*mount*)할 수 없습니다. 또한 풀에 대해 overflow 속성이 설정된 경우 유효한 매체 유형을 가진 스크래치 카트리지가 없습니다.

표시 영역 메시지

- *Pool pool_id: low water mark warning.*

설명: 지정된 스크래치 풀의 카트리지 수가 카트리지 하위 임계값보다 작거나 같습니다.

변수: *low_water_mark*는 지정된 스크래치 풀의 하위 임계값입니다.

- *Pool pool_id: high water mark warning.*

  설명: 지정된 스크래치 풀의 카트리지 수가 카트리지 상위 임계값보다 크거나 같습니다.

  변수: *high_water_mark*는 지정된 스크래치 풀의 카트리지 상위 워터마크 임계값입니다.

move

move 명령은 지정된 카트리지를 지정된 LSM의 사용 가능한 스토리지 셀이나 특정 스토리지 셀로 이동합니다.

주:

SL3000에서는 카트리지를 특정 셀로 이동할 수 있습니다. 자세한 내용은 "라이브러리 분할 또는 분할 영역 ID 변경 "을 참조하십시오.

형식

move vol_id lsm_id or move vol_id cell_id

옵션

- *vol_id*

  카트리지를 지정합니다.

- *lsm_id*

  이동된 카트리지를 포함할 LSM을 지정합니다.

- *cell_id*

  카트리지를 이동할 셀을 지정합니다.

사용법

move 명령을 사용하면 지정된 카트리지를 다음의 사용 가능한 스토리지 셀로 이동할 수 있습니다.

- 동일한 LSM의 다른 패널. 예를 들어 LSM의 전체 패널을 비우려면 해당 패널의 모든 카트리지를 동일한 LSM의 다른 위치로 이동하십시오.
- 다른 LSM

move 명령을 입력할 때마다 카트리지를 하나만 이동할 수 있습니다. 카트리지가 현재 상주하는 LSM을 지정하면 ACSLS는 카트리지를 해당 LSM 내의 다른 패널로 이동합니다. 그렇지 않으면 ACSLS는 사용자가 지정한 LSM으로 카트리지를 이동합니다. *move* 명령은 취소할 수 없습니다.

이동에 성공하려면 다음 사항을 충족해야 합니다.

- 카트리지가 사용 가능한 상태이고 지정된 LSM과 동일한 ACS에 있어야 합니다.
- 카트리지가 현재 상주하는 LSM과 지정된 LSM이 모두 온라인 상태여야 합니다. 지정된 LSM에 사용 가능한 스토리지 셀이 하나 이상 있어야 합니다. 동일한 LSM 내로 이동하는 경우 해당 LSM의 다른 패널에 사용 가능한 셀이 하나 이상 있어야 합니다. 이동에 전달이 필요한 경우 사용되는 모든 LSM도 온라인 상태여야 합니다.

예

- 카트리지 EDU010(LSM 0,1에 상주함)을 이 LSM의 다른 패널로 이동하려는 경우

  ```
  move EDU010 0,1
  ```
- 카트리지 EDU010(LSM 0,1에 상주함)을 LSM 0,2로 이동하려는 경우

  ```
  move EDU010 0,2
  ```

참조:

찾을 정보	참조 항목
드라이브 또는 카트리지의 잠금 상태 표시	“query lock ”
LSM의 상태 표시	“query lsm ”
카트리지 위치 및 매체 유형 표시	“query volume ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

- *Move: vol_id moved to location cell_id*

  설명: ACSLS가 지정된 카트리지를 지정된 셀 위치로 이동했습니다.

  변수:
  - *vol_id*는 ACSLS가 이동한 카트리지의 카트리지 식별자입니다.
  - *cell_id*는 지정된 카트리지의 새 셀 위치입니다.

중간 메시지

없음

오류 메시지

Move: Move failed

query 명령

query 명령은 라이브러리 구성 요소의 상태를 표시합니다. 형식, 옵션 및 사용법을 포함하여 각 *query* 명령에 대한 자세한 내용은 다음 절을 참조하십시오.

형식

다음은 *query* 명령의 일반 형식을 보여줍니다.

```
query type [subtype | *] identifier... | all
```

주:

보류 중 또는 현재 *query* 요청에 대해 *cancel* 요청이 실행되면 정보 표시가 정지됩니다.

참조:

찾을 정보	참조 항목
명령 취소	"cancel "
Display 명령	"display 명령 옵션 사용 "

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

없음

주:

*query*가 성공하면 요청된 상태가 나타납니다.

중간 메시지

없음

오류 메시지

- *Library not available.*

  설명: ACSLS가 query server를 제외한 모든 질의를 복구하고 있어서 *query*가 실패했습니다.

- *Cartridge identifier vol_id not found.*

  설명: 지정된 카트리지가 라이브러리에 없어서 *query mount ** 명령이 해당 카트리지의 상태를 표시할 수 없습니다.

  변수: *vol_id*는 지정된 카트리지입니다.

- *Invalid media type*

  설명: 지정된 카트리지의 매체 유형이 잘못되었거나 명령에 잘못된 매체 유형이 지정되어서 *query mount **명령이 해당 카트리지의 상태를 표시할 수 없습니다.

  변수: *vol_id*는 지정된 카트리지입니다.

표시 영역 메시지

없음

query acs

query acs 명령은 ACS 상태를 표시합니다.

형식

query acs acs_id... | all

옵션

acs_id | all

*query*할 ACS를 지정하거나 모든 ACS에 대해 all을 지정합니다.

사용법

query acs 명령을 사용하면 다음 형식으로 ACS의 상태를 표시할 수 있습니다.

```
yyy-mm-dd hh:mm:ss ACS Status
Identifier  State    Free Cell  Audit  Mount  Dismount  Enter  Eject
                     Count      C/P    C/P    C/P       C/P    C/P

acs_id      state    count      n/n    n/n    n/n       n/n    n/n
```

설명:

- *acs_id*

  ACS 식별자입니다.
- *state*는 다음 ACS 상태 중 하나입니다.
  - *online*

    ACS가 온라인 상태입니다.
  - *offline*

    ACS가 오프라인 상태입니다.
  - *offline pending*

ACS가 현재 및 보류 중인 요청을 처리한 다음 오프라인 상태로 전환됩니다. ACS가 모든 새 요청을 거부합니다.

- *diagnostic*

  ACS가 현재 및 보류 중인 요청만 처리하고 모든 새 요청을 거부(*reject*)합니다. ACS를 클라이언트 응용 프로그램에서 사용할 수 없으며 *cmd_proc*를 사용해서만 제어할 수 있습니다. *vary* 명령을 사용하면 ACS를 온라인 상태로 전환할 수 있습니다.

- *recovery*

  ACS가 초기화 중이거나 오류에서 복구 중입니다. ACS가 온라인 상태로 전환될 때까지 기다리십시오.

- *count*

  ACS에 있는 사용 가능한 셀의 수입니다.

- *n*

  라이브러리 리소스가 필요한 각 명령(*audit*, *mount*, *dismount*, *enter*, *eject*)에 대해 ACS에 대한 현재(C) 및 보류 중인(P) 요청의 수입니다.

예

- ACS 1을 질의(*query*)하려는 경우

  ```
  query acs 1
  ```

- 라이브러리의 모든 ACS를 질의(*query*)하려는 경우

  ```
  query acs all
  ```

참조:

찾을 정보	참조 항목
라이브러리 구성 요소 상태 변경	“vary ”
요청 상태 표시	“query request ”

query cap

query cap 명령은 CAP 상태를 표시합니다.

형식

```
query cap cap_id... | all
```

옵션

- *cap_id* | *all*

  *query*할 CAP를 지정하거나 모든 CAP에 대해 all을 지정합니다.

주:

별표가 포함된 *cap_id*는 지정할 수 없습니다.

사용법

query cap 명령을 사용하면 CAP의 상태를 표시할 수 있습니다.

query cap 명령은 다음 형식으로 CAP 상태를 표시합니다.

```
yyy-mm-dd   hh:mm:ss   CAP Status
Identifier   Priority      Size        State       Mode     Status

cap_id       cap_priority  cap_size     cap_state  cap_mode status
```

- *cap_id*

  CAP 식별자입니다.
- *cap_priority*

  CAP 우선 순위입니다.
- *cap_size*

  CAP에 있는 셀의 수입니다.
- *cap_state*

  다음 CAP 상태 중 하나입니다.
  - *online*

    CAP가 온라인 상태입니다.
  - *offline*

    CAP가 오프라인 상태입니다.
  - *offline-pending*

    CAP가 현재 및 보류 중인 요청을 처리한 다음 오프라인 상태로 전환됩니다. CAP가 모든 새 요청을 거부합니다.
  - *diagnostic*

    CAP가 현재 및 보류 중인 요청만 처리하고 모든 새 요청을 거부합니다. CAP를 클라이언트 응용 프로그램에서 사용할 수 없으며 *cmd_proc*를 사용해서만 제어할 수 있습니다. *vary* 명령을 사용하면 CAP를 온라인 상태로 전환할 수 있습니다.
  - *recovery*

    CAP가 초기화 중이거나 오류에서 복구 중입니다. CAP가 온라인 상태로 전환될 때까지 기다리십시오.
- *cap_mode*

다음 CAP 넣기 모드 중 하나입니다.

- *manual*

  카트리지를 넣기 전에 CAP를 잠금 해제해야 합니다.
- *automatic*

  CAP가 카트리지를 넣을 준비가 되었습니다.

- *status*

  다음 CAP 상태 중 하나입니다.
  - *available*

    CAP가 사용 가능합니다.
  - *enter*

    CAP를 사용할 수 없습니다(카트리지 넣기용으로 예약됨).
  - *eject*

    CAP를 사용할 수 없습니다(카트리지 꺼내기용으로 예약됨).
  - *audit*

    CAP를 사용할 수 없습니다(감사 처리용으로 예약됨).

예

- CAP 0,1,0을 질의(*query*)하려는 경우

  ```
  query cap 0,1,0
  ```
- 라이브러리의 모든 CAP를 질의(*query*)하려는 경우

  ```
  query cap all
  ```

참조:

찾을 정보	참조 항목
요청 상태 표시	“query request ”
CAP 모드(수동 또는 자동) 설정	“set cap mode ”
CAP 선택 우선 순위 설정	“set cap priority ”
라이브러리 구성 요소 상태 변경	“vary ”

query clean

query clean 명령은 청소 카트리지 상태를 표시합니다. 존재하지 않거나 꺼낸 카트리지는 표시되지 않습니다. 테이프 드라이브에서 소모된(모두 사용됨) 것으로 보고된 청소 카트리지는 보고되지 않습니다.

형식

query clean vol_id... | all

옵션

- *vol_id | all*

  질의할 청소 카트리지를 지정하거나 모든 카트리지를 질의하려는 경우 all을 지정합니다.

사용법

query clean 명령을 사용하면 다음 형식으로 청소 카트리지의 상태를 표시할 수 있습니다.

```
yyy-mm-ddhh:mm:ssDrive Status
Identifier   State   Status   volumeType

drive_id     state   status   vol_idtype
```

설명:

- *vol_id*

  청소 카트리지의 카트리지 식별자입니다.
- *cell_id*

  청소 카트리지의 위치입니다.
- *max_usage*

  청소 카트리지를 사용할 수 있는 횟수입니다.
- *current_usage*

  청소 카트리지가 사용된 횟수입니다.
- *status*

  청소 카트리지의 위치입니다.
  - *home*

    카트리지가 스토리지 셀에 있습니다.
  - *in drive*

    카트리지가 드라이브에 있습니다.
  - *in transit*

    카트리지가 이동 중입니다.
- *type*

  카트리지 매체 유형(예: *3480*, *DD3D*, *DLTIII* 또는 *STK1R*)입니다.

예

- 청소 카트리지 J35992의 상태 정보를 표시하려는 경우

  query clean J35992
- 모든 청소 카트리지를 질의하려는 경우

  query clean all

참조:

찾을 정보	참조 항목
드라이브 청소 지침 및 절차	“LSM 채우기 ”
청소 카트리지 속성 설정	“set clean ”
청소 카트리지 및 소모된 청소 카트리지 표시	“display 명령 옵션 사용 ”

query drive

query drive 명령은 드라이브 상태를 표시합니다.

형식

query drive drive_id... | all

옵션

- *drive_id | all*

  질의할 드라이브를 지정하거나 모든 드라이브를 질의하려는 경우 all을 지정합니다.

사용법

query drive 명령을 사용하면 다음 형식으로 드라이브 상태를 표시할 수 있습니다.

설명:

- *drive_id*

  드라이브 식별자입니다.
- *state*

  다음 중 하나입니다.
  - *online*

    드라이브가 온라인 상태입니다.
  - *offline*

드라이브가 오프라인 상태입니다.

- *diagnostic*

  드라이브가 현재 및 보류 중인 요청만 처리하고 모든 새 요청을 거부합니다. 드라이브를 클라이언트 응용 프로그램에서 사용할 수 없으며 *cmd_proc*를 사용해서만 제어할 수 있습니다. *vary* 명령을 사용하면 드라이브를 온라인 상태로 전환할 수 있습니다.

- *recovery*

  드라이브가 초기화 중이거나 오류에서 복구 중입니다. 드라이브가 온라인 상태로 전환될 때까지 기다리십시오.

- *status*

  다음 드라이브 상태 중 하나입니다.

- *In use*

  드라이브에 카트리지가 마운트되었거나 드라이브가 마운트용으로 예약되었습니다.

  가능한 시나리오: *query drive all*을 수행하여 드라이브가 사용 중임을 나타내는 메시지를 받습니다. 그런 다음 *display drive **를 수행하여 드라이브가 예약되었음을 나타내는 메시지를 받습니다. 즉, 드라이브의 예약됨 상태는 *mount* 요청이 처리 중이며 카트리지가 드라이브로 이동 중임을 나타냅니다. 동시에 드라이브가 사용 중인 것으로 간주됩니다.

- *Available*

  드라이브를 마운트할 수 있습니다.

- *vol_id*

  드라이브에 있는 카트리지의 식별자입니다. 드라이브에 카트리지가 없거나 카트리지의 외부 레이블을 읽을 수 없거나 알 수 없으면 이 필드는 비어 있습니다.

- *drive_type*

  드라이브 유형입니다.

예

- 드라이브 0,3,1,0을 질의하려는 경우

  ```
  query drive 0,3,1,0
  ```

- 모든 드라이브를 질의하려는 경우

  ```
  query drive all
  ```

참조:

찾을 정보	참조 항목
라이브러리 구성 요소 상태 변경	“vary ”

찾을 정보	참조 항목
드라이브에서 카트리지 마운트 해제	“dismount ”
드라이브 일련 번호	“display 명령 옵션 사용 ” 및 “display 명령 옵션 사용 ”

query lmu

query lmu 명령은 단일 LMU 및 이중 LMU ACS 구성에 대한 LMU 및 포트 상태와 ACS 및 포트의 원하는 상태를 표시합니다. 라이브러리가 분할된 경우 분할 영역 ID도 표시합니다.

query lmu 명령은 ACSLS와 ACSLS가 관리하는 라이브러리 간의 통신을 모니터링할 수 있는 가장 좋은 방법입니다. *query lmu*:

- ACSLS와 라이브러리 간의 연결 상태를 표시합니다.
- 문자열의 라이브러리가 RE(중복 전자 부품)를 보고할 경우 RE를 표시합니다.
- 비RE 라이브러리 또는 비SL8500에 대한 단일 또는 이중 LMU를 표시합니다.
- ACS, 포트 연결, LSM 및 테이프 드라이브에 대한 원하는 상태를 표시합니다.
- 분할된 라이브러리의 분할 영역 ID를 표시합니다.

> **주:**
>
> ACSLS는 호스트/LMU 마이크로코드 호환성 레벨 12인 9330 LMU에 대해서만 이중 LMU 구성을 지원합니다. 두 LMU에 동일한 마이크로코드 레벨을 로드해야 합니다.

형식

query lmu acs_id... | all

옵션

- *acs_id | all*

  질의할 LMU가 있는 ACS를 지정하거나 모든 ACS에 대한 LMU를 질의하려는 경우 *all*을 선택합니다.

사용법

query lmu 명령을 사용하면 LMU 및 포트 상태와 단일 LMU 및 이중 LMU ACS 구성에 대한 원하는 상태를 표시할 수 있습니다. 다음 예에서는 중복 전자 부품이 없는 라이브러리와 중복 전자 부품이 있는 라이브러리에 대한 출력을 보여줍니다.

```
Output Example without Redundant Electronics
ACSSA> q lmu all
2010-04-02 14:43:54              LMU Status
ACS: 0     Mode: Single LMU      Active Status: Communicating
Not Partitioned                  Standby Status: -
      ACS State    Desired State
      online       online
```

```
Port  Port State   Desired State Role       CL
0,0   online       online        -          21 springtime:9997
ACS: 1    Mode: Dual LMU            Active Status:  Communicating
Not Partitioned                      standby Status: Communicating
      ACS State    Desired State
      online       online
Port  Port State   Desired State Role       CL
1,0   online       online        Active(A)  13 springtime:51100
1,1   online       online        standby(B) 13 springtime:51101
ACSSA>
```

```
Output Example of library with Redundant Electronics
ACSSA> q lmu all
2010-05-03 11:03:11        LMU Status
ACS: 0     Mode: Redundant   Active Status: Communicating
Not Partitioned              Standby Status: Communicating
          ACS State      Desired State
          Online         Online
Port      Port State    Desired State   Role       CL
0,0       online        online          Standby(A) 21 10.80.92.43
0,1       online        online          Standby(A) 21 10.80.93.33
0,2       online        online          Active(B)  21 10.80.92.52
0,3       online        online          Active(B)  21 10.80.93.47
0,4       online        online          Standy(B)  21 10.80.92.44
0,5       online        online          Active(A)  21 10.80.92.53
```

설명:

- *acs_id*

  ACS 식별자입니다.

- *mode*

  LMU 모드(*Dual LMU*, *Single LMU* 또는 *SCSI LMU*)입니다.

- *status*

  활성 또는 대기 LMU 상태(*Communicating*, *Not Communicating* 또는 *Offline*)입니다.

- *partition_status*

  라이브러리가 분할되었는지 여부를 표시합니다. 라이브러리가 분할된 경우 연결된 분할 영역이 표시됩니다. 분할 상태는 다음과 같습니다.

  - *Not Partitioned*
  - *Partition 1-n*

    분할 영역 번호입니다.

- *acs_state*

  실제 ACS 상태입니다. 상태는 다음과 같습니다.

  - *online*

    ACS가 온라인 상태입니다.

  - *diagnostic*

    ACS가 현재 및 보류 중인 요청만 처리하고 모든 새 요청을 거부합니다. ACS를 클라이언트 응용 프로그램에서 사용할 수 없으며 *cmd_proc*를 사용해서만 제어할 수 있습니다. *vary* 명령을 사용하면 ACS를 온라인 상태로 전환할 수 있습니다.
  - *offline*

    ACS가 오프라인 상태입니다.
  - *offline pending*

    ACS가 현재 및 보류 중인 요청을 처리한 다음 오프라인 상태로 전환됩니다. ACS가 모든 새 요청을 거부합니다.
- *acs_desired_state*

  원하는 ACS 상태입니다. 원하는 상태는 다음과 같습니다.
  - *online*
  - *diagnostic*
  - *offline*
- *port_id*

  포트 식별자입니다.
- *port_state*

  다음 실제 포트 상태 중 하나입니다.
  - *online*

    포트가 온라인 상태입니다.
  - *offline*

    포트가 오프라인 상태입니다.
- *port_desired_state*

  원하는 포트 상태입니다. 원하는 상태는 다음과 같습니다.
  - *online*
  - *offline*
- *role (des)*

  LMU의 역할 및 지정(A 또는 B)이며 역할은 다음과 같습니다.
  - *Active*

    LMU가 활성 역할입니다(LMU가 ACS를 관리함).
  - *Standby*

    LMU가 대기 *role*입니다(ACS를 관리하지 않음, 활성 LMU와 통신하며 전환에 사용 가능함).

주:

전환 중에는 *role* 필드의 정보가 최신이 아니며 대시(-)로 표시될 수 있습니다. 정보가 최신 상태가 되면 ACSLS가 각 LMU의 실제 역할로 *role* 필드를 새로 고칩니다.

- *compat_level*

  호스트/LMU 마이크로코드 호환성 레벨입니다. 이중 LMU 구성에는 레벨 11 이상이 필요합니다.

- *dev_name*

  포트 장치 이름입니다.

예

- 모든 ACS를 관리 중인 모든 LMU에 대한 LMU 및 포트 상태를 표시하려는 경우

  query lmu all

- ACS 0과 1을 관리 중인 모든 LMU에 대한 LMU 및 포트 상태를 표시하려는 경우

  query lmu 0 1

참조:

찾을 정보	참조 항목
ACS 관리를 ACS의 활성 LMU에서 대기 LMU로 수동으로 전환	“switch lmu ”

query lock

query lock 명령은 드라이브 또는 카트리지의 잠금 상태를 표시합니다.

형식

query lock type identifier... | all

옵션

- *type identifier | all*

  다음 표와 같이 질의할 드라이브 또는 카트리지를 지정하거나 모든 드라이브 또는 카트리지를 질의하려는 경우 all을 지정합니다.

표 13.5. query lock에 유효한 잠금 유형

라이브러리 구성 요소	유형	식별자
드라이브	*drive*	*drive_id*

라이브러리 구성 요소	유형	식별자
볼륨	*volume*	*vol_id*

사용법

query lock 명령을 사용하면 다음 형식으로 드라이브 또는 카트리지의 잠금 상태를 표시할 수 있습니다.

```
yyy-mm-ddhh:mm:ssLock            Status
Identifier   Lock-id   Duration   Pending   StatusUser Identifier

vol_id       lock_id   duration   pending   status   user_id
or
drive_id     lock_id   duration   pending   status   user_id
```

설명:

- *vol_id*

  지정된 카트리지의 식별자입니다.
- *drive_id*

  지정된 드라이브의 식별자입니다.
- *lock_id*

  잠금 ID입니다.
- *duration*

  잠금이 활성 상태였던 시간(초)입니다.
- *pending*

  카트리지 또는 드라이브를 대기 중인 잠금 요청의 수입니다.
- *status*는 다음 상태 중 하나입니다.
  - *available*

    카트리지 또는 드라이브가 사용 가능합니다.
  - *in use*

    카트리지 또는 드라이브가 사용 중이거나 마운트용으로 예약되었습니다.
- *user_id*

  카트리지 또는 드라이브를 잠근 사용자 ID입니다. *user_id*는 80자를 초과하면 다음 줄로 넘어갑니다.

예

- 드라이브 1,0,4,0에 대한 잠금 상태 정보를 표시하려는 경우

q loc dr 1,0,4,0

- 모든 드라이브에 대한 잠금 상태 정보를 표시하려는 경우

  query lock drive all
- 카트리지 SL4493에 대한 잠금 상태 정보를 표시하려는 경우

  query lock cartridge SL4493
- 모든 카트리지에 대한 잠금 상태 정보를 표시하려는 경우

  query lock cartridge all

참조:

찾을 정보	참조 항목
지정한 드라이브 또는 카트리지의 모든 활성 또는 보류 중인 잠금 제거	“clear lock ”
드라이브 및 카트리지 잠금	“lock ”
잠금 ID 설정	“set lock ”
잠금 또는 사용자 ID 표시	“show ”
활성 잠금 제거	“unlock ”

query lsm

query lsm 명령은 LSM 상태를 표시합니다.

형식

query lsm lsm_id... | all

옵션

- *lsm_id | all*

  질의할 LSM을 지정하거나 모든 잠금을 질의하려는 경우 all을 지정합니다.

  query lsm 명령을 사용하면 다음 형식으로 LSM의 상태를 표시할 수 있습니다.

```
yyy-mm-ddhh:mm:ssLSM Status
Identifier   State   Free Cell   Audit   Mount    Dismount   Enter   Eject
                     Count       C/P     C/P      C/P        C/P     C/P
```

설명:

- *lsm_id*

  LSM 식별자입니다.
- *state*는 다음 LSM 상태 중 하나입니다.

- *diagnostic*

  LSM이 현재 및 보류 중인 요청만 처리하고 모든 새 요청을 거부합니다. LSM을 클라이언트 응용 프로그램에서 사용할 수 없으며 *cmd_proc*를 사용해서만 제어할 수 있습니다. *vary* 명령을 사용하면 LSM을 온라인 상태로 전환할 수 있습니다.

- *offline*

  LSM이 오프라인 상태입니다.

- *offline pending*

  LSM이 현재 및 보류 중인 요청을 처리한 다음 오프라인 상태로 전환됩니다. LSM이 모든 새 요청을 거부합니다.

- *online*

  LSM이 온라인 상태입니다.

- *recovery*

  LSM이 초기화 중이거나 오류에서 복구 중입니다. LSM이 온라인 상태로 전환될 때까지 기다리십시오.

- *count*

  LSM에 있는 빈 스토리지 셀의 수입니다.

- *n*

  라이브러리 리소스가 필요한 각 명령(*audit, mount, dismount, enter* 및 *eject*)에 대해 LSM에 대한 현재(C) 및 보류 중인(P) 요청의 수입니다.

예

- ACS 0의 LSM 1에 대한 상태 정보를 표시하려는 경우

  ```
  query lsm 0,1
  ```

- 모든 LSM에 대한 상태 정보를 표시하려는 경우

  ```
  query lsm all
  ```

참조:

찾을 정보	참조 항목
요청 상태 표시	“query request ”
라이브러리 구성 요소 상태 변경	“vary ”

query mount

query mount 명령을 사용하면 여러 개의 LSM(예: SL8500 또는 일련의 연결된 SL8500)을 사용하여 라이브러리 성능을 최적화할 수 있습니다. 지정된 데이터 카트리지에 대한 매체

호환 드라이브의 상태를 표시합니다. 카트리지가 존재하지 않거나 거부된 경우에는 이러한 드라이브가 표시되지 않습니다. 또한 다음 사항을 고려하십시오.

- *query mount* 요청에 대해 반환할 드라이브 목록을 선택할 때 선택한 드라이브는 지정된 볼륨과 호환되어야 합니다.
- 드라이브는 기본적으로 전달 거리를 기준으로 정렬됩니다. 카트리지와 가장 가까운 LSM의 드라이브가 먼저 나열됩니다.
- ACSLS 7.3 이상 릴리스에서는 카트리지에서 동일한 전달 거리에 있는 드라이브는 가장 오래 전에 사용된 순서로 나열됩니다.

  예: 카트리지가 마운트 해제된 이후 시간이 가장 오래된 호환되는 드라이브가 가장 먼저 나열되고 그 다음으로 시간이 오래된 드라이브가 두번째로 나열됩니다.

형식

query mount vol_id

옵션

- *vol_id*

  질의할 카트리지를 지정합니다.

사용법

query mount 명령을 사용하면 카트리지와 동일한 ACS에 연결되어 있고 지정된 카트리지의 매체 유형과 호환되는 모든 라이브러리 드라이브의 상태를 표시할 수 있습니다. *query mount*는 LSM의 드라이브 상태(온라인, 오프라인, 오프라인 보류 중 또는 진단)를 표시합니다.

호환되는 드라이브는 지정된 카트리지와의 근접성을 기준으로 다음 형식으로 정렬됩니다.

```
yyy-mm-dd  hh:mm:ss  Mount Status
Identifier   Status   Drive     State   Status       Volume   Drive Type

vol_id      vol_stat  drive_id  state   drive_stat   inu_id   drive_type
```

설명:

- *vol_id*

  지정된 카트리지의 식별자입니다.
- *vol_stat*

  카트리지의 위치입니다.
  - *home*

    카트리지가 스토리지 셀에 있습니다.

- *in drive*

  카트리지가 드라이브에 있습니다.

- *in transit*

  카트리지가 이동 중이거나 누락되었습니다.

- *drive_id*

  지정된 카트리지의 매체 유형과 호환되는 모든 라이브러리 드라이브의 목록입니다.

- *drive_id*

  드라이브 식별자입니다.

- *state*

  다음 드라이브 상태 중 하나입니다.

  - *online*

    드라이브가 온라인 상태입니다.

  - *offline*

    드라이브가 오프라인 상태입니다.

  - *diagnostic*

    드라이브가 현재 및 보류 중인 요청만 처리하고 모든 새 요청을 거부합니다. 드라이브를 클라이언트 응용 프로그램에서 사용할 수 없으며 *cmd_proc*를 사용해서만 제어할 수 있습니다. *vary* 명령을 사용하면 드라이브를 온라인 상태로 전환할 수 있습니다.

  - *recovery*

    드라이브가 초기화 중이거나 오류에서 복구 중입니다. 드라이브가 온라인 상태로 전환될 때까지 기다리십시오.

- *status*

  다음 드라이브 상태 중 하나입니다.

  - *In use*

    드라이브에 카트리지가 마운트되었거나 드라이브가 마운트용으로 예약되었습니다.

  - *Available*

    드라이브를 마운트할 수 있습니다.

- *inu_id*

  드라이브에 있는 카트리지의 식별자입니다. 카트리지 ID는 *drive_stat*가 사용 중인 경우에만 나타납니다.

- *drive_type*

  드라이브 유형입니다.

예

- 카트리지 ZUNI14와의 근접성을 기준으로 정렬된 드라이브의 상태 정보를 표시하려는 경우

  ```
  query mount ZUNI14
  ```

참조:

찾을 정보	참조 항목
드라이브에 데이터 카트리지 마운트	“mount ”
드라이브 또는 카트리지의 잠금 상태 표시	“query lock ”
카트리지 위치 및 매체 유형 표시	“query volume ”
라이브러리 구성 요소 상태 변경	“vary ”

query mount *

*query mount **명령은 하나 이상의 지정된 스크래치 풀에 있는 매체와 호환되는 드라이브(필요한 경우 풀 내의 특정 카트리지 매체 유형과 호환되는 드라이브만)의 상태를 표시합니다.

형식

```
query mount * pool_id... [media media_type | media *]
```

옵션

- *pool_id*

  질의할 스크래치 풀을 지정합니다.
- *media media_type | media **

  매체 유형을 지정합니다.

사용법

*query mount **명령을 사용하면 지정된 스크래치 풀 및 카트리지와 동일한 ACS에 있는 모든 카트리지 매체 유형과 호환되는 모든 라이브러리 드라이브의 상태를 표시할 수 있습니다. 풀 0은 공통 스크래치 풀입니다. 풀 내의 특정 카트리지 매체 유형과 호환되는 드라이브로 표시를 제한하려면 *media_type* 옵션을 지정하십시오. 표시된 드라이브는 밀도가 가장 높은 스크래치 풀과의 근접성을 기준으로 정렬됩니다. *query mount **는 LSM의 드라이브에 대한 드라이브 상태(온라인, 오프라인, 오프라인 보류 중 또는 진단)를 표시합니다.

*query mount **명령은 다음 형식으로 드라이브 상태를 표시합니다.

```
yyy-mm-dd  hh:mm:ss  Mount Scratch Status
```

```
Identifier     Drive     State    Volume     Status        Drive Type
pool_id        drive_id  state    vol_id     drive_stat    drive_type
```

설명:

- *pool_id*

  지정된 스크래치 풀입니다.
- *drive_id*

  지정된 풀의 모든 매체 유형 또는 매체 유형이 지정된 경우 풀 내의 특정 매체 유형과 호환되는 모든 라이브러리 드라이브의 목록입니다.
- *state*

  다음 드라이브 상태 중 하나입니다.
  - *online*

    드라이브가 온라인 상태입니다.
  - *offline*

    드라이브가 오프라인 상태입니다.
  - *diagnostic*

    드라이브가 현재 및 보류 중인 요청만 처리하고 모든 새 요청을 거부합니다. 드라이브를 클라이언트 응용 프로그램에서 사용할 수 없으며 *cmd_proc*를 사용해서만 제어할 수 있습니다. *vary* 명령을 사용하면 드라이브를 온라인 상태로 전환할 수 있습니다.
  - *recovery*

    드라이브가 초기화 중이거나 오류에서 복구 중입니다. 드라이브가 온라인 상태로 전환될 때까지 기다리십시오.
- *vol_id*

  드라이브에 있는 카트리지의 식별자입니다. 카트리지 ID는 *drive_stat*가 사용 중인 경우에만 나타납니다.
- *drive_stat*

  다음 드라이브 상태 중 하나입니다.
  - *In use*

    드라이브에 카트리지가 마운트되었거나 드라이브가 마운트용으로 예약되었습니다.
  - *Available*

    드라이브를 마운트할 수 있습니다.
- *drive_type*

  드라이브 유형입니다.

예

풀 5에서 집중도가 가장 높은 스크래치 테이프와의 근접성을 기준으로 나열된 호환되는 드라이브의 상태를 표시하려는 경우

```
query mount * 5
```

공통 풀 0에서 집중도가 가장 높은 3480 스크래치 테이프와의 근접성을 기준으로 나열된 호환되는 드라이브의 상태를 표시하려는 경우

```
query mount * 0 media 3480
```

참조:

찾을 정보	참조 항목
스크래치 풀 만들기 또는 수정	“define pool ”
빈 스크래치 풀 삭제	“delete pool ”
드라이브에 스크래치 카트리지 마운트	“mount * ”
드라이브 또는 카트리지의 잠금 상태 표시	“query lock ”
스크래치 풀 속성 표시	“query pool ”
스크래치 카트리지 상태 표시	“query scratch ”
카트리지 스크래치 속성 설정 또는 지우기	“set scratch ”
라이브러리 구성 요소 상태 변경	“vary ”

query pool

query pool 명령은 스크래치 풀 속성을 표시합니다.

형식

```
query pool pool_id... | all
```

옵션

- *pool_id* | *all*

  질의할 스크래치 풀을 지정하거나 모든 풀을 질의하려는 경우 all을 지정합니다. 풀 0은 공통 풀입니다.

사용법

pool 명령을 사용하면 다음 형식으로 스크래치 풀 속성을 표시할 수 있습니다.

```
yyy-mm-dd hh:mm:ss Pool Status
Identifier Volume Count Low Water Mark High Water Mark  Attributes

pool_id    vol_count    low_water_mark high_water_mark attribute
```

설명:

- *pool_id*

  지정된 스크래치 풀입니다.
- *vol_count*

  풀의 스크래치 카트리지 수입니다.

  존재하지 않은 스크래치 카트리지 및 꺼낸 스크래치 카트리지와 풀의 데이터 카트리지는 계산되지 않습니다.
- *low_water_mark*

  카트리지 하위 경고 임계값입니다. 스크래치 카트리지 수가 이 임계값 아래로 떨어지면 ACSLS가 이벤트 로그에 경고 메시지를 기록합니다.

  값 뒤에 "-"가 나오면 스크래치 카트리지 수가 카트리지 하위 임계값보다 낮음을 의미합니다.
- *high_water_mark*

  카트리지 상위 경고 임계값입니다. 스크래치 카트리지 수가 이 임계값에 도달하거나 초과하면 ACSLS가 이벤트 로그에 경고 메시지를 기록합니다. 값 뒤에 "+"가 나오면 스크래치 카트리지 수가 카트리지 상위 임계값보다 높거나 같음을 의미합니다.
- *attribute*

  지정된 스크래치 풀에 대해 *overflow*가 설정된 경우(*set scratch* 명령 사용) 표시됩니다. overflow는 지정된 스크래치 풀의 카트리지로 *mount scratch ** 요청을 충족할 수 없는 경우 공통 스크래치 풀(풀 0)에서 스크래치 카트리지가 선택되도록 지정합니다.

예

- 스크래치 풀 5의 상태 정보를 표시하려는 경우

  ```
  query pool 5
  ```
- 모든 스크래치 풀의 상태 정보를 표시하려는 경우

  ```
  query pool all
  ```

참조:

찾을 정보	참조 항목
스크래치 풀 만들기 또는 수정	“define pool ”
빈 스크래치 풀 삭제	“delete pool ”
스크래치 카트리지 상태 표시	“query scratch ”
카트리지 스크래치 속성 설정 또는 지우기	“set scratch ”

query port

query port 명령은 포트 상태를 표시합니다.

형식

query port port_id... | all

옵션

- *port_id | all*

  질의할 포트를 지정하거나 모든 포트를 질의하려는 경우 all을 지정합니다.

사용법

query port 명령을 사용하면 다음 형식으로 포트 상태를 표시할 수 있습니다.

```
yyy-mm-dd  hh:mm:ss  Port Status
State   Identifier

state   port_id
```

설명:

- *state*

  다음 포트 상태 중 하나입니다.
  - *online*

    포트가 온라인 상태입니다.
  - *offline*

    포트가 오프라인 상태입니다.
- *port_id*

  포트 식별자입니다.

예

- 포트 0,0의 상태 정보를 표시하려는 경우

  query port 0,0
- 모든 포트를 질의하려는 경우

  query port all

참조:

찾을 정보	참조 항목
라이브러리 구성 요소 상태 변경	“vary ”

query request

query request 명령은 요청 상태를 표시합니다.

형식

query request request_id... | all

옵션

- *request_id | all*

  질의할 요청을 지정하거나 모든 요청을 질의하려는 경우 all을 지정합니다.

사용법

query request 명령을 사용하면 다음 형식으로 요청 상태를 표시할 수 있습니다.

설명:

- *request_id*

  ACSLS 요청 식별자입니다.
- *command*

  요청 식별자에 해당하는 ACSLS 명령입니다.
- *status*

  다음 요청 상태 중 하나입니다.
  - *Current*

    ACSLS가 요청을 처리 중입니다.
  - *Pending*

    요청이 처리를 기다리는 중입니다.
  - *Not found*

    지정된 요청이 유효한 ACSLS 요청이 아닙니다.

예

- 요청 33179의 상태 정보를 표시하려는 경우

  query request 33179
- 현재 및 보류 중인 모든 요청을 표시하려는 경우

  query request all

참조:

찾을 정보	참조 항목
명령 취소	“cancel ”

query scratch

query scratch 명령은 풀의 스크래치 카트리지(액세스 날짜를 기준으로 오름차순 정렬됨)의 상태를 표시합니다. 액세스 날짜가 가장 이른 카트리지가 목록의 맨 위에 표시되고 가장 최근에 사용된 카트리지가 목록 맨 아래에 표시됩니다. 첫번째 열의 풀 ID는 특정 순서로 나타나지 않습니다. 액세스 제어를 통해 허용된 카트리지만 표시됩니다.

형식

query scratch pool_id... | all

옵션

pool_id | all

질의할 스크래치 풀을 지정하거나 모든 풀을 질의하려는 경우 all을 지정합니다. 풀 0은 공통 풀입니다.

사용법

query scratch 명령을 사용하면 다음 형식으로 풀의 스크래치 카트리지 상태를 표시할 수 있습니다.

설명:

- *pool_id*

  지정된 스크래치 풀입니다.
- *vol_id*

  스크래치 카트리지의 식별자입니다.
- *cell_id*

  카트리지가 포함된 스토리지 셀입니다.
- *status*

  카트리지의 위치입니다.
  - *home*

    카트리지가 스토리지 셀에 있습니다.
  - *in drive*

    카트리지가 드라이브에 있습니다.

- *in transit*

  카트리지가 이동 중입니다.

- *media_type*

  카트리지의 매체 유형(예: 3480, 3490E, DD3D 또는 DLTIV)입니다.

  주:

  카트리지가 존재하지 않거나, 꺼내졌거나, 누락된 상태의 스크래치 카트리지는 *query scratch* 출력에 포함되지 않습니다.

  풀을 기준으로 정렬된 카트리지를 보려면 각 풀에 대해 *query scratch* 명령을 연속 실행하십시오. 또는 *display volume* 명령을 실행하여 정보를 보고 풀을 기준으로 정렬할 수 있습니다. 단, 가장 오래 전에 사용된 스크래치 카트리지를 가져온다는 보장은 없습니다.

예

- 스크래치 풀 29015에 있는 스크래치 카트리지의 상태 정보를 표시하려는 경우

  query scratch 29015

- 모든 스크래치 풀에 있는 스크래치 카트리지의 상태 정보를 표시하려는 경우

  query scratch all

참조:

찾을 정보	참조 항목
스크래치 풀 만들기 또는 수정	“define pool ”
빈 스크래치 풀 삭제	“delete pool ”
카트리지 스크래치 속성 설정 또는 지우기	“set scratch ”

query server

query server 명령은 ACSLS 및 라이브러리 상태를 표시합니다.

형식

query server

옵션

없음

사용법

query server 명령을 사용하면 다음 형식으로 ACSLS 및 라이브러리의 상태를 표시할 수 있습니다.

설명:

- *Identifier*

  비어 있습니다.
- *state*

  다음 ACSLS 상태 중 하나입니다.
  - *idle*

    ACSLS가 유휴 상태(요청을 처리하지 않음)입니다.
  - *idle pending*

    ACSLS가 현재 및 보류 중인 요청을 처리하고 새 요청을 거부한 다음 유휴 상태로 전환됩니다.
  - *recovery*

    ACSLS가 초기화 중(실행 상태로 전환 중)이거나 오류에서 복구 중입니다. ACSLS가 요청을 처리하지 않습니다.
  - *run*

    ACSLS가 실행 중(요청을 처리하고 있음)입니다.
- *count*

  라이브러리에 있는 빈 스토리지 셀의 수입니다.
- *n*

  라이브러리 리소스가 필요한 각 명령(*audit, mount, dismount, enter* 및 *eject*)에 대한 현재(C) 및 보류 중인(P) ACSLS 요청의 수입니다.

힌트: *query server* 요청은 취소할 수 없습니다.

예

- 서버에 대한 상태 정보를 표시하려는 경우

  query server

참조:

찾을 정보	참조 항목
라이브러리 구성 요소 상태 변경	“vary ”
요청 상태 표시	“query request ”

query volume

query volume 명령은 카트리지의 위치를 표시합니다.

형식

query volume vol_id... | all

옵션

- *vol_id | all*

  질의할 볼륨을 지정하거나 모든 볼륨을 질의하려는 경우 all을 지정합니다.

사용법

query volume 명령을 사용하면 다음 형식으로 볼륨의 위치를 표시할 수 있습니다.

설명:

- *vol_id*

  볼륨 식별자입니다.
- *status*

  카트리지의 위치입니다.
  - *home*

    카트리지가 스토리지 셀에 있습니다.
  - *in drive*

    카트리지가 드라이브에 있습니다.
  - *in transit*

    카트리지가 이동 중이거나 누락되었습니다.
  - *absent*

    카트리지를 찾을 수 없습니다.
  - *ejected*

    라이브러리에서 카트리지를 꺼냈습니다.
- *location*

  위치를 다음 중 하나로 지정합니다.
  - 상태가 home이면 위치가 스토리지 셀 식별자입니다.
  - 상태가 *in transit*이면 위치가 셀 식별자 또는 드라이브 식별자입니다.
  - 상태가 *in drive*이면 위치가 드라이브 식별자입니다.
- *media_type*

  볼륨의 매체 유형(예: 3480, 3490E, DD3D 또는 DLTIV)입니다.

예

- 볼륨 2903B의 상태 정보를 표시하려는 경우

  ```
  query volume 2903B
  ```

- 라이브러리의 모든 볼륨을 표시하려는 경우

  ```
  q volume all
  ```

참조:

찾을 정보	참조 항목
기타 볼륨 정보	“로깅 볼륨 통계 보고서 작성”
Display 명령	“display 명령 옵션 사용 ” 및 “display 명령 옵션 사용 ”
드라이브에 데이터 볼륨 마운트	“mount ”
드라이브 또는 볼륨의 잠금 상태 표시	“query lock ”
지정한 데이터 볼륨에 대한 매체 호환 드라이브의 상태 표시	“query mount ”
스크래치 풀 속성 표시	“query pool ”
볼륨 스크래치 속성 설정 또는 지우기	“set scratch ”
라이브러리 구성 요소 상태 변경	“vary ”

set 명령

set 명령은 여러 라이브러리 구성 요소의 다양한 속성을 설정합니다. 형식, 옵션, 사용법 및 메시지를 포함하여 각 *set* 명령에 대한 자세한 내용은 다음 절을 참조하십시오.

형식

다음은 set 명령의 일반 형식을 보여줍니다.

```
set type [off | subtype] [*] identifier...
```

참조:

찾을 정보	참조 항목
명령 취소	“cancel ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

set 명령이 성공하면 다음 메시지가 나타납니다.

```
Set: Set completed, Success.
```

해당 성공 메시지는 각 *set* 명령을 참조하십시오.

중간 메시지

없음

오류 메시지

해당 메시지는 각 *set* 명령을 참조하십시오.

표시 영역 메시지

해당 메시지는 각 *set* 명령을 참조하십시오.

set cap mode

set cap mode 명령은 CAP의 넣기 모드를 설정합니다.

형식

set cap mode cap_mode cap_id

옵션

- *cap_mode*

  수동 또는 자동 CAP 모드를 지정합니다.
- *cap_id*

  CAP 식별자를 지정합니다.

  우선 순위가 설정되어 있지 않으면 별표(*)가 포함된 CAP 식별자를 지정할 수 없습니다.

사용법

set cap mode 명령을 사용하면 CAP 넣기 모드를 다음 모드 중 하나로 설정할 수 있습니다.

- *manual*

  카트리지를 넣기 전에 *enter* 명령을 입력해야 합니다.
- *automatic*

  먼저 *enter* 명령을 입력하지 않고 카트리지를 넣을 수 있습니다.

  주:

  분할된 라이브러리에서 CAP 모드를 자동으로 설정할 수 없습니다.

힌트: CAP가 사용 중인 동안에는 CAP의 모드를 변경할 수 없습니다. 즉, 수동 또는 자동 넣기 작업 중 도어가 열려 있으면 *enter* 작업을 완료할 때까지 해당 모드를 변경할 수 없습니다.

예

- CAP 0,3,1을 수동 모드로 설정하려는 경우

  ```
  set cap mode manual 0,3,1
  ```
- CAP 0,3,1을 자동 모드로 설정하려는 경우

  ```
  set cap mode automatic 0,3,1
  ```

참조:

찾을 정보	참조 항목
CAP(수동 모드)가 레이블이 지정된 카트리지를 라이브러리에 넣을 수 있도록 준비	“enter ”
CAP 상태 표시	“query cap ”
라이브러리 구성 요소 상태 변경	“vary ”
CAP가 레이블이 없는 카트리지를 라이브러리에 넣을 수 있도록 준비	“venter ”

명령 영역 메시지

이 절에서는 명령 메시지에 대해 설명합니다.

성공 메시지

- *Set: CAP cap_id, mode changed to cap_mode.*

  설명: ACSLS가 지정된 CAP의 모드를 변경했습니다.

  변수:
  - *cap_id*는 모드가 변경된 CAP입니다.
  - *cap_mode*는 CAP의 새 넣기 모드입니다.

중간 메시지

없음

오류 메시지

- *Set: CAP cap_id Set failed, Incorrect attribute.*

  설명: 잘못된 CAP 모드가 지정되어서 ACSLS가 지정된 CAP의 모드를 변경할 수 없습니다.

  변수: *cap_id*는 모드가 변경되지 않은 CAP입니다.

- *CAP cap_id: Automatic mode.*

  설명: ACSLS가 지정된 CAP의 모드를 자동으로 변경했습니다.

  변수: *cap_id*는 모드가 변경된 CAP입니다.
- *CAP cap_id: Manual mode.*

  설명: ACSLS가 지정된 CAP의 모드를 수동으로 변경했습니다.

  변수: *cap_id*는 모드가 변경된 CAP입니다.

set cap priority

set cap priority 명령은 CAP의 자동 선택 우선 순위를 설정합니다.

형식

set cap priority cap_priority cap_id

옵션

- *cap_priority*

  CAP 우선 순위를 지정합니다. 유효한 값은 0 ~ 16입니다. 16이 가장 높은 우선 순위입니다. 모든 CAP는 처음에는 우선 순위가 0입니다. 즉, ACSLS가 자동으로 CAP를 선택하지 않습니다.

  주:

  이는 AEM 옵션을 사용한 경우에 해당합니다. AEM은 소규모 넣기 및 꺼내기 작업에 사용되지 않아야 하므로 최대 CAP 우선 순위는 1입니다. 이 경우 *audit*, *enter* 또는 *eject*에 CAP ID가 별표를 사용한 와일드카드로 지정된 경우 AEM이 선택되지 않도록 합니다. AEM에 대한 자세한 내용은 “AEM 사용 ”을 참조하십시오.
- *cap_id*

  CAP 식별자를 지정합니다. 특정 CAP를 지정해야 합니다. 별표(*)를 지정하여 모든 CAP에 대해 동일한 우선 순위를 설정할 수 없습니다.

사용법

set cap priority 명령을 사용하면 CAP의 자동 선택 우선 순위를 설정할 수 있습니다.

CAP 요청에서 CAP ID에 대해 별표(*)를 지정하면 ACSLS는 요청에 지정된 각 ACS에 대해 우선 순위가 0이 아닌 가장 높은 사용 가능한 CAP를 자동으로 선택합니다.

예

- CAP 0,3,1에 우선 순위 16을 지정하려는 경우

```
set cap priority 16 0,3,1
```

참조:

찾을 정보	참조 항목
라이브러리 카트리지의 실제 인벤토리와 일치하도록 ACSLS 데이터베이스 업데이트	“audit ”
라이브러리에서 카트리지 꺼내기	“eject ”
CAP(수동 모드)가 레이블이 지정된 카트리지를 라이브러리에 넣을 수 있도록 준비	“enter ”
CAP 상태 표시	“query cap ”
라이브러리 구성 요소 상태 변경	“vary ”
CAP가 레이블이 없는 카트리지를 라이브러리에 넣을 수 있도록 준비	“venter ”

명령 영역 메시지

라이브러리 카트리지의 실제 인벤토리와 일치하도록 ACSLS 데이터베이스를 업데이트합니다.

성공 메시지

- *`Set: CAP cap_id, priority changed to cap_priority.`*

  설명: ACSLS가 지정된 CAP의 우선 순위를 변경했습니다.

  변수:
  - *`cap_id`*는 우선 순위가 변경된 CAP입니다.
  - *`cap_priority`*는 CAP의 새 우선 순위입니다.

중간 메시지

없음

오류 메시지

- *`Set: CAP cap_id Set failed, Incorrect attribute.`*

  설명: 잘못된 CAP 우선 순위가 지정되어서 ACSLS가 지정된 CAP의 모드를 변경할 수 없습니다.

  변수: *`cap_id`*는 우선 순위가 변경되지 않은 CAP입니다.

표시 영역 메시지

없음

set clean

set clean 명령은 청소 카트리지 속성을 설정합니다.

모든 최신 청소 카트리지의 경우 *audit*, *enter* 또는 *Cartridge Recovery*를 통해 청소 카트리지가 추가될 때 청소 카트리지 속성이 자동으로 설정됩니다. 여기에는 청소 카트리지 *max_usage*를 설정하는 작업이 포함됩니다.

형식

set clean max_usage | off vol_id | volrange

옵션

- *max_usage | off*

  ACSLS가 드라이브 청소를 위해 카트리지 선택을 중지하기 전에 청소 카트리지가 사용되는 횟수를 지정합니다. *off*는 ACSLS가 카트리지를 선택하지 않고 카트리지를 데이터 카트리지로 재정의하도록 지정합니다.

  주:

  데이터 카트리지 전용으로 예약된 매체 유형에 대해서는 청소 최대 사용 횟수를 지정할 수 없습니다. 청소 카트리지에 해당하는 매체 유형에 대해서만 청소를 *off*로 설정하십시오.

- *vol_id | volrange*

  청소 카트리지 또는 카트리지 범위를 지정합니다.

사용법

set clean 명령을 사용하면 ACSLS가 청소 카트리지를 선택하는 횟수를 설정할 수 있습니다. 또한 set clean을 사용하여 카트리지의 청소 카트리지 속성을 해제할 수 있습니다. 이 작업은 데이터 카트리지를 청소 카트리지로 잘못 정의한 경우 수행합니다.

예

- 청소 카트리지 *CLN108 - CLN112*의 최대 사용 횟수를 10으로 설정하려는 경우

  set clean 10 CLN108-CLN112
- 청소 카트리지 속성을 해제하고 카트리지 *HRR234 - HRR244*를 데이터 카트리지로 재정의하려는 경우

  set clean off HRR234-HRR244

참조:

찾을 정보	참조 항목
드라이브 청소 지침 및 절차	“LSM 채우기 ”

찾을 정보	참조 항목
청소 카트리지 상태 표시	“query clean ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

- *Set: volume vol_id is a cleaning cartridge.*

  설명: ACSLS가 청소 카트리지를 드라이브 청소에 유효한 것으로 지정했습니다.

  변수: *vol_id*는 유효한 청소 카트리지입니다.
- *Set: volume vol_id is not a cleaning cartridge.*

  설명: ACSLS가 청소 카트리지를 드라이브 청소에 유효하지 않은 것으로 지정했습니다.

  변수: *vol_id*는 잘못된 청소 카트리지입니다.

중간 메시지

없음

오류 메시지

- *Set: Clean vol_id Set failed, Incorrect attribute.*

  설명: 지정된 카트리지가 청소 카트리지가 아니므로 ACSLS가 청소 속성을 변경할 수 없습니다.

  변수: *vol_id*는 데이터 또는 스크래치 카트리지입니다.

표시 영역 메시지

없음

set lock

set lock 명령은 잠금 ID를 설정합니다.

형식

set lock lock_id

옵션

- *lock_id*

잠금 ID를 지정합니다. 유효한 잠금 ID는 0 ~ 32767입니다.

사용법

set lock 명령을 사용하면 잠금 ID를 설정하거나 변경할 수 있습니다. 다음과 같이 잠금 ID가 현재 잠금 ID와 일치하지 않는 드라이브 또는 카트리지에서 잠금을 제거하려는 경우 현재 잠금 ID를 변경하십시오.

- 지정된 드라이브 또는 카트리지의 모든 잠금을 지우려는 경우 잠금 ID를 0으로 설정한 다음 *clear lock* 명령을 입력합니다.
- 드라이브 또는 카트리지의 활성 잠금을 제거하려는 경우 잠금 ID를 잠긴 구성 요소의 잠금 ID로 설정한 다음 *unlock* 명령을 입력합니다.

주:

lock 명령을 입력하여 카트리지나 드라이브를 잠그고 잠금 ID가 0인 경우 ACSLS는 카트리지 또는 드라이브에 잠금 ID를 지정한 다음 잠금 ID를 카트리지 또는 드라이브의 잠금 ID로 변경합니다. *set lock* 명령을 사용하여 잠금 ID를 설정한 다음 *lock* 명령을 사용하여 *set lock*으로 설정한 잠금 ID를 가진 카트리지 또는 드라이브를 잠글 수 없습니다.

예

- 현재 잠금 ID를 새 잠금 ID 354로 변경하려는 경우

```
set lock 354
```

참조:

찾을 정보	참조 항목
지정한 드라이브 또는 카트리지의 모든 활성 또는 보류 중인 잠금 제거	“clear lock ”
드라이브 및 카트리지 잠금	“lock ”
드라이브 또는 카트리지의 잠금 상태 표시	“query lock ”
잠금 또는 사용자 ID 표시	“show ”
활성 잠금 제거	“unlock ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

- *Set: Changed lock identifier from previous_lock_id to lock_id.*

  설명: ACSLS가 잠금 ID를 변경했습니다.

  변수:

- *previous_lock_id*는 이전 잠금 ID입니다.
- *lock_id*는 새 잠금 ID입니다.

중간 메시지

없음

오류 메시지

없음

표시 영역 메시지

없음

set owner

set owner 명령은 볼륨 소유권을 설정합니다.

형식

set owner owner_id volume vol_id | volrange

옵션

- *owner_id*

  소유자 식별자를 지정합니다. 이 값을 따옴표(" ")로 묶어야 합니다.
- *volume vol_id | volrange*

  볼륨 또는 볼륨 범위를 지정합니다.

사용법

set owner 명령을 사용하면 볼륨 소유권을 설정할 수 있습니다. *cmd_proc*에서 set owner를 입력하십시오. 클라이언트 응용 프로그램에서는 볼륨 소유권을 설정할 수 없습니다.

예

- 볼륨 YUMA06의 소유권을 지정하려는 경우

  set owner "cray" volume YUMA06

  주:

  볼륨 YUMA06의 볼륨 소유권을 제거하려는 경우

  set owner:"" volume 0YUMA06

명령 영역 메시지

없음

표시 영역 메시지

없음

set scratch

set scratch 명령은 볼륨의 스크래치 속성을 설정하거나 지우고 볼륨을 풀에 지정합니다.

또한 *audit*, *enter* 또는 *Cartridge Recovery*로 스크래치 카트리지가 추가되거나 재활성화될 때 *watch_vols* 유틸리티를 사용하여 스크래치 카트리지 속성을 자동으로 설정할 수 있습니다.

형식

set scratch [off] pool_id vol_id | volrange

옵션

- *off*

  볼륨이 데이터 카트리지임을 지정합니다.
- *pool_id*

  카트리지의 스크래치 풀을 지정합니다. 데이터 카트리지를 현재 풀에 다시 지정하려면 별표(*)를 지정하십시오.
- *vol_id | volrange*

  볼륨 또는 볼륨 범위를 지정합니다.

사용법

set scratch 명령을 사용하면 볼륨의 스크래치 속성을 설정하거나 지우고 볼륨을 풀에 지정할 수 있습니다.

예

- 볼륨 YUMA10-YUMA20을 스크래치 볼륨으로 정의하고 스크래치 풀 5에 지정하려는 경우

  set scratch 5 YUMA10-YUMA20
- 스크래치 볼륨 YUMA10-YUMA15를 풀 10으로 이동하려는 경우

  set scratch 10 YUMA10-YUMA15

- 볼륨 YUMA16-YUMA20을 "스크래치 해제"(스크래치에서 데이터로 변경)하고 공통 풀(풀 0)로 이동하려는 경우

  set scratch off 0 YUMA16-YUMA20
- 데이터 볼륨 YUMA16-YUMA20으로 스크래치하고 현재 풀에 유지하려는 경우

  *set scratch * YUMA16-YUMA20*

참조:

찾을 정보	참조 항목
스크래치 카트리지 상태 표시	“query scratch ”
스크래치 풀 속성 표시	“query pool ”
스크래치 풀 만들기 또는 수정	“define pool ”
빈 스크래치 풀 삭제	“delete pool ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

- *Set: volume vol_id in tape pool pool_id is a scratch cartridge.*

  설명: ACSLS가 지정된 데이터 카트리지를 스크래치 카트리지로 변경하고 풀에 지정했습니다.

  변수:
  - *vol_id*는 지정된 볼륨입니다.
  - *pool_id*는 볼륨이 지정된 풀입니다.
- *Set: volume vol_id in tape pool pool_id is a data volume.*

  설명: ACSLS가 지정된 스크래치 카트리지를 데이터 카트리지로 변경했습니다.

  변수: *vol_id*는 지정된 볼륨입니다.

중간 메시지

없음

오류 메시지

- *Set: Scratch vol_id Set failed, Incorrect attribute.*

  설명: 지정된 카트리지가 청소 카트리지이므로 ACSLS가 스크래치 속성을 변경할 수 없습니다.

변수: *vol_id*는 청소 카트리지입니다.

표시 영역 메시지

- *Pool pool_id: low water mark warning.*

  설명: 지정된 스크래치 풀의 카트리지 수가 하위 임계값보다 작거나 같습니다.

  변수: *low_water_mark*는 지정된 스크래치 풀의 하위 임계값입니다.
- *Pool pool_id: high water mark warning.*

  설명: 지정된 스크래치 풀의 카트리지 수가 상위 임계값보다 크거나 같습니다.

  변수: *high_water_mark*는 지정된 스크래치 풀의 상위 임계값입니다.

show

show 명령은 잠금 ID 또는 사용자 ID를 표시합니다.

형식

show type

옵션

- *type*

  다음 유형 중 하나를 지정합니다.
- *lock*

  잠금 ID입니다.
- *user*

  사용자 ID입니다.

사용법

show 명령을 사용하면 잠금 ID 또는 사용자 ID를 표시할 수 있습니다.

예

- 요청자의 *user_id*를 표시하려는 경우

  show user
- 현재 *lock_id*를 표시하려는 경우

show lock

참조:

찾을 정보	참조 항목
지정한 드라이브 또는 카트리지의 모든 활성 또는 보류 중인 잠금 제거	“clear lock ”
드라이브 및 카트리지 잠금	“lock ”
드라이브 또는 카트리지의 잠금 상태 표시	“query lock ”
잠금 ID 설정	“set lock ”
활성 잠금 제거	“unlock ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

- *Show: Current lock identifier is lock_id*

  설명: 현재 잠금 ID가 *lock_id*입니다.
- *Show: User identifier is user_id*

  설명: 현재 사용자 ID가 *user_id*입니다.

중간 메시지

없음

오류 메시지

없음

표시 영역 메시지

없음

start

start 명령은 ACSLS 요청 처리를 시작합니다.

형식

start

옵션

없음

사용법

start 명령을 사용하면 ACSLS를 실행 상태로 전환하고 ACSLS 요청 처리를 시작할 수 있습니다. ACSLS가 유휴 상태인 경우 일반적으로 start 명령을 사용하여 요청 처리를 다시 시작합니다.

예

- ACSLS 요청 처리를 다시 시작하려는 경우

  start

ACSLS가 실행 상태인 경우 *start* 명령을 입력해도 아무런 영향을 주지 않습니다.

참조:

찾을 정보	참조 항목
ACSLS의 새 요청 처리 중지	“idle ”
라이브러리 구성 요소의 상태 표시	“query 명령 ”
라이브러리 구성 요소 상태 변경	“vary ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

- *ACSLM Request Processing Started: Success.*

  설명: ACSLS가 요청 처리를 시작했습니다.

중간 메시지

없음

오류 메시지

- *ACSLM Request Processing Not Started: status*

  설명: ACSLS가 요청 처리를 시작하지 않았습니다.

  변수: *status*는 오류의 원인입니다. 일반 상태 메시지에 대한 자세한 내용은 ACSLS Messages를 참조하십시오.

표시 영역 메시지

- *Server system running*

  설명: ACSLS가 요청 처리를 시작했습니다.

switch lmu

RE(중복 전자 부품) 또는 이중 LMU 구성에서 *switch lmu* 명령을 사용하면 ACS 관리를 수동으로 활성 LC(라이브러리 컨트롤러)에서 대기 LC로 전환하거나 ACS의 활성 LMU에서 대기 LMU로 전환합니다. 다음 사항에 유의하십시오.

- switch lmu는 단일 RE 라이브러리가 있는 ACS(SL3000 또는 독립형 SL8500)만 지원합니다.
- switch lmu는 분할된 SL8500 또는 SL3000을 지원하지 않습니다.
- ACSLS는 호스트/라이브러리 인터페이스 호환성 레벨 11 이상이 로드된 9330 LMU에 대해서만 이중 LMU 구성을 지원합니다. 두 LMU에 동일한 마이크로코드 레벨을 로드해야 합니다.

형식

switch lmu acs_id

옵션

- *acs_id*

  라이브러리 관리를 활성에서 대기 LC 또는 LMU로 전환할 ACS를 지정합니다.

사용법

switch lmu 명령을 사용하면 라이브러리 관리를 라이브러리의 활성 LC 또는 LMU에서 대기 LC 또는 LMU로 수동으로 전환할 수 있습니다. switch lmu 명령을 입력하기 전에 다음 사항을 확인하십시오.

- ACSLS가 실행 상태입니다.
- 지정하는 ACS가 온라인 또는 진단 상태입니다.
- 각 LC 또는 LMU에 대해 하나 이상의 포트가 온라인 상태입니다.

예

RE 또는 이중 LMU 구성에서 *switch lmu* 명령을 사용하면 ACS 관리를 라이브러리의 활성 LC 또는 LMU에서 대기 LC 또는 LMU로 전환할 수 있습니다. 다음 RE 구성을 사용한다고 가정합니다.

- 라이브러리에서 LC A는 활성 역할이고 LC B는 대기 역할입니다.

ACSLS가 LC A와 통신이 끊어져도 LC B와 통신할 수 있는 경우에는 switch lmu를 사용하여 LC B를 활성 LC로 설정하십시오.

참조:

찾을 정보	참조 항목
ACS 상태 표시	“query acs ”
세부정보와 함께 ACS 및 포트 상태 표시	“query lmu ”
포트 상태 표시	“query port ”
ACSLS 및 라이브러리 상태 표시	“query server ”
ACSLS 요청 처리 시작	“start ”
라이브러리 구성 요소 상태 변경	“vary ”
중복 전자 부품	“개요 ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

- Switch: Switch lmu completed for acs_id, Success.

  설명: ACSLS가 라이브러리 관리를 활성에서 대기 LMU로 전환했습니다.

  변수: *acs_id*는 LC 또는 LMU가 역할을 전환한 ACS입니다.
- *Switch: Switch lmu initiated for acs_id, Success.*

  설명: ACSLS가 라이브러리 관리를 활성에서 대기 LC 또는 LMU로 전환하기 시작했습니다.

  변수: *acs_id*는 LC 또는 LMU가 역할을 전환하고 있는 ACS입니다.
- *Switch: Switch lmu failed, acs_id not found.*

  설명: 잘못된 ACS를 지정했습니다.

  변수: *acs_id*는 *switch lmu* 명령에 지정한 ACS 식별자입니다.
- *Switch: Switch lmu failed, acs_id is offline.*

  설명: 오프라인 상태의 ACS를 지정했습니다. *vary* 명령을 사용하면 ACS를 온라인 또는 진단 모드로 전환할 수 있습니다.

  변수: *acs_id*는 *switch lmu* 명령에 지정한 ACS 식별자입니다.
- *Switch: Switch lmu failed, acs_id not configured for dual lmu.*

  설명: 활성 및 대기 LC 또는 LMU로 구성되지 않은 ACS를 지정했습니다. 다음 중 하나를 수행합니다.

- 지정한 ACS를 재구성한 다음 *switch lmu* 명령을 다시 입력합니다.
- 활성 및 대기 LC 또는 LMU로 구성된 라이브러리를 지정하는 *switch lmu* 명령을 입력합니다.

변수: *acs_id*는 *switch lmu* 명령에 지정한 ACS 식별자입니다.

- *Switch: Switch lmu failed, not communicating.*

  설명: 대기 LC 또는 LMU가 통신 중이 아니라 switch lmu 전환이 실패했습니다. 대기 LC 또는 LMU에 통신 문제가 있는지 확인하십시오.

- *Switch: Switch lmu failed, switch already active.*

  설명: 지정한 ACS에 대해 switch lmu가 진행 중입니다. 전환이 완료될 때까지 기다린 다음 switch lmu 명령을 다시 입력하십시오.

- *Switch: Switch lmu failed, port is offline.*

  설명: 온라인 상태의 포트가 없는 ACS를 지정했습니다. 각 LC 또는 LMU에 대해 하나 이상의 포트를 온라인 상태로 전환(*Vary*)하십시오.

표시 영역 메시지

없음

unlock

unlock 명령은 지정된 드라이브 또는 카트리지의 활성 잠금(현재 잠금 ID와 연관됨)을 제거하거나 모든 활성 잠금을 제거합니다.

형식

unlock type identifier...| all

옵션

- *type identifier*

  라이브러리 구성 요소를 지정합니다. 다음 표에는 잠금 해제할 수 있는 구성 요소가 나와 있습니다.

표 13.6. Unlock에 유효한 구성 요소

라이브러리 구성 요소	유형	식별자
드라이브	*drive*	*drive_id*
볼륨	*volume*	*vol_id*

- *all*

  모든 활성 잠금을 지정합니다.

사용법

unlock 명령을 사용하면 지정된 카트리지 및 드라이브의 활성 잠금을 제거하거나 모든 활성 잠금을 제거할 수 있습니다. 구성 요소의 *lock ID*가 현재 잠금 ID와 일치해야 합니다.

사용 중이 아닌 구성 요소를 잠금 해제하면 ACSLS가 구성 요소에 보류 중인 잠금이 있는지 확인합니다. 보류 중인 잠금이 있으면 ACSLS가 보류 중인 잠금의 잠금 ID를 사용하여 구성 요소를 잠급니다.

주:

unlock 명령은 항상 잠금 ID를 0으로 재설정합니다.

예

- 드라이브 0,0,2,0의 활성 잠금을 제거하려는 경우

  ```
  unlock drive 0,0,2,0
  ```
- 모든 잠긴 카트리지의 활성 잠금을 제거하려는 경우

  ```
  unlock volume all
  ```

참조:

찾을 정보	참조 항목
지정한 드라이브 또는 볼륨의 모든 활성 또는 보류 중인 잠금 제거	“clear lock ”
드라이브 및 볼륨 잠금	“lock ”
드라이브 또는 카트리지의 잠금 상태 표시	“query lock ”
잠금 ID 설정	“set lock ”
잠금 또는 사용자 ID 표시	“show ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

- *Unlock: Unlock completed, Success.*
- *Unlock: drive drive_id unlocked.*

  설명: ACSLS가 지정된 드라이브를 잠금 해제했습니다.

  변수: *drive_id*는 잠금 해제된 드라이브입니다.
- *Unlock: Unlock completed, Success.*
- *Unlock: volume vol_id unlocked.*

  설명: ACSLS가 지정된 볼륨을 잠금 해제했습니다.

변수: *vol_id*는 잠금 해제된 볼륨입니다.

중간 메시지

없음

오류 메시지

- *Unlock: Unlock of drive drive_id failed, status.*

  설명: ACSLS가 드라이브를 잠금 해제할 수 없습니다.

  변수: status는 오류의 원인입니다. 일반 상태 메시지에 대한 자세한 내용은 ACSLS Messages를 참조하십시오.
- *Unlock: Unlock of volume vol_id failed, status.*

  설명: ACSLS가 볼륨을 잠금 해제할 수 없습니다.

  변수: *status*는 오류의 원인입니다. 일반 상태 메시지에 대한 자세한 내용은 ACSLS Messages를 참조하십시오.

표시 영역 메시지

없음

vary

vary 명령은 라이브러리 구성 요소의 상태를 변경합니다.

형식

vary type identifier...state [force]

옵션

- *type [identifier]*

  라이브러리 구성 요소를 지정합니다. 다음 표에는 전환할 수 있는 구성 요소가 나와 있습니다.

표 13.7. Vary에 유효한 구성 요소

ACS 구성 요소	유형	식별자
ACS	*acs*	*acs_id*
LSM	*lsm*	*acs_id,lsm_id*
CAP	*cap*	*cap_id*
드라이브	*drive*	*drive_id*
포트	*port*	*port_id*

- *state*

  다음 장치 상태 중 하나를 지정합니다. *online*(클라이언트 및 *cmd_proc*에 대해 온라인으로 전환), *offline* 또는 *diagnostic*(*cmd_proc*에 대해서만 온라인으로 전환)

  ACS, LSM, CAP 또는 드라이브를 온라인, 오프라인 또는 진단 상태로 전환(*vary*)할 수 있습니다. 포트를 온라인 또는 오프라인 상태로 전환(*vary*)할 수 있습니다. 장치 상태에 대한 자세한 내용은 "사용법 "을 참조하십시오.
- *force*

  현재 로봇 요청만 처리한 후 ACS, LSM, CAP 또는 드라이브를 *offline* 상태로 전환합니다.

사용법

vary 명령을 사용하면 ACS, LSM, CAP, 드라이브 또는 포트의 상태를 변경할 수 있습니다. 다음 절에서는 각 장치 상태가 라이브러리 구성 요소에 어떤 영향을 주는지에 대해 설명합니다.

- *vary offline*

  다음 표에서는 각 ACS 구성 요소에 대한 *vary offline*의 결과를 보여줍니다. 구성 요소를 오프라인 상태로 전환하면 먼저 오프라인 보류 중 상태로 전환됩니다. 그러면 ACSLS가 구성 요소에 대한 모든 활성 및 보류 중인 요청을 처리하고 새 요청을 거부한 다음 구성 요소를 오프라인 상태로 전환합니다.

표 13.8. Vary Offline 결과

ACS 구성 요소	결과
ACS	ACS 및 하위 구성 요소가 오프라인 보류 중 상태로 전환된 다음 오프라인 상태로 전환됩니다. ACS가 오프라인 상태로 전환되기 전에 LSM이 오프라인 상태로 전환되어야 합니다.
LSM	LSM이 오프라인 보류 중 상태로 전환된 다음 오프라인 상태로 전환됩니다.
CAP	CAP가 오프라인 보류 중 상태로 전환된 다음 오프라인 상태로 전환됩니다.
드라이브	사용 가능한 경우 드라이브가 즉시 오프라인 상태로 전환됩니다. 사용 중인 경우 드라이브가 온라인 상태로 유지됩니다.
포트	다음과 같은 경우 포트가 즉시 오프라인 상태로 전환됩니다. • ACS에 온라인 상태의 다른 포트가 있는 경우 또는 • ACS가 오프라인 상태인 경우 그렇지 않은 경우 포트가 온라인 상태로 유지됩니다.

- *vary offline force*

  다음 표에는 각 ACS 구성 요소에 대한 *vary offline force*의 결과가 나와 있습니다. 이 옵션은 장치가 확장 작업(예: 감사 중)에 사용되는 동안 장치를 오프라인 상태로 전환해야 하는 경우 유용합니다.

표 13.9. Vary Offline Force 결과

ACS 구성 요소	결과
ACS	현재 로봇 요청만 완료되고 ACS 및 하위 구성 요소가 즉시 오프라인 상태로 전환됩니다. 보류 중인 요청이 삭제되고 새 요청이 거부됩니다. ACS가 오프라인 상태로 전환되기 전에 LSM이 오프라인 상태로 전환되어야 합니다.
LSM	현재 로봇 요청만 완료되고 LSM이 즉시 오프라인 상태로 전환됩니다. 보류 중인 요청이 삭제되고 새 요청이 거부됩니다.
CAP	현재 로봇 요청만 완료되고 CAP가 즉시 오프라인 상태로 전환됩니다. 보류 중인 요청이 삭제되고 새 요청이 거부됩니다.
드라이브	현재 로봇 요청만 완료되고 드라이브가 즉시 오프라인 상태로 전환됩니다. 보류 중인 요청이 삭제되고 새 요청이 거부됩니다.
포트	유효하지 않습니다.

- *`vary diagnostic`*

  `vary diagnostic` 요청은 지정된 구성 요소를 진단 상태(*`cmd_proc`*에 대해서만 온라인으로 전환)로 전환합니다. ACSLS가 구성 요소에 대한 모든 활성 및 보류 중인 요청을 처리하고 새 클라이언트 응용 프로그램 요청을 거부한 다음 구성 요소를 진단 상태로 전환합니다. ACS의 경우 모든 하위 LSM도 진단 상태로 전환됩니다.
- *`vary online`*

  다음 표에는 각 ACS 구성 요소에 대한 *`vary online`*의 결과가 나와 있습니다. 구성 요소를 온라인 상태로 전환(*`vary`*)하면 먼저 복구 상태로 전환됩니다. 그러면 ACSLS가 구성 요소에 대한 모든 활성 및 보류 중인 요청을 처리하고 새 요청을 거부한 다음 구성 요소를 온라인 상태로 전환합니다. 구성 요소가 온라인 상태로 전환되면 ACSLS가 구성 요소에 대한 모든 요청을 처리합니다.

표 13.10. Vary Online 결과

ACS 구성 요소	결과
ACS	ACS가 오프라인 상태이면 ACS 및 해당 LSM이 복구 상태로 전환된 다음 온라인 상태로 전환됩니다. ACS가 진단 상태이면 ACS 및 해당 LSM이 즉시 온라인 상태로 전환됩니다.
LSM	LSM이 복구 상태로 전환된 다음 온라인 상태로 전환됩니다. LSM이 오프라인 ACS에 연결된 경우에는 LSM을 온라인 상태로 전환할 수 없습니다.
CAP	CAP가 복구 상태로 전환된 다음 온라인 상태로 전환됩니다.
드라이브	드라이브가 복구 상태로 전환된 다음 온라인 상태로 전환됩니다.
포트	포트가 즉시 온라인 상태로 전환됩니다.

예

- 드라이브 0,0,9,3을 오프라인 상태로 전환(*`vary`*)하려는 경우

  `vary drive 0,0,9,3 offline`
- CAP 0,0,0을 진단 상태로 전환(*`vary`*)하려는 경우

  `vary cap 0,0,0 diagnostic`

- 0,1을 오프라인 상태로 *force lsm*하려는 경우

  vary lsm 0,1 offline force

  주:

  시스템을 IPLing해도 이러한 구성 요소의 상태는 변경되지 않습니다. ACSLS를 설치하거나 재구성하면 가능한 경우 모든 구성 요소가 온라인 상태로 전환됩니다.

  ACSLS가 동일한 라이브러리에 있는 둘 이상의 분할 영역을 관리하는 경우 별도의 *vary* 명령을 사용하여 동일한 CAP(다른 ACS에 있는 것으로 식별됨)를 전환하십시오.

  예: ACS 0과 ACS 1이 동일한 라이브러리의 두 분할 영역인 경우 각 분할 영역 및 각 ACS에 대해 독립적으로 CAP를 온라인 또는 오프라인 상태로 전환(*vary*)하십시오. 동일한 CAP를 한 분할 영역(ACS)에 대해 온라인 상태로 전환하고 다른 분할 영역(ACS)에 대해 오프라인 상태로 전환할 수 있습니다.

참조:

찾을 정보	참조 항목
라이브러리 구성 요소의 상태 표시	"query 명령 "

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

- *Vary: type identifier Varied state*

  설명: ACSLS가 지정된 라이브러리 구성 요소의 상태를 변경했습니다.

  변수:
  - *type*는 라이브러리 구성 요소 유형입니다.
  - *identifier*는 라이브러리 구성 요소 식별자입니다.
  - *state*는 라이브러리 구성 요소의 새 상태입니다.

중간 메시지

없음

오류 메시지

- *In-transit volume recovery incomplete.*

  설명: LSM 또는 해당 ACS가 온라인 상태로 전환된 동안에는 LSM 로봇이 이동 중인 볼륨을 폐기할 수 없습니다. ACS를 온라인 상태로 전환하는 경우 ACS 상태가 즉시 온라인 상태로 변경되지만 이동 중인 볼륨 복구를 완료할 수 없는 모든 LSM은 오프라인 상태로 유지됩니다.
- *Vary: Vary type identifier failed, Drive in use.*

설명: 명령에 사용 중인 드라이브가 포함되어서 ACSLS가 지정된 라이브러리 구성 요소의 상태를 변경할 수 없습니다.

변수:

- *type*는 라이브러리 구성 요소 유형입니다.
- *identifier*는 라이브러리 구성 요소 식별자입니다.

- *Vary: Vary type identifier failed, Incorrect lockid*

설명: *drive lock_id*가 *lock_id*와 일치하지 않아서 ACSLS가 지정된 라이브러리 구성 요소의 상태를 변경할 수 없습니다.

변수:

- *type*는 라이브러리 구성 요소 유형입니다.
- *identifier*는 라이브러리 구성 요소 식별자입니다.

- *Vary: Vary type identifier failed, State unchanged.*

설명: 구성 요소가 이미 요청된 상태여서 ACSLS가 지정된 라이브러리 구성 요소의 상태를 변경할 수 없습니다.

변수:

- *type*는 라이브러리 구성 요소 유형입니다.
- *identifier*는 라이브러리 구성 요소 식별자입니다.

- *Vary: Vary type identifier failed, Vary disallowed.*

설명: 클라이언트 응용 프로그램에서 *vary diagnostic* 요청이 실행되어서 ACSLS가 지정된 라이브러리 구성 요소의 상태를 변경할 수 없습니다.

변수:

- *type*는 라이브러리 구성 요소 유형입니다.
- *identifier*는 라이브러리 구성 요소 식별자입니다.

- *Vary: Vary type identifier failed, Incorrect state.*

설명: 다음과 같이 전환(*vary*)하려고 했기 때문에 ACSLS가 지정된 라이브러리 구성 요소의 상태를 변경할 수 없습니다.

- LSM 또는 ACS가 진단 또는 오프라인 상태일 때 CAP를 온라인 상태로 전환하려고 함
- LSM 또는 ACS가 오프라인 상태일 때 CAP를 진단 상태로 전환하려고 함
- ACS가 진단 또는 오프라인 상태일 때 LSM을 온라인 상태로 전환하려고 함
- ACS가 오프라인 상태일 때 LSM을 진단 상태로 전환하려고 함

변수:

- *type*는 라이브러리 구성 요소 유형입니다.
- *identifier*는 라이브러리 구성 요소 식별자입니다.

- *Vary: Vary type identifier failed, Vary in progress.*

설명: 다음과 같은 이유로 ACSLS가 지정된 라이브러리 구성 요소의 상태를 변경할 수 없습니다.

- CAP 및 해당 LSM 또는 ACS가 이동 중(복구 또는 오프라인 보류 중) 상태입니다.
- LSM 및 해당 ACS 또는 하위 CAP가 이동 중 상태입니다.
- ACS 및 하위 LSM 또는 하위 CAP가 이동 중 상태입니다.
- ACS, LSM 또는 CAP를 진단 또는 온라인 상태로 전환하려고 했지만 *vary offline/force* 요청이 시도를 대체했습니다.
- 드라이브가 복구의 이동 중 상태입니다.

변수:

- *type*는 라이브러리 구성 요소 유형입니다.
- *identifier*는 라이브러리 구성 요소 식별자입니다.

- *Vary: Vary type identifier failed, State unchanged.*

  설명: ACS, LSM 또는 CAP를 오프라인 상태로 전환(*vary*)하려고 시도했지만 *vary offline/force* 요청이 이 시도를 대체해서 ACSLS가 지정된 라이브러리 구성 요소의 상태를 변경할 수 없습니다.

  변수:

  - *type*는 라이브러리 구성 요소 유형입니다.
  - *identifier*는 라이브러리 구성 요소 식별자입니다.

- *Vary: Vary type identifier failed, ACS online.*

  설명: 온라인 ACS에서 마지막 포트를 오프라인 상태로 전환(*vary*)하려고 시도해서 ACSLS가 지정된 라이브러리 구성 요소의 상태를 변경할 수 없습니다.

  변수:

  - *type*는 라이브러리 구성 요소 유형입니다.
  - *identifier*는 라이브러리 구성 요소 식별자입니다.

- *Vary: Vary type identifier failed, No ports online.*

  설명: ACS에 온라인 상태의 포트가 없어서 ACSLS가 지정된 라이브러리 구성 요소의 상태를 변경할 수 없습니다.

  변수:

  - *type*는 라이브러리 구성 요소 유형입니다.
  - *identifier*는 라이브러리 구성 요소 식별자입니다.

표시 영역 메시지

ACS, LSM, 드라이브 또는 포트의 상태가 변경되면 다음 메시지 중 하나가 나타납니다.

- *ACS acs_id: state*

- *LSM lsm_id: state*
- *Drive drive_id: state*
- *Port port_id: state*

이동 중 볼륨을 복구하는 동안 카트리지 레이블을 읽을 수 없거나 레이블이 중복되면 다음 메시지가 나타납니다. 카트리지는 CAP로 이동됩니다.

CAP cap_id. Remove volumes from CAP.

- *cap_id*

  카트리지가 포함된 CAP입니다.

venter

venter 명령은 CAP가 레이블이 없는 카트리지를 라이브러리에 넣을 수 있도록 준비합니다.

> **주:**
>
> L5500은 매체 레이블이 없는 카트리지를 지원하지 않습니다. SL500, SL3000 및 SL8500 라이브러리는 레이블이 없는 카트리지(venter)를 지원하지 않습니다. 이러한 라이브러리는 volser 레이블이 없는 매체 레이블을 읽을 수 없으며 ACSLS는 알 수 없는 매체 유형의 카트리지를 venter하지 않기 때문입니다.

형식

venter cap_id vol_id

옵션

- *cap_id*

  카트리지를 넣을 CAP를 지정합니다.
- *vol_id*

  넣을 카트리지의 가상 레이블을 지정합니다.

사용법

venter 명령을 사용하면 CAP가 레이블이 누락되거나 레이블을 읽을 수 없는 카트리지를 라이브러리에 넣을 수 있도록 준비할 수 있습니다. *venter* 명령에 가상 레이블(볼륨 ID)을 지정합니다. 이렇게 하면 카트리지 정보가 ACSLS 데이터베이스에 추가됩니다. 한 번에 1 ~ 42개의 카트리지를 넣을 수 있습니다.

매체 유형이 "3480"이 아니면 별도의 매체 유형 레이블이 없는 카트리지를 *venter*할 수 없습니다.

주의:

venter할 카트리지에 매체 레이블이 있고 카트리지의 매체 레이블이 SCSI 라이브러리에 연결된 드라이브와 호환되는 경우에만 ACSLS가 SL500보다 먼저 SCSI/광 섬유 연결 LSM에 대해 *venter* 명령을 지원합니다. 카트리지의 매체 유형이 3480이 아니면 별도의 매체 레이블이 없는 카트리지를 venter할 수 없습니다.

venter 명령을 사용하면 레이블이 누락되거나 레이블을 읽을 수 없는 카트리지를 넣을 수 있습니다. LSM 도어를 열고 누락되거나 읽을 수 없는 레이블이 있는 카트리지를 스토리지 셀에 놓지 마십시오. ACSLS에서 이러한 카트리지를 관리할 수 없습니다. *audit* 중 ACSLS는 레이블이 누락되거나 레이블을 읽을 수 없는 카트리지 및 가상 레이블이 없는 카트리지를 꺼냅니다.

venter 명령은 넣을 카트리지의 매체 유형을 지정할 수 있는 옵션을 제공하지 않습니다. 혼합 매체 환경에서는 ACSLS가 가상으로 넣은 카트리지에 대해 드라이브/매체 비호환성을 방지할 수 없습니다.

예

- 볼륨 *MAINT1* 및 *MAINT2*에 가상 레이블을 지정하고 이러한 볼륨을 CAP 0,2,2를 통해 *enter*를 수행하려는 경우

  ```
  venter 0,2,2 MAINT1 MAINT2
  ```

참조:

찾을 정보	참조 항목
명령 취소	“cancel ”
스크래치 풀 만들기 또는 수정	“define pool ”
라이브러리에서 카트리지 꺼내기	“eject ”
CAP(수동 모드)가 레이블이 지정된 카트리지를 라이브러리에 넣을 수 있도록 준비	“enter ”
드라이브에 스크래치 카트리지 마운트	“mount * ”
드라이브에 데이터 카트리지 마운트	“mount ”
CAP 상태 표시	“query cap ”
CAP 모드(수동 또는 자동) 설정	“set cap mode ”
CAP 선택 우선 순위 설정	“set cap priority ”

명령 영역 메시지

이 절에서는 명령 영역 메시지에 대해 설명합니다.

성공 메시지

- *Venter: Enter complete, nn cartridges entered.*

  설명: ACSLS가 표시된 수의 카트리지를 넣었습니다.

변수: *nn*은 넣은 총 카트리지 수입니다.

- *Venter: vol_id Entered through cap_id*

  설명: ACSLS가 지정된 가상 카트리지를 넣었습니다.

  변수:
  - *vol_id*는 가상 카트리지 레이블입니다.
  - *cap_id*는 카트리지를 넣을 때 사용된 CAP입니다.

중간 메시지

없음

오류 메시지

- *Venter: vol_id Enter failed, status*

  설명: ACSLS가 지정된 가상 카트리지를 넣지 않았습니다.

  변수:
  - *vol_id*는 가상 카트리지 레이블입니다.
  - *status*는 ACSLS가 반환한 요청의 변환된 완료 상태입니다.

표시 영역 메시지

- *CAP cap_id: Place cartridges in CAP.*

  설명: CAP가 카트리지를 넣을 준비가 되었습니다. CAP를 열고 카트리지 넣기(*enter*)를 수행하십시오.

  변수: *cap_id*는 카트리지를 넣는 데 사용된 CAP입니다.
- *CAP cap_id: Unknown media type label.*

  설명: 매체 유형을 알 수 없어서 하나 이상의 카트리지를 넣을 수 없습니다.

  변수: *cap_id*는 카트리지를 넣는(*enter*) 데 사용된 CAP입니다. CAP를 열고 카트리지를 제거하십시오.

14

14장. display 명령 참조

이 장에서는 *display* 명령을 사용하는 방법에 대해 배웁니다.

SQL의 관점에서 이 옵션을 검토하면 *display* 명령의 강력한 기능과 유연성을 발견하게 될 것입니다. *display* 명령은 ACSLS 데이터베이스에서 정보를 보고하기 위한 강력한 도구입니다. SQL SELECT 문과 마찬가지로 display를 사용하여 다음을 지정할 수 있습니다.

- 표시할 데이터베이스 테이블(ACSLS 객체)
- 와일드카드
- 객체 선택 조건
- 보고할 필드와 보고 순서
- 출력을 표시할 정렬 순서
- 선택 조건을 충족하는 객체의 수만 표시하도록 설정

결과는 표시할 항목과 표시 방법을 사용자 정의할 수 있는 *query* 기능입니다. *display* 명령을 사용하면 ACSLS *query* 명령을 사용할 때보다 훨씬 더 많은 정보를 확인할 수 있습니다. 예를 들어, *display volume*을 사용하면 사용자 정의 *volrpt. display lsm*을 통해서만 확인 가능한 볼륨 정보에 액세스할 수 있고 *display drive*는 LSM 및 테이프 드라이브 일련 번호를 제공합니다.

display 명령은 다음과 같습니다.

- *"display cap "*

  선택한 옵션에 따라 특정 CAP 정보를 표시합니다.
- *"display cell "*

  선택한 옵션에 따라 특정 셀 정보를 표시합니다.
- *"display drive "*

  선택한 옵션에 따라 특정 드라이브 정보를 표시합니다.
- *"display lock "*

  사용자 ID에 따라 특정 잠금 정보를 표시합니다.
- *"display lsm "*

  선택한 옵션에 따라 특정 LSM 정보를 표시합니다.

- *“display panel ”*

  선택한 옵션에 따라 특정 라이브러리 패널 정보를 표시합니다.
- *“display pool ”*

  선택한 옵션에 따라 특정 스크래치 풀 정보를 표시합니다.
- *“display port ”*

  선택한 옵션에 따라 특정 포트 정보를 표시합니다.
- *“display volume ”*

  선택한 옵션에 따라 특정 볼륨 정보를 표시합니다.

display 명령 옵션 사용

display 명령은 여러 라이브러리 구성 요소에 대한 다양한 정보 필드를 표시합니다. *display* 명령을 사용하여 출력을 표시할 순서와 형식을 선택할 수 있습니다.

표시되는 정보에는 여러 열과 긴 행이 포함될 수 있습니다. 행의 줄 바꿈을 방지하려면 표시에 사용되는 터미널 창의 크기를 조정합니다. *cmd_proc -lq* 창이 최상의 선택일 수 있습니다.

와일드카드, 숫자 또는 알파 범위, 선택 목록 등을 *display* 명령과 함께 사용할 수 있습니다.

와일드카드 문자 사용

- * 와일드카드 문자를 특정 *display* 피연산자 또는 복합 display 피연산자의 하위 필드 하나 이상 대신 사용하거나 볼륨 ID의 시작 또는 끝에 사용할 수 있습니다. 하나 이상의 문자와 일치해야 합니다.
- 많은 ACSLS 라이브러리 구성 요소에 여러 필드로 구성된 기본 키가 있습니다.

  예:
  - LSM의 기본 키: *acs,lsm*
  - 드라이브의 기본 키: *acs,lsm,panel,drive*
- 여러 필드를 포함하는 기본 키를 와일드카드로 지정할 경우 키의 모든 필드를 별표(*)로 지정하거나 개별 키 필드를 와일드카드로 지정할 수 있습니다. 일부 키 필드를 (*)로 지정하고 다른 필드를 와일드카드로 지정할 경우 모든 키 필드를 지정합니다.
- *vol_id*에 대해 별표(*)를 사용하면 안됩니다. *vol_id*의 단일 문자를 일치시키려면 밑줄(_)을 사용합니다. 밑줄은 해당 위치의 단일 문자를 나타냅니다. *vol_id*에서 밑줄을 여러 번 사용하여 여러 문자를 나타낼 수 있습니다.

예

* 문자를 사용하는 유효한 와일드카드의 예:

- *display drive **(모든 드라이브 표시)

- *display drive 0,1,*,**(LSM 0,1의 모든 드라이브 표시)
- *display volume *100*(100으로 끝나는 모든 볼륨 표시)
- *display drive * -volume RB0001*(볼륨 RB0001을 포함하는 드라이브 표시)

* 문자를 사용하는 잘못된 와일드카드의 예:

- *display drive 0,1,**(LSM 0,1의 모든 드라이브를 선택할 경우 패널 위치와 드라이브 위치 모두에 대해 별표를 지정해야 함)
- *display cap * **(모든 CAP를 선택할 경우 별표가 하나만 허용됨)
- *display volume 1*111*(중간에 포함된 별표가 허용되지 않으며, *display volume 1_111*을 사용해야 함)

범위 사용

- 범위는 '-'(하이픈 또는 대시)로 표시합니다.
- 범위 사용 규칙은 다음과 같습니다.
  - 볼륨 홈 LSM을 제외하고 음수가 허용되지 않습니다.
  - <n 및 >n은 허용됩니다. <n은 n보다 작은 모든 숫자입니다. >n은 n보다 큰 모든 숫자입니다.
  - 여러 숫자 또는 범위를 공백으로 구분하여 사용할 수 있습니다.
  - 볼륨 범위는 영숫자일 수 있습니다.

예

유효한 범위의 예:

- *display volume * -drive 0,1,2,2-5*(acs 0, lsm 1, 패널 2, 드라이브 2~5에 있는 모든 볼륨 표시)
- *display lsm * -state online -free_cells >50 -type 9730*(사용 가능한 셀 수가 50개를 초과하고 온라인 상태인 모든 9730 lsm 표시)

잘못된 범위의 예:

- *display drive 0,1,1,1-**(*는 범위에서 유효하지 않음)
- *display volume * -drive 0,1,?,1-5*(?는 범위에서 유효하지 않음)

형식

display type arg ... [-selection ...] [[-c] | [-f field ...] [-s sort_field ...] [-n n]]

옵션

- *type*

표시할 객체를 지정합니다. 유효한 유형은 cap, 셀, 드라이브, 잠금, LSM, 패널, 풀, 포트 및 볼륨입니다.

- *arg*

  (인수) 객체 유형에 대한 식별자를 지정합니다. 하나 이상의 식별자와 일치하는 객체만 반환됩니다.

- *-selection*

  추가 선택 조건을 지정합니다. 구성 요소가 각 선택 조건에 대한 하나 이상의 선택 식별자와 일치해야 선택될 수 있습니다.

- *-c*

  (수) 요청이 *arg* 및 *selection* 조건을 충족하는 객체 수만 표시하게 만듭니다. 이 옵션은 *-f field, -s sort_field* 및 *-n* n 옵션과 함께 사용할 수 없습니다.

- *-f field*

  반환할 정보 필드와 선택된 각 객체에 대해 필드를 반환할 순서를 지정합니다. 각 표시 유형에는 사용 가능한 필드, 키 필드 및 기본 필드 목록이 있습니다. *-f*를 지정한 경우 기본 필드는 키 필드이거나 지정된 경우가 아닌 한 표시되지 않습니다. *-f*를 지정하지 않은 경우 기본 필드가 반환됩니다. 일반적으로 각 유형에 대한 키 필드가 먼저 표시됩니다. 하지만 *-f* 옵션 뒤에 키 필드를 지정하여 키 필드가 표시되는 순서를 변경할 수 있습니다.

  필드 목록은 공백으로 구분됩니다.

- *-s sort_field*

  필드에 의해 반환되는 객체를 지정된 순서로 선택하여 정렬합니다. 각 display 명령의 기본 정렬 순서는 각 명령 아래에 설명되어 있습니다. 내부 데이터베이스 값을 기준으로 정렬되며 영숫자 순서와 다를 수도 있습니다.

  유형에 대해 반환되는 모든 필드는 유효한 sort_field입니다.

- *-n n*

  표시할 최대 객체 수를 지정합니다.

80자를 초과하는 행을 표시할 경우 행 모드에서 cmd_proc 시작

curses 모드에서 시작되는 *cmd_proc* 명령은 80자를 초과하는 행을 표시할 수 없으며, 80자를 초과하는 행을 표시하려고 시도하면 *cmd_proc* 창이 정지됩니다.

따라서 많은 선택적 필드를 표시할 경우 *cmd_proc*를 행 모드(*–l* 옵션 사용)에서 시작하는 것이 좋습니다. 예: *display drive * -f volume type state serial_num wwn*, *cmd_proc* 사용, *cmd_proc –l*로 시작

명령

이 절에서는 display 명령에 대해 설명합니다.

display cap

display cap 명령은 선택한 옵션에 따라 특정 CAP 정보를 표시합니다.

주:

4.70 이상의 펌웨어가 설치된 SL3000 및 SL8500에서는 SL3000에 모듈을 더 추가하거나 추가 CAP를 설치할 때 CAP ID가 변경되지 않도록 라이브러리에서 구성 정보에 가능한 모든 CAP 위치를 반환합니다. 설치되지 않은 CAP는 CAP 상태에 "not installed"로 보고됩니다.

ACSLS에서는 CAP가 실제로 설치된 CAP 위치에 대해서만 데이터베이스 레코드를 추가합니다.

주:

표시 선택 조건 지정 및 표시할 데이터 선택에 대한 자세한 내용은 “display 명령 옵션 사용 ”을 참조하십시오.

형식

```
display cap cap_id ... ][ -availability cap_availability ...] [ -
status cap_status ... ][-priority cap_priority ... ] [ -state cap
_state ... ] [ -manual | -automatic ] [ -condition cap_condition .
.. ] [ [ -c ] | [ -f field ... ] [ -s sort_field ... ] [ -n n ] ]
```

필드

display cap에 대한 필드는 다음과 같습니다.

- 키 필드: *acs, lsm, cap*
- 기본 필드: *acs, lsm, cap, status, priority, state, mode, size, availability*
- 사용 가능한 필드: *acs, lsm, cap, status, priority, state, desired _state, mode, size, condition, availability*

옵션

- *cap_id*

  표시할 CAP를 지정합니다. *cap_id*는 acs, lsm, cap 형식입니다.

  * 와일드카드 문자를 임의의 *cap_id* 하위 필드에 사용하거나 모든 하위 필드를 나타내는 데 사용할 수 있습니다. 하지만 모든 *cap_ids*를 나타내는 경우(예: display cap * 또는 display cap *, *, *) 추가 *cap_ids*(숫자 또는 *)가 허용되지 않습니다. 숫자 범위는 *cap_id*의 모든 하위 필드에 적용됩니다.
- *availability cap_availability*

  하나 이상의 cap 가용성을 추가 선택 조건으로 지정합니다. CAP 가용성은 *lib_not _partn, cap_shared, cap_dedicated, dedicated_other, not_installed*입니다.

dedicated_other 또는 *not_installed*인 CAP는 ACSLS 데이터베이스에 정의되지 않습니다. 이러한 값은 구성된 이후에 CAP의 가용성이 변경된 경우에만 표시됩니다. 이러한 CAP를 제거하려면 재구성합니다.

- *-status cap_status*

  하나 이상의 CAP 상태를 추가 선택 조건으로 지정합니다. CAP 상태는 *audit, available, eject, enter, insert_magazines*입니다.

- *-priority cap_priority*

  하나 이상의 CAP 우선순위를 지정합니다. 숫자 범위 규칙이 적용됩니다. CAP 숫자 범위: 0~16

- *-state cap_state*

  하나 이상의 CAP 상태를 지정합니다. CAP 상태는 *diagnostic, offline, offline_pending, online, recovery*입니다.

- *-manual*

  수동 모드에서 CAP를 선택합니다.

- *-automatic*

  자동 모드에서 CAP를 선택합니다.

- *-condition cap_condition*

  선택할 하나 이상의 CAP 조건을 지정합니다. 유효한 조건은 *inoperative, maint_required* 또는 *operative*입니다.

  주:

  inoperative 또는 *maint_required* 조건은 드라이브, LSM 또는 ACS가 온라인으로 전환되는 경우에만 지워집니다. 따라서 *inoperative* 또는 *maint_required* CAP 조건이 부정확할 수 있습니다.

- *-c*

  (수) 요청이 `arg` 및 `selection` 조건을 충족하는 객체 수만 표시하게 만듭니다. 이 옵션은 -f field, -s sort_field 및 -n n 옵션과 함께 사용할 수 없습니다.

- *-f field*

  반환할 정보 필드와 선택된 각 객체에 대해 필드를 반환할 순서를 지정합니다. 각 표시 유형에는 사용 가능한 필드, 키 필드 및 기본 필드 목록이 있습니다. *-f*를 지정한 경우 기본 필드는 키 필드이거나 지정된 경우가 아닌 한 표시되지 않습니다. *-f*를 지정하지 않은 경우 기본 필드가 반환됩니다. 일반적으로 각 유형에 대한 키 필드가 먼저 표시됩니다. 하지만 *-f* 옵션 뒤에 키 필드를 지정하여 키 필드가 표시되는 순서를 변경할 수 있습니다.

  필드 목록은 공백으로 구분됩니다.

  다음은 *query cap*에 의해 보고되지 않는 새로운 CAP 필드입니다.

  - *desired_state*

ACS, 포트, LSM, 드라이브 또는 CAP에 대한 *desired state*는 구성 요소에 대해 원하는 가용성입니다. ACSLS에서는 명시적 *vary* 작업이 수행될 때 원하는 상태를 설정합니다. 이는 *cmd_proc* 또는 *ACSAPI client* 명령의 vary이며, 라이브러리 상태가 변경되어 ACSLS에서 내부적으로 생성하는 *vary*가 아닙니다.

ACS, 포트, LSM, 드라이브 또는 CAP에 대한 현재 상태("state"로 지정됨)는 원하는 상태에 의해 제한되는 구성 요소의 현재 가용성입니다. 드라이브가 온라인으로 전환될 경우 원하는 상태는 온라인입니다. 하지만 드라이브가 작동하지 않거나 오프라인 또는 준비되지 않은 LSM에 있기 때문에 현재 상태가 오프라인일 수 있습니다. ACS, 포트, LSM, 드라이브 또는 CAP의 현재 상태를 *query* 및 *display* 명령의 결과에서는 구성 요소의 "상태"라고도 합니다.

현재 가용성에 따른 CAP의 현재 상태는 유지되지 않습니다.

논리적 라이브러리와 논리적 라이브러리 내 테이프 드라이브의 가용성은 기본 물리적 라이브러리와 논리적 라이브러리 모두에 대해 설정하는 원하는 상태에 의해서도 제어됩니다. 물리적 라이브러리와 논리적 라이브러리 모두에 대한 원하는 상태가 온라인인 경우 논리적 라이브러리와 논리적 테이프 드라이브의 현재 상태에 기본 물리적 라이브러리 및 드라이브의 현재 상태가 반영됩니다.

- *-s sort_field*

  필드에 의해 반환되는 객체를 지정된 순서로 선택하여 정렬합니다. 내부 데이터베이스 값을 기준으로 정렬되며 영숫자 순서와 다를 수도 있습니다.

  유형에 대해 반환되는 모든 필드는 유효한 *sort_fields*입니다.

- *-n n*

  표시할 최대 객체 수를 지정합니다.

예

- 모든 CAP 데이터를 표시(*display*)하려면 다음 명령을 사용합니다.

  ```
  display cap *
  ```

- ACS 1, LSM 1, CAP 1에 대해 CAP 데이터를 표시(*display*)하려면 다음 명령을 사용합니다.

  ```
  display cap 1,1,1
  ```

- LSM 1, 0에서 모든 수동 모드 CAP를 표시(*display*)하려면 다음 명령을 사용합니다.

  ```
  display cap 1,0,* -manual
  ```

- 오프라인 보류 중 상태의 모든 CAP를 표시(*display*)하려면 다음 명령을 사용합니다.

  ```
  display cap * -state offline_pending
  ```

display cell

display cell 명령은 선택한 옵션에 따라 셀에 대한 특정 정보를 표시합니다.

주:

표시 선택 조건 지정 및 표시할 데이터 선택에 대한 자세한 내용은 “display 명령 옵션 사용 ”을 참조하십시오.

형식

display cell cell_loc ... [-status cell_status ...] [[-c] | [-f field ...] [-s sort_field ...] [-n n]]

필드

display cell에 대한 필드는 다음과 같습니다.

- 키 필드: *acs*, *lsm*, *panel*, *row*, *column*
- 기본 필드: *acs*, *lsm*, *panel*, *row*, *column*, *status*
- 사용 가능한 필드: *acs*, *lsm*, *panel*, *row*, *column*, *status*

옵션

- *-status cell_status*

  하나 이상의 셀 상태를 추가 선택 조건으로 지정합니다. 유효한 셀 상태: *empty, full, inaccessible, reserved*

  * 와일드카드 문자를 임의의 *cell_id* 하위 필드에 사용하거나 모든 하위 필드를 나타내는 데 사용할 수 있습니다. 하지만 모든 cell_ids를 나타내는 경우(예: d*isplay cell ** 또는 *display cell_id* *,*,*,*) 추가 cell_ids(숫자 또는 *)가 허용되지 않습니다. 숫자 범위는 cell_id의 모든 하위 필드에 적용됩니다.
- *-c*

  (수) 요청이 *arg* 및 *selection* 조건을 충족하는 객체 수만 표시하게 만듭니다. 이 옵션은 *-f field, -s sort_field* 및 *-n* n 옵션과 함께 사용할 수 없습니다.
- *-f field*

  정보 필드와 선택된 각 객체에 대해 필드를 반환할 순서를 지정합니다. 각 표시 유형에는 사용 가능한 필드, 키 필드 및 기본 필드 목록이 있습니다. *-f*를 지정한 경우 기본 필드는 키 필드이거나 지정된 경우가 아닌 한 표시되지 않습니다. *-f*를 지정하지 않은 경우 기본 필드가 반환됩니다. 일반적으로 각 유형에 대한 키 필드가 먼저 표시됩니다. 하지만 *-f* 옵션 뒤에 키 필드를 지정하여 키 필드가 표시되는 순서를 변경할 수 있습니다.

  필드 목록은 공백으로 구분됩니다.
- *-s sort_field*

필드에 의해 반환되는 객체를 지정된 순서로 선택하여 정렬합니다. 내부 데이터베이스 값을 기준으로 정렬되며 영숫자 순서와 다를 수도 있습니다.

유형에 대해 반환되는 모든 필드는 유효한 *sort_field*입니다.

- *-n n*

  표시할 최대 객체 수를 지정합니다.

예

상태가 예약됨인 셀에 대한 정보를 표시하려면 다음 명령을 사용합니다.

```
display cell * -status reserved
```

display drive

display drive 명령은 선택한 옵션에 따라 특정 드라이브 정보를 표시합니다.

> **주:**
>
> 표시 선택 조건 지정 및 표시할 데이터 선택에 대한 자세한 내용은 “display 명령 옵션 사용 ”을 참조하십시오.

형식

```
display drive drive_id ... [ -status drive_status ... ][-state drive
_state ... ] [ -type drive_type ... ][ -volume vol_id ... ] [ -lock
lock_id... ] [ -serial drive_serial_num ... ] [ -condition drive
_condition ...][ [ -c ] | [ -f field ... ] [ -s sort_field ... ] [ -n
n ] ]
```

필드

display drive에 대한 필드는 다음과 같습니다.

- 키 필드: *acs, lsm, panel, drive*

  기본 필드: *acs, lsm, panel, drive, status, state, volume, type*

- 사용 가능한 필드: *acs, lsm, panel, drive, status, state, volume, type, lock, desired_state, serial_num, condition, wwn, last_dismount _time, error*

옵션

- *drive_id*

*drive_id*를 `acs, lsm, panel, drive` 형식으로 표시합니다.

* 와일드카드 문자를 임의의 `drive_id` 하위 필드에 사용하거나 모든 하위 필드를 나타내는 데 사용할 수 있습니다. 하지만 모든 drive_ids를 나타내는 경우(예: *`display drive *`* 또는 *`display drive_id`* *,*,*,*) 추가 drive_ids(숫자 또는 *)가 허용되지 않습니다. 숫자 범위는 `drive_id`의 모든 하위 필드에 적용됩니다.

- *`-status drive_status`*

  `available, in_use, reserved` 상태 중 하나를 추가 선택 조건으로 표시합니다.

  가능한 시나리오: *`query drive all`*을 수행할 때 드라이브가 사용 중이라는 메시지가 표시됩니다. 그런 다음 *`display drive`* *를 수행하면 드라이브가 예약되어 있다는 메시지가 표시됩니다. 의미: 드라이브에 대한 예약됨 상태가 마운트 요청이 진행 중이고 볼륨이 드라이브로 전달되고 있음을 암시합니다. 동시에 드라이브가 사용 중인 것으로 간주됩니다.

- *`-state drive_state`*

  하나 이상의 드라이브에 대해 *`diagnostic, online, offline, recovery`* 상태를 표시합니다.

- *`-type drive_type`*

  전송 유형별로 드라이브를 표시합니다.

- *`-volume vol_id`*

  선택한 드라이브에 마운트된 볼륨을 표시합니다. *`vol_id`*는 테이프 볼륨 또는 볼륨 ID 범위를 지정하는 6자리 영숫자 문자열 또는 와일드카드 문자열일 수 있습니다.

- *`-lock lock_id`*

  잠금 ID별로 잠긴 드라이브에 대한 드라이브 정보를 표시합니다.

- *`-serial drive_serial_num`*

  드라이브 일련 번호를 지정합니다.

- *`-condition drive_condition`*

  지정된 드라이브의 조건을 표시합니다. 유효한 조건은 *`operative, inoperative`* 또는 *`maint_required`*입니다.

  주:

  inoperative 또는 *`maint_required`* 조건은 드라이브, LSM 또는 ACS가 온라인으로 전환되는 경우에만 지워집니다. 따라서 inoperative 또는 *`maint_required`* 드라이브 조건이 부정확할 수 있습니다.

- *`-c`*

  (수) 요청이 *arg* 및 *selection* 조건을 충족하는 객체 수만 표시하게 만듭니다. 이 옵션은 *`-f field`*, *`-s sort_field`* 및 *`-n n`* 옵션과 함께 사용할 수 없습니다.

- *-s sort_field*

  필드에 의해 반환되는 객체를 지정된 순서로 선택하여 정렬합니다. 내부 데이터베이스 값을 기준으로 정렬되며 영숫자 순서와 다를 수도 있습니다.

  유형에 대해 반환되는 모든 필드는 유효한 sort_field입니다.

- *-f field*

  정보 필드와 선택된 각 객체에 대해 필드를 반환할 순서를 지정합니다. 각 표시 유형에는 사용 가능한 필드, 키 필드 및 기본 필드 목록이 있습니다. *-f*를 지정한 경우 기본 필드는 키 필드이거나 지정된 경우가 아닌 한 표시되지 않습니다. *-f*를 지정하지 않은 경우 기본 필드가 반환됩니다. 일반적으로 각 유형에 대한 키 필드가 먼저 표시됩니다. 하지만 *-f* 옵션 뒤에 키 필드를 지정하여 키 필드가 표시되는 순서를 변경할 수 있습니다.

  필드 목록은 공백으로 구분됩니다. 다음은 *query drive*에 의해 보고되지 않는 새 테이프 드라이브 필드 설명의 목록입니다.

  - *desired_state*

    ACS, 포트, LSM, 드라이브 또는 CAP에 대한 원하는 상태는 구성 요소에 대해 원하는 가용성입니다. ACSLS에서는 명시적 *vary* 작업이 수행될 때 원하는 상태를 설정합니다. 이는 *cmd_proc* 또는 *ACSAPI client* 명령의 vary이며, 라이브러리 상태가 변경되어 ACSLS에서 내부적으로 생성하는 *vary*가 아닙니다.

    ACS, 포트, LSM, 드라이브 또는 CAP에 대한 현재 상태("state"로 지정됨)는 원하는 상태에 의해 제한되는 구성 요소의 현재 가용성입니다. 드라이브가 온라인으로 전환될 경우 원하는 상태는 온라인입니다. 하지만 드라이브가 작동하지 않거나 오프라인 또는 준비되지 않은 LSM에 있기 때문에 현재 상태가 오프라인일 수 있습니다. ACS, 포트, LSM, 드라이브 또는 CAP의 현재 상태를 *query* 및 *display* 명령의 결과에서는 구성 요소의 "상태"라고도 합니다.

    논리적 라이브러리와 논리적 라이브러리 내 테이프 드라이브의 가용성은 기본 물리적 라이브러리와 논리적 라이브러리 모두에 대해 설정하는 원하는 상태에 의해서도 제어됩니다. 물리적 라이브러리와 논리적 라이브러리 모두에 대한 원하는 상태가 온라인인 경우 논리적 라이브러리와 논리적 테이프 드라이브의 현재 상태에 기본 물리적 라이브러리 및 드라이브의 현재 상태가 반영됩니다.

  - *serial_num*

    드라이브에서 라이브러리에 드라이브의 일련 번호를 보고하고 라이브러리에서 ACSLS에 라이브러리의 일련 번호를 보고할 경우에만 ACSLS에서 드라이브 일련 번호를 보고할 수 있습니다. 최신 라이브러리 및 드라이브에서만 ACSLS에 드라이브 일련 번호를 보고합니다. 여기에는 T9840, T9940, T10000, LTO, DLT 7000, SDLT 이상 드라이브가 포함됩니다. SCSI 연결 라이브러리, L5500, T10000 드라이브를 지원하는 9310 이상 라이브러리는 ACSLS에 일련 번호를 보고합니다.

    ACSLS는 드라이브 트레이의 일련 번호가 아닌 테이프 드라이브 "블록"의 일련 번호를 보고합니다.

드라이브 블록의 일련 번호는 드라이브에 의해 라이브러리에 보고되고 SLConsole 및 ACSLS를 통해 사용자에게 보고되는 일련 번호이자 백업 애플리케이션 등에 대한 SCSI/광 섬유 데이터 경로를 통해 보고되는 일련 번호입니다.

드라이브 트레이에는 자체 일련 번호가 레이블로 붙습니다. 드라이브 트레이 레이블의 일련 번호에 따라 StorageTek을 통해 서비스 등에 대한 사용을 허가합니다. 드라이브 트레이의 번호는 활성화하는 데 사용되므로, 활성화된 일련 번호를 업데이트할 필요 없이 드라이브(드라이브 "블록")를 교체할 수 있습니다.

드라이브 트레이와 드라이브 블록 간에는 최소한의 통신만 진행됩니다. 이는 몇 가지 준비됨/준비되지 않음 표시기를 사용하여 표시됩니다. 드라이브 트레이는 기본적으로 스마트 기능이 없는 드라이브에 대한 전원 공급 장치입니다.

◦ *wwn*

드라이브의 World Wide Name입니다. WWN은 8바이트(이진)로, 16진수 문자 쌍(바이트) 사이에 점 구분 기호가 있는 16개의 16진수 문자(0~9 및 A~F)로 표시됩니다.

표시되는 WWN은 노드의 WWN입니다. 첫번째 포트(포트 A)의 WWN은 한 숫자가 더 큽니다. 드라이브에 포트가 2개 있는 경우 포트 B의 WWN은 두 숫자가 더 큽니다.

드라이브의 WWN은 SL3000 라이브러리와 펌웨어가 3.94 이상인 SL8500 라이브러리에 사용할 수 있습니다.

◦ *last_dismount_time*

드라이브에서 카트리지를 마지막으로 마운트 해제한 날짜 및 시간입니다. ACSLS에서는 이 정보를 사용하여 *query mount*에 보고되는 드라이브를 정렬합니다. 드라이브는 기본적으로 마운트 중인 볼륨에 대한 LSM 근접도(최소 전달)를 기준으로 나열되고 보조적으로 가장 빠른 *last_dismount_time*을 기준으로 나열됩니다. 가장 빠른 *last_dismount_time*을 기준으로 드라이브를 정렬하면 드라이브 사용이 균등해집니다.

카트리지가 드라이브에서 마운트 해제되면 이러한 통계가 ACSLS에 보고됩니다. 다음은 이러한 라이브러리 및 드라이브에 대해서만 보고되는 통계 필드입니다.

라이브러리:

› 모든 SL3000
› 펌웨어가 4.13 이상인 SL8500

테이프 드라이브:

› 드라이브 펌웨어가 1.42 이상인 T9840A, T9840C 및 T9840D. 메모리 제한으로 인해 T9840B는 지원되지 않습니다.
› 드라이브 펌웨어가 1.42 이상인 T9940A 및 T9940B
› 드라이브 펌웨어가 1.38 이상인 T10000A 및 T10000B

◦ *error* - (드라이브 유지 관리 필요)

이는 현재 T10000 드라이브에 대해서만 보고됩니다. 값은 다음과 같습니다.

› none - 보고된 드라이브 오류가 없습니다.
› *maint_reqd* - 테이프 드라이브에 대해 유지 관리가 필요합니다.

이는 테이프 드라이브에서 드라이브를 피하고 오프라인으로 전환해야 한다고 결정한 경우에만 보고됩니다. 서비스를 요청해야 합니다.

기본 드라이브 오류 시나리오:

이러한 시나리오에서 드라이브와 라이브러리는 수동 개입을 기다립니다.

표 14.1. 수동 개입을 기다리는 드라이브 및 라이브러리

언로드 실패	카트리지가 드라이브에 끼어 있음
드라이브 리더 오프 후크 실패	매체를 꺼내야 드라이브에 드라이브 오류가 표시됩니다.
드라이브 IPL 실패	드라이브와 라이브러리의 통신이 중지됩니다.

- *-s sort_field*

  필드에 의해 반환되는 객체를 지정된 순서로 선택하여 정렬합니다. 내부 데이터베이스 값을 기준으로 정렬되며 영숫자 순서와 다를 수도 있습니다. 예를 들어, *drive type*은 리터럴(display)이 아니라 내부 숫자 드라이브 유형을 기준으로 정렬됩니다.

  유형에 대해 반환되는 모든 필드는 유효한 *sort_fields*입니다.
- *-n n*

  표시할 최대 객체 수를 지정합니다.

예

- 0,1,1 패널의 모든 드라이브를 표시하려면 다음 명령을 사용합니다.

  ```
  display drive 0,1,1,*
  ```
- LSM 0,1의 모든 드라이브를 표시하려면 다음 명령을 사용합니다.

  ```
  display drive 0,1,*,*
  ```
- ACS 1의 모든 드라이브에 대해 최대 56개의 행을 패널 및 드라이브별로 정렬하여 표시하려면 다음 명령을 사용합니다.

  ```
  display drive 1,*,*,* -s panel drive -n 56
  ```
- 상태가 사용할 수 있음인 ACS 1, LSM 1에 대한 드라이브 데이터를 표시하려면 다음 명령을 사용합니다.

  ```
  display drive 1,1,*,* -status available
  ```
- 모든 드라이브에 대한 드라이브 유형, 일련 번호 및 WWN(World Wide Name)을 표시하려면 다음 명령을 사용합니다.

  ```
  display drive * -f type serial_num wwn
  ```

- 원하는 상태의 모든 드라이브를 표시하려면 다음 명령을 사용합니다.

```
display drive * -f state desired_state
```

display lock

display lock 명령은 선택한 옵션에 따라 *lock_id*별로 특정 잠금 정보를 표시합니다.

> **주:**
>
> 표시 선택 조건 지정 및 표시할 데이터 선택에 대한 자세한 내용은 "display 명령 옵션 사용 "을 참조하십시오.

형식

display lock lock_id ... [-user user_id ...] [[-c] | [-f field ...] [-s sort_field ...] [-n n] *]*

필드

display lock에 대한 필드는 다음과 같습니다.

- 키 필드: *lock_id*
- 기본 필드: *lock_id*, *user_id*
- 사용 가능한 필드: *lock_id*, *user_id*

옵션

- *lock_id*

  숫자 *lock_id*를 표시합니다.

  * 와일드카드 문자를 임의의 *lock_id* 하위 필드에 사용하거나 모든 하위 필드를 나타내는 데 사용할 수 있습니다. 하지만 모든 *lock_ids*를 나타내는 경우(예*: display lock **display lock_id* *,*,*) 추가 lock_ids(숫자 또는 *)가 허용되지 않습니다. 숫자 범위는 *lock_id*의 모든 하위 필드에 적용됩니다.
- *user_id*

  볼륨 또는 전송을 잠근 사용자 이름(예: *acsss*)을 지정합니다. *user_id*는 80자 이후부터 줄 바꿈됩니다.
- *-c*

  (수) 요청이 *arg* 및 *selection* 조건을 충족하는 객체 수만 표시하게 만듭니다. 이 옵션은 *-f field*, *-s sort_field* 및 -n *n* 옵션과 함께 사용할 수 없습니다.
- *-f field*

  정보 필드와 선택된 각 객체에 대해 필드를 반환할 순서를 지정합니다. 각 표시 유형에는 사용 가능한 필드, 키 필드 및 기본 필드 목록이 있습니다. *-f*를 지정한 경우 기본 필드는

키 필드이거나 지정된 경우가 아닌 한 표시되지 않습니다. *-f*를 지정하지 않은 경우 기본 필드가 반환됩니다. 일반적으로 각 유형에 대한 키 필드가 먼저 표시됩니다. 하지만 *-f* 옵션 뒤에 키 필드를 지정하여 키 필드가 표시되는 순서를 변경할 수 있습니다.

필드 목록은 공백으로 구분됩니다.

- *-s sort_field*

  필드에 의해 반환되는 객체를 지정된 순서로 선택하여 정렬합니다. 내부 데이터베이스 값을 기준으로 정렬되며 영숫자 순서와 다를 수도 있습니다.

  유형에 대해 반환되는 모든 필드는 유효한 정렬 필드입니다.

- *-n n*

  표시할 최대 객체 수를 지정합니다.

예

- *lock_id 2*에 대한 잠금 정보를 표시하려면 다음 명령을 사용합니다.

  ```
  display lock 2
  ```

display lsm

display lsm 명령은 선택한 옵션에 따라 특정 LSM 정보를 표시합니다.

> **주:**
>
> 표시 선택 조건 지정 및 표시할 데이터 선택에 대한 자세한 내용은 “display 명령 옵션 사용 ”을 참조하십시오.

형식

display lsm lsm_id ... [-status lsm_status ...][-state lsm_state ...] [-free_cells cell_count ...][-type lsm_type ...] [-serial lsm_serial_num ...][-condition lsm_condition] [-door_open | -door_closed] [[-c] |[-f field ...] [-s sort_field ...][-n n *]]*

필드

display lsm에 대한 필드는 다음과 같습니다.

- 키 필드: *acs, lsm*
- 기본 필드: *acs, lsm, status, state, free_cells*
- 사용 가능한 필드: *acs, lsm, status, state, desired_state, free_cells, type, serial_num, activity, condition, door_status*
- *lsm_id*

lsm ID를 *acs*, *lsm* 형식으로 표시합니다.

* 와일드카드 문자를 임의의 *lsm_id* 하위 필드에 사용하거나 모든 하위 필드를 나타내는 데 사용할 수 있습니다. 하지만 모든 *lsm_ids*를 나타내는 경우(예: *display lsm ** 또는 *display lsm_id *,**) 추가 *lsm_ids*(숫자 또는 *)가 허용되지 않습니다. 숫자 범위는 *lsm_id*의 모든 하위 필드에 적용됩니다.

- *-status lsm_status*

  선택할 하나 이상의 LSM 상태를 지정합니다. 유효한 상태는 *audit*, *normal*입니다.

- *-state lsm_state*

  선택할 하나 이상의 LSM 상태를 지정합니다. 유효한 상태는 *diagnostic, online, offline, offline_pending, recovery*입니다.

- *-free_cells cell_count*

  사용 가능한 셀의 수를 기준으로 LSM을 선택합니다.

- *-type lsm_type*

  LSM을 유형별로 표시합니다.

- *-serial lsm_serial_num*

  선택한 LSM의 일련 번호를 표시합니다.

  SL8500 6.0 이상 및 SL3000 3.0 이상 펌웨어의 경우 라이브러리에서 사용이 허가된 일련 번호를 보고합니다. SL8500 라이브러리 번호를 구하려면 LSM 번호에 1을 더하고 4로 나눕니다.

- *-condition lsm_condition*

  선택한 LSM의 조건을 표시합니다. 유효한 조건은 *operative*, *inoperative* 또는 *maint_required*입니다.

  주:

  inoperative 또는 *maint_required* 조건은 드라이브, LSM 또는 ACS가 온라인으로 전환되는 경우에만 지워집니다. 따라서 *inoperative* 또는 *maint_required* LSM 조건이 부정확할 수 있습니다.

- *-door_open*

  도어가 열려 있는 LSM을 표시합니다.

- *-door_closed*

  도어가 닫힌 LSM을 표시합니다.

- *-c*

  (수) 요청이 *arg* 및 *selection* 조건을 충족하는 객체 수만 표시하게 만듭니다. 이 옵션은 *-f field, -s sort_field* 및 *-n* n 옵션과 함께 사용할 수 없습니다.

- *-f field*

  정보 필드와 선택된 각 객체에 대해 필드를 반환할 순서를 지정합니다. 각 표시 유형에는 사용 가능한 필드, 키 필드 및 기본 필드 목록이 있습니다. *-f*를 지정한 경우 기본 필드는 키 필드이거나 지정된 경우가 아닌 한 표시되지 않습니다. *-f*를 지정하지 않은 경우 기본 필드가 반환됩니다. 일반적으로 각 유형에 대한 키 필드가 먼저 표시됩니다. 하지만 *-f* 옵션 뒤에 키 필드를 지정하여 키 필드가 표시되는 순서를 변경할 수 있습니다.

  필드 목록은 공백으로 구분됩니다. 다음은 *query lsm*에 의해 보고되지 않는 새로운 LSM 필드입니다.

  - *desired_state*

    ACS, 포트, LSM, 드라이브 또는 CAP에 대한 원하는 상태는 구성 요소에 대해 원하는 가용성입니다. ACSLS에서는 명시적 *vary* 작업이 수행될 때 원하는 상태를 설정합니다. 이는 *cmd_proc* 또는 *ACSAPI client* 명령의 *vary*이며, 라이브러리 상태가 변경되어 ACSLS에서 내부적으로 생성하는 vary가 아닙니다.

    ACS, 포트, LSM, 드라이브 또는 CAP에 대한 현재 상태("state"로 지정됨)는 원하는 상태에 의해 제한되는 구성 요소의 현재 가용성입니다. 드라이브가 온라인으로 전환될 경우 원하는 상태는 온라인입니다. 하지만 드라이브가 작동하지 않거나 오프라인 또는 준비되지 않은 LSM에 있기 때문에 현재 상태가 오프라인일 수 있습니다. ACS, 포트, LSM, 드라이브 또는 CAP의 현재 상태를 *query* 및 *display* 명령의 결과에서는 구성 요소의 "상태"라고도 합니다.

    논리적 라이브러리와 논리적 라이브러리 내 테이프 드라이브의 가용성은 기본 물리적 라이브러리와 논리적 라이브러리 모두에 대해 설정하는 원하는 상태에 의해서도 제어됩니다. 물리적 라이브러리와 논리적 라이브러리 모두에 대한 원하는 상태가 온라인인 경우 논리적 라이브러리와 논리적 테이프 드라이브의 현재 상태에 기본 물리적 라이브러리 및 드라이브의 현재 상태가 반영됩니다.

  - *serial_num*

    ACSLS에서 LSM에 대해 보고하는 일련 번호는 SL3000 및 SL8500 라이브러리의 HBC 카드에 기록되는 라이브러리의 일련 번호입니다(SL8500 6.00 펌웨어 또는 SL3000 3.0 펌웨어 이전). 이후 버전의 펌웨어가 설치된 SL8500 및 SL3000 라이브러리는 프레임 일련 번호, 즉 활성화된 일련 번호를 보고합니다.

    SL8500 및 SL3000 라이브러리는 기본 모듈 프레임의 일련 번호를 통해 활성화됩니다. 이 일련 번호는 프레임의 UL 레이블에 있으며 라이브러리의 HBK 카드에 기록됩니다. 제조 과정에서 교체 HBK 카드가 재생성될 수 있습니다. 이러한 일련 번호는 516(SL8500) 및 571(SL3000)로 시작합니다. SLConsole(System Detail -> Properties -> General)을 통해 프레임 일련 번호를 표시할 수 있습니다.

    앞으로는 라이브러리 펌웨어에서 이 프레임 일련 번호를 ACSLS에 보고할 예정입니다. LSM 일련 번호가 변경될 때마다 ACSLS에서는 LSM이 온라인으로 전환될 때 일련 번호를 자동으로 업데이트합니다.

- *-s sort_field*

필드에 의해 반환되는 객체를 지정된 순서로 선택하여 정렬합니다. 내부 데이터베이스 값을 기준으로 정렬되며 영숫자 순서와 다를 수도 있습니다.

유형에 대해 반환되는 모든 필드는 유효한 정렬 필드입니다.

- *-n n*

  표시할 최대 객체 수를 지정합니다.

예

- 9714 유형에 대해 라이브러리에 있는 모든 LSM에 대한 정보를 표시하려면 다음 명령을 사용합니다.

  *display lsm * -type 9714*
- 모든 LSM 유형을 표시하려면 다음 명령을 사용합니다.

  *display lsm * - f type*
- 라이브러리 일련 번호를 표시하려면 다음 명령을 사용합니다.

  *display lsm * -f serial_num*
- 모든 LSM 상태 및 *desired_state*를 표시하려면 다음 명령을 사용합니다.

  display lsm * -f state *desired_state*

display panel

display panel 명령은 선택한 옵션에 따라 특정 라이브러리 패널 정보를 표시합니다.

> **주:**
>
> 표시 선택 조건 지정 및 표시할 데이터 선택에 대한 자세한 내용은 “display 명령 옵션 사용 ”을 참조하십시오.

형식

display panel panel_id ...[-type panel_type ...] [[-c] |[-f field ...] [-s sort_field ...] [-n n *]]*

필드

display panel에 대한 필드는 다음과 같습니다.

- 키 필드: *acs*, *lsm*, *panel*
- 기본 필드: *acs*, *lsm*, *panel*, *type*
- 사용 가능한 필드: *acs*, *lsm*, *panel*, *type*

옵션

- *panel_id*

  패널 ID를 *acs, lsm, panel* 형식으로 표시합니다.

  * 와일드카드 문자를 임의의 *panel_id* 하위 필드에 사용하거나 모든 하위 필드를 나타내는 데 사용할 수 있습니다. 하지만 모든 panel_ids를 나타내는 경우(예: *display panel ** 또는 *display panel_id *,*,**) 추가 panel_ids(숫자 또는 *)가 허용되지 않습니다. 숫자 범위는 panel_id의 모든 하위 필드에 적용됩니다.
- *-type panel_type*

  하나 이상의 유효한 패널 유형을 지정합니다.
- *-c*

  (수) 요청이 *arg* 및 *selection* 조건을 충족하는 객체 수만 표시하게 만듭니다. 이 옵션은 *-f field, -s sort_field* 및 *-n* n 옵션과 함께 사용할 수 없습니다.
- *-f field*

  정보 필드와 선택된 각 객체에 대해 필드를 반환할 순서를 지정합니다. 각 표시 유형에는 사용 가능한 필드, 키 필드 및 기본 필드 목록이 있습니다. *-f*를 지정한 경우 기본 필드는 키 필드이거나 지정된 경우가 아닌 한 표시되지 않습니다. *-f*를 지정하지 않은 경우 기본 필드가 반환됩니다. 일반적으로 각 유형에 대한 키 필드가 먼저 표시됩니다. 하지만 *-f* 옵션 뒤에 키 필드를 지정하여 키 필드가 표시되는 순서를 변경할 수 있습니다.

  필드 목록은 공백으로 구분됩니다.
- *-s sort_field*

  필드에 의해 반환되는 객체를 지정된 순서로 선택하여 정렬합니다. 내부 데이터베이스 값을 기준으로 정렬되며 숫자 순서와 다를 수도 있습니다.

  유형에 대해 반환되는 모든 필드는 유효한 정렬 필드입니다.
- *-n n*

  표시할 최대 객체 수를 지정합니다.

예

모든 패널에 대한 패널 데이터 표시:

```
display panel *
```

display pool

display pool 명령은 선택한 옵션에 따라 특정 스크래치 풀 정보를 표시합니다.

주:

표시 선택 조건 지정 및 표시할 데이터 선택에 대한 자세한 내용은 “display 명령 옵션 사용 ”을 참조하십시오.

형식

display pool pool_id ... [-low_water low_water_mark ... | -high_water high_water_mark...][-overflow | -no_overflow] [[-c] | [-f field ...][-s sort_field ...] [-n n]]

필드

display pool에 대한 필드는 다음과 같습니다.

- 키 필드: *pool_id*
- 기본 필드: *pool_id, low_water, high_water, overflow*
- 사용 가능한 필드: *pool_id, low_water, high_water, overflow*

옵션

- *pool_id*

  스크래치 풀의 풀 ID를 표시합니다.

  * 와일드카드 문자를 임의의 *pool_id* 하위 필드에 사용하거나 모든 하위 필드를 나타내는 데 사용할 수 있습니다. 하지만 모든 pool_ids를 나타내는 경우(예: *display pool ** 또는 *display pool_id *, *, **) 추가 pool_ids(숫자 또는 *)가 허용되지 않습니다. 숫자 범위는 *pool_id*의 모든 하위 필드에 적용됩니다.
- *-low_water low_water_mark*

  풀의 최소 스크래치 카트리지 수를 지정합니다. *low_water* 마크가 하이픈(-)이 추가되어 표시되는 경우 선택한 스크래치 풀의 라이브러리에 있는 카트리지 수가 지정된 *low_water* 마크 내에 있는 것입니다.
- *-high_water high_water_mark*

  풀의 최대 스크래치 카트리지 수를 지정합니다. *high_water* 마크가 하이픈(-)이 추가되어 표시되는 경우 선택한 스크래치 풀의 라이브러리에 있는 카트리지 수가 지정된 *high_water* 마크 내에 있는 것입니다.
- *-overflow*

  스크래치 마운트가 요청되고 풀이 비어 있을 때 일반 풀이 요청을 처리하는 풀을 선택합니다.
- *-no_overflow*

  스크래치 마운트가 요청되고 풀이 비어 있을 때 일반 풀에 사용 가능한 테이프가 있는지 확인하지 않고 마운트가 실패하는 풀을 선택합니다.

- *-c*

  (수) 요청이 *arg* 및 *selection* 조건을 충족하는 객체 수만 표시하게 만듭니다. 이 옵션은 *-f field, -s sort_field* 및 *-n n* 옵션과 함께 사용할 수 없습니다.
- *-f field*

  정보 필드와 선택된 각 객체에 대해 필드를 반환할 순서를 지정합니다. 각 표시 유형에는 사용 가능한 필드, 키 필드 및 기본 필드 목록이 있습니다. *-f*를 지정한 경우 기본 필드는 키 필드이거나 지정된 경우가 아닌 한 표시되지 않습니다. *-f*를 지정하지 않은 경우 기본 필드가 반환됩니다. 일반적으로 각 유형에 대한 키 필드가 먼저 표시됩니다. 하지만 *-f* 옵션 뒤에 키 필드를 지정하여 키 필드가 표시되는 순서를 변경할 수 있습니다.

  필드 목록은 공백으로 구분됩니다.
- *-s sort_field*

  필드에 의해 반환되는 객체를 지정된 순서로 선택하여 정렬합니다. 내부 데이터베이스 값을 기준으로 정렬되며 숫자 순서와 다를 수도 있습니다.

  유형에 대해 반환되는 모든 필드는 유효한 정렬 필드입니다.
- *-n n*

  표시할 최대 객체 수를 지정합니다.

예

- 풀 5에 대한 모든 정보를 표시하려면 다음 명령을 사용합니다.

  ```
  display pool 5
  ```
- 오버플로우가 발생한 모든 풀을 표시하려면 다음 명령을 사용합니다.

  ```
  display pool * -overflow
  ```

display port

display port 명령은 선택한 옵션에 따라 특정 포트 정보를 표시합니다.

> **주:**
>
> 표시 선택 조건 지정 및 표시할 데이터 선택에 대한 자세한 내용은 “display 명령 옵션 사용 ”을 참조하십시오.

형식

display port port_id *... [-online | -offline][-name* port_name *. ..] [[-c] | [-f* field *...][-s* sort_field *...] [-n* n *]]*

필드

display port에 대한 필드는 다음과 같습니다.

- 키 필드: *acs, port*
- 기본 필드: *acs, port, name, state*
- 사용 가능한 필드: *acs, port, name, state, desired_state*

옵션

- port_id

  포트 ID를 *acs*, *port* 형식으로 표시합니다.

  * 와일드카드 문자를 임의의 *port_id* 하위 필드에 사용하거나 모든 하위 필드를 나타내는 데 사용할 수 있습니다. 하지만 모든 port_ids를 나타내는 경우(예: *display port ** 또는 *display port_id *,*,**) 추가 port_ids(숫자 또는 *)가 허용되지 않습니다. 숫자 범위는 *port_id*의 모든 하위 필드에 적용됩니다.

  - *online*

    온라인 상태인 포트를 선택하여 표시합니다.

  - *offline*

    오프라인 상태인 포트를 선택하여 표시합니다.

  - *name* port_name

    포트 이름을 지정합니다.

- *-c*

  (수) 요청이 *arg* 및 *selection* 조건을 충족하는 객체 수만 표시하게 만듭니다. 이 옵션은 *-f* field, *-s* sort_field 및 *-n* n 옵션과 함께 사용할 수 없습니다.

- *-f* field

  정보 필드와 선택된 각 객체에 대해 필드를 반환할 순서를 지정합니다. 각 표시 유형에는 사용 가능한 필드, 키 필드 및 기본 필드 목록이 있습니다. *-f*를 지정한 경우 기본 필드는 키 필드이거나 지정된 경우가 아닌 한 표시되지 않습니다. *-f*를 지정하지 않은 경우 기본 필드가 반환됩니다. 일반적으로 각 유형에 대한 키 필드가 먼저 표시됩니다. 하지만 *-f* 옵션 뒤에 키 필드를 지정하여 키 필드가 표시되는 순서를 변경할 수 있습니다.

  필드 목록은 공백으로 구분됩니다. 다음은 *query port*에 의해 보고되지 않는 새로운 포트 필드입니다.

  - *desired_state*

    ACS, 포트, LSM, 드라이브 또는 CAP에 대한 원하는 상태는 구성 요소에 대해 원하는 가용성입니다. ACSLS에서는 명시적 *vary* 작업이 수행될 때 원하는 상태를 설정합니다. 이는 *cmd_proc* 또는 *ACSAPI client* 명령의 *vary*이며, 라이브러리 상태가 변경되어 ACSLS에서 내부적으로 생성하는 *vary*가 아닙니다.

ACS, 포트, LSM, 드라이브 또는 CAP에 대한 현재 상태는 원하는 상태에 의해 제한되는 구성 요소의 현재 가용성입니다. 드라이브가 온라인으로 전환될 경우 원하는 상태는 온라인입니다. 하지만 드라이브가 작동하지 않거나 오프라인 또는 준비되지 않은 LSM에 있기 때문에 현재 상태가 오프라인일 수 있습니다. ACS, 포트, LSM, 드라이브 또는 CAP의 현재 상태를 *query* 및 *display* 명령의 결과에서는 구성 요소의 "상태"라고도 합니다.

논리적 라이브러리와 논리적 라이브러리 내 테이프 드라이브의 가용성은 기본 물리적 라이브러리와 논리적 라이브러리 모두에 대해 설정하는 원하는 상태에 의해서도 제어됩니다. 물리적 라이브러리와 논리적 라이브러리 모두에 대한 원하는 상태가 온라인인 경우 논리적 라이브러리와 논리적 테이프 드라이브의 현재 상태에 기본 물리적 라이브러리 및 드라이브의 현재 상태가 반영됩니다.

- *-s* sort_field

  필드에 의해 반환되는 객체를 지정된 순서로 선택하여 정렬합니다. 내부 데이터베이스 값을 기준으로 정렬되며 영숫자 순서와 다를 수도 있습니다.

  유형에 대해 반환되는 모든 필드는 유효한 정렬 필드입니다.

- *-n* n

  표시할 최대 객체 수를 지정합니다.

예

- 온라인 상태인 모든 포트를 표시하려면 다음 명령을 사용합니다.

  *display port * -online*

display volume

display volume 명령은 선택한 옵션에 따라 특정 볼륨 정보를 표시합니다.

주:

표시 선택 조건 지정 및 표시할 데이터 선택에 대한 자세한 내용은 “display 명령 옵션 사용 ”을 참조하십시오.

형식

display volume vol_id *... [-home acs,lsm,panel,row,column...] [-drive* drive_loc *...] [-data | -scratch | -clean]*
[-media media_type *...] [-pool* pool_id*...] [-standard |*
-virtual] [-status vol_status *...] [-entry* entry_date *...] [-access* access_date *...] [-lock* lock_id *...] [[-c] |*
[-f field ...][-s sort_field ...][-n n *]]*
[-max_use max_use*] [-lock_time* lock_time*]*

필드

*display volume*에 대한 필드는 다음과 같습니다.

- 키 필드: *vol_id*
- 기본 필드: *vol_id, acs, lsm, panel, row, column, pool, status, media, type*
- 사용 가능한 필드: *vol_id, acs, lsm, panel, row, column, drive_lsm, drive_panel, drive*(카트리지의 마운트 위치를 표시하기 위해 3개 모두 표시), *type, media, pool, label_type, status, entry_date, access_date, access_count, max_use, lock, lock_time, recording_format_family, recording_format_model, encrypt_status, volsafe_status, media _status, warranty_life, end_of_life, load_limit_alert*

옵션

- *vol_id*

  선택한 볼륨을 표시합니다.

  *vol_id*는 테이프 카트리지를 지정하는 하나 이상의 1~16자 영숫자 문자열일 수 있습니다.

  * 와일드카드 문자를 모든 vol_ids를 나타내거나(예:
  *display vol**) 카트리지 그룹을 나타내는 데 사용할 수 있습니다. 예를 들어, *display* vol_id *VAP**는 VAP로 시작하는 모든 카트리지를 표시합니다.

  *vol_id*는 모든 유효한 볼륨 ID 문자와 *를 포함할 수 있습니다.

- *home home_loc*

  볼륨 정보를 검색할 볼륨 홈 위치를 *acs,lsm,panel,row,column* 형식으로 지정합니다.
  -drive와 함께 사용할 수 없습니다.

  home_loc 하위 필드에서는 숫자 범위를 사용할 수 있습니다. 모든 *home_loc* 하위 필드를 지정해야 합니다. * 와일드카드 문자를 하나 이상의 drive_loc 하위 필드 대신 사용할 수 있습니다. 숫자 범위 규칙은 모든 하위 필드 또는 drive_loc에 적용됩니다.

- *acs <acs_id>*

  볼륨 정보를 검색해야 하는 ACS를 지정합니다.

- *drive drive_loc*

  볼륨 정보를 검색할 드라이브 위치를 *acs*,*lsm*,*panel*,*drive* 형식으로 지정합니다. -home과 함께 사용할 수 없습니다.

drive_loc 하위 필드에서는 숫자 범위를 사용할 수 있습니다. 모든 *home_loc* 하위 필드를 지정해야 합니다. * 와일드카드 문자를 하나 이상의 *drive_loc* 하위 필드 대신 사용할 수 있습니다. 숫자 범위 규칙은 모든 하위 필드 또는 *drive_loc*에 적용됩니다.

- *data*

  데이터 카트리지를 선택합니다.

- *scratch*

  스크래치 모드에서 카트리지를 선택합니다.

- *clean*

  청소 카트리지를 선택합니다.

- spent_clean

  드라이브 유형별로 모두 사용된 것으로 보고된 청소 카트리지를 선택합니다.

- *media media_type*

  지정한 매체 유형을 가진 카트리지만 선택합니다.

- *pool pool_id*

  스크래치 풀에서 카트리지를 선택하여 표시합니다. 카트리지는 데이터 카트리지로 변경된 후에도 스크래치 풀의 ID를 유지합니다.

- *standard*

  스캔 가능한 볼륨 ID *label_type*을 가진 테이프를 선택합니다.
  -virtual과 함께 사용할 수 없습니다.

- *virtual*

  venter 명령을 통해 입력된 외부 볼륨 ID label_type이 없는 테이프를 선택합니다. -standard와 함께 사용할 수 없습니다.

- *status vol_status*

  유효한 항목: *dismount*, *eject*, *enter*, mount, home, in_drive, move, missing, absent, ejected

- *entry entry_date*

  카트리지를 라이브러리에 넣은 날짜 및 시간을 지정합니다. ISO 표준 형식(yyyy-mm-dd)으로 입력 및 표시합니다.

- *access* access_date

  카트리지를 마지막으로 사용한 날짜 및 시간을 지정합니다. ISO 표준 형식(yyyy-mm-dd)으로 입력 및 표시합니다.

- *lock lock_id*

  볼륨에 대한 잠금 ID를 지정합니다.

- *max_use max_use*

  청소 카트리지의 최대 사용 수를 지정합니다. 비청소 카트리지의 *max_use* 값은 0입니다.
- *lock_time lock_time*

  잠금이 설정된 날짜 및 시간을 지정합니다. 시스템 기본 형식에 따라 표시됩니다.
- *recording_format_family drive_family* -recording_format_model *drive_model*

  기록 형식 드라이브 제품군 및 드라이브 모델을 지정합니다. **예:** T10000 및 T10000C
- *c*

  (수) 요청이 *arg* 및 *selection* 조건을 충족하는 객체 수만 표시하게 만듭니다. 이 옵션은 *-f* field,
  -s sort_field 및 *-n* n 옵션과 함께 사용할 수 없습니다.
- *-f* field

  정보 필드와 선택된 각 객체에 대해 필드를 반환할 순서를 지정합니다. 각 표시 유형에는 사용 가능한 필드, 키 필드 및 기본 필드 목록이 있습니다. *-f*를 지정한 경우 기본 필드는 키 필드이거나 지정된 경우가 아닌 한 표시되지 않습니다. *-f*를 지정하지 않은 경우 기본 필드가 반환됩니다. 일반적으로 각 유형에 대한 키 필드가 먼저 표시됩니다. 하지만 *-f* 옵션 뒤에 키 필드를 지정하여 키 필드가 표시되는 순서를 변경할 수 있습니다.

  필드 목록은 공백으로 구분됩니다. 다음은 *query volume*에 의해 보고되지 않는 새 볼륨 필드의 목록입니다. 다음은 이러한 라이브러리 및 드라이브에 대해서만 보고되는 통계 필드입니다.

  다음은 이러한 라이브러리 및 드라이브에 대해서만 보고되는 통계 필드입니다.

  라이브러리:
  - 모든 SL3000
  - 펌웨어가 4.13 이상인 SL8500
  - 테이프 드라이브:
  - 드라이브 펌웨어가 1.42 이상인 T9840A, T9840C 및 T9840D. 메모리 제한으로 인해 T9840B는 지원되지 않습니다.
  - 드라이브 펌웨어가 1.42 이상인 T9940A 및 T9940B
  - 드라이브 펌웨어가 1.38 이상인 T10000A 및 T10000B
  - T10000C 및 T10000D 이상의 드라이브
- *recording_format_family*

  이 카트리지에서 데이터를 마지막으로 기록한 드라이브 제품군을 표시합니다.
- r*ecording_format_model*

  이 카트리지에서 데이터를 마지막으로 기록한 드라이브 모델을 표시합니다. 이 정보는 다음에 유용합니다.

  - 카트리지에서 데이터를 읽을 수 없는 드라이브에 대한 카트리지 마운트 방지(예: T10000A에서는 T10000B 드라이브에 의해 기록되는 데이터를 읽을 수 없음)
  - 카트리지에 데이터 추가(예: T10000B에서는 T10000A에 의해 작성된 카트리지에 데이터를 추가할 수 없음)
- *encrypt_status*

  카트리지의 암호화 상태를 표시합니다(알려진 경우).
- *volsafe_status*

  volsafe 카트리지인지 여부를 표시합니다. *Volsafe*는 StorageTek의 WORM(Write Once Read Many) 카트리지입니다.
- *media_status*

  테이프 드라이브에서 카트리지를 "의심스러운" 것으로 보고할 경우 카트리지를 검사해야 합니다.
- *warranty_life* 및 *end_of_life*

  볼륨의 *warranty_life* 및 *end_of_life* 백분율은 볼륨 사용을 해당 end_of _life 설정의 백분율로 나타낸 것입니다. 000.0%부터 100.0% 사이의 값입니다. *warranty_life*는 *end_of_life*보다 작습니다.

  이 정보를 사용하여 데이터를 새 카트리지로 마이그레이션한 후 폐기해야 할 카트리지를 식별할 수 있습니다.
- *load_limit_alert*

  볼륨의 *load_limit_alert*는 부울 값입니다. 여기서 True=1이고 False=0입니다. *end_of_life* 백분율을 보고하는 볼륨이 100%에 도달하면 해당 *load_limit_alert*가 1로 설정됩니다. 일부 드라이브 유형은 *load_limit_alert*만 보고합니다.
- *-s sort_field*

  필드에 의해 반환되는 객체를 지정된 순서로 선택하여 정렬합니다. 내부 데이터베이스 값을 기준으로 정렬되며 영숫자 순서와 다를 수도 있습니다. 예를 들어, *media type*은 리터럴(*display*)이 아니라 내부 숫자 매체 유형을 기준으로 정렬됩니다.

  유형에 대해 반환되는 모든 필드는 유효한 정렬 필드입니다.
- *-n* n

  표시할 최대 객체 수를 지정합니다.

예

- 위치를 기준으로 볼륨을 표시하려면 다음 명령을 사용합니다.

  이 예에서는 패널 0,1,9에 홈 위치가 있는 모든 볼륨을 표시합니다.

  ```
  display volume * -home 0,1,9,*,*
  ```

- 모든 9840 데이터 볼륨을 표시하려면 다음 명령을 사용합니다.

  ```
  display volume * -media STK1R
  ```
- 9840 데이터 볼륨을 기록 형식(알려진 경우)과 함께 표시하려면 다음 명령을 사용합니다.

  ```
  display volume * -media STK1R -f media recording_format_family
  recording_format_model
  ```
- ACS 및 매체 유형별로 청소 볼륨을 표시하려면 다음 명령을 사용합니다.

  ```
  display volume * -home acs,*,*,*,* -media media_type
  -f vol_id acs lsm media max_use access_count
  ```
- 날짜별로 볼륨을 표시하려면 다음 명령을 사용합니다.
  - 2011년에 넣은 모든 볼륨을 표시하려면 다음 명령을 사용합니다.

    ```
    display volume * -entry 2011
    ```
  - 2011년 1월에 넣은 모든 볼륨을 표시하려면 다음 명령을 사용합니다.

    ```
    display volume * -entry 2011-01
    ```
  - 2011년 1월 27일에 넣은 모든 볼륨을 표시하려면 다음 명령을 사용합니다.

    ```
    display volume * -entry 2011-01-27
    ```
  - 2011년 1월 27일 오전 10시에서 오전 11시 사이에 넣은 모든 볼륨을 표시하려면 다음 명령을 사용합니다.

    ```
    display volume * -entry 2011-01-27:10
    ```
  - 2011년 1월에서 2011년 4월 사이에 넣은 모든 볼륨을 표시하려면 다음 명령을 사용합니다.

    ```
    display volume * -entry 2011-01-2011-04
    ```
  - 4월 10일 오전 6시 33분에서 6시 57분 45초 사이에 넣은 모든 볼륨을 표시하려면(예: 장애 후) 다음 명령을 사용합니다.

    ```
    display volume * -entry 2011-04-10:06:33:00-2011-04-10:6:57:45
    ```
  - 2011년 2월 25일 이후에 넣은 모든 볼륨을 표시하려면 다음 명령을 사용합니다.

    ```
    display volume * -entry >2011-02-25
    ```
  - T10000 데이터 볼륨을 *end_of_life* 사용량별로 정렬하여 표시하려면 다음 명령을 사용합니다.

    ```
    display volume * -media T10000T1 T10000TS -f media end_of_life
    warranty_life -s end_of_life
    ```
  - 모든 볼륨의 보증 기간과 수명 종료를 end_of_life 사용량별로 정렬하여 표시하려면 다음 명령을 사용합니다.

    ```
    display volume * -f warranty_life end_of_life -s end_of_life
    ```

- 모두 사용된 청소 카트리지를 표시하려면 다음 명령을 사용합니다.

  ```
  display volume * -spent_clean
  ```

15

15장. lib_cmd

이 장에서는 선택적 CLI(명령줄 인터페이스)인 *lib_cmd*에 대해 설명합니다. 이 도구는 ACSLS에서 논리적 라이브러리를 관리하는 데 주로 사용되지만, 일부 명령은 특정 컨텍스트에서 ACSLS GUI 또는 *cmd_proc*에 대한 대안으로 사용할 수도 있습니다. *lib_cmd* CLI는 42개를 초과하는 긴 볼륨 목록을 지정하는 기능을 비롯한 꺼내기 작업을 지원합니다.

소개

ACSLS 버전 8.0에서는 논리적 라이브러리 개념을 도입했습니다. 논리적 라이브러리 클라이언트 작업은 광 섬유 채널 연결을 통해 SMCE(SCSI Media Changer Emulation)에서 SCSI 명령을 사용하여 액세스합니다. 클라이언트는 테이프 *mount*, *dismount*, *enter* 및 *eject*를 수행할 수 있으며, SMCE 인터페이스를 사용하여 자체 테이프 인벤토리를 관리하고, 라이브러리가 클라이언트 응용 프로그램에 직접 연결된 것처럼 작동합니다.

ACSLS 8.2 이전 버전에서는 ACSLS 8.0에서 도입된 ACSLS 그래픽 사용자 인터페이스를 통해 논리적 라이브러리 관리 설정 기능을 수행했습니다. 관련 작업으로는 원하는 수의 스토리지 셀과 테이프 드라이브 슬롯을 포함하는 논리적 테이프 라이브러리 만들기, 클라이언트 개시자-대상 관계 설정, 라이브러리에 물리적 드라이버 지정, 테이프 볼륨 지정 등이 있습니다.

이러한 관리 작업은 UNIX CLI(명령줄 인터페이스)를 통해 수행할 수도 있습니다. CLI는 논리적 라이브러리의 유용성을 확장하여 중요 관리 기능을 더 빠르게 실행할 수 있는 경로를 제공하고, 일괄 처리 UNIX 셸 스크립트를 사용하여 논리적 라이브러리를 관리할 수 있도록 해 줍니다.

명령

lib_cmd

경로 이름

$ACS_HOME/bin/lib_cmd

개요

```
lib_cmd
lib_cmd [-f infile]
lib_cmd assign drive drive_id lib_id
lib_cmd assign volume [vol_id | vol-range] lib_id
lib_cmd unassign volume [ vol_id | vol-range ] lib_id
lib_cmd unassign drive lib_id  logical_drive_id
lib_cmd create library lib_name backing_acs cell_capacity drive_capacity
```

```
lib_cmd create mapping initiator_id target_id library_id
lib_cmd edit initiator initiator_id alias
lib_cmd edit library lib_id [-n name ]
                            [-c capacity ]
                            [-d drive_slots ]
                            [-f volume_label_format [6|8p|8s|all]
                            [-x imp/exp_cell_count ]
lib_cmd edit target target_id alias
lib_cmd refresh initiator
lib_cmd refresh target
lib_cmd delete initiator initiator_id (y/n)
lib_cmd delete library lib_id (y/n)
lib_cmd delete mapping lib_id (y/n)
lib_cmd delete target target_id  (y/n)
lib_cmd display drive [drive_id ]
                      [-p acs_id | all ]
                      [-l lib_id | all ]
                      [-t drive_type ]
                      [-u acs_id ]
lib_cmd display initiator
lib_cmd display library [ -p acs_id | all ]
                        [ -l lib_id | all ]
lib_cmd display mapping [ lib_id | all ]
lib_cmd display target
lib_cmd display volume [ vol_id ] | vol-range ]
                       [ -p acs_id | all ]
                       [ -l lib_id | all ]
                       [ -u acs_id ]
lib_cmd eject cap <cap_id> [-verbose] volume <vol_id...vol_id> | file <path_to_volume
_list>
lib_cmd vary library lib_id [online|offline|diagnostic]
lib_cmd vary drive lib_id drive_id  [online|offline|diagnostic]
lib_cmd [ exit | quit | log ]
```

lib_cmd 사용

lib_cmd 명령줄 유틸리티는 ACSLS 논리적 라이브러리의 리소스 관리 및 모니터링을 위한 ACSLS GUI를 대체합니다. ACSLS 관리자는 lib_cmd를 사용하여 논리적 라이브러리를 생성, 편집 또는 삭제하거나, 논리적 라이브러리에 볼륨 또는 테이프 드라이브를 지정 또는 지정 해제하거나, 개시자 또는 대상을 새로 고치거나, 클라이언트 통신을 위해 개시자-대상 매핑을 만들거나, 논리적 라이브러리의 상태를 표시하거나, 구성된 물리적 라이브러리 또는 논리적 라이브러리 내의 볼륨 또는 드라이브 상태를 표시할 수 있습니다.

논리적 라이브러리 관리 외에도 lib_cmd는 라이브러리의 컨텐츠를 표시하거나 단일 CAP(예: SL3000 AEM)에 대한 많은 양의 볼륨을 꺼내기 위해 cmd_proc 대신 사용할 수 있는 몇 가지 일반적인 라이브러리 명령 세트를 제공합니다.

이러한 작업은 *root*를 비롯하여 ACSLS 그룹 내의 모든 사용자가 수행할 수 있습니다.

*lib_cmd*는 대화식 모드 또는 일괄 처리 모드로 작동할 수 있습니다. 대화식 모드는 인수 없이 명령에 의해서만 호출됩니다.

```
lib_cmd
```

대화식 모드에서는 모든 가능한 하위 명령과 인수를 나열하는 'help' 기능을 지원합니다. 각 하위 명령의 컨텍스트 내에 요청 시 가능한 인수 및 옵션 목록을 표시할 수 있습니다. 가능한

유효 입력 문자열이 확실치 않은 경우 *<Enter>* 키를 눌러 해당 컨텍스트에 대해 사용 가능한 모든 인수 목록을 가져올 수 있습니다.

사용자 생성 스크립트에서 이 유틸리티의 효율적인 사용을 지원하기 위해 일괄 처리 모드에서는 상호 작용이 제공되지 않습니다. 일괄 처리 모드에서는 명령을 사용하여 모든 원하는 옵션 및 인수를 제출해야 합니다. 그렇지 않으면 오류 메시지가 반환됩니다.

옵션

- *-f*(명령 파일 입력 사양)

  -f 옵션이 요청되면 유틸리티는 지정된 입력 파일에서 찾은 일련의 명령을 실행합니다. 입력 파일에는 완전한 형태의 요청만 포함되어야 합니다.

하위 명령

이 절에서는 하위 명령에 대해 설명합니다.

assign drive drive_id lib_id

지정된 라이브러리에 지정된 드라이브를 지정합니다. *drive_id*는 물리적 라이브러리의 드라이브에 대한 표준 표기법으로 표시됩니다(*acs,lsm,panel,drive*). *lib_id* 인수는 논리적 라이브러리의 지정된 ID입니다. 논리적 라이브러리 ID는 1001부터 시작하며 라이브러리가 생성되면 증분적으로 증가합니다.

unassign drive lib_id logical_drive_id

논리적 라이브러리 구성에서 지정된 논리적 드라이브를 제거합니다. 드라이브는 논리적 주소(주소 500부터 시작)로 지정됩니다. 논리적 라이브러리 ID는 1001부터 시작하는 4자리 숫자입니다.

assign volume vol_id | vol-range lib_id

논리적 라이브러리 내에서 배타적 사용을 위해 볼륨을 지정합니다. 볼륨은 표준 6자리 또는 바코드 레이블 형식으로 표시됩니다. 라이브러리는 숫자 논리적 라이브러리 ID로 표시됩니다. 볼륨은 '지정 가능'해야 합니다. 즉, 아직 지정되지 않았고, 소유자가 없으며, 마운트, 예약, 전송 또는 사용 중이 아니어야 합니다.

범위는 대시로 구분된 하한 ACSII 값 문자열과 상한 ASCII 값 문자열로 지정됩니다. 볼륨 범위가 요청되면 *lib_cmd*는 지원 ACS에서 지정된 범위에 속하는 볼륨을 검색합니다. 범위 내에 있는 모든 지정 가능한 볼륨이 지정된 논리적 라이브러리에 지정됩니다.

unassign volume vol_id | vol-range lib_id

논리적 라이브러리 인벤토리에서 볼륨을 제거합니다. 볼륨은 홈 셀 내에 있어야 하며 보류 중인 이동 작업에 사용하도록 예약할 수 없습니다.

범위는 대시로 구분된 하한 ACSII 값 문자열과 상한 ASCII 값 문자열로 지정됩니다. 볼륨 범위가 요청되면 *lib_cmd*는 지정된 논리적 라이브러리에서 지정된 범위에 속하는 볼륨을 검색합니다. 범위 내에서 사용 중이 아닌 모든 지정된 볼륨이 라이브러리에서 지정 해제됩니다.

create library lib_name backing_acs cell_capacity drive_capacity

새 논리적 라이브러리를 만듭니다. *create library* 뒤에 4개의 인수가 라이브러리 별칭 이름(문자열), 지원 ACS ID(정수), 스토리지 셀 수(정수), 드라이브 슬롯 수(정수)의 순서로 와야 합니다.

create mapping initiator_id target_id library_id

지정된 라이브러리에 대한 개시자-대상(클라이언트-서버) 관계를 설정합니다.

edit library lib_id [-n name] [-c capacity] [-d drive_slots] [-f volume_label_format [6|8p|8s|all] [-x imp/exp_cell_count]

기존 논리적 라이브러리의 구성을 변경합니다. 특정 라이브러리는 4자리 숫자 ID로 표시됩니다. 모든 단일 옵션이 제공될 수 있으며, 해당 옵션과 함께 제공된 인수를 사용하여 단일 필드만 변경됩니다. 옵션은 다음과 같습니다.

- *-n* 라이브러리 별칭 이름
- *-c* 논리적 스토리지 셀 용량
- *-d* 논리적 드라이브 슬롯 수
- *-f* 볼륨 레이블 형식
- *-x* 논리적 가져오기/내보내기 (CAP) 셀 수

볼륨 레이블 형식은 다음 중 하나로 표현될 수 있습니다.

- *6* 6자 레거시 볼륨 레이블
- *8p* 매체 유형 코드가 앞에 나오는 8자 문자
- *8s* 매체 유형 코드가 뒤에 나오는 8자 문자
- *all* 모든 레이블 형식 허용

refresh initiator

현재 ACSLS로 알려진 모든 개시자를 검색합니다.

refresh target

현재 ACSLS로 알려진 모든 대상을 검색합니다.

delete initiator initiator_id (y/n)

구성에서 지정된 개시자를 제거합니다. 지정된 삭제를 커밋하려면 확인(y)이 필요합니다.

delete library lib_id (y/n)

ACSLS 구성에서 지정된 논리적 라이브러리를 제거합니다. 이렇게 하면 계단식 *delete*, 라이브러리에서 지정된 볼륨 및 드라이브 연결 해제를 만듭니다. 삭제를 시도하기 전에 라이브러리를 오프라인 상태로 전환해야 합니다. 라이브러리에서 하나 이상의 논리적 볼륨이 현재 사용 중이면 제거가 실패합니다. 지정된 삭제를 커밋하려면 확인(y)이 필요합니다.

delete mapping lib_id (y/n)

지정된 논리적 라이브러리에서 모든 개시자-대상 매핑을 제거합니다. 이 작업은 현재 클라이언트 연결을 사용 안함으로 설정합니다. 지정된 삭제를 커밋하려면 확인(y)이 필요합니다.

delete target target_id (y/n)

구성에서 지정된 대상을 제거합니다. 지정된 삭제를 커밋하려면 확인(**y**)이 필요합니다.

display drive drive_id [-t drive_type] [-p acs_id | all] [-l lib_id | all] [-u acs_id]

지정된 라이브러리의 지정된 드라이브 또는 모든 드라이브 요약을 생성합니다.

표시에는 드라이브의 물리적 주소와 논리적 주소, 드라이브 유형, 드라이브 상태 등이 표시되고, *drive_status*가 *in_use*를 표시하는 경우 마운트된 카트리지의 *VOLSER*가 표시됩니다.

```
----------------------- Example ----------------------------
$ lib_cmd display drive 0,0,10,1
 Physical   Logical    Drive     Drive     Drive      Mounted
 Location   Address    Type      State     Status     Volume
 0,0,10,1   1001:502   9840      Online    in use     RIFF21
------------------------------------------------------------
```

- *display drive all*이 옵션 없이 제출된 경우 유틸리티는 각 물리적 ACS에 포함되어 있는 모든 볼륨 목록을 생성합니다.
- *-t* 옵션이 전달된 경우 지정된 드라이브 유형의 드라이브만 표시됩니다.
- *-p* 옵션은 지정된 물리적 ACS의 드라이브에 대한 표시를 제한합니다.
- *-l* 옵션이 포함된 경우 지정된 논리적 라이브러리에 지정된 드라이브만 표시됩니다. *all*이 *-l* 옵션 뒤에 지정된 경우 구성된 각 논리적 라이브러리와 연관된 모든 드라이브를 보여주는 표시가 생성됩니다.
- *-u* 옵션은 ACS와 연관된 지정되지 않은 드라이브만 표시합니다. 이 옵션은 원하는 ACS를 지정하는 인수와 함께 나와야 합니다.

display initiator

ACSLS에서 식별하는 모든 개시자 목록을 생성합니다. 각 개시자는 World Wide 이름 및 별칭 이름을 기준으로 나열됩니다.

display library [-l lib_id | all][-p acs_id | all]

요청된 라이브러리 ID의 속성을 표시합니다. *-l*(논리적 라이브러리의 경우) 또는 '*-p*'(물리적 ACS의 경우)를 지정합니다.

- *-l*을 지정하면 유틸리티는 다음과 같이 논리적 라이브러리 구성에 대한 요약을 생성합니다.

  숫자 ID, 이름, 상태(원하는 상태)

지원 ACS ID, 상태(원하는 상태)

할당된 가져오기/내보내기 셀 수

할당된 드라이브 슬롯 수

지정된 드라이브 수

할당된 스토리지 셀 수

사용 가능한 셀 수

지정된 볼륨 수(액세스할 수 없는 볼륨 수)

볼륨 레이블 형식

```
-----------------Example-----------------------------------
ACS 1001 'logLib01' Offline (Desired Online)
   Backing ACS 0 Offline (Desired Online)
   2 import/export cells
   5 Tape Drive Slots
   2 Tape Drives
   999 Storage Cells
   993 Free Cells
   106 Volumes (6 Inaccessible):
   Volume Label Format: Six character (classic)
-----------------------------------------------------
```

액세스할 수 없는 볼륨은 다음과 같은 이유로 논리적 라이브러리의 범위를 벗어난 지정된 볼륨입니다.

- 볼륨을 물리적 라이브러리에서 꺼냈습니다.
- 클라이언트 응용 프로그램에 볼륨이 *eject*로 표시되었습니다.
- 지정된 볼륨이 물리적 라이브러리에 있지만, 논리적 라이브러리가 스토리지 용량을 초과했습니다.
- *all* 인수를 숫자 *lib_id* 대신 지정한 경우 구성된 각 논리적 라이브러리에 대해 표시가 반복됩니다. 일괄 처리 모드에서 *-l*을 인수 없이 전달하면 *all* 인수를 사용한 것과 같습니다.
- *-p* 옵션을 지정된 ACS ID와 함께 지정한 경우 유틸리티는 해당 ACS에 대한 라이브러리 구성을 요약하는 표시를 생성합니다.

```
------------------------------ Example ---------------------------------
$ lib_cmd display lib -p 0
Physical ACS 0 Online 56 Drives:
1 LSM:
LSM  Library   LSM     LSM    Drive  Vol     Free Cell
```

```
  ID    Type    Status  State   Count  Count   Count
  0,0   SL3000  Normal  Online   56    62      4321
  10 CAPs:
   ID     Mode        State    Status     Condition    Size     Availability
  0,0,1   Manual     Online   Available   Operative    26       shared
  0,0,2   Manual     Online   Available   Operative    26       shared
  0,0,3   Manual     Online   Available   Operative    26       shared
  0,0,4   Manual     Online   Available   Operative    26       shared
  0,0,5   Manual     Online   Available   Operative    26       shared
  0,0,6   Automatic  Online   Available   Operative    26       shared
  0,0,7   Manual     Online   Available   Operative    26       shared
  0,0,8   Manual     Online   Available   Operative    26       shared
  0,0,9   Manual     Online   Available   Operative    26       shared
  0,0,10  Manual     Online   Available   Operative    26       shared
  ------------------------------------------------------------------------
```

- *all* 인수를 *-p* 옵션과 함께 전달하면 구성된 각 물리적 ACS에 대해 표시가 반복됩니다.

  all 인수는 일괄 처리 모드의 요청에 대한 기본 표시입니다. 인수가 없는 *-p* 옵션은 *all*을 요청한 것과 같습니다.

display mapping [lib_id | all]

각 라이브러리 ID(또는 모든 라이브러리)에 대한 개시자-대상 매핑 목록을 생성합니다.

display volume [vol_id] | vol-range] [-p [acs_id | all] [-l [lib_id | all] [-u acs_id]

지정된 볼륨 또는 볼륨 세트에 대한 요약 정보를 생성합니다.

```
----------------------Example ----------------------------
$ lib_cmd display volume ST0212
   Volume       media     current      physical     logical
     ID         type      status       location     address
   ST0212       STK1R     Home        0,0,2,6,0     1001:100
-------------------------------------------------------------
```

- 볼륨이 마운트되면 논리적 주소와 물리적 위치에 볼륨이 마운트되는 드라이브의 주소가 반영됩니다. 그렇지 않으면 볼륨의 논리적 홈 주소와 물리적 홈 주소가 나열됩니다. 논리적 주소는 볼륨이 논리적 라이브러리에 지정된 경우에만 채워집니다.
- 볼륨 범위는 대시로 구분된 하한 ACSII 값 문자열과 상한 ASCII 값 문자열로 지정할 수 있습니다. 표시에는 지정된 범위 내의 각 볼륨에 대한 상태 요약이 표시됩니다.
- *display volume all*을 옵션 없이 제출한 경우 표시에는 각 물리적 ACS에 포함된 모든 볼륨이 표시됩니다.
- 물리적 ACS ID를 *-p* 옵션과 함께 전달하면 목록은 지정된 물리적 ACS 내에 있는 볼륨으로 제한됩니다. *all*을 *-p*와 함께 제출하면 모든 물리적 ACS에 포함된 볼륨이 표시됩니다.

- 논리적 라이브러리의 숫자 ID(*lib_id*)를 *-l* 옵션과 함께 전달하면 특정 논리적 라이브러리와 연관된 볼륨 세트만 표시됩니다. 단어 *all*을 *-l* 옵션과 함께 전달하면 유틸리티는 구성된 각 논리적 라이브러리와 연관된 모든 볼륨을 표시합니다.
- *-u* 옵션은 지정된 물리적 ACS에 포함된 지정되지 않은 볼륨으로 제한되는 표시를 생성합니다.

eject cap <cap_id> [-verbose] volume <vol_id...vol_id> | file <path_to_volume_list>

*lib_cmd eject*는 42개를 초과하는 긴 볼륨 목록을 지정하는 기능을 비롯한 꺼내기 작업을 지원합니다. 볼륨 목록을 포함하는 텍스트 파일에 대한 경로를 지정하거나 사용자 입력으로 볼륨을 입력할 수 있습니다. 여러 볼륨을 한 라인에 대문자 또는 소문자로 입력할 수 있습니다.

- <*cap_id*>
  - 라이브러리 구성에 있는 특정한 기존 CAP여야 합니다.
  - 와일드카드는 지원되지 않습니다.
  - CAP 목록을 사용할 수 있습니다.
- *[-verbose | -v]*
  - *verbose* 옵션은 모든 볼륨을 표시하도록 지정합니다.
  - 기본값은 verbose가 아니며 최대 10개의 볼륨을 개수와 함께 표시합니다.
- <*vol_id*>
  - 라인당 하나 이상의 볼륨 식별자를 지정할 수 있습니다.
  - 대화식 모드로 실행 중인 경우 <*Return*> 키를 누르면 목록이 종료됩니다.
- <*path_to_volume_list*>
  - 텍스트 파일에 대한 상대 경로 또는 전체 경로로 지정합니다.
  - 라인당 여러 볼륨 식별자를 지정할 수 있습니다.
  - 모든 설명 라인("#"으로 시작) 또는 빈 라인은 무시됩니다.

어떠한 형태로든 지정 가능한 볼륨 수에 대한 알려진 제한이 없습니다. 모든 볼륨이 제공되면 ACSLS에서 단일 꺼내기 작업이 수행되고, 가능한 볼륨을 모두 꺼낼 때까지 필요한 대로 지정된 CAP를 계속해서 채웁니다.

출력 표시에는 볼륨 수(지정된 볼륨 수, 꺼낸 볼륨 수, 꺼내지 않은 볼륨 수 등) 보고가 포함됩니다. *verbose* 옵션은 모든 볼륨을 표시하는 출력을 생성합니다. 기본적으로 각 목록에 대해 10개의 볼륨이 표시된 미리보기가 생성됩니다. 꺼내지 않은 볼륨의 경우 볼륨 식별자와 이유를 나타내는 라인이 표시됩니다.

vary drive lib_id drive_id [online|offline|diagnostic]

논리적 드라이브의 원하는 상태를 지정된 상태(온라인, 오프라인, 진단)로 변경합니다.

vary library lib_id [online|offline|diagnostic]

논리적 라이브러리의 원하는 상태를 지정된 상태(온라인, 오프라인, 진단)로 변경합니다.

일괄 처리 모드에서 lib_cmd 사용

일괄 처리 모드에서 *lib_cmd*를 사용하면 더 복잡한 관리 작업을 수행할 수 있습니다. 예를 들어, 여러 볼륨 또는 여러 드라이브를 지정하거나 지정 해제하는 작업은 대체로 일괄 처리 모드에서 가장 효율적으로 수행됩니다.

이 예에서는 SL8500 모듈의 레일 2에 있는 모든 9840 드라이브를 논리적 라이브러리 1002에 지정합니다.

1. *lib_cmd*를 사용하여 acs-0에서 지정되지 않은 모든 드라이브를 나열하고, *grep*를 사용하여 레일 2에 있는 9840 드라이브만 필터링합니다(acs-0, lsm-1).

   ```
   $ lib_cmd display drive -u 0  |  grep 9840 | grep 0,1,.,.
   ```

   레일 2에 있는 모든 지정되지 않은 드라이브는 5개 필드 표시에 나열됩니다.

   ```
   Physical Logical   Drive  Drive   Drive      Mounted
   Location Address   Type   State   Status     Volume
   0,1,1,0            9840   Online  available
   0,1,1,1            9840   Online  available
   0,1,1,2            9840   Online  available
   0,1,1,3            9840   Online  available
   0,1,1,4            9840   Online  available
   ```

   여기서는 이 표시의 드라이브 ID만 필요하므로 *awk*를 사용하여 첫번째 필드만 인쇄합니다.

   ```
   $ lib_cmd display drive -u 0 | grep 9840|grep 0,1,.,.|awk ’{print $1}’
   0,1,1,0
   0,1,1,1
   0,1,1,2
   0,1,1,3
   0,1,1,4
   ```

2. 동일한 *awk* 인쇄 문을 사용하여 논리적 라이브러리 1002에 드라이브를 지정하는 데 필요한 명령을 구성하는 텍스트를 추가할 수 있습니다.

   ```
   $ lib_cmd display drive -u 0 | grep 9840|grep 0,1,.,.|awk ’{print ”assign drive
    ”$1”1002}’
   assign drive 0,1,1,0 1002
   assign drive 0,1,1,1 1002
   assign drive 0,1,1,2 1002
   assign drive 0,1,1,3 1002
   assign drive 0,1,1,4 1002
   ```

3. 명령 텍스트에 만족하는 경우 이제 텍스트 파일에 대한 출력을 지시할 수 있습니다.

```
$ lib_cmd display drive -u 0 | grep 9840 | grep 0,1,.,. | awk '{print "assign
 drive "$1" 1002"}'  > /tmp/assignDr
```

4. 마지막으로, 명령 파일에 대한 변경사항을 편집한 다음 *lib_cmd*를 사용하여 파일을 실행할 수 있습니다.

```
$ lib_cmd -f ./tmp./assignDr
--ACSLS 8.2.0--
Copyright (c) 2012 Oracle and/or its affiliates. All rights reserved.
Drive 0,1,1,0 now assigned to logLib02 at 1002:500
Drive 0,1,1,1 now assigned to logLib02 at 1002:501
Drive 0,1,1,2 now assigned to logLib02 at 1002:502
Drive 0,1,1,3 now assigned to logLib02 at 1002:503
Drive 0,1,1,4 now assigned to logLib02 at 1002:504
```

5. *lib_cmd* 표시 명령을 사용하여 지정을 확인합니다.

```
$ lib_cmd  display  drives  -l  1002
ACS 1002 LOG lIB02:20 Drive Slots 5 Drives
Physical   Logical      Drive  Drive   Drive       Mounted
Location   Address      Type   State   Status      Volume
0,1,1,0    1002:500     9840   Online  available
0,1,1,1    1002:501     9840   Online  available
0,1,1,2    1002:502     9840   Online  available
0,1,1,3    1002:503     9840   Online  available
0,1,1,4    1002:504     9840   Online  available
```

부록 A. ACSLS 백업 및 복구 도구

이 부록의 내용:

- 각 유틸리티, 해당 유틸리티의 용도 및 해당 유틸리티가 중요한 이유에 대한 개요와 설명을 제공합니다.
- 재해 복구 시나리오에 대한 간략한 설명을 제공합니다.

ACSLS 백업 도구

ACSLS는 해당 데이터베이스와 ACSLS 제어 파일을 모두 백업하기 위해 강력하면서 확연히 다른 세 가지 방법을 제공합니다. 각 유틸리티는 서로 다른 기능을 수행하고 모든 방법은 전체 재해 복구 계획에서 중요한 역할을 합니다.

자동 백업

ACSLS는 자동화된 데이터베이스 보호 서비스를 제공합니다. 이러한 자동화된 보호 서비스는 의도하지 않은 결과를 생성하거나 데이터베이스 손상으로 인해 발생할 수 있는 변경사항에 대비하여 ACSLS 데이터베이스의 일간 작업을 안전하게 보호합니다.

결과적으로 자동화된 백업 보호 서비스를 통해 데이터베이스를 현재에서 보존 기간의 끝까지 모든 백업 시간으로 다시 복원할 수 있습니다. 복원 도구는 이 부록의 뒷부분에서 설명합니다.

이 절에서는 자동화된 백업 방법 및 해당 방법이 사용되는 이유에 대해 설명합니다.

- ACSLS 기본 백업 디렉토리

  ACSLS 초기 설치 시 백업에 사용할 디렉토리 이름을 제공해야 합니다(기본값: */export/backup*). 백업 작동이 발생하는 이 디렉토리에 있습니다.
- 전체 데이터베이스 백업이 수행되고 날짜 이름 지정 규칙을 사용하여 디렉토리에 배치됩니다.

  /export/backup/yyyy-mm-dd-hh:mm:ss.tar

  일 단위 백업이 수행되는 시간은 *acsss_config* 내에서 "Automatic Backup Variables"를 변경하여 수정할 수 있습니다.

  기본 백업 동작 변경에 대한 자세한 내용은 6장. *ACSLS* 동작을 제어하는 변수 설정을 참조하십시오.
- 데이터베이스 보존 기간

자동 백업에 영향을 주는 ACSLS 내 다른 구성 가능 매개변수는 데이터베이스 보존 기간입니다. 이는 ACSLS가 백업을 보존하는 기간으로 정의됩니다.

보존 기간에 대한 기본값은 8일입니다.

기본 백업 동작 변경에 대한 자세한 내용은 6장. *ACSLS* 동작을 제어하는 변수 설정을 참조하십시오.

보존 기간은 *acsss_config*를 사용하여 수정할 수도 있습니다.

수동 백업

ACSLS는 명령줄을 사용하여 중요한 ACSLS 데이터를 백업하는 *bdb.acsss* 유틸리티를 제공합니다. 환경이 같거나 동일한 하드웨어, OS 레벨 및 ACSLS 버전으로 구성된 ACSLS 데이터베이스를 복원하는 데 사용되는 방법이기도 합니다. “bdb.acsss ”를 참조하십시오.

명령줄 옵션 없이 사용되는 *bdb.acsss*는 데이터베이스 백업을 만들고 해당 백업을 기본 백업 디렉토리에 저장하는 기능을 제공합니다. 중요한 ACSLS 데이터베이스 및 ACSLS 제어 파일이 단일 파일로 모두 백업됩니다. 그런 다음 이 파일은 내부 디스크 또는 마더보드 결함과 같은 시나리오가 발생한 경우 같거나 동일한 하드웨어의 이전 상태로 ACSLS를 복원하는 데 사용될 수 있습니다.

rdb_acsss 유틸리티는 파일 및 위치(*rdb.acsss -f /path/my_file*) 또는 테이프 장치(*-f /dev/rmt/0mn*)를 지정하는 데 사용할 수 있는
'*-f*' 옵션을 허용합니다. 테이프 장치를 사용할 때 테이프 장치에 파일 이름을 제공하지 않습니다.

수동 데이터베이스 내보내기

ACSLS는 *db_export.sh* 유틸리티를 제공하여 ACSLS 데이터베이스, ACSLS 제어 파일 및 사용자 정의된 모든 동적 변수를 내보냅니다. *db_export.sh* 유틸리티는 ACSLS 데이터베이스를 쉼표로 구분된 ACSII 파일에 덤프하고, ACSLS 제어 파일의 복사본을 만들고, 동적 변수의 복사본을 만듭니다. 이는 최신 버전의 ACSLS로 마이그레이션하는 데 사용되는 방법이며, 내보내기를 수행하기 전에 ACSLS와 데이터베이스가 모두 작동 중지되어야 하므로 일 단위 백업 작업에 권장되지 않습니다.

db_export.sh 명령줄 유틸리티는 서로 다른 레벨의 서버 하드웨어, OS 버전 및 서로 다른 ACSLS 릴리스 간에 데이터베이스를 마이그레이션하는 경우 선호되는 방법입니다. 옵션 없이 */dev/0mn*과 같은 로컬 기본 테이프 장치와 함께 사용할 수 있습니다. 그런 다음 이 테이프는 모든 위치로 이동될 수 있으며 ACSLS 및 연관된 ACSLS 제어 파일은 모든 OS 버전 또는 모든 레벨의 ACSLS로 복원될 수 있습니다.

주:

모든 테이프 장치를 선택할 수 있지만 *no-rewind* 장치가 사용되어야 합니다. `db_export` 유틸리티는 두 개의 파일을 만듭니다. 되감기 장치가 선택된 경우 첫번째 파일(데이터 파일)은 두번째 파일이 생성될 때 덮어쓰여집니다.

bdb.acsss 유틸리티처럼 '*-f*' 옵션은 시스템 기본값 이외의 테이프 장치를 지정하는 데 사용할 수 있습니다. 이 옵션을 사용하려면 *db_export.sh /dev/0mn* 또는 연결된 테이프 장치를 실행하기만 하면 됩니다.

또한 *-f* 옵션을 사용하면 데이터베이스를 이름이 지정된 파일로 내보낼 수 있습니다. 이 방법을 사용하면 두 개의 파일이 생성되었음을 확인할 수 있습니다. 하나는 이름이 지정된 파일이고 다른 하나는 확장자가 *.misc*인 파일입니다. 두 파일은 가져오기를 성공적으로 수행하기 위해 가져오기를 수행할 서버로 전송해야 합니다.

-f 옵션을 사용하거나 사용하지 않고 *db_export.sh* 유틸리티를 실행하면 내보낼 ACSLS 버전을 선택하라는 프롬프트가 표시됩니다.

*db_export.sh*의 메뉴 선택은 다음과 같습니다.

```
1: ACSLS 7.3
2: ACSLS 8.0, 8.0.1, 8.0.2, 8.1
3. ACSLS 8.2 or 8.3
4. ACSLS 8.4
E: Exit
Please select by number (or E to exit):
```

ACSLS 복구 도구

ACSLS는 두 개의 서로 다른 복구 도구를 사용하여 모든 백업 및 내보내기를 복원합니다. 두 도구는 메뉴 기반 사용자 인터페이스 및 쉽게 선택할 수 있는 옵션을 제공합니다. 두 유틸리티는 다음과 같습니다.

- *rdb.acsss* - 자동 백업과 수동 백업 모두를 위한 복구 도구입니다.
- *db_import.sh* - 내보낸 데이터베이스 및/또는 ACSLS 제어 파일을 동일한 버전의 ACSLS, 다른 버전의 ACSLS 또는 다른 하드웨어 플랫폼에서 복원합니다. 또한 이 옵션으로 사용자 정의된 동적 변수를 복구할 수 있습니다.

rdb.acsss 사용

rdb.acsss 유틸리티는 자동 백업 기능 또는 *bdb.acsss* 유틸리티로 만든 백업을 사용하여 ACSLS 데이터베이스 및 ACSLS 제어 파일을 복원합니다. ACSLS 제어 파일은 *$ACS_HOME/data*에 있으며 ACSLS에 대해 서로 다른 여러 환경 변수를 정의합니다. 이러한 파일은 액세스 제어 설정, 스크래치 환경 설정, Extended Store LSM, 사용자 정의 *volrpt* 설정, 볼륨 속성(*watch_vols* 유틸리티의 경우) 등을 지정합니다.

옵션 및 절차는 “rdb.acsss ”를 참조하십시오.

db_import.sh 사용

ACSLS는 *db_import.sh* 유틸리티를 제공하여 동일한 버전의 ACSLS, 다른 버전의 ACSLS 또는 다른 하드웨어 플랫폼에서 내보낸 데이터베이스를 복원할 수 있습니다. 이 유틸리티는 *rdb.acsss*와 같이 수행할 작업을 선택할 수 있는 읽기 쉬운 메뉴 기반 사용자 인터페이스를 제공합니다.

db_import.sh 유틸리티를 옵션 없이 작동하거나 경로 및 파일 이름에 *'-f'* 옵션을 인수로 제공할 수 있습니다. 옵션 없이 명령줄에서 *db_import.sh*를 실행하면 유틸리티가 로컬 테이프 장치에서 내보낸 데이터베이스를 검색합니다. 먼저 내보낸 데이터베이스의 존재 여부를 검사하고, 해당 데이터베이스가 유효한 데이터베이스 내보내기 파일인지 확인한 다음 4개의 옵션이 있는 메뉴를 표시합니다.

주:

또한 기본값이 아닌 장치의 경우 테이프 장치에 *-f* 옵션을 제공(*-f /dev/rmt/0mn*)할 수 있습니다. 유효한 테이프 장치를 제공하더라도 되감기가 아닌 장치를 제공해야 합니다. *db_import.sh* 유틸리티는 두 개의 파일을 사용합니다. 하나는 데이터용이고, 다른 하나는 제어 파일용입니다. 되감기 장치를 사용하는 경우 데이터 파일이 복구된 후 테이프가 되감기고 제어 파일이 실패합니다.

경로 및 파일 이름에 *-f* 옵션을 제공하는 경우 *db_import.sh*는 제공된 파일 이름을 내보낸 데이터베이스 파일로 사용합니다. 로컬 테이프 장치처럼 먼저 파일이 존재하는지 여부를 확인한 다음 제공된 파일 이름이 내보낸 데이터베이스 파일인지 검증합니다. 제공된 파일이 유효한 내보내기인 경우 메뉴를 표시합니다. 메뉴 옵션은 다음과 같습니다.

- 옵션 1 - 내보낸 파일에 대한 데이터베이스 테이블, 제어 파일 및 동적 변수를 가져옵니다.

  이 옵션은 내보낸 버전에서 유지된 모든 사용자 정의 업데이트와 라이브러리 데이터베이스를 가져옵니다.
- 옵션 2 - 내보낸 파일에서 데이터베이스 테이블만 가져옵니다.

  이 옵션은 전체 라이브러리 구성 및 볼륨 데이터 세트를 가져오지만 내보낸 버전에서 수행된 시스템 사용자 정의는 적용하지 않습니다.
- 옵션 3 - 내보낸 파일에서 제어 파일만 가져옵니다.

  이 옵션은 현재 라이브러리 데이터베이스를 변경하지 않고 이전 버전에서 내보낸 사용자 정의만 가져옵니다.
- 옵션 4 - 내보낸 파일에서 사용자 정의된 동적 변수를 병합합니다.

  이 옵션은 내보낸 버전의 모든 사용자 정의된 설정을 현재 버전과 병합합니다. [71]6장. *ACSLS* 동작을 제어하는 변수 설정을 참조하십시오.

재해 시나리오

이 절에서는 재해 시나리오에 대해 설명합니다.

데이터베이스가 손상된 경우

1. 사용자 *acsss*로 복구를 실행하기 전에 ACSLS를 중지합니다.

   ```
   $ acsss db
   $ rdb.acsss
   ```

2. 옵션 2를 선택합니다. “rdb.acsss ”을 참조하십시오.
3. 복구가 완료되면 ACSLS를 시작합니다(*acsss enable*).

잘못된 라이브러리에 대해 acsss_config가 실행된 경우

1. 옵션 2를 선택합니다. “rdb.acsss ”을 참조하십시오.
2. ACSLS를 시작하고 데이터베이스 백업 및 복원 절차에 따라 테스트합니다.

서버 실패 – 새 하드웨어로 동일한 서버 재구축

1. 운영체제를 설치합니다.
2. 이전 서버의 설정을 사용하여 새 서버 및 OS를 구성합니다.
3. ACSLS를 설치합니다.
4. 백업 테이프 또는 FTP 백업 파일을 서버에 삽입합니다.
5. *rdb.acsss* 유틸리티를 시작합니다.
6. 옵션 2를 선택합니다. “rdb.acsss ”을 참조하십시오.
7. *rdb.acsss*를 종료합니다.
8. ACSLS를 시작하고 데이터베이스 백업 및 복원 절차에 따라 테스트합니다.

서버 실패 – 새 하드웨어로 다른 ACSLS 서버 재구축

1. 운영체제를 설치합니다.
2. ACSLS를 설치합니다.
3. ACSLS 서버 간 백업 파일을 적합한 위치에 배치합니다.
4. *rdb.acsss*를 입력합니다. “rdb.acsss ”를 참조하십시오.
5. 옵션 3을 선택합니다.
6. 복구 유틸리티가 완료되면 ACSLS를 시작하고 데이터베이스 백업 및 복원 절차에 따라 테스트합니다.

부록 B. 엔터프라이즈 라이브러리 연결 옵션

이 장은 다음 항목으로 구성됩니다.

개요

SL8500 및 SL3000 라이브러리에 ACSLS를 연결하기 위한 여러 옵션이 있습니다. ACSLS와 SL8500 또는 SL3000 간의 통신을 위해 이러한 옵션을 개별적으로 또는 함께 사용할 수 있습니다.

연결된 SL8500의 문자열에서 이중 TCP/IP, 다중 라이브러리 TCP/IP 및/또는 중복 전자 부품을 구현할 수 있습니다.

SL3000 또는 SL8500에서 이중 TCP/IP 및/또는 RE(중복 전자 부품)를 구현할 수 있습니다. IPv4를 통해 SL3000 또는 SL8500에 연결할 수 있습니다.

연결 옵션을 요약하면 다음과 같습니다.

- 이중 TCP/IP

  이중 TCP/IP는 ACSLS와 라이브러리 컨트롤러 카드 간에 두 개의 개별적인 독립 TCP/IP 연결을 제공합니다. 이 통신 경로 중 하나가 실패하면 ACSLS에서는 자동으로 두번째 경로를 사용하여 통신합니다.

  이중 TCP/IP 지원을 구현하려면 "*route*" 명령을 사용하여 ACSLS 서버와 라이브러리의 경로 지정 테이블을 정의하고 관리해야 합니다. 이러한 경로 지정 테이블은 정의된 네트워크 통신 경로를 사용하여 ACSLS 서버와 라이브러리의 포트 쌍 간에 통신이 이루어지도록 합니다.

  SL8500과 SL3000은 라이브러리와의 이중 TCP/IP 통신을 지원합니다.
- 다중 TCP/IP 지원

  ACSLS 서버에서 다중 TCP/IP 지원을 사용하여 일련의 연결된 SL8500에 있는 여러 SL8500 라이브러리에 연결할 수 있습니다. 한 라이브러리와의 통신을 실패하면 ACSLS에서는 다른 라이브러리와의 연결로 라이브러리 통신을 자동으로 보냅니다. 라이브러리는 메시지를 다른 라이브러리에 자동으로 전달합니다.

다중 TCP/IP 통신을 구성하여 관리하면 ACSLS 서버 또는 SL8500 라이브러리에서 경로 지정 테이블을 정의할 필요가 없으므로 이중 TCP/IP보다 더 간단합니다. 하지만 다중 TCP/IP를 사용하려면 일련의 연결된 SL8500 라이브러리가 필요합니다. 단일 독립형 SL8500 또는 SL3000 라이브러리에는 적용되지 않습니다.

- RE(중복 전자 부품)

  RE는 중복 라이브러리 컨트롤러 카드 세트를 사용합니다. 특정 시간에 한 세트는 활성 상태이고 다른 세트는 대기 상태입니다. 활성 라이브러리 컨트롤러는 ACSLS 또는 SLConsole의 명령에 응답하여 대기 라이브러리 컨트롤러로 페일오버할 수 있습니다. 라이브러리 카드에 장애가 발생할 경우 라이브러리에서 자동 페일오버를 시작할 수 있습니다.

  RE를 사용하면 라이브러리 펌웨어(마이크로코드) 다운로드 중단을 최소화할 수 있습니다. 일련의 연결된 SL8500 내에서 라이브러리 단위로 RE를 구현할 수 있습니다. 컴플렉스 내의 임의 또는 모든 라이브러리에서 RE를 구현할 수 있습니다.

  라이브러리에서 RE를 지원하려면 ACSLS 7.3.1 또는 8.0.2 이상이 필요합니다.

라이브러리와의 ACSLS 통신 상태 표시

query lmu 명령을 사용하여 관리되는 라이브러리와의 ACSLS 통신 상태를 보고 모니터할 수 있습니다. 또한 *query lmu* 명령은 라이브러리에 대한 ACS 및 포트 연결 상태를 표시합니다.

이중 TCP/IP 지원

이중 TCP/IP는 SL8500 및 SL3000 라이브러리(여기서는 라이브러리라고 함)를 위해 구매할 수 있는 옵션이며, 라이브러리에 대한 두 가지 TCP/IP 연결을 제공합니다. 하지만 두 연결 작업 중 하나를 통해서만 라이브러리를 계속 사용할 수 있습니다.

이중 TCP/IP의 용도는 실패한 통신 경로를 자동으로 인식하여 피하기 위한 것입니다. 이 작업은 자동으로 수행되므로 작동하지 않는 연결을 수동으로 전환할 필요가 없습니다.

라이브러리에서 이중 TCP/IP 지원을 사용하려면 "*route*" 명령을 사용하여 ACSLS 서버와 라이브러리의 경로 지정 테이블을 관리해야 합니다. 그러면 라이브러리의 정의된 네트워크 인터페이스에 대한 경로가 강제 지정되어 인터페이스 간 일대일 관계가 생성됩니다. CSA(Customer Systems Administrator)는 ACS 서버에서 경로 지정 테이블을 변경하고 CSE(Customer Systems Engineer)는 라이브러리에서 경로 지정 테이블을 업데이트합니다. UNIX "*route*" 명령에 대한 자세한 내용은 ACSLS 서버의 매뉴얼 페이지를 참조하십시오.

요구 사항

- 시스템 관리자 및 네트워크 관리자와 협력하여 현재 네트워크 환경을 이해하고 필요한 모든 IP 주소를 미리 식별합니다.

- 시스템 관리자와 협력하여 네트워크 인터페이스를 구성하거나 네트워크 인터페이스가 올바르게 구성되었는지 검증합니다.

구성

ACSLS에서는 모든 활성 연결을 사용하므로 ACSLS에서 라이브러리에 대한 두 연결을 계속 열어 두는 것이 좋습니다. 연결 중 하나가 작동하지 않을 경우 ACSLS에서는 작동하는 나머지 연결을 사용하면서 실패한 연결에 대한 통신을 다시 설정하기 위해 계속 시도합니다.

시나리오 1과 같이 이중 TCP/IP 구현을 위한 선호 구성은 ACSLS 서버의 개별 서브넷 두 개에 네트워크 인터페이스 두 개를 사용하는 것입니다. 이 구성은 네트워크 통신과 관련된 최대 처리량과 최소 리소스 경합을 제공하고, 향상된 안정성을 위한 보조 물리적 연결을 추가합니다.

단일 라이브러리에 대해 TCP/IP 연결 두 개를 구성하려면 *acsss_config* 유틸리티 또는 동적 구성(*config*)을 사용합니다. 라이브러리에 대한 연결 수(2)와 네트워크 장치의 IP 주소를 입력합니다. SL3000은 IPv4 연결을 지원합니다.

다음 시나리오에서는 ACSLS 서버 구성을 위한 예를 제공합니다. 라이브러리 이중 TCP/IP 기능을 구성하는 방법에 대한 지침은 적절한 *Library System Dual TCP/IP Feature* 문서를 참조하십시오.

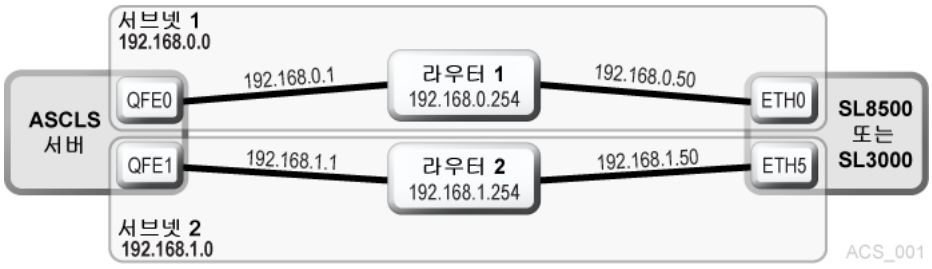
다음 시나리오에서는 전용 서브넷 IP 주소를 사용하므로 사용자 환경에서는 서브넷 IP 주소가 다를 수 있습니다. 이 시나리오에서는 네트워크 장치가 구성되어 있고 제대로 작동 중이라고 간주합니다.

시나리오 1 - 선호 구성

시나리오 1은 이중 TCP/IP 기능에 대한 선호 구성입니다.

이 구성에서 ACSLS 서버는 두 개의 개별 서브넷에 있는 두 개의 네트워크 인터페이스로 구성됩니다. SL8500 또는 SL3000은 ACSLS 서버와 동일한 두 서브넷에 두 개의 네트워크 인터페이스가 있습니다.

그림 B.1. 선호 구성



이 시나리오에서는 라이브러리가 ACSLS 서버의 네트워크 인터페이스와의 일대일 관계를 사용하며, 다음과 같이 통신합니다.

- ACSLS 서버의 qfe0 인터페이스는 SL8500 또는 SL3000의 eth0 인터페이스와만 통신합니다.
- ACSLS 서버의 qfe1 인터페이스는 SL8500 또는 SL3000의 eth5 인터페이스와만 통신합니다.

UNIX "*route*" 명령을 사용하여 이 관계를 강제로 적용합니다.

- Solaris의 경우 *root* 사용자가 다음 명령을 입력합니다.

  ```
  route -p add 7.0.50 -ifp qfe0 192.168.0.254

  route -p add 192.168.1.50 -ifp qfe1 192.168.1.254
  ```

  첫번째 route 명령은 *192.168.0.50*과의 모든 통신이 ACSLS 서버의 qfe0을 통과한 다음 라우터 1을 통과하도록 경로 지정합니다.

  두번째 route 명령은 *192.168.1.50*과의 모든 통신이 ACSLS 서버의 qfe1을 통과한 다음 라우터 2를 통과하도록 경로 지정합니다.

  다음을 입력하여 경로가 경로 지정 테이블에 있는지 검증할 수 있습니다.

  ```
  # netstat –r
  ```

예 B.1. IPv4 경로 지정 테이블

```
Destination               Gateway       Flags  Ref  Use  Interface
_______________           ________      _____  ___  ___  _________
192.168.0.50              192.168.0.254  UGH    1    0    qfe0
192.168.1.50              192.168.1.254  UGH    1    0    qfe1
192.168.0.0               192.168.0.1     U     1    7    qfe0
192.168.1.0               192.168.1.1     U     1    0    qfe1
BASE-ADDRESS.MCAST.NET    192.168.0.1     U     1    0    qfe0
default                   192.168.0.254   UG    1   33
localhost                 localhost       UH    4   77    lo0
```

처음 두 항목은 방금 추가된 항목입니다. *192.168.0.50*과의 모든 통신은 QFE0을 통과하고 *192.168.1.50*과의 모든 통신은 QFE1을 통과합니다.

참고: *StorageTek SL8500 Modular Library System Dual TCP/IP Feature* 문서의 지침에 따라 라이브러리의 경로 지정 테이블을 구성합니다.

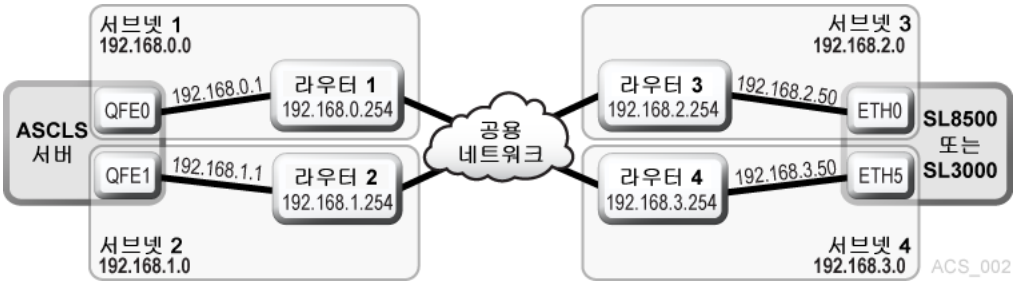
시나리오 2

시나리오 2에서는 다음을 보여줍니다.

- 라이브러리의 개별 서브넷에 두 개의 인터페이스가 있는 ACSLS 서버
- ACSLS의 개별 서브넷에 두 개의 네트워크 인터페이스가 있는 SL8500 또는 SL3000 라이브러리

- 공용 네트워크를 사용하는 ACSLS와 SL8500 또는 SL3000 모두

그림 B.2. 공용 네트워크를 사용하는 ACSLS와 SL8500 또는 SL3000



UNIX "route" 명령을 사용하여 이 관계를 강제로 적용합니다.

- Solaris의 경우 *root* 사용자가 다음 명령을 입력합니다.

  ```
  #route add 192.168.2.50 -ifp qfe0 192.168.0.254

  #route add 192.168.3.50 -ifp qfe1 192.168.1.254
  ```

  ACSLS에 대한 기본 경로는 동일하게 유지됩니다. 서브넷 내의 경로는 공용 LAN을 통한 라이브러리 통신의 경로 지정 방법을 알려주며, 인터페이스와의 일대일 관계를 계속해서 강제로 지정합니다. 다음 명령을 사용하면 이 내용이 다시 표시됩니다.

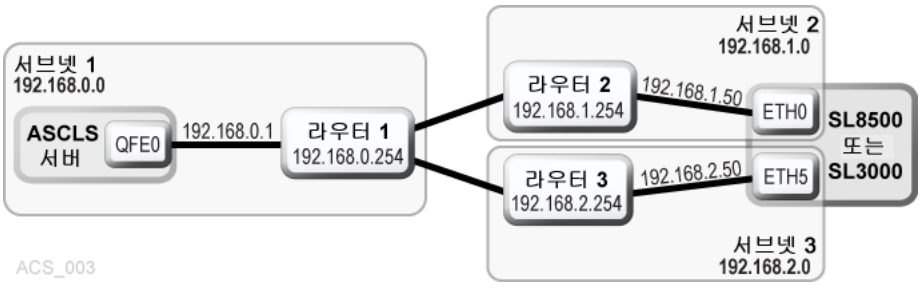
  ```
  # netstat –r
  ```

참고: *StorageTek SL8500 or SL3000 Modular Library System Dual TCP/IP Feature* 문서의 지침에 따라 라이브러리의 경로 지정 테이블을 구성합니다.

시나리오 3

이 시나리오에는 개별 서브넷에 한 개의 네트워크 인터페이스를 가진 ACSLS 서버 한 개가 있습니다. SL8500 또는 SL3000 라이브러리는 ACSLS 서버와 별개인 두 개의 서브넷에 두 개의 네트워크 인터페이스가 있습니다.

그림 B.3. 네트워크 인터페이스 두 개가 있는 SL8500 또는 SL3000



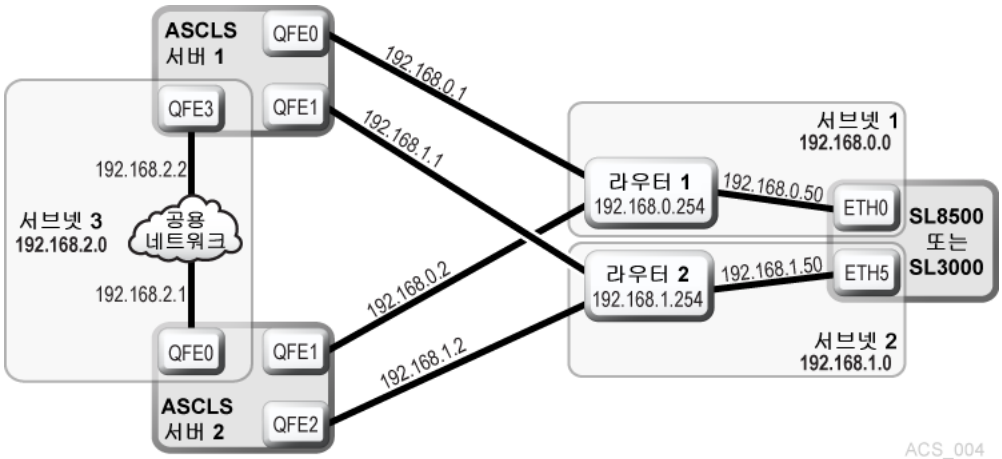
참고: "StorageTek SL8500 or SL3000 Modular Library System Dual TCP/IP Feature" 문서의 지침에 따라 라이브러리의 경로 지정 테이블을 구성합니다.

시나리오 4

시나리오 4에서는 다음을 보여줍니다.

- 네트워크 인터페이스 세 개, SL8500 또는 SL3000을 포함하는 개별 전용 서브넷 두 개, 세번째 공용 네트워크로 구성된 고가용성(ACSLS HA) 서버 두 개
- ACSLS 서버와 동일한 전용 서브넷 두 개에 네트워크 인터페이스 두 개가 있는 SL8500 또는 SL3000 라이브러리 하나

그림 B.4. ACSLS HA



이 시나리오에서는 ACSLS HA가 각각 서로 다른 네트워크 인터페이스를 사용하는 두 개의 다른 서버를 사용합니다. 즉, 두 ACSLS 서버에 사용자 정의 경로 항목을 추가해야 합니다.

Solaris 사용자

- ACSLS 서버 1에서 다음을 입력합니다.

  ```
  route add 192.168.0.50 –ifp qfe0 192.168.0.254

  route add 192.168.1.50 –ifp qfe1 192.168.1.254
  ```

- ACSLS 서버 2에서 다음을 입력합니다.

  ```
  route add 192.168.0.50 –ifp qfe1 192.168.0.254

  route add 192.168.1.50 –ifp qfe2 192.168.1.254
  ```

  두 서버의 IP 주소를 라이브러리의 구성에 추가해야 합니다. *StorageTek SL8500 or SL3000 Modular Library System Dual TCP/IP Feature* 문서를 참조하십시오.

  ACSLS HA에 있을 때 서로 다른 두 개의 서브넷을 통해 라이브러리의 네트워크 인터페이스를 분리해야 합니다. 고가용성 환경의 용도는 중복으로 작성하여 단일 실패 지점을 제거하는 것입니다.

참고: "StorageTek SL8500 or SL3000 Modular Library System Dual TCP/IP Feature" 문서의 지침에 따라 라이브러리의 경로 지정 테이블을 구성합니다.

재부트 후 사용자 정의 경로 지정 테이블 항목 유지

시스템이 재부트된 후 사용자 정의 경로 지정 테이블 항목이 손실됩니다. 이는 시스템 경로 지정 테이블의 특성으로, 예상된 동작입니다.

SL8500 또는 SL3000에서 이중 TCP/IP 기능을 지원하려면 ACSLS 서버의 경로 지정 테이블에 사용자 정의 항목을 추가해야 합니다. ACSLS 서버가 재부트되면 모든 경로 지정 테이블 항목이 비워지고 라이브러리에 대한 모든 필수 경로가 제거됩니다. 이는 운영체제의 특성이므로 이 상황을 처리할 수 있는 몇 가지 방법이 있습니다.

스크립트 만들기

부트 시 초기화할 사용자 정의 경로를 추가하기 위한 스크립트를 만들 수 있습니다. 절차는 "부트 시 초기화할 사용자 정의 경로 추가 "를 참조하십시오.

그런 다음 부트 시 자동으로 실행하도록 이러한 스크립트를 *rc* 디렉토리 구조에 저장할 수 있습니다. 가장 좋은 구현 방법에 대한 자세한 내용은 시스템 설명서를 참조하십시오.

ACSLS 시작 스크립트를 사용하여 부트 시 사용자 정의 경로 지정 항목을 추가합니다. 시작 스크립트는 사용자 정의 경로 지정 테이블 항목을 포함하는 파일을 확인합니다. 발견된 항목은 UNIX *route* 명령을 사용하여 경로 지정 테이블에 자동으로 추가됩니다. 독립형 ACSLS 설치의 경우 라이브러리 지원에 필요한 경로 지정 항목을 유지하려면 이 방법을 사용하는 것이 좋습니다.

중요: ACSLS 설치가 고가용성 ACSLS(ACSLS HA) 환경인 경우에는 이 솔루션이 **적용되지 않습니다**.

이 경우 첫번째 방법을 사용하여 경로 지정 테이블을 유지해야 합니다.

ACSLS HA는 Solaris Cluster에 의존하여 클러스터화된 리소스를 관리하므로 독립형 ACSLS 서버와는 다른 방법으로 시스템 초기화를 처리합니다. 즉, 부트 시 시스템 RC 메커니즘을 통해 ACSLS를 자동으로 시작할 수 없습니다. 이는 사용된 적이 없는 S87ACSLS 시작 스크립트를 사용하여 Solaris Cluster 에이전트에서 엄격하게 처리됩니다. 적절한 "route add" 명령을 사용하여 스크립트를 추가하고 */etc/rc2.d* 디렉토리 구조 내에서 찾습니다. ACSLS HA 환경을 사용하는 경우 오라클 고급 고객 지원팀(가급적 ACSLS HA 시스템을 원래 설치한 컨설턴트)과 상담하는 것이 좋습니다.

부트 시 초기화할 사용자 정의 경로 추가

사용자 지정 경로 지정 항목을 추가하려면 다음을 수행합니다.

1. cd를 수행하여 다음 디렉토리로 이동합니다.

   $ACS_HOME/data/external/ custom_routing.

이 디렉토리에는 *custom_routing_tables.tpl* 템플리트 파일이 포함되어 있습니다.

2. 이 파일을 복사하고 파일 이름을 *custom_routing_tables*로 변경합니다.

   ```
   # cp custom_routing_tables.tpl custom_routing_tables
   ```

3. *custom_routing_tables* 파일을 편집(vi)하고 항목을 추가합니다.

   파일에는 새 개의 필드가 포함되어 있습니다.
   - SL8500 또는 SL3000에 대한 IP 주소
   - 일대일 관계를 설정할 ACSLS 서버의 인터페이스 이름
   - 서브넷에 대한 기본 경로의 IP 주소

4. 형식은 *custom_routing_tables* 설명 섹션의 지침을 따릅니다.

   주:

   빈 라인이 없는지 확인하십시오.

   서버가 재부트되면 ACSLS가 자동으로 초기화되고 사용자 정의 경로가 경로 지정 테이블에 추가됩니다.

5. 다음 명령을 사용하여 경로 지정 테이블의 모든 경로를 확인합니다.

   ```
   # netstat -r
   ```

route 및 *netstat* 명령에 대한 전체 설명서는 UNIX 매뉴얼 페이지를 참조하십시오.

경로 지정 명령 제거

실수로 추가되거나 이전 구성에 더 이상 필요하지 않은 특별 경로 지정 명령을 제거하려면 *route* 명령을 사용합니다.

예: *root* 사용자가 다음 명령을 입력합니다.

```
# route delete 192.168.0.50 192.168.0.254
```

이 명령은 기본 경로 *192.168.0.254*를 사용하여 *192.168.0.50*(SL8500 또는 SL3000)에 대한 경로를 제거하도록 지시합니다. 그러면 경로가 제거됩니다.

다중 TCP/IP 지원

SL8500 3.97 이상의 펌웨어가 설치된 경우 ACSLS는 ACS(라이브러리 컴플렉스)에 있는 둘 이상의 SL8500에 연결할 수 있습니다.

ACSLS는 ACS에 대한 연결을 최대 15개까지 지원합니다. 예를 들어, 4개의 SL8500에 대한 15개 연결, 각 2개의 SL8500에 대한 2개 연결, 1개의 SL8500에 대한 2개 연결과 다른 2개의 SL8500에 대한 2개 연결, 2개 또는 3개의 라이브러리에 대한 3개 연결 등이 가능합니다.

ACSLS가 두 개 이상의 라이브러리에 연결된 경우 중복성을 위해 서로 다른 서브넷을 통해 연결해야 합니다. 한 서브넷이 실패하면 ACSLS와 라이브러리 간 통신이 다른 서브넷을 통해 계속됩니다.

ACSLS에 SL8500 HBC 카드 하나에 대한 두 개의 연결이 있는 경우 "이중 TCP/IP 지원"에 설명된 대로 SL8500 및 ACSLS 서버 경로 지정 테이블을 구성해야 합니다. ACSLS 서버와 각 SL8500 HBC 카드 간의 연결이 하나만 있는 경우 ACSLS 및 SL8500 경로 지정 테이블을 구성할 필요가 **없습니다**.

라이브러리 성능을 최적화하고 SL8500 사이의 라이브러리 간 통신을 최소화하려면 작동이 가장 많은 라이브러리에 대한 첫번째 연결(포트 0)을 정의합니다.

다중 TCP/IP 통신을 구성하여 관리하면 ACSLS 서버 또는 SL8500 라이브러리에서 경로 지정 테이블을 정의할 필요가 없으므로 이중 TCP/IP보다 더 간단합니다. 하지만 다중 TCP/IP를 사용하려면 일련의 연결된 SL8500 라이브러리가 필요합니다. 단일 독립형 SL8500 또는 SL3000 라이브러리에는 적용되지 않습니다.

자세한 내용은 *StorageTek SL8500 Modular Library System Technical Brief - Host to Library Communications*를 참조하십시오.

그림 B.5. "다중 TCP/IP를 사용하는 ACSLS "에서는 다중 TCP/IP 구성을 사용하는 ACSLS를 보여주고, 그림 B.6. "다중 TCP/IP 및 이중 TCP/IP를 사용하는 ACSLS "에서는 다중 TCP/IP 및 이중 TCP/IP 구성을 사용하는 ACSLS를 보여줍니다.

그림 B.5. 다중 TCP/IP를 사용하는 ACSLS

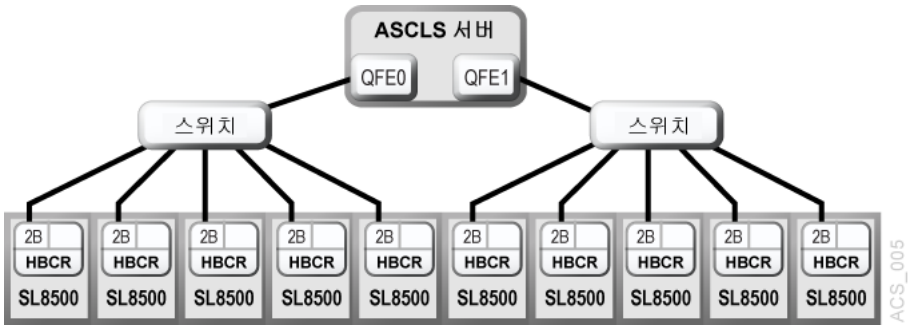
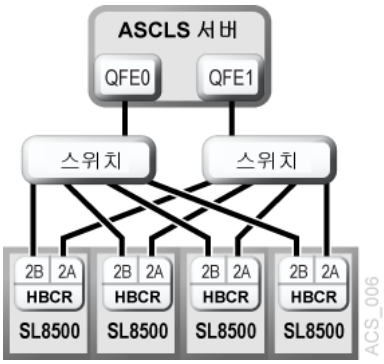


그림 B.6. 다중 TCP/IP 및 이중 TCP/IP를 사용하는 ACSLS



중복 전자 부품

선택적 SL8500 또는 SL3000 RE(중복 전자 부품) 기능은 엔터프라이즈 라이브러리에서 페일오버 보호 기능을 제공합니다. 라이브러리 컨트롤러에서 오류가 발생할 경우 대체 라이브러리 컨트롤러로 자동으로 전환되어 라이브러리 및 호스트 작업 중단을 최소화합니다. 이 기능은 라이브러리가 계속해서 정상 작동하는 동안 오라클 고객지원센터 담당자가 고장 난 카드를 교체할 수 있도록 지원합니다.

또한 RE는 펌웨어 업그레이드 중에 라이브러리 작업 중단을 최소화합니다.

주:

라이브러리는 로봇, 전원 시스템을 비롯한 다양한 구성 요소에서 중복 기능을 제공합니다. 특히 "중복 전자 부품"이라는 용어는 라이브러리 및 드라이브 컨트롤러 구성 요소의 중복을 나타냅니다.

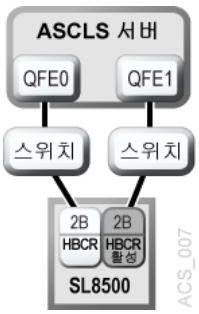
RE에는 다음 하드웨어 구성 요소가 모두 필요합니다.

- 활성 드라이브 컨트롤러(HBT)와 연결된 활성 라이브러리 컨트롤러(HBC 또는 HBCR)
- 대기 HBT와 연결된 대기 HBC 또는 HBCR
- 기타 중복 구성 요소

자세한 내용은 *StorageTek SL8500 or SL3000 User's Guide*를 참조하십시오.

그림 B.7. "RE를 사용하는 ACSLS "에서는 단일 라이브러리에서 RE를 사용하는 ACSLS를 보여줍니다.

그림 B.7. RE를 사용하는 ACSLS

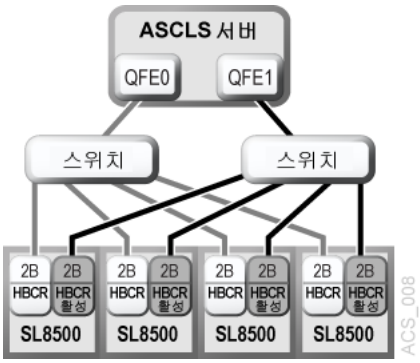


RE에 대한 ACSLS 지원

ACSLS는 단일 라이브러리 컴플렉스(전달 포트를 통해 연결된 라이브러리의 ACS) 내에서 활성 및 대기 SL8500 LC(라이브러리 컨트롤러) 카드를 함께 처리합니다.

그림 B.8. "RE 및 다중 TCP/IP를 사용하는 ACSLS "과 같이 각 SL8500의 HBCR 카드 중 하나가 활성 컨트롤러 카드일 수 있습니다.

그림 B.8. RE 및 다중 TCP/IP를 사용하는 ACSLS



일련의 연결된 SL8500에 있는 각 라이브러리는 이제 중복 라이브러리 컨트롤러의 자체 쌍을 포함할 수 있습니다. 라이브러리 컴플렉스에서 일부 라이브러리는 RE를 사용으로 설정하여 라이브러리 컨트롤러 카드 쌍을 사용하고, 일부 라이브러리는 단일 라이브러리 컨트롤러만 사용할 수 있습니다. ACSLS에서는 모든 활성 LC와 동시에 통신할 수 있어야 합니다.

ACSLS는 이중 TCP/IP를 사용하는 RE(그림 B.9. “RE 및 이중 TCP/IP를 사용하는 ACSLS ” 참조) 또는 이중 및 다중 TCP/IP를 사용하는 RE(그림 B.10. “이중 TCP/IP 및 다중 TCP/IP를 사용하는 RE ” 참조)를 지원합니다.

그림 B.9. RE 및 이중 TCP/IP를 사용하는 ACSLS

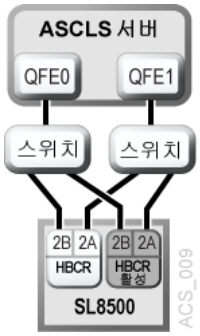
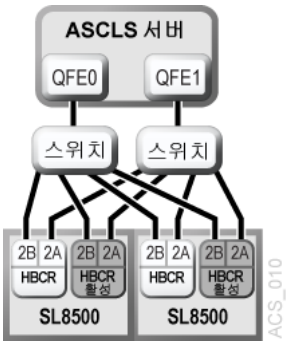


그림 B.10. 이중 TCP/IP 및 다중 TCP/IP를 사용하는 RE



마운트 및 마운트 해제 질의 및 재시도

RE를 지원하기 위해 ACSLS에서는 임시 라이브러리 및 드라이브 작동 중단 중에 마운트 및 마운트 해제에 대한 질의 및 재시도를 구현했습니다. 자세한 내용은 “라이브러리가 임시로 사용 불가능한 경우 마운트와 마운트 해제 큐에 넣기 및 재시도 ”를 참조하십시오.

단일 라이브러리 전용 switch lmu

`switch lmu` 명령을 사용하여 SL3000 또는 단일 SL8500 라이브러리에서 라이브러리 컨트롤러 간을 강제로 전환할 수 있습니다. 라이브러리 컴플렉스에 있는 다른 SL8500에 연결된 SL8500 하나를 전환하려는 경우에는 *`switch lmu`* 명령을 사용할 수 없습니다.

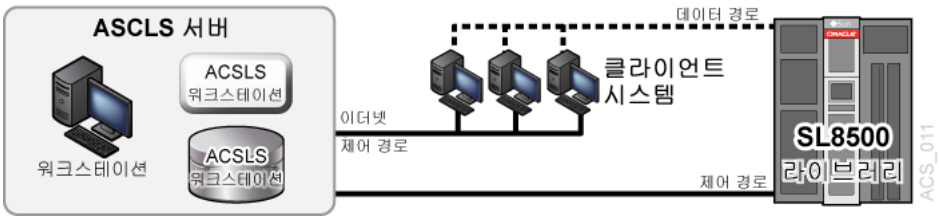
부록 C

부록 C. SL8500의 ACSLS 지원

이 부록에서는 다음 항목에 대해 설명합니다.

다음 그림은 ACSLS 서버가 포함된 SL8500 라이브러리를 보여줍니다.

그림 C.1. SL8500에 대해 ACSLS를 구성하기 전의 ACSLS 서버가 포함된 SL8500



SL8500 라이브러리는 이더넷 물리적 인터페이스를 통해 TCP/IP 프로토콜을 사용해서 호스트 및 ACSLS를 관리하고 이와 통신합니다. 이 인터페이스는 ACSLS가 SL8500에 연결하고 이와 통신할 수 있게 해줍니다. SL8500에 대해 ACSLS를 구성하기 전:

- ACSLS에 하나 이상의 SL8500을 연결합니다.
- SL8500의 모든 구성 요소가 작동하는지 확인합니다.

  ACSLS는 라이브러리에서 보고된 정보로부터 라이브러리 구성을 작성합니다. SL8500 구성 요소가 작동하지 않으면 라이브러리 정보가 ACSLS에 보고되지 않을 수 있으며 SL8500의 ACSLS 구성이 완전하지 않게 됩니다.

주:

드라이브 또는 CAP와 같은 구성 요소가 작동하지 않을 경우에는 동적 구성(*config acs, config lsm* 또는 *config drives*)을 사용해서 ACSLS가 작동 중이고 라이브러리가 온라인인 상태에서 이를 쉽게 추가 또는 업데이트할 수 있습니다.

다중 TCP/IP를 사용해서 여러 SL8500에 연결

SL8500 3.97 이상의 펌웨어가 설치된 경우 ACSLS는 ACS(라이브러리 컴플렉스)에 있는 둘 이상의 SL8500에 연결할 수 있습니다.

ACSLS는 ACS에 대한 연결을 최대 15개까지 지원합니다. 예를 들어, 4개의 SL8500에 대한 15개 연결, 각 2개의 SL8500에 대한 2개 연결, 1개의 SL8500에 대한 2개 연결과 다른 2개의 SL8500에 대한 2개 연결, 2개 또는 3개의 라이브러리에 대한 3개 연결 등이 가능합니다.

ACSLS가 두 개 이상의 라이브러리에 연결된 경우 중복성을 위해 서로 다른 서브넷을 통해 연결해야 합니다. 한 서브넷이 실패하면 ACSLS와 라이브러리 간 통신이 다른 서브넷을 통해 계속됩니다.

ACSLS에 하나의 SL8500 HBC 카드에 대해 2개의 연결이 포함된 경우, “개요 ”에 설명된 대로 SL8500 및 ACSLS 서버 경로 지정 테이블을 구성합니다. ACSLS 서버와 각 SL8500 라이브러리 사이에 단일 연결만 있는 경우에는 ACSLS 및 SL8500 경로 지정 테이블 구성이 필수가 **아닙니다**.

라이브러리 성능을 최적화하고 SL8500 사이의 라이브러리 내 통신을 최소화하려면 활동이 가장 많은 라이브러리에 연결하십시오. *acsss_config* 또는 *config acs*에서 지정한 첫번째 연결을 가장 활동이 많은 SL8500에 대해 새 연결로 설정합니다.

자세한 내용은 *SL8500 Modular Library System Technical Brief - Host to Library Communications*를 참조하십시오.

모든 SL8500 구성 요소가 작동하는지 확인

SL8500의 모든 구성 요소가 작동하는지 확인하려면 다음을 수행합니다.

1. StorageTek 라이브러리 콘솔(SL 콘솔)에 로그온합니다.

   SL8500의 콘솔 또는 원격 라이브러리 콘솔을 사용할 수 있습니다.
2. *Tools -> System Detail*을 선택합니다.
   - 모든 SL8500 구성 요소는 녹색이어야 합니다.

     예외사항: 노란색 드라이브는 동적 구성을 사용해서 지금 또는 나중에 구성할 수 있습니다(“bdb.acsss ”).
   - 누락된 구성 요소는 동적 구성(*config acs* 또는 *config lsm*) 유틸리티를 사용해서 추가할 수 있습니다.
   - **중요:** SL8500을 구성하기 전에 엘리베이터(Elevator 폴더)가 **녹색이어야 합니다**. 엘리베이터가 녹색이 아니면 ACSLS에 대해 SL8500을 구성하지 **마십시오**. 엘리베이터

는 논리적 PTP(전달 포트)입니다. PTP가 없으면 ACSLS에서 SL8500 레일이 연결되었는지 확인할 수 없습니다.

3. SL8500 구성 요소가 작동하면 “CSI 조정 변수 설정” 또는 “acsss 매크로 ”에 설명된 대로 ACSLS에 대해 SL8500을 구성합니다.

SL8500 내부 주소 및 ACSLS 주소 이해

SL8500의 내부 주소와 ACSLS 및 HSC에서 지원되는 다른 라이브러리 사이에는 차이가 있습니다.

- SL8500은 *1*부터 시작되며 음수가 사용됩니다.
- 다른 라이브러리에서는 *0*부터 시작되며 음수가 사용되지 않습니다.
- SL8500에서는 라이브러리, 레일, 열, 측면 및 행의 *5*개 매개변수가 사용됩니다.
- Legacy StorageTek 라이브러리(예: 9310)에서는 ACS, LSM, 패널, 행 및 열이 사용됩니다(HLI–PRC).

표 C.1. 주소 지정 설명

HLI-PRC	SL8500	설명
ACS	라이브러리	라이브러리 컴플렉스에 있는 특정 SL8500 라이브러리의 번호입니다. ACS는 SL8500 라이브러리 컴플렉스입니다. 라이브러리 컴플렉스에는 SL8500이 여러 개 있을 수 있습니다.
LSM LSM 0 LSM 1 LSM 2 LSM 3	레일 레일 1 레일 2 레일 3 레일 4	SL8500 라이브러리에는 HandBot이 이동하는 4개의 레일이 있습니다. 이러한 레일은 위에서 아래의 순서로 1–4(1부터 시작)로 번호가 지정됩니다. ACSLS는 각 레일을 개별 LSM으로 고려합니다. 이러한 LSM은 위에서 아래의 순서로 0–3(0부터 시작)으로 번호가 지정됩니다.
패널 패널 0 패널 1 패널 2–n	열 CAP 드라이브 스토리지 슬롯	열은 라이브러리에서 가로 위치를 나타냅니다. 라이브러리 전면에서 봤을 때 열 및 패널 번호는 드라이브 패널(1)의 가운데에서 시작하여 순차적으로 번호가 증가합니다. (SL8500에서는 패널이 주소로 사용되지 않습니다.) 라이브러리 너비에 걸쳐 있는 HLI 패널은 각 LSM에 대해 양쪽(왼쪽 및 오른쪽)과 양벽(내벽 및 외벽)을 포함합니다.
	측면	벽면 위치: 외벽 내벽 HandBot 번호: 왼쪽(–) 오른쪽(+)
행	행	행은 테이프 카트리지의 세로 위치를 나타내며 위에서 아래의 순서로 번호가 지정됩니다.
열		HLI 주소에 대한 행은 다음과 같습니다. 스토리지 패널은 열 0 = 왼쪽

	및 열 1 = 오른쪽이 있는 2에서 시작합니다. 행 0–12 외벽 행 13–26 내벽 일반 스토리지 패널의 각 열에는 27개 행이 포함됩니다. 패널 당 총 용량은 54개 카트리지입니다. SL8500 주소에 대한 행은 다음과 같습니다. 스토리지 슬롯은 열 -3 = 왼쪽 열 +3 = 오른쪽에서 시작합니다. 행 1–13 외벽 행 1–14 내벽

- 0 기준 번호 지정(HLI)에서는 0부터 번호 지정이 시작됩니다.
- 1 기준 번호 지정(SL8500)에서는 1부터 번호 지정이 시작됩니다.
- 이러한 차이는 소프트웨어(ACSLS 또는 HSC) 및 하드웨어(물리적 SL8500 주소) 사이의 번호 지정 시퀀스에서 중요한 차이점입니다.

SL 콘솔을 사용하여 주소 변환

SL 콘솔 검색 유틸리티를 사용하면 SL8500 내부 주소와 ACSLS 또는 HSC 패널, 행 및 열 사이의 변환을 수행할 수 있습니다. 카트리지를 찾으려면 다음을 수행합니다.

1. SL 콘솔에 로그인합니다.
2. Tools > *Diagnostics* > *Search*를 선택합니다.
3. Location을 선택합니다.
4. *Location* 필드에서 다음 연산 중 하나를 선택합니다.

contains	예: 1,1,-9는 양측에 있는 모든 행에 대해 라이브러리 1, 레일 1, 열 -9에 있는 컨텐츠를 나열합니다.
endsWith	예: 1,5는 모든 레일에 대한 슬롯 컨텐츠 및 측면 1, 행 5에 대한 열을 나열합니다.
equals	예: 1,1,-9,1,1은 특정 위치에 있는 컨텐츠를 나열합니다(L,R,C,S,W).
startsWith	예: 1, 3은 라이브러리 1, 레일 3에 있는 모든 열, 측면 및 행에 대한 슬롯 컨텐츠를 나열합니다.

5. *Requestor* 풀다운 메뉴에서 다음 중 하나를 선택합니다.
   - *default*

     라이브러리 내부의 물리적 위치입니다(셀, 드라이브, CAP).

     물리적 위치(내부 주소)를 알고 있고, HLI-PRC 주소를 찾아야 하는 경우 해당 주소를 *location*에 입력(*enter*)하고 *default*를 요청자로 선택합니다.
   - *hli#*

그러면 라이브러리 관리 소프트웨어에서 카트리지의 HLI-PRC 주소가 선택됩니다. 여기서 #은 다음 중 하나입니다.

- 분할되지 않은 라이브러리의 경우 hli0.
- 분할된 라이브러리의 경우 hli1-8. 이 번호는 분할 영역 번호입니다.

이 옵션은 내부 주소와 *hli# Requester*를 모두 표시합니다.

6. SL 콘솔의 오른쪽 위 모서리에 있는 *Search* 버튼을 누릅니다.

   검색 결과에는 슬롯 유형별 위치가 나열됩니다(셀, 드라이브 또는 CAP).

7. *Details (...)* 필드를 누릅니다.

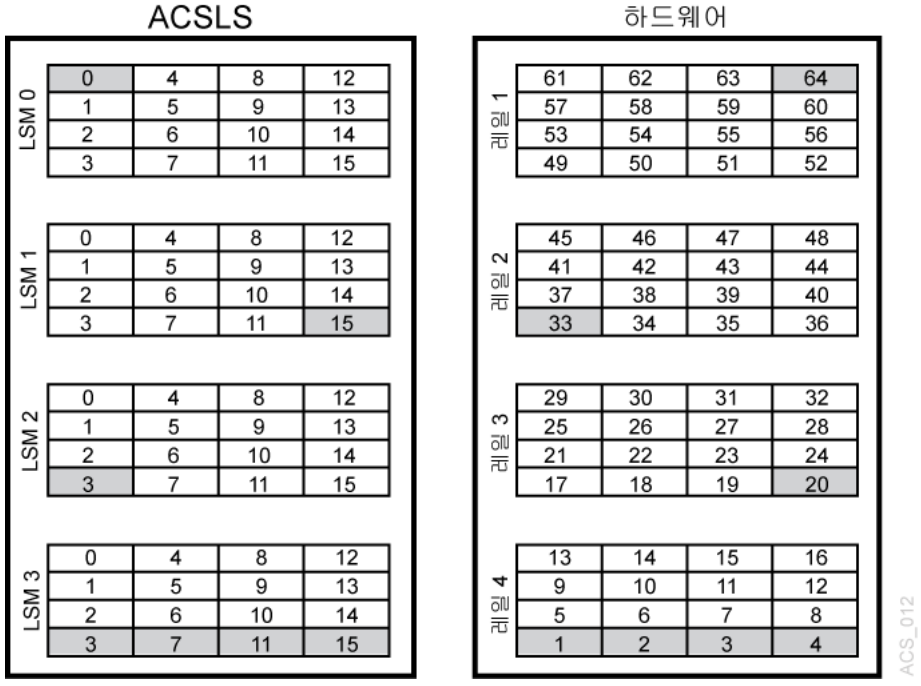
   팝업 창은 VOLID, 매체 및 카트리지에 대한 카트리지 유형(LTO, SDLT 테이프 및 T-Series, 데이터, 청소 또는 진단)과 같은 자세한 정보를 제공하고 위치에 대한 내부 및 HLI 주소를 모두 보여줍니다.

테이프 드라이브 위치

테이프 드라이브는 *LSM*과 연관되어 있으며 LSM에 포함됩니다. 다른 LSM에서 카트리지 테이프를 *mount*하기 위해서는 카트리지가 내부 전달 작업(이 경우에는 엘리베이터)을 통해 해당 드라이브로 이동해야 합니다.

다음 표에서는 **내부**(소프트웨어) 매핑(테이프 드라이브의 라이브러리 내에서 바라볼 경우) 및 **외부**(드라이브의 물리적 번호 지정) 매핑(드라이브 및 전자 부품 모듈의 후면 밖에서 바라볼 경우)을 보여줍니다.

그림 C.2. 소프트웨어 및 물리적 드라이브 번호 지정



강조 표시된 드라이브는 일치하는 드라이브를 보여줍니다. **예:**

- 내부/소프트웨어 LSM 0 드라이브 0은 외부/물리적 드라이브 64와 일치합니다.
- 내부 LSM 1 드라이브 15는 외부/물리적 드라이브 33과 일치합니다.
- 내부 LSM 2 드라이브 3은 외부 물리적 드라이브 20과 일치합니다.

분할 영역에서 셀을 제거하기 전에 카트리지 이동

SL8500은 고급 분할을 통해 드라이브 및 셀 배열 레벨로 분할할 수 있습니다. 자세한 내용은 분할 영역에서 셀을 제거하기 전에 카트리지 이동 을 참조하십시오.

SL8500 CAP

ACSLS 8.4부터는 ACSLS에서 두 가지 SL8500 CAP 유형이 지원됩니다. 레거시 회전식 유형은 각 SL8500 라이브러리에 한 개 또는 두 개의 39셀 CAP를 설치할 수 있습니다. 새로운 CAP 유형인 대량 CAP에서는 각 라이브러리에 8개의 36셀 CAP가 설치됩니다.

대량 CAP

새로운 SL8500 라이브러리는 대규모의 빈번한 저장 활동이 일반적인 데이터 센터에서 넣기 및 꺼내기 작업을 보다 빠르고 효율적으로 수행할 수 있도록 설계되었습니다. 각 SL8500 라이브러리에는 8개의 대량 CAP가 있고 각 레일의 각 측면에 하나의 CAP가 포함됩니다. 각 CAP에는 3개의 이동식 12슬롯 매거진이 포함됩니다.

ACSLS는 대량 CAP를 사용해서 카트리지 넣기 및 꺼내기를 효율적으로 수행합니다. CAP에서 넣은 볼륨은 CAP와 동일한 측면 및 동일한 레일에 있는 슬롯으로 이동합니다. 해당 측면이 가득 차면 다른 측면의 빈 슬롯이 선택됩니다. 레일이 가득 차면 인접한 레일이 선택됩니다. 이러한 전략은 로봇 이동을 최소화하고 로봇 간 충돌을 방지합니다. 비슷한 방식으로, *ejecting.sh*가 볼륨 목록을 꺼낼 경우, 각 볼륨은 사용자가 지정한 CAP 사이에 가장 인접한 CAP로 꺼내집니다. “ejecting.sh ”를 참조하십시오.

꺼내기 중 핸들로 작동할 수 있도록 각 매거진에서 열린 슬롯을 예약하려면 *dv_config*를 사용해서 동적 변수 *BULK_CAP_EJECT_HANDLE*을 **TRUE**로 설정합니다.

```
$ dv_config -p BULK_CAP_EJECT_HANDLE  -u

When ejecting cartridges to an SL8500 Bulk CAP, leave a slot in each CAP magazine
 empty so it can be used as a handle. (TRUE/FALSE) [FALSE]: TRUE

Updating configuration file.
```

이 변수가 *TRUE*로 설정되었으면 각 매거진의 11개 스토리지 셀이 꺼내기 작업에 사용됩니다. 위쪽 3개 레일에 있는 각 매거진의 아래쪽 셀은 빈 상태로 유지되고, 아래쪽 레일의 각 매거진에 있는 위쪽 셀은 비어 있습니다. 그 결과 빈 슬롯을 핸들로 사용할 수 있습니다. 이 설정은 넣기 작업 중 동작에 영향을 주지 않습니다.

모듈당 8개 CAP가 있어서 10 문자열 SL8500 구성에 8개 대량 CAP가 있을 수 있습니다. 대형 SL8500 컴플렉스에서는 여러 넣기 및 꺼내기가 진행 중일 때 CAP 작업으로 인해 마운트 및 마운트 해제 작업이 지연될 수 있습니다. 이 문제를 완화하기 위해 동적 변수 LIMIT_CAP_CONCURRENT_MOVES는 동시 넣기 및 꺼내기 로봇 이동 수를 제한하여 마운트 및 마운트 해제가 진행되도록 할 수 있습니다. 이 기능을 사용하기 위해서는 *dv_config*를 사용해서 동적 변수 *LIMIT_CAP_CONCURRENT_MOVES*를 **TRUE**로 설정합니다.

```
$ dv_config -p LIMIT_CAP_CONCURRENT_MOVES  -u

When using large numbers of CAPs for ejects and/or enters in an ACS with multiple
 libraries, limit the number of concurrent moves to/from CAPs to reserve library
 resources for mounts and dismounts. (TRUE/FALSE). [FALSE]: TRUE

Updating configuration file
```

ACSLS를 사용해서 대량 CAP를 처리하도록 SL8500 업그레이드

하나 이상의 SL8500에 대량 CAP를 설치할 때 ACSLS를 업데이트하려면 다음을 수행합니다.

1. ACSLS 8.4를 설치합니다.

   이 작업은 대량 CAP 설치 전에 미리 수행할 수 있습니다.
2. Oracle FSE(현장 서비스 엔지니어)는 해당 SL8500에서 대량 CAP를 지원하는 SL8500 펌웨어를 로드하고 활성화해야 합니다.

   최소 SL8500 펌웨어 레벨은 8.50입니다.
3. 대량 CAP 하드웨어를 설치하기 전 ACSLS *cmd_proc*를 사용해서 대량 CAP가 설치된 위치에서 라이브러리를 오프라인으로 전환합니다.
   - 독립형 SL8500에 대량 CAP를 설치하거나 하나의 문자열로 모든 SL8500에서 대량 CAP를 설치하는 경우에는 전체 ACS(라이브러리 컴플렉스)를 오프라인으로 전환합니다.
   - 라이브러리 컴플렉스에서 일부 SL8500에서만 대량 CAP를 설치할 경우에는 관련 LSM만 오프라인으로 전환하면 됩니다.
4. 이 단계 중에는 FSE가 해당 라이브러리에서 대량 CAP 하드웨어를 설치합니다.
   a. FSE가 대량 CAP 하드웨어를 설치하기 전에 사용자는 서비스 도어에 가장 가까운 3개 셀 배열 열에서 카트리지를 분리하고 라이브러리 외부에 보관해야 합니다. (설치가 완료된 후 카트리지를 다시 넣습니다.)

      이유: 대량 CAP를 설치하기 위해서는 3팩 배열 외에도 시스템 셀 배열의 2개 열을 제거해야 합니다. 세번째 열에 있는 대부분의 스토리지 셀은 더 이상 ACSLS에서 액세스할 수 없는 시스템 셀이 됩니다.
   b. FSE가 라이브러리에서 대량 CAP 하드웨어를 설치합니다.
5. 모든 SL8500이 재부트되고 라이브러리 하드웨어 감사가 완료된 다음에는 ACSLS *cmd_proc*를 사용해서 SL8500 ACS를 진단 모드로 전환합니다.

   진단 모드에서는 ACSLS 구성을 업데이트하고 라이브러리를 감사하는 동안 ACSLS 클라이언트가 이러한 라이브러리에 액세스할 수 없습니다.

6. ACSLS가 실행 중인 상태에서 *config acs acs_id* 유틸리티를 사용해서 데이터베이스에 기록된 ACSLS 구성에 대량 CAP를 추가합니다.

   주:

   또한 ACSLS를 사용 안함으로 설정하고 *acsss_config*, 옵션 8을 실행해서 구성을 업데이트할 수 있습니다. 이렇게 할 경우에는 *cmd_proc*에서 *query lmu all*을 실행하고 ACSLS를 종료하기 전에 출력을 저장합니다. 그런 다음 동일한 ACS 번호 및 동일한 포트 연결을 사용해서 ACS를 *acsss_config*로 지정합니다. *acsss_config*가 수행된 다음 ACSLS를 사용으로 설정합니다.

7. 다음과 같이 ACSLS *cmd_proc*를 사용해서 CAP 상태 및 유형을 표시하고 확인합니다.

   ```
   display cap * -f  state mode status size type
   ```

   샘플 출력:

   ```
   0  0  0     online    automatic   available     36    SL8500-Bulk
   0  0  1     offline   manual      available     36    SL8500-Bulk
   0  1  0     online    automatic   available     36    SL8500-Bulk
   0  1  1     offline   manual      available     36    SL8500-Bulk
   0  2  0     online    automatic   available     36    SL8500-Bulk
   0  2  1     offline   manual      available     36    SL8500-Bulk
   0  3  0     online    automatic   available     36    SL8500-Bulk
   0  3  1     offline   manual      available     36    SL8500-Bulk
   ```

8. *cmd_proc*를 사용해서 대량 CAP가 설치된 라이브러리를 감사합니다. 다음 중 하나를 수행할 수 있습니다.
   - 전체 ACSLS 감사:

     ```
     audit <cap_id> acs <acs_id>
     ```

   - 대량 CAP가 설치된 LSM만 감사:

     ```
     audit <cap_id> lsm  <lsm_id> <lsm_id> <lsm_id> <lsm_id> ...
     ```

9. *vary* 명령을 사용해서 ACSLS에 대해 ACS 및 LSM을 온라인으로 전환합니다.

   ACSLS 클라이언트가 이제 대량 CAP를 사용할 수 있습니다.

10. CAP를 사용해서 위 4단계에서 라이브러리에서 제거한 카트리지를 모두 다시 넣습니다.

넣기 및 꺼내기 목적을 보여주는 사용자 정의 SL 콘솔 메시지

SL 콘솔은 CAP 상태 화면에서 대량 CAP에 대한 넣기 및 꺼내기 목적을 보여주는 사용자 정의 작업자 메시지를 표시할 수 있습니다. 이러한 메시지는 또한 카트리지를 넣는 중인 분할 영역 또는 카트리지를 꺼낸 분할 영역을 보고할 수도 있습니다.

이러한 작업자 메시지는 선택사항입니다. 메시지는 기본 넣기 및 꺼내기 처리에 영향을 주지 않으며, SL8500 대량 CAP에 대해서만 지원됩니다.

사용자 정의 작업자 메시지를 사용하려면 다음을 수행합니다.

1. SL 콘솔을 사용해서 *opmsg* 번호에 대해 표시할 메시지를 정의합니다. 다음 옵션을 선택합니다.

   ```
   Tools
      Configuration
         CAP Usage Message
   ```

   4-99까지의 메시지 번호 및 연관된 메시지를 정의합니다. 가능한 경우 사용 가능한 공간에 들어갈 수 있도록 메시지를 20자로 제한합니다.
2. 수동 넣기 또는 꺼내기에 대해 선택적인 작업자 메시지 번호를 입력합니다.

   enter <cap_id> [opmsg <opmsg_nbr>]

   eject <cap_id> [opmsg <opmsg_nbr>] vol_id | vol_range ...

   사용자 정의 *opmsg* 번호는 ACSAPI 클라이언트, ACSLS GUI 또는 *lib_cmd eject*에서의 꺼내기에 대해 지정할 수 없습니다. 이 경우 기본 메시지가 표시됩니다.

   opmsg 메시지는 넣거나 삽입할 카트리지 또는 제거하기 위해 꺼내는 중인 카트리지에 대해 CAP가 잠금 해제된 다음 System Details, CAP Status 페이지에 표시됩니다.

   예: 대량 CAP 1,2,1을 통해 카트리지를 넣을 때 사용자 정의 작업자 패널 메시지 번호 55를 지정하려면 다음을 수행합니다.

   enter 1,2,1 opmsg 55

회전식 CAP

SL8500 회전식 CAP는 LSM 1, 2, 3에 해당하는 3개 레일(2,3,4)에 걸쳐 있습니다. 기본 구성에는 SL8500 모듈당 하나의 CAP와 옵션으로 설치할 수 있는 보조 CAP가 포함됩니다.

각 회전식 CAP에는 각각 13개 셀이 포함된 매거진이 3개 있습니다. 이러한 매거진은 해당 레일의 HandBot에서만 액세스할 수 있는 서로 다른 레일에 각각 배치됩니다. 넣기 중 ACSLS는 각 매거진에서 인접한 LSM(레일)으로 카트리지를 이동하려고 시도합니다. 인접한 레일이 가득 찬 경우에만 볼륨이 다른 레일로 이동합니다. 비슷한 방식으로, 지정된 레일의 볼륨은 해당 레일에서 인접한 매거진으로 꺼내기가 수행됩니다.

위쪽 레일(LSM 0)에는 인접한 CAP 매거진이 없기 때문에 엘리베이터가 자동으로 이동해서 위쪽 레일에서 꺼낸 볼륨을 사용합니다. 넣기 시, 위쪽 레일은 아래쪽 레일이 해당 용량으로 채워질 때까지 채워지지 않습니다. 위쪽 레일의 드라이브에 마운트된 볼륨은 마운트 해제할 때 결국 위쪽 LSM으로 마이그레이션됩니다. 그렇지 않으면 위쪽 레일의 LSM에 볼륨을 배치하기 위해 명시적인 이동 작업이 필요합니다. 이러한 추가 이동은 *watch_vols* 유틸리티를 사용해서 넣기 후 자동으로 처리할 수 있습니다. “watch_vols ”를 참조하십시오.

단일 회전식 CAP는 여러 LSM을 처리하기 때문에 회전식 CAP 상태는 LSM의 온라인 또는 오프라인 상태와 연결되지 않습니다. CAP는 1개 또는 모든 인접한 LSM이 오프라인 상태인

지 여부에 관계없이 온라인으로 유지될 수 있습니다. 반대로, CAP가 오프라인이면 LSM이 온라인으로 전환될 때 자동으로 온라인으로 전환되지 않습니다.

여러 LSM이 CAP에 액세스하더라도 SL8500 회전식 CAP는 LSM 1(예: 1,5,9,13)에 있는 것처럼 처리됩니다. 각 분할 영역이 서로 다른 호스트에 지정되는 분할된 라이브러리에서 사용자는 CAP가 공유 리소스라는 것을 인식해야 합니다. 회전식 CAP는 넣기 또는 꺼내기 작업 시 즉시 예약됩니다. 공유 환경의 작업자는 CAP를 즉시 채우거나 비우고 CAP 작업 완료 시 도어를 닫아야 합니다. [461]부록 I. 라이브러리 분할 을 참조하십시오.

SL8500의 이전 릴리스에서는 선택적인 CAP가 제공된 것으로 보고했지만, 두번째 CAP가 실제로 설치되지 않은 경우 작동하지 않는 것으로 보고했습니다. 임시해결책으로 ACSLS 사용자는 존재하지 않는 CAP의 상태를 오프라인으로 유지해야 했습니다. 6.07 이상의 라이브러리 펌웨어 레벨에서는 더 이상 이 문제가 발생하지 않습니다.

넣기 또는 꺼내기 작업

enter 중 ACSLS는 항상 CAP 매거진에 인접한 LSM(레일)으로 카트리지를 이동하려고 시도합니다. 꺼내기의 경우 ACSLS는 항상 카트리지가 포함된 LSM에 인접한 CAP 셀로 카트리지를 꺼내려고 시도합니다.

이러한 두 작업을 수행할 수 없는 경우, 라이브러리 컨트롤러는 엘리베이터를 통해 카트리지를 다른 LSM으로 이동합니다. 이를 위해서는 2개의 HandBot과 엘리베이터 사이의 이동이 필요합니다.

일부 ACSLS 클라이언트에 대한 넣기, 꺼내기 및 감사 작업

다른 라이브러리와 달리 SL8500은 SL8500 라이브러리의 각 LSM ID에 대해 정의된 CAP가 없습니다. SL8500의 CAP에는 해당 CAP ID의 LSM 1이 포함됩니다. SL8500에는 LSM ID 0, 2 또는 3의 CAP가 없습니다. 분할 시에는 LSM 1(SL8500 CAP ID의 LSM ID)이 분할 영역에 지정되지 않을 수 있기 때문에 이 문제가 더 복잡해집니다. (CAP는 모든 분할 영역에 대해 공유 리소스로 계속 제공됩니다.)

일부 ACSLS 클라이언트는 *enter*, *eject* 또는 *audit*를 위해 CAP를 선택하기 전에 존재하고 사용 가능한 CAP를 식별하기 위해 ACSLS에 대해 *query*를 수행하지 않습니다. 이러한 클라이언트는 존재하지 않는 `cap_ids` 또는 온라인이 아닌 CAP를 지정할 수 있습니다. 예를 들어, 일부 ACSAPI 클라이언트는 모든 LSM ID에 대해 CAP가 존재한다고 가정합니다. 이러한 클라이언트는 이들이 관리하는 카트리지 또는 드라이브 위치와 동일한 LSM ID를 사용해서 CAP를 자동으로 지정할 수 있습니다. 존재하지 않는 CAP ID를 지정하는 넣기, 꺼내기 또는 감사는 실패합니다.

ACSLS *cmd_proc*를 사용해서 다음을 수행해야 합니다.

- 존재하지 않는 CAP ID를 지정하는 클라이언트에 대해 카트리지를 넣고(*enter*) 꺼냅니다(*eject*).
- 이러한 클라이언트에서 사용되는 ACS 및 분할 영역에 대해 감사를 실행합니다.

넣기, 꺼내기 또는 감사를 수행한 후에는 클라이언트 응용 프로그램의 데이터베이스를 ACSLS 데이터베이스와 다시 동기화해야 합니다.

엘리베이터 및 PTP 작업 최소화

엘리베이터 및 PTP 작업을 최소화하기 위해서는 사용할 수 있는 방법은 다음과 같습니다.

카트리지 마운트	테이프를 마운트할 때는 가능한 한 동일한 LSM에 있는 카트리지 및 테이프 드라이브를 사용합니다. LSM은 SL8500 라이브러리 내에 있는 **단일 레일**을 의미합니다. 각 SL8500에는 4개의 LSM이 포함됩니다.
Float 사용	각 LSM 내에서 일부 비어 있는 셀을 유지 관리하여 ACSLS "*float*" 옵션(ACSLS에서 기본적으로 사용을 설정됨)을 활용합니다. 카트리지 Float는 동일 LSM에서 또는 테이프가 원래 전달 작업을 사용해서 다른 LSM으로부터 비롯된 경우 테이프 드라이브에 가까운 LSM에서 ACSLS가 마운트 해제된 테이프 카트리지를 빈 슬롯에 배치할 수 있게 해주는 기능입니다. 카트리지가 마운트 해제되면 카트리지의 이전 홈 셀이 다른 LSM에 있을 때마다 ACSLS가 새 홈 셀을 지정하여 LSM 간 엘리베이터(전달) 작업을 방지하려고 시도합니다. ACSLS는 카트리지를 다음과 같은 위치에 배치하려고 시도합니다. • 이전에 마운트 해제된 테이프 드라이브와 동일한 LSM • 또는 드라이브에 가장 가까운 LSM(빈 스토리지 셀이 있는)
카트리지 넣기	넣는 중인 매체에 대해 호환되는 테이프 드라이브가 포함된 LSM으로 카트리지 *Enter*를 수행합니다. 예: LSM 2 및 3에 LTO 드라이브만 있고, LTO 카트리지를 이러한 LSM에 배치하려고 합니다. 이러한 카트리지를 넣을 때는 이를 LSM 2 및 3과 인접한 CAP 매거진에 배치해야 합니다. 그런 다음 ACSLS는 모든 방법을 동원해서 CAP 매거진에 인접한 LSM에 카트리지를 배치합니다.
스크래치 카트리지	사용되는 각 LSM에서 사용 가능한 스크래치 카트리지의 수량이 충분한지 확인합니다. SL8500의 경우에는 라이브러리의 각 레일(LSM)에서 스크래치 카트리지를 사용할 수 있습니다.
사용 가능한 셀	각 LSM에에 인접한 사용 가능한 셀이 있는지 확인합니다.

작업 로드를 지원하도록 테이프 드라이브 구성

SL8500에서 테이프 드라이브의 구성 방법에 따라 테이프 작업 로드를 지원하면서 엘리베이터 및 PTP 작업을 최소화할 수 있습니다. SL8500에서 테이프 드라이브의 배치 위치를 결정하는 데 사용할 수 있는 전략은 다음과 같습니다.

- 작업 로드에 필요한 최대 드라이브를 지원할 수 있도록 충분한 드라이브를 사용해서 작업 로드별로 카트리지를 클러스터화합니다. 각 작업 로드에 사용되는 카트리지를 개별 레일로 구분하고, 작업 로드 전용으로 지정된 레일에 해당 작업 로드의 최대 사용량에 맞는 최대 동시 마운트를 충족할 수 있도록 드라이브가 충분한지 확인합니다. 레일에 작업 로드에 대한 테이프 카트리지뿐만 아니라 필요한 스크래치 카트리지도 있는지 확인합니다.
- 각 주요 응용 프로그램 작업 로드에 개별 레일을 할당합니다. Symantic NetBackup 및 Tivoli와 같은 일부 응용 프로그램은 고유 매체 및 드라이브를 사용할 수 있기 때문입니다.
- 단일 레일에 대한 드라이브 및 매체 클러스터화는 시간당 마운트 임계값에 도달하거나, 모든 드라이브가 사용 중이거나, 레일 하나에 들어갈 수 없을 정도로 활성 카트리지가 너무 많아질 때까지 작동합니다. 작업 로드에 필요한 리소스가 레일 용량을 초과할 경우에는 카트리지 및 드라이브를 2개 이상의 레일로 확산합니다.
- 드라이브를 유형별로 클러스터화하고 서로 다른 매체 유형을 사용하는 드라이브는 별도의 레일(LSM)에 배치합니다. 예를 들어, T9840 드라이브와 T10000 드라이브를 서로 다른 레일에 배치합니다.

- 라이브러리 구성의 성능 제한을 초과하지 않도록 중요 테이프 응용 프로그램을 구성합니다.
- 8개 HandBot(레일당 2개의 HandBot)을 사용해서 SL8500을 구성하여 중복성을 제공합니다. 이렇게 하면 작업 로드를 지원하는 카트리지 및 드라이브에 항상 액세스할 수 있습니다.

카트리지 위치 관리

라이브러리에서 처음에 카트리지를 넣은 방법 또는 라이브러리에서의 카트리지 상태는 ACSLS 성능에 영향을 줄 수 있습니다. 고려할 사항은 다음과 같습니다.

카트리지 넣기	권장 사항: CAP를 통해 카트리지를 넣습니다. 전면 액세스 도어가 열린 상태로 라이브러리에서 카트리지를 수동으로 배치하면 라이브러리 작업이 중단되고 ACSLS이 전체 감사를 수행하여 라이브러리의 실제 컨텐츠와 일치하도록 라이브러리 데이터베이스를 업데이트해야 합니다. 성능을 극대화하려면*:* CAP(카트리지 액세스 포트)를 통해 카트리지를 넣습니다. 넣기 중에 라이브러리는 온라인 상태로 유지되며, 마운트가 계속 수행되고, 라이브러리 관리 소프트웨어는 항상 CAP 매거진에 인접한 LSM으로 카트리지를 이동하려고 시도하여 전달 작업을 최소화합니다. 이렇게 할 수 없는 경우에는 라이브러리 컨트롤러가 엘리베이터를 통해 다른 LSM으로 카트리지를 이동하므로, 2개의 HandBot과 엘리베이터 사이에 추가 이동이 필요합니다.
카트리지 클러스터화	해당 작업 로드에 필요한 최대한의 활동(최대 사용량)을 지원할 수 있도록 충분한 테이프 드라이브가 포함된 개별 레일에서 작업 로드별로 카트리지를 클러스터화합니다.
float 사용	권장 사항: *`float`*가 설정된 경우, ACSLS는 마운트 해제 시 가능한 한 드라이브와 가까운 LSM에 있는 카트리지에 대해 새로운 홈 셀을 선택합니다. 이 옵션은 해당 작업 로드에 대해 드라이브별로 카트리지를 자동으로 클러스터화합니다. 해당 LSM에서 새 홈 셀을 선택할 수 있도록 각 LSM에 사용 가능한 셀이 충분히 포함되어 있는지 확인합니다.
스크래치 카트리지 공급	각 레일에서 데이터 카트리지의 수량 및 유형이 올바른지 확인하고 작업 로드를 지원하기에 충분한 스크래치 카트리지가 있는지 확인합니다.

누락된 카트리지 찾기

카트리지가 배치되지 않았거나 ACSLS에서 고려되지 않은 경우:

1. SL 콘솔을 사용해서 SL8500에 대한 물리적 감사를 수행합니다.

   SL8500의 물리적 감사는 마운트 및 기타 라이브러리 작업 요청을 처리하는 사이에 백그라운드 작업으로 수행됩니다.

주의:

SL8500 컨텐츠가 카트리지 직접 로드와 같은 수동 작업으로 인해 ACSLS와 동기화되지 않은 경우에는 계속 작업을 수행하지 않는 것이 좋습니다.

테이프를 수동으로 추가하려는 경우 SL8500 내에서 특정 LSM에 추가하는 방법이 더 좋습니다. 특정 LSM에 테이프를 추가하고 해당 LSM만 감사하는 것이 더 빠르고 더 안정적인 방법입니다.

감사를 진행하는 동안 해당 LSM을 ACSLS에 대해 진단 상태로 전환(*vary*)해야 합니다. SL8500 라이브러리 감사가 수행된 다음 LSM을 ACSLS에 대해 온라인으로 전환(*vary*)합니다.

2. ACSLS *audit*를 실행해서 라이브러리 카트리지의 실제 인벤토리와 일치하도록 ACSLS 데이터베이스를 업데이트합니다.

SL8500 오프라인 전환

SL8500 구성 요소가 작동하지 않을 경우 전원을 켜기 전 그리고 SL8500 액세스 도어를 열기 전에 ACSLS에 대해 SL8500 구성 요소를 오프라인으로 전환(*vary*)합니다. 그러면 ACSLS에서 이러한 구성 요소의 사용 불가 상태가 인식됩니다. 사용 가능해지면 다시 온라인으로 전환(*vary*)합니다.

SL 콘솔이 아닌 ACSLS를 사용하여 SL8500 구성 요소를 오프라인으로 전환

SL 콘솔이 아닌 ACSLS에 대해 SL8500 구성 요소(ACS, LSM 및 CAP)를 오프라인으로 전환(*vary*)합니다.

ACSLS에서는 오프라인 전환이 강제 적용되지 않는 한 구성 요소를 오프라인으로 전환하기 전에 대기 중인 요청을 완료할 수 있습니다. SL 콘솔에서는 ACSLS에 대해 대기 중인 요청이 인식되지 않습니다.

SL 콘솔을 사용해서 구성 요소를 오프라인으로 전환하면 진행 중인 요청이 실패할 수 있습니다.

ACSLS에 대해 SL8500 구성 요소를 오프라인으로 전환해야 하는 경우

이 절에서는 ACSLS에 대해 SL8500 구성 요소를 오프라인으로 전환해야 하는 경우에 대해 설명합니다.

액세스 도어를 열기 전

SL8500 액세스 도어를 열기 전에 ACS 또는 4개의 모든 LSM을 오프라인으로 전환합니다.

- 독립형 SL8500의 경우 다음 명령을 사용해서 ACS를 오프라인으로 전환(*vary*)합니다.

  vary acs acs_id offline
- PTP를 통해 연결된 SL8500의 경우 다음 명령을 네 번 사용해서(4개의 LSM에 대해 각각 한 번씩 사용) 4개의 모든 LSM(액세스 도어가 열리는 SL8500에 있는)을 오프라인으로 전환(*vary*)합니다.

vary lsm lsm_id offline

주:

SL8500의 CAP가 자동 모드인 경우 다음을 수행해야 합니다.

1. 액세스 도어를 열기 전에 수동 모드로 설정합니다.
2. 액세스 도어를 닫고 SL8500을 온라인으로 전환한 다음 자동 모드로 다시 설정합니다.

CAP가 작동하지 않는 경우

CAP가 작동하지 않으면 다음 명령을 사용해서 오프라인으로 전환(*vary*)합니다.

vary cap cap_id offline

서비스 안전 도어를 닫을 때

하드웨어 교체 시 서비스 안전 도어를 사용해야 할 때는 항상 서비스 안전 도어를 가능한 한 최소한의 시간 동안만 닫은 상태로 유지하는 것이 좋습니다. 서비스 안전 도어는 특정 요청을 완료하기 위해 액세스가 필요할 수 있는 다른 하드웨어 구성 요소(엘리베이터, CAP 및 셀)를 차단합니다.

- SL8500의 왼쪽 또는 오른쪽 측면에서 서비스 안전 도어를 닫으려면 먼저 SL 콘솔을 통해 해당 측면에 있는 엘리베이터를 오프라인으로 전환(*vary*)합니다.

  서비스 안전 도어가 열린 다음에는 SL 콘솔을 통해 해당 측면에 있는 엘리베이터를 다시 온라인으로 전환(*vary*)합니다.
- 서비스 안전 도어가 오른쪽 측면에서 닫힌 경우, CAP에 대한 액세스가 차단됩니다.
- SL8500의 오른쪽 측면에서 서비스 안전 도어를 닫으려면 먼저 ACSLS를 통해 CAP를 오프라인으로 전환(*vary*)합니다.
- 서비스 안전 도어가 열린 다음에는 ACSLS를 통해 CAP를 온라인으로 전환(*vary*)합니다.

주:

서비스 베이를 나머지 라이브러리와 구분하도록 SL8500 서비스 안전 도어가 닫혀 있으면, CSE가 LSM 또는 ACS를 오프라인으로 전환하지 않고도 해당 측면에서 액세스 도어를 열 수 있습니다.

서비스 안전 도어 사용 시 사용하지 않아야 하는 ACSLS 명령 및 유틸리티

서비스 안전 도어를 사용 중일 때는 진행 또는 시작하지 않아야 하는 일부 ACSLS 명령 및 유틸리티가 있습니다. 이러한 명령은 다음과 같습니다.

서비스 안전 도어가 어느 한쪽 측면에서 닫혀 있을 때는 다음 유틸리티를 사용하지 마십시오.

- *acsss_config*

- *config(config drives*는 사용 가능*)*

서비스 안전 도어가 오른쪽(CAP) 측면에서 닫혀 있을 때는 다음 명령을 사용하지 마십시오.

- *enter*
- *eject*
- *set cap mode auto <cap_id>*

서비스 안전 도어가 오른쪽(CAP) 측면에서 닫혀 있을 때는 다음 명령을 사용할 수 있지만, 특별한 고려 사항이 적용됩니다.

- *audit*

  audit 명령은 사용할 수 있습니다. 하지만 감사 결과로 인해(감사 시 중복 항목 또는 읽을 수 없는 레이블이 발견된 경우) 카트리지에 대해 *eject*를 수행해야 할 경우, 감사가 완료되고 ACSLS 데이터베이스가 업데이트되지만, 카트리지는 꺼내지지 **않습니다**.
- *vary acs* 및 *vary lsm*

  이러한 명령은 실행이 성공하지만 *cmd_proc*에 메시지가 표시되고 이벤트 로그에 CAP 오류 및 작동하지 않는 CAP가 보고됩니다.

동적 구성(config) 유틸리티 사용

동적 구성(*config*) 유틸리티를 사용하면 ACSLS가 온라인 상태로 실행되는 동안 ACSLS 라이브러리(및 구성 요소)에 대한 구성 변경을 구현할 수 있습니다. 이러한 구성 변경사항은 *acsss_config.log* 파일에 기록됩니다.

다음과 같은 동적 구성 유틸리티가 지원됩니다.

- *config acs*
- *config drives*
- *config lsm*
- *config ports*

config 유틸리티를 사용하면 다음과 같은 이점이 있습니다.

- ACSLS가 실행을 계속하여 사용자가 영향을 받지 않는 라이브러리 구성 요소에 대해 *mount* 요청을 수행할 수 있습니다.
- 다른 모든 구성 정보가 변경되지 않은 상태에서 지정된 라이브러리 구성 요소를 다시 구성할 수 있습니다. 예를 들어, 다음과 같습니다.
  - 특정 ACS를 지정할 때는 다른 ACS의 구성 요소가 영향을 받지 않습니다.
  - 특정 LSM을 지정할 때는 다른 LSM의 구성 요소가 영향을 받지 않습니다.
  - 모든 기존 드라이브에 대한 드라이브 패널(패널에 있는 드라이브) 마운트 및 마운트 해제가 영향을 받지 않습니다.

SL8500 확장

SEM(스토리지 확장 모듈)이 SL8500에 추가되어 용량이 늘어납니다. CAP 및 SEM 또는 CIM에 현재 연결되어 있는 RIM(로봇 인터페이스 모듈)이 포함된 CIM(고객 인터페이스 모듈) 사이에 SEM이 삽입됩니다.

SL8500이 확장되는 경우:

- 라이브러리 구성이 변경되며, ACSLS가 작동 중인 동안에는 ACSLS 동적 구성 유틸리티를 사용해서 ACSLS를 다시 구성하고, ACSLS가 작동 중지된 동안에는 *acsss_config*를 실행해야 합니다.
- SL8500 확장을 위해 카트리지를 제거해야 합니다. **이러한 카트리지를 다시 라이브러리에 배치할 때는 이전에 사용된 셀 배열을 빈 상태로 둡니다**.
- 확장이 완료된 다음에는 SL8500을 두 번 재부트해야 합니다. 먼저 새 구성을 검색하고, 두번째로, 업데이트된 구성을 사용해서 모든 라이브러리 구성 요소를 다시 시작합니다.
- 라이브러리는 물리적 감사를 통해 모든 카트리지의 위치를 업데이트해야 합니다.

  액세스 도어가 닫히면 라이브러리가 물리적 감사를 자동으로 시작합니다. 라이브러리의 물리적 감사는 모든 로봇 이동이 최소한 1분 이상 중지될 때 완료됩니다.
- 라이브러리의 물리적 감사가 완료되었으면 “확장된 SL8500 감사 ” 절차에 따라 ACSLS 데이터베이스를 업데이트합니다.

> **주:**
>
> ACSLS 감사가 완료되고 ACSLS 데이터베이스가 새로운 카트리지 위치로 업데이트되기 전까지는 자동 라이브러리 작업을 시작하지 마십시오.

포함된 작업

SL8500의 물리적 확장에는 다음과 같은 작업이 포함됩니다.

- 기존 SEM 또는 RIM과 CIM 사이에 새로운 SEM이 삽입됩니다.

  새로운 SEM 및 CIM의 패널 번호는 이제 기존 SEM 및 RIM의 패널 번호보다 높습니다.
- CIM이 외부로 이동해야 하기 때문에 CIM의 3개 셀 패널(열)에 새로운 더 높은 패널 번호가 지정됩니다. CIM의 셀 패널이 더 높은 패널 번호로 지정된 경우 CIM의 모든 카트리지 주소가 변경됩니다.
- SL8500을 확장하려면 많은 카트리지를 제거해야 합니다. CIM을 잭으로 들어올리기 위해 기존 레일 볼트를 풀고 새 레일의 볼트를 조이려면 셀 배열을 제거해야 합니다.
- 새 레일을 설치하고 SL8500이 확장된 다음에는 제거한 카트리지를 다시 라이브러리에 로드할 수 있습니다.**라이브러리를 확장하기 위해 임시로 제거한 셀 배열은 빈 상태로 둡니다.**

확장 후에는 ACSLS가 이러한 카트리지의 새 주소로 데이터베이스를 업데이트할 수 있도록 라이브러리에 대해 *audit*를 수행합니다. 카트리지가 새 위치에 배치되었으면 ACSLS가 이

전에 셀에 있던 카트리지를 찾으려고 시도하여 감사 중 라이브러리 성능이 크게 저하될 수 있습니다. ACSLS와 라이브러리 성능을 모두 최적화하기 위해서는 다음 절차를 수행합니다.

1. ACSLS 감사를 수행하기 전에 라이브러리의 물리적 감사가 완료되도록 기다립니다. 라이브러리가 해당 데이터베이스에서 ACSLS에 대한 카트리지 위치를 보고할 수 있으면 조합된 감사가 더 빠르게 완료됩니다. 그렇지 않으면, ACSLS의 *audit* 요청에 응답하기 전에 라이브러리가 카트리지 위치를 다시 확인해야 합니다.
2. 라이브러리 확장을 위해 제거된 카트리지에 대해서는 다음 전략 중 하나를 사용해서 라이브러리에 다시 삽입합니다.
   a. 라이브러리에서 카트리지를 제거하고, ACSLS에서 라이브러리를 감사(*audit*)하고, 제거된 카트리지를 라이브러리에 다시 넣습니다(*enter*).
   b. 라이브러리에 추가된 패널 번호에만 카트리지를 삽입하고, 이러한 패널을 먼저 감사(*audit*)합니다.

      ACSLS 스크래치 풀을 사용해서 스크래치 카트리지를 추적하지 않을 경우, 빈 볼륨 보존을 사용으로 설정한 경우(기본값) 특별한 절차가 필요하지 않습니다. 라이브러리를 *audit*할 때, 이전에 이동된 카트리지는 비어 있는 것으로 표시되는 경우가 많지만, *audit*를 통해 카트리지의 새로운 위치가 확인되고 다시 활성화됩니다. 카트리지 위치가 업데이트되고 중요한 정보도 손실되지 않습니다.

      카트리지를 ACSLS 스크래치 풀에 지정하여 스크래치 카트리지를 관리하고 있고, 카트리지의 스크래치 상태를 지울 필요가 없으면, 새로 추가된 패널에 카트리지를 배치하고, 이러한 패널을 먼저 감사(*audit*)합니다.
      - 이전에 제거된 카트리지를 라이브러리에 추가된 패널에 삽입합니다. 즉, 새로운 첫 번째 SEM에서 드라이브에 가장 가까운 3개 패널을 넘어서 다른 패널(열)에만 삽입합니다. (첫번째 SEM의 처음 3개 패널에는 CIM의 3개 패널에 이전에 지정된 패널 번호가 사용됩니다.)
      - 그런 다음 추가된 모든 새 패널 번호를 통해 가장 높은 패널 번호부터 아래쪽으로 ACS 또는 LSM에 있는 패널을 감사(*audit*)합니다.

        audit cap_id panel panel_id 명령을 사용합니다.
      - 마지막으로 나머지 ACS 또는 LSM을 감사(*audit*)합니다.

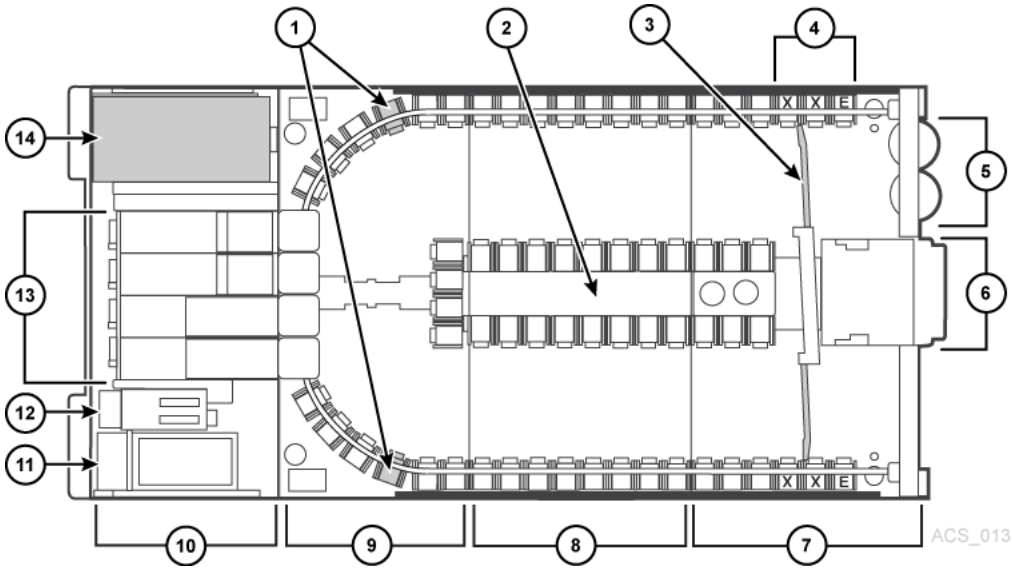
        audit cap_id acs acs_id 또는
        audit cap_id lsm lsm_id 명령을 사용합니다.

      이 절차에 대해서는 "확장된 SL8500 감사 "에서 자세히 설명합니다.

SL8500 모듈의 다이어그램:

다음 다이어그램에서 고객 확장 모듈(#5)의 3개 셀 패널은 CEM의 드라이브 측면에 있는 외벽 및 내벽 모두의 3개 카트리지 열입니다. 이러한 주소는 CIM에서 서비스 안전 도어가 활성화되었을 때 HandBot이 액세스할 수 있는 유일한 셀 주소입니다.

그림 C.3. SL8500 고객 확장 모듈



범례:

1. 카트리지 액세스 포트(2) CAP
2. 전면 운영자 패널(선택사항) 아이스 패드
3. 고객 인터페이스 모듈
4. 스토리지 확장 모듈
5. 로봇 인터페이스 모듈
6. PTP(전달 포트)
7. 드라이브 전자 부품 모듈
8. AC 전원 공급 장치 전자 부품 제어 모듈
9. DC 전원 공급 장치
10. 테이프 드라이브
11. 부속품 랙
12. 내벽
13. 서비스 도어
14. 예약된 열
    - E = 중단 중지점
    - S = 시스템 셀

확장된 SL8500 감사

SL8500 확장 전:

1. 카트리지의 스크래치 상태를 보존하려는 경우:

a. 다음 명령을 실행하여 라이브러리에 있는 LSM에 대해 가장 높은 패널 번호를 확인합니다.

*display panel acs,lsm,**

설명:

- *acs*는 ACS입니다.
- *lsm*은 LSM 중 하나입니다. SL8500의 모든 레일(LSM)은 동일한 패널 번호를 갖습니다.
- *는 LSM에 있는 모든 패널을 표시합니다.

b. 가장 높은 패널 번호를 기록합니다.

2. 구성이 업데이트되고 카트리지 주소가 *audit*로 업데이트될 때까지 LSM을 진단 상태로 전환(*vary*)합니다.
3. ACSLS 데이터베이스를 백업합니다.
4. SL8500 확장:

확장 중에는 신규 또는 확장된 레일을 설치하고 스토리지 확장 모듈을 추가하기 위해 카트리지를 제거해야 합니다.

주:

라이브러리 확장을 위해 임시로 제거한 셀 배열에는 이러한 카트리지를 다시 삽입하지 마십시오. 이렇게 하면 이러한 위치를 먼저 감사할 때 라이브러리 성능이 심각하게 저하되는 것을 방지할 수 있습니다.

5. 이전에 제거된 카트리지를 라이브러리에 추가된 패널에 삽입합니다.

즉, 새로운 첫번째 스토리지 확장 모듈에서 드라이브에 가장 가까운 3개 패널을 넘어서 다른 패널(열)에만 이러한 카트리지를 삽입해야 합니다. (첫번째 스토리지 확장 모듈의 처음 3개 패널은 CIM의 패널에 이전에 지정된 번호가 보존됩니다.)

SL8500의 물리적 확장이 완료된 후:

6. 라이브러리를 두 번 재부트합니다.
7. LSM을 진단 상태로 둡니다.
8. 다음 방법 중 하나를 사용해서 ACSLS 데이터베이스에서 구성을 업데이트합니다.
   - ACSLS가 실행 중인 경우:

     config acs acs_id

     또는 SL8500의 각 LSM(레일)의 경우:

     config lsm lsm_id
   - ACSLS 종료:
     - ACSLS 작동 중지: *acsss disable*
     - ACSLS 구성 업데이트: *acsss_config*
     - ACSLS 작동: *acsss enable*

카트리지의 스크래치 상태를 보존하려는 경우에는 9~11단계를 사용해서 라이브러리를 감사(*audit*)합니다. ACSLS 스크래치 풀을 사용해서 카트리지 스크래치 상태를 추적하지 않을 경우에는 12단계로 건너뜁니다.

9. 라이브러리에 있는 LSM에 대해 새로운 가장 높은 패널 번호를 확인하고 다음 명령을 사용해서 이 번호를 기록합니다.

   *display panel acs,lsm,**

   설명:

   - *acs*는 ACS입니다.
   - *lsm*은 LSM 중 하나입니다. SL8500의 모든 레일(LSM)은 동일한 패널 번호를 갖습니다.
   - *는 LSM에 있는 모든 패널을 표시합니다.

10. 새로운 가장 높은 패널 번호에서 이전의 가장 높은 패널 번호+1까지 확장 중 추가된 패널을 감사(*audit*)합니다. 다음 명령을 사용해서 내림차순으로 이러한 패널을 감사(*audit*)합니다.

    audit cap_id panel panel_id

    설명:

    - *cap_id*는 중복된 *vol_ids*를 갖는 카트리지 또는 읽을 수 없는 레이블을 꺼내야 하는 CAP입니다.
    - *panel_id*는 감사 중인 패널입니다(*acs*,*lsm*,*panel*).

11. 각 LSM에 있는 추가된 각 패널에 대해 별도의 감사를 수행합니다.

    주:

    라이브러리 컴플렉스에 있는 CAP 수 만큼만 동시 감사를 실행할 수 있습니다. 각 동시 감사에는 별도의 CAP가 필요합니다. 감사 작업으로 CAP에 대해 카트리지 꺼내기가 수행된 경우, 감사가 완료되기 전에 이를 제거해야 합니다.

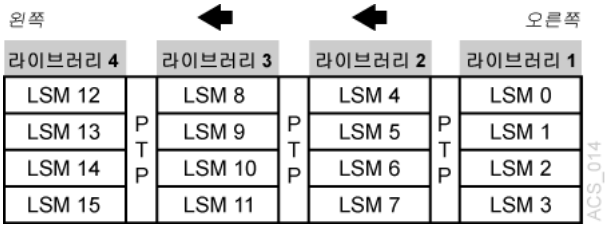
12. 확장된 SL8500에서 전체 LSM을 각각 감사(*audit*)합니다(또는 전체 ACS 감사).
13. ACSLS 데이터베이스를 백업합니다.
14. LSM을 온라인으로 전환(*vary*)합니다.
15. 자동화된 라이브러리 작업을 재개합니다.

전달 포트로 SL8500 연결

SL8500 라이브러리를 기존 SL8500 라이브러리 컴플렉스에 추가할 때는 카트리지 주소 업데이트를 위해 SL8500을 *audit*해야 합니다.

SL8500 컴플렉스의 LSM은 다음 그림에 표시된 것처럼 CAP 끝쪽에서 볼 때 위에서 아래로 그리고 오른쪽에서 왼쪽의 순서로 번호가 지정됩니다.

그림 C.4. 4개의 연결된 SL8500 라이브러리



사이트 설정 방법에 따라 기존 SL8500의 오른쪽 또는 왼쪽에 새로운 SL8500을 추가할 수 있습니다. 결과는 다음과 같습니다.

- 새로운 SL8500이 왼쪽에 추가된 경우, 새로운 LSM은 높은 번호를 갖게 되고, 기존 LSM은 번호가 다시 지정되지 않습니다. 작업 중단은 최소한으로 유지됩니다.
- 새로운 SL8500이 오른쪽에 추가된 경우에는 모든 기존 LSM의 번호가 다시 지정됩니다. 이로 인해 모든 기존 카트리지의 홈 셀 주소가 변경됩니다.

SL8500 PTP 연결을 설치하기 전

1. SL8500 연결을 설치하기 전에 ACSLS에 모든 관련 유지 관리를 적용합니다.
2. 비어 있는 카트리지 지원을 사용으로 설정합니다.

   비어 있는 카트리지 지원이 사용으로 설정된 경우, ACSLS는 삭제된 LSM에서 또는 *audit*로 이전 주소에서 카트리지를 찾을 수 없을 경우, 카트리지를 삭제하지 않습니다. 이러한 카트리지는 비어 있는 것으로 표시되며, 스크래치 상태를 제외하고, 이에 대한 모든 정보가 보존됩니다. 이러한 카트리지가 나중에 *audit*로 발견되면, 다시 활성화됩니다.
   - 비어 있는 카트리지 지원은 *ABSENT_VOLUME_RETENTION_PERIOD* 변수가 0이 **아닌 경우**에 사용으로 설정됩니다. 기본값은 5일입니다.
   - *acsss_config*(옵션 3)를 사용합니다. 절차는 “CSI 조정 변수 설정” 및 *ABSENT_VOLUME RETENTION_PERIOD* 변수를 참조하십시오.
3. 다음 방법 중 하나를 사용해서 구성을 업데이트합니다.
   - ACSLS가 작동 중이면 다음 명령을 사용합니다.

     config acs acs_id
   - ACSLS가 작동 중이 아니면 다음 명령을 사용합니다.
     - ACSLS 작동 중지: *acsss disable*
     - ACSLS 구성 업데이트: *acsss_config*
     - ACSLS 작동: *acsss enable*

     *acsss_config*를 실행할 때는 ACSLS가 실행 중일 수 없습니다.

     자세한 내용은 “CSI 조정 변수 설정” 또는 “acsss 매크로 ”를 참조하십시오.

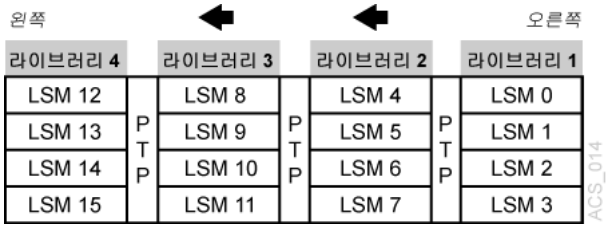
새로운 SL8500 추가

추가 SL8500이 기존 SL8500 라이브러리 컴플렉스에 추가된 경우, 새로운 ACSLS 구성을 업데이트해야 합니다. 새로운 SL8500 추가로 기존 SL8500의 LSM 번호가 다시 지정될 경우, 이러한 LSM에 있는 카트리지 주소를 업데이트해야 합니다.

카트리지와 연관된 다른 정보를 손실하지 않으면서 카트리지 주소를 업데이트해야 합니다. 여기에는 스크래치 상태, 풀, 잠금 및 소유권 및 청소 카트리지 사용 횟수와 같은 카트리지 정보와 카트리지를 넣은 날짜 및 시간이 포함됩니다.

연결된 SL8500의 LSM은 CAP 끝쪽에서 볼 때 위에서 아래로 그리고 오른쪽에서 왼쪽의 순서로 번호가 지정됩니다.

그림 C.5. 4개의 연결된 SL8500 라이브러리



새로운 SL8500을 왼쪽에 추가

새로운 SL8500을 원본 SL8500의 왼쪽(CAP 끝쪽에서 볼 때)에 무중단 방식으로 추가할 때는 호스트 소프트웨어(ACSLS)를 사용해서 추가 드라이브 및 라이브러리를 동적으로 구성합니다. 동적 구성이 설치된 경우에는 ACSLS를 재사용할 필요가 없습니다. 이 단계 중에는 기존 SL8500에 대해 *Mount* 요청이 정상적으로 계속 수행됩니다.

카트리지가 새로운 SL8500의 셀에 배치된 경우 새로운 SL8500의 LSM에서 ACSLS *audit*를 실행하여 이러한 카트리지를 ACSLS 데이터베이스에 추가해야 합니다.

기존 SL8500의 LSM은 감사 중 온라인일 수 있습니다.

새로운 ACSLS 구성을 동적으로 구성

1. 라이브러리 컴플렉스에 새로운 SL8500을 추가합니다.
2. 구성을 변경하기 전에 ACSLS를 백업합니다.
3. 다음 명령을 사용해서 ACSLS 구성을 동적으로 업데이트합니다.

   config acs acs_id

   또한 ACSLS가 작동 중지되었을 때에는 다음 명령을 사용해서 ACSLS 구성을 업데이트할 수 있습니다.

   acsss_config

4. 구성을 변경한 후 ACSLS를 백업합니다.

주:

새로운 SL8500에 카트리지가 있는 경우 새로운 SL8500에서 LSM(레일)을 *audit*하여 이러한 카트리지를 ACSLS 데이터베이스에 추가합니다.

새로운 SL8500을 오른쪽에 추가

새로운 SL8500을 오른쪽에 추가하려는 경우에는 다음 그림에 표시된 것처럼 기존 LSM의 번호가 다시 지정됩니다.

SL8500을 오른쪽에 추가할 때의 고려 사항

새로운 SL8500을 오른쪽에 추가하면 기존의 모든 LSM 번호가 다시 지정되고 카트리지 주소가 변경됩니다. LSM 번호를 변경하면 기존의 모든 카트리지 주소가 변경됩니다. ACSLS가 주소가 변경된 카트리지를 *mount*하려고 시도하면 ACSLS가 카트리지를 찾을 수 없기 때문에 *mount*가 실패합니다.

- 모든 카트리지 주소가 업데이트될 때까지 마운트 작업을 중지하십시오.
- 마운트를 방지하려면 ACS의 모든 LSM을 진단 상태로 전환(*vary*)합니다.
- 특정 순서에 따라 기존 및 신규 SL8500을 감사하여 카트리지 주소를 업데이트합니다.

새로운 ACSLS 구성을 동적으로 구성

새로운 SL8500을 추가한 후 ACSLS 구성을 동적으로 업데이트하려면 다음을 수행합니다.

1. 기존 LSM을 진단 상태로 전환(*vary*)합니다(*vary lsm_id diag*).

주의:

이러한 LSM은 감사될 때까지 진단 상태로 유지되어야 합니다. 그렇지 않으면 다음과 같은 문제가 발생합니다.

   - 마지막으로 알려진 주소에서 카트리지를 찾을 수 없기 때문에 마운트가 실패합니다.
   - 번호가 다시 지정된 LSM에 있는 빈 셀의 ACSLS 맵이 감사 작업으로 업데이트될 때까지:
     - 새 카트리지를 넣으면 기존 카트리지와 충돌합니다.
     - 기존(번호가 다시 지정된) LSM에 대한 카트리지 이동 시 셀에 이미 있는 카트리지와 충돌합니다.
2. 라이브러리 컴플렉스에 새로운 SL8500을 추가합니다.
3. (구성을 변경하기 전에) ACSLS를 백업합니다.
4. 다음 명령을 사용해서 ACSLS 구성을 동적으로 업데이트합니다.

   config acs acs_id

주:

또는 ACSLS가 작동 중지된 상태에서 ACSLS 구성을 업데이트합니다. 먼저 *acsss disable* 명령을 사용해서 ACSLS를 작동 중지합니다. *acsss_config*를 사용해서 구성을 변경하고, 마지막으로 *acsss enable*을 사용해서 ACSLS를 다시 작동합니다.

주:

온라인 상태가 되면 새로운 LSM이 추가됩니다. 감사될 때까지 이러한 LSM을 진단 상태로 전환(*vary*)합니다.

5. (구성 변경 후) ACSLS를 백업합니다.
6. 라이브러리를 *Audit*해서 카트리지 주소를 업데이트합니다(다시 번호가 지정되었기 때문).

   다음 순서를 따릅니다.

   a. 번호가 다시 지정된 기존 SL8500에서 각 LSM을 감사(*audit*)합니다.
      - 카트리지 손실을 방지하기 위해 올바른 순서로 기존 SL8500을 감사(*audit*)합니다.

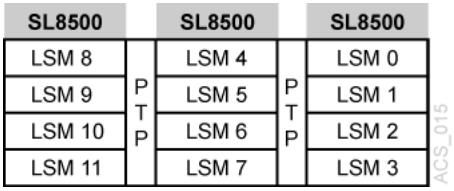
        번호가 가장 높은 LSM에서 번호가 가장 낮은 LSM ID의 순서로 각 LSM(레일)을 감사(*audit*)합니다. 감사를 수행하면 해당(번호가 다시 지정된) LSM 주소로 모든 카트리지가 검색됩니다.
      - ACSLS에 대해서는 각 LSM을 개별적으로 감사(*audit*)하고, 하나의 LSM 감사가 완료된 후에 다음 LSM을 감사해야 합니다.

        *Audit*를 수행하면 카트리지의 이전 홈 셀 주소(LSM에서 오른쪽으로)가 확인되고, 카트리지 주소가 업데이트됩니다. 가장 왼쪽의 SL8500에 있는 LSM에 대한 감사는 시간이 오래 걸립니다. 다른 모든 SL8500에서의 감사는 더 빠르게 수행됩니다.

        가장 왼쪽의 LSM 감사 시간이 오래 걸리는 이유는 연쇄적인 *Cartridge Recovery* 요청이 트리거되기 때문입니다. 감사 작업으로 홈 셀 주소가 다른 카트리지가 발견되면 *audit* 작업이 데이터베이스에 기록된 셀 주소를 확인합니다. 해당 셀이 다른 카트리지에 있으면 *Cartridge Recovery* 작업이 해당 카트리지를 조사합니다. 모든 LSM 주소가 변경되었으므로, SL8500 라이브러리 컴플렉스 내에서 이러한 복구 작업이 연쇄적으로 수행됩니다.
      - LSM은 감사될 때까지 진단 상태로 유지해야 합니다. 감사된 다음에는 이를 온라인으로 전환(*vary*)할 수 있습니다. *Audit*를 수행해서 이 LSM에 있는 카트리지 주소가 업데이트되었으므로, 감사된 LSM의 카트리지를 사용해서 자동화된 *mount* 작업을 재개할 수 있습니다.

   b. 마지막으로 새로 추가된 SL8500(가장 낮은 LSM ID)에서 LSM을 감사(*audit*)합니다.

      이러한 LSM이 감사된 후에는 온라인으로 전환할 수 있으며, 다음 그림에 표시된 것처럼 여기에 포함된 카트리지를 테이프 드라이브에 마운트할 수 있습니다.



7. 감사가 완료된 후 ACSLS를 백업합니다.

ACS 병합 절차

SL8500 PTP에서는 여러 개의 개별 SL8500을 단일 ACS로 병합하는 기능이 지원됩니다. 카트리지에 대한 정보 손실 및 작동 중지 시간을 최소화하기 위해서는 다음과 같은 권장 절차를 따르십시오.

주:

ACSLS 구성이 업데이트될 때 전역 카트리지 주소는 변경되지 **않습니다**.

아래에서는 ACSLS에 대해 다음 두 가지 시나리오를 설명합니다. 첫째, 병합할 ACS는 오른쪽에서 왼쪽(CAP 측면에서 바라볼 때)의 순서로 번호가 지정됩니다. 둘째, 왼쪽에서 오른쪽으로 번호가 지정됩니다. 여기에서는 결과 ACS에 낮은/가장 낮은 ACS ID가 사용된다고 가정합니다.

오른쪽에서 왼쪽으로 번호가 지정되는 ACS 병합

다음 시나리오에서 병합할 ACS는 오른쪽에서 왼쪽(CAP 측면에서 바라볼 때)의 순서로 번호가 지정됩니다.

그림 C.6. 병합할 ACS

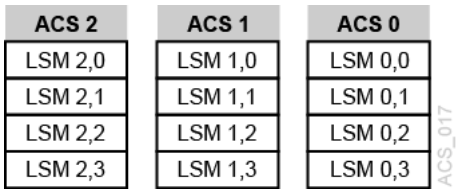
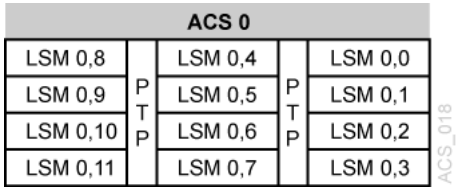


그림 C.7. 필요한 구성: 단일 ACS



오른쪽에서 왼쪽으로 번호가 지정되는 ACS 병합 절차

1. 병합 중인 가장 오른쪽에 있는 ACS를 제외하고 모든 ACS를 온라인으로 전환(*vary*)합니다.

   그러면 카트리지 주소를 업데이트하는 동안 마운트 및 마운트 해제가 방지됩니다.
2. ACSLS *Stop*: `acsss disable`
3. 구성을 변경하기 전에 ACSLS를 백업합니다.
4. (ACSLS가 작동 중지된 상태에서) `acsss_config`를 사용하여 ACSLS 구성을 업데이트합니다.
5. ACSLS 작동: `acsss enable`
6. ACS에 추가된 새로운 LSM을 진단 상태로 전환(*vary*)합니다.

   이러한 LSM은 온라인 상태일 때 추가되었습니다.
7. 구성 변경 후 ACSLS를 백업합니다.
8. ACS에 추가된 LSM을 감사(*audit*)합니다.

   감사되는 순서는 중요하지 않습니다. 전체 ACS 또는 모든 LSM을 한 번에 감사할 수 있습니다.

   카트리지가 다시 활성화됩니다.
9. 감사가 완료된 후 ACSLS를 백업합니다.
10. 새로운 LSM을 온라인으로 전환(*vary*)하고 정상적인 자동화 처리를 재개합니다.

왼쪽에서 오른쪽으로 번호가 지정되는 ACS 병합

다음 시나리오에서 병합할 ACS는 왼쪽에서 오른쪽의 순서로 번호가 지정됩니다.

그림 C.8. 기존 구성: 3개의 개별 ACS

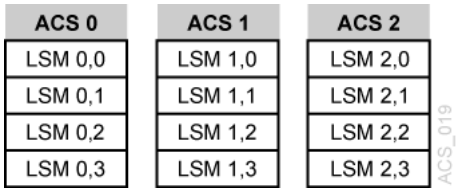
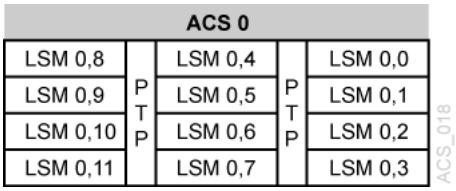


그림 C.9. 필요한 구성: 단일 ACS



왼쪽에서 오른쪽으로 번호가 지정되는 ACS 병합 절차

1. 모든 ACS를 오프라인으로 전환(*vary*)합니다.

   그러면 카트리지 주소를 업데이트하는 동안 마운트 및 마운트 해제가 방지됩니다.
2. ACSLS *Stop*: `acsss disable`.
3. 구성을 변경하기 전에 ACSLS를 백업합니다.
4. (ACSLS가 작동 중지된 상태에서) `acsss_config`를 사용하여 ACSLS 구성을 업데이트합니다.
5. ACSLS 작동: `acsss enable`.
6. ACS에 추가된 새로운 LSM을 진단 상태로 전환(*vary*)합니다.

   (이러한 LSM은 온라인 상태일 때 추가되었습니다.)
7. 구성 변경 후 ACSLS를 백업합니다.
8. 왼쪽에 추가된 새로운 LSM을 진단 상태로 전환(*vary*)합니다.

   이러한 LSM을 감사하기 전까지는 마운트 및 마운트 해제를 수행하지 않아야 합니다.
9. 새로 구성된 LSM을 감사(*audit*)합니다.

   감사되는 순서가 중요합니다. 다음 순서로 감사(*audit*)합니다.
   a. 첫째, SL8500에서 동일한 ACS 번호로 유지되는 LSM을 먼저 감사합니다.
      - *audit*를 수행해서 새로운 LSM 주소로 해당 카트리지를 찾은 후, 이러한 카트리지의 이전 주소로 지정된 LSM을 감사(*audit*)합니다.
      - 이러한 SL8500에 있는 모든 카트리지에 대해 LSM 주소가 업데이트됩니다.
      - ACSLS에 대해서는 각 LSM을 개별적으로 감사(*audit*)하고, 하나의 LSM 감사가 완료된 후에 다음 LSM을 감사합니다.
      - LSM은 감사될 때까지 진단 상태로 유지해야 합니다.

      감사된 다음에는 이를 온라인으로 전환(*vary*)할 수 있습니다. *Audit*를 수행해서 카트리지의 주소가 업데이트되었으므로, 감사된 LSM의 카트리지를 사용해서 자동화된 *mount* 작업을 재개할 수 있습니다.
   b. 마지막으로 가장 낮은 ACS로 병합된 SL8500을 감사(*audit*)합니다. 이러한 SL8500의 카트리지가 다시 활성화됩니다.
      - 이러한 모든 LSM은 동시에 감사할 수 있습니다.
      - 이러한 LSM이 감사되는 순서는 중요하지 않습니다.
10. (감사가 완료된 후) ACSLS를 백업합니다.

11. 모든 LSM을 온라인으로 전환(*vary*)하고 정상적인 자동화 처리를 재개합니다.

PTP 제거 및 ACS 분할

2개의 SL8500을 연결하는 PTP 메커니즘을 제거하고 단일 ACS를 2개의 개별 ACS로 분할해야 할 수 있습니다. 이러한 구성 변경은 2개의 ACS를 단일 ACS로 병합하는 과정의 반대입니다.

SL8500에 새로운 ACS를 추가하는 작업은 분할의 왼쪽 측면에서 수행하는 것이 더 쉽습니다. 이렇게 하면 새로운 ACS에 더 높은 번호로 지정된 LSM이 지정되어 기존 ACS에서 유지되는 LSM의 번호 지정을 방지합니다.

왼쪽의 SL8500에서 새로운 ACS가 생성되는 ACS 분할 - 가능한 시나리오

그림 C.10. 기존 구성: 1개의 ACS

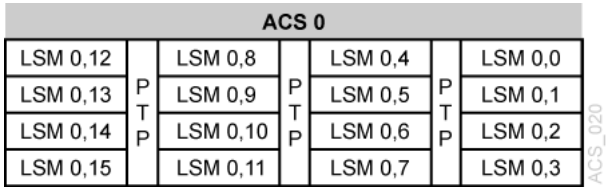
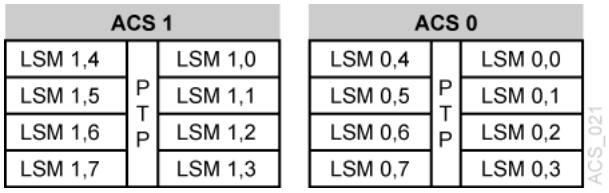


그림 C.11. 필요한 구성: 2개의 ACS



ACS 분할을 위한 ACSLS 절차

동적 구성의 경우, ACSLS가 실행되는 동안 ACS를 추가할 수 있습니다. 하지만 동적 구성에서는 전역 카트리지 주소가 변경되지 않습니다.

1. 구성을 변경하기 전에 ACSLS를 백업합니다.
2. 새로운 ACS로 이동할 LSM을 오프라인으로 전환(*vary*)합니다.

   그러면 카트리지 주소를 업데이트하는 동안 마운트 및 마운트 해제가 방지됩니다.

   기존 ACS에 남아 있는 LSM은 온라인 상태로 유지될 수 있습니다.

   이러한 LSM에서는 마운트 및 마운트 해제가 계속 수행될 수 있습니다.
3. 구분하려는 SL8500을 연결하는 4개의 PTP 메커니즘을 제거합니다.
4. 전달 포트가 제거된 다음 기존 ACS를 다시 구성합니다.

그러면 새로운 ACS로 이동 중인 LSM이 제거됩니다. 다음 명령을 사용합니다.

config acs acs_id

5. 다음 명령을 사용해서 새 ACS를 추가합니다.

   config acs acs_id new

   또는 ACSLS가 작동 중지된 상태에서 ACSLS 구성을 업데이트할 수 있습니다.

   acsss_config

   새로운 LSM이 온라인 상태로 추가됩니다.
6. 감사될 때까지 이러한 LSM을 진단 상태로 전환(*vary*)합니다.
7. 새로운 ACS를 진단 상태로 전환(*vary*)합니다.
8. 구성 변경 후 ACSLS를 백업합니다.
9. 새로운 ACS에서 LSM을 감사(*audit*)합니다.

   이러한 LSM을 감사하기 전까지는 마운트 및 마운트 해제를 수행하지 않아야 합니다. 감사되는 순서는 중요하지 않습니다. 모두 한 번에 감사할 수 있습니다.

   카트리지가 다시 활성화됩니다.
10. 감사가 완료된 후 ACSLS를 백업합니다.
11. 새로운 LSM을 온라인으로 전환(*vary*)하고 정상적인 자동화 처리를 재개합니다.

    해당 LSM의 *audit*가 완료되는 즉시 LSM을 온라인으로 전환할 수 있습니다.

분할의 오른쪽 측면에서 새로운 ACS 추가

새로운 ACS가 분할의 오른쪽에 추가된 경우, 모든 카트리지 주소가 다시 매핑됩니다.

이 방식은 권장되지 않으므로, 이에 대한 자세한 절차에 대해 설명하지 않습니다.

하지만, 일부 고려할 사항이 있습니다.

- 오른쪽 SL8500의 LSM이 기존 ACS에서 제거될 때는 이러한 LSM의 카트리지가 비어 있는 것으로 표시됩니다(빈 카트리지 보존이 활성 상태인 경우). 이러한 LSM은 새로운 ACS가 감사될 때 다시 활성화됩니다.
- 기존 ACS에 있는 LSM은 이러한 LSM에 있는 카트리지 주소를 업데이트하기 위해 한 번에 하나씩 감사를 수행해야 합니다. 먼저 가장 높은 번호의 LSM을 감사(*audit*)하고, 그 다음 LSM의 순서로 진행합니다. 하나의 LSM *audit*를 완료한 후 다음 감사를 시작해야 합니다.

부록 D

부록 D. SL3000의 ACSLS 지원

SL3000은 다음과 같은 이점을 제공합니다.

- 200~4500개 스토리지 셀의 스토리지 용량
- 1~56개 테이프 드라이브의 성능
- 혼합 매체 및 다양한 테이프 드라이브 지원
- 스토리지 셀 및 드라이브 레벨에서 SL3000 분할 기능
- AEM(액세스 확장 모듈) 설치 옵션(ACSLS 7.3 PUT0801 이상). 이 옵션은 “AEM 사용”에서 설명합니다.

> **주의:**
>
> ACSLS에 대해 SL3000을 구성 또는 다시 구성한 다음에는 SL3000을 *audit*해야 합니다. 또한 라이브러리의 활성 용량을 늘리거나 변경한 후에도 SL3000을 *audit*해야 합니다.
>
> 라이브러리는 ACSLS가 이를 감사할 때 ACSLS에서 액세스할 수 없는 잠재적인 셀 위치를 보고합니다. 액세스할 수 없는 셀 위치에는 CAP, 드라이브 및 운영자 패널이 설치된 위치, 로봇이 액세스할 수 없는 셀 위치, 활성화되지 않은 셀 위치 및 이 분할 영역에 없는 위치가 포함됩니다.

ACSLS 지원

ACSLS는 다음과 같은 SL3000 기능에 대한 지원을 제공합니다.

- 새로운 LSM 및 패널 유형과 SL3000 라이브러리에 대한 패널 맵
- 최대 8개 분할 영역

  SL3000은 드라이브 및 셀 레벨까지 분할될 수 있습니다. 셀이 하나의 분할 영역에서 다른 분할 영역으로 다시 지정된 경우, 해당 셀의 카트리지가 고립되고 이전에 있던 분할 영역에서 더 이상 액세스할 수 없게 됩니다. 이를 방지하기 위해서는 SL3000을 다시 분할하기 전에 분할 영역에 남아 있게 되는 셀로 카트리지를 이동하십시오.
- SL3000에 대한 CAP(12), 패널당 드라이브(32) 및 셀 행(52)의 새로운 최대 개수
- SL3000 내에서 분할 영역에 CAP를 전용으로 지정하는 기능

  CAP는 SL 콘솔을 사용해서 분할 영역에 전용으로 지정됩니다. 라이브러리는 ACSLS에 전용 CAP를 보고합니다.

  ACSLS에서 관리되는 분할 영역에 **전용으로 지정된** CAP는 자동 모드로 설정할 수 있습니다.
- SL3000 CAP 별칭 지정

SL3000은 최대 12개의 CAP를 포함할 수 있지만, 일부 백업 응용 프로그램에서는 LSM 당 최대 3개의 CAP만 지원됩니다. CAP 별칭 지정을 통해 3-10번의 CAP를 CAP 0, 1, 2로 별칭을 지정할 수 있습니다.

PUT0801에서는 더 이상 CAP 0으로 별칭을 지정할 수 없습니다. CAP 0은 AEM(액세스 확장 모듈)용으로 예약되며, 특별한 성격을 갖습니다. 따라서 CAP 번호 1과 2만 별칭으로 사용할 수 있습니다.

- PUT0801이 포함된 ACSLS 7.3에서는 대량 넣기 및 꺼내기에 대한 AEM이 지원됩니다.

SL3000에 대해 ACSLS를 구성하기 전

SL3000에 대해 ACSLS를 구성하기 전:

- SL3000을 ACSLS에 연결합니다.
- SL3000의 모든 구성 요소가 작동하는지 확인합니다.

  ACSLS는 라이브러리에서 보고된 정보로부터 라이브러리 구성을 작성합니다. SL3000 구성 요소가 작동하지 않으면 라이브러리 정보가 ACSLS에 보고되지 않을 수 있으며 SL3000의 ACSLS 구성이 완전하지 않게 됩니다.

SL3000을 ACSLS에 연결

SL3000 라이브러리는 이더넷 물리적 인터페이스를 통해 TCP/IP 프로토콜을 사용해서 호스트 및 ACSLS를 관리하고 이와 통신합니다. 이 인터페이스는 ACSLS가 SL3000에 연결하고 이와 통신할 수 있게 해줍니다.

SL3000에 대해 TCP/IP 연결을 사용하면 다음과 같은 이점이 있습니다.

- ACSLS가 라이브러리에 여러 동시 요청을 전송할 수 있고, 라이브러리가 이를 병렬로 처리할 수 있습니다. 그 결과 라이브러리 성능이 향상됩니다.
- TCP/IP를 사용해서 연결된 경우 SL3000은 이중 TCP/IP 및 중복 전자 부품을 지원합니다.
- TCP/IP 호스트/라이브러리 인터페이스는 광 섬유 채널을 통한 SCSI 매체 교환기보다 기능이 뛰어난 인터페이스입니다. 라이브러리 상태, 오류 등에 대해 더 많은 정보가 ACSLS에 제공됩니다.

ARP 브로드캐스트 범람으로부터 보호하기 위해서는 SL3000 라이브러리를 별도의 서브넷 또는 제어되는 네트워크에 배치하는 것이 좋습니다.

모든 SL3000 구성 요소가 작동하는지 확인

SL3000의 모든 구성 요소가 작동하는지 확인하기 위해서는 StorageTek 라이브러리 콘솔(SL 콘솔)에 로그온합니다.

절차는 *SL3000* 사용 설명서를 참조하십시오.

ACSLS에 대해 SL3000 구성

SL3000 구성 요소가 작동하면 ACSLS에 대해 SL3000을 구성합니다. *acsss_config*("CSI 조정 변수 설정" 참조) 또는 동적 구성("동적 구성(config) 유틸리티 사용" 참조)을 사용할 수 있습니다.

SL3000 감사

ACSLS에 대해 구성 또는 다시 구성한 다음 SL3000을 감사합니다. 감사를 수행하면 이 라이브러리에서 관리 중이고 패널 드라이브 또는 CAP에 실제로 있는 셀을 표시하는 분할 영역에 포함되는 셀을 ACSLS가 확인할 수 있습니다.

- 먼저 ACSLS에 대해 SL3000을 구성하거나 다시 구성합니다.
- ACS를 감사(*audit*)합니다.
- 다음 명령을 입력하여 할당된 셀 및 비어 있는 셀을 조사합니다.
  - *free_cells.sh -a*
  - *get_license_info*
  - *display cell * -f status*
- 다음 중 하나를 사용해서 드라이브를 조사합니다.
  - *query drive all*
  - *display drive * -f type* serial_num *wwn*

SL3000 주소 지정

SL3000은 드라이브 및 스토리지 셀에 대한 ACSLS 주소를 변경하지 않고도 확장할 수 있습니다. ACSLS 주소는 가능한 한 가장 큰 SL3000 라이브러리에 매핑되므로 기존 셀, 드라이브 및 CAP의 주소를 변경하지 않고도 모듈을 추가할 수 있습니다.

모듈이 설치되지 않은 경우, 이러한 모듈의 패널은 단순히 "not installed"로 매핑됩니다. 패널이 설치된 경우에는 이러한 패널의 주소를 사용할 수 있습니다. 이러한 방식에 따라 새 모듈을 설치해도 기존 모듈의 패널 주소가 변경되지 않습니다.

그림 D.1. SL3000 주소 지정

가장 작은 SL3000 구성은 단일 BDM(기본 드라이브 모듈)입니다. 단일 BDM 구성에서는 패널 12 및 13이 설치되며, 다른 모든 잠재적인 패널 위치는 설치되지 않습니다. DEM(드라

이브 확장 모듈)을 추가하면 패널 10과 11도 설치됩니다. BDM의 오른쪽에 CEM(카트리지 확장 모듈)을 추가하면 패널 14 및 15가 설치됩니다. BDM에 있는 셀 및 드라이브의 주소는 변경되지 않습니다. 그 결과 SL3000 용량을 늘릴 수 있습니다. 드라이브 및 스토리지 셀의 ACSLS 주소는 변경되지 않습니다.

CAP 번호 지정

CAP는 다음과 같이 번호 지정됩니다.

- CAP 0은 왼쪽의 AEM에 대해 예약되어 있습니다.
- CAP 1-5는 왼쪽의 CEM과 DEM에 있습니다.
- CAP 6은 기본 모듈에 있습니다(유일한 필수 CAP).
- CAP 7-10은 오른쪽의 CEM에 있습니다.
- CAP 11은 오른쪽의 AEM에 대해 예약되어 있습니다.

CAP가 없으면 ACSLS가 이를 ”not installed”로 보고합니다. 그 결과 새 CAP를 추가할 때 SL3000에서 CAP 위치가 변경되지 않습니다.

CAP는 분할 영역에 전용으로 지정할 수 있습니다.

SL3000 모듈

SL3000 라이브러리에는 5가지 유형의 모듈이 있습니다.

- 기본 모듈 1개, 필수

  기본 모듈은 단일 프레임으로 구성됩니다. 이 모듈은 전원 구성, 로봇(TallBot), 전자 모듈, 카트리지 액세스 포트, 스토리지 셀, 테이프 드라이브 및 운영자 컨트롤을 포함하여 라이브러리에 있는 모든 다른 모듈에 대한 기반구조를 중앙화합니다.

  모든 라이브러리 설치에는 하나의 기본 모듈(하나만)이 필요합니다.

  이 모듈의 전면에는 다음 항목이 있습니다.
  - 단일 26 카트리지 듀얼 매거진 CAP(카트리지 액세스 포트)
  - 라이브러리 액세스를 위한 서비스 도어
  - 3개 LED가 포함된 전면 패널: 라이브러리 활성, 서비스 필요 및 대기
  - 터치 스크린 운영자 패널 또는 창에 대한 선택적 기능
- DEM(드라이브 확장 모듈) 1개, 왼쪽 측면만

  DEM(드라이브 확장 모듈)은 왼쪽에만 있는 기본 모듈에 인접하여 연결할 수 있습니다. 이 모듈은 테이프 드라이브의 추가 확장을 허용하고 추가적인 데이터 카트리지 용량을 제공합니다.
- CEM(카트리지 확장 모듈) 왼쪽 또는 오른쪽 측면

  CEM(카트리지 확장 모듈)은 추가적인 카트리지 셀 용량 및 성장을 제공합니다. 이 모듈에는 테이프 드라이브가 없습니다. 기본 모듈(필수)과 선택적인 드라이브 확장 모듈 외에도 단일 라이브러리에서 최대 4개의 CEM이 지원됩니다.

- PEM(장착 확장 모듈) 왼쪽 끝 및 오른쪽 끝 측면 모듈

  PEM(장착 확장 모듈)은 카트리지 확장 모듈과 동일하며, 라이브러리가 결함이 있는 로봇을 장착할 수 있도록 액세스할 수 없는 배열이 6개(전면 벽에 3개 그리고 후면 벽에 3개) 있습니다. 왼쪽 PEM에는 3개의 왼쪽 끝 열(전면과 후면)이 있습니다. 오른쪽 PEM에는 3개의 오른쪽 끝 열이 있습니다.

  주:

  이중 로봇이 있는 SL3000에는 AEM 또는 PEM이 포함됩니다.

- AEM(액세스 확장 모듈)

  AEM은 “AEM 사용 ”에서 설명합니다.

새 패널 유형

SL3000에서 가능한 각 셀 위치를 정의하는 특정 패널 정의를 사용하는 대신 일반 패널 정의가 있습니다.

3개의 일반 패널 유형은 다양한 특정 패널 맵에 매핑됩니다. 다음과 같습니다.

- 단일 SL3000 셀 패널 유형

  라이브러리에 있는 특정 셀과 활성화된 용량에 포함된 항목 및 분할 영역에 지정된 항목을 확인하려면 라이브러리를 *audit*해야 합니다.
- 드라이브 패널 유형

  ACSLS는 라이브러리를 구성하여 제공되는 드라이브를 식별합니다. 이렇게 하려면 먼저 라이브러리 구성 전에 모든 드라이브가 전원이 켜져 있고 준비되어 있는지 확인해야 합니다.
- 설치되지 않음

  이 항목은 모듈에서 아직 설치되지 않은 전면 및 후면 패널에 대한 위치 표시자입니다.

내부 SL3000 주소 지정 이해

SL3000의 내부 주소와 ACSLS 및 HSC에서 지원되는 다른 라이브러리 사이에는 차이가 있습니다.

- SL3000에서는 *1*부터 시작되며 음수가 사용됩니다.
- 다른 라이브러리에서는 *0*부터 시작되며, 음수가 사용되지 않습니다.
- SL3000에서는 라이브러리, 레일, 열, 측면 및 행의 *5*개 매개변수가 사용됩니다.
- 다른 라이브러리에서는 ACS, LSM, 패널, 행 및 열이 사용됩니다(HLI–PRC).

라이브러리의 모듈은 카트리지, 테이프 드라이브, 카트리지 액세스 포트 및 로봇 장치가 포함된 벽면, 열 및 행으로 구성됩니다.

SL 콘솔을 사용하면 SL3000 내부 주소와 ACSLS 패널, 행 및 열 사이의 변환을 수행할 수 있습니다.

자세한 내용 및 절차는 *SL3000* 사용 설명서를 참조하십시오.

AEM 사용

AEM(액세스 확장 모듈)은 온라인/오프라인 상태 및 분할 영역에서 공유할 수 있는 기능과 같이 CAP의 모든 특징을 갖고 있는 매우 큰 CAP입니다.

AEM에서는 라이브러리에 대해 최대 234개까지 카트리지 대량 로드 또는 언로드를 수행할 수 있습니다. 또한 TallBot 라이브러리의 무중단 유지 관리도 허용됩니다.

AEM은 회전식 CAP와 같은 많은 특징 및 기능을 공유하지만, 고유한 특징도 갖고 있습니다. 다음과 같습니다.

- 액세스 도어
- 안전 도어
- CAP 작업
- 무중단 유지 관리

AEM에 대한 자세한 내용은 *StorageTek SL3000 Modular Library System User's Guide*, 부품 번호 316194401을 참조하십시오.

액세스 도어

AEM 액세스 도어는 라이브러리에서 카트리지를 대량 로드 또는 언로드하기 위해 여는 외부 도어입니다.

안전 도어

안전 도어는 AEM을 라이브러리의 나머지 부분과 분리하기 위해 아래로 내리는 내부의 "차고문"입니다. 이 도어를 사용하면 AEM의 내부에 안전하게 액세스할 수 있으며, 라이브러리 운영자가 카트리지를 로드 또는 언로드하거나 스토리지 CSE가 Tallbot 또는 다른 AEM 구성 요소에서 서비스를 수행할 수 있습니다.

CAP 작업

AEM의 이점으로, 사용자는 다음 작업을 수행할 수 있습니다.

- 대량 넣기

  대량 넣기를 통해 한 번에 최대 234개의 카트리지를 넣을 수 있습니다.

  AEM *cap_id*를 지정하여 정상 넣기를 시작해야 합니다. 수동 넣기로 수행하거나 AEM을 자동 넣기 모드로 설정할 수도 있습니다.

  분할된 라이브러리에서는 CAP가 분할 영역에 전용으로 지정된 경우에만 CAP를 자동 넣기 모드로 설정할 수 있습니다. 특정 분할 영역으로만 AEM을 전용으로 지정하면 다른 분할 영역에서 사용할 수 없습니다. 이를 수행하기 전에 다른 SL3000 사용자를 고려하십시오.

- 대량 꺼내기

  AEM에서 42개를 초과하는 볼륨을 꺼낼 때, 더 큰 꺼내기 작업에서 선호되는 도구는 *lib_cmd* 또는 *ejecting.sh*입니다. *cmd_proc*, *xeject* 또는 ACSAPI 꺼내기를 포함한 다른 모든 사용자 인터페이스에서는 꺼내기 로드가 한 번에 42개로 제한됩니다. “lib_cmd 사용” 및 “ejecting.sh ”를 참조하십시오.
- 무중단 로봇 유지 관리

주의:

AEM을 사용해서 대량 꺼내기를 수행할 때는 모든 CAP 매거진이 있는지 확인합니다. ACSLS가 카트리지 *eject*를 시도하는 위치에 CAP 매거진이 누락된 경우 *eject*가 실패합니다.

주:

소량의 카트리지 넣기 또는 꺼내기를 수행할 때는 AEM을 사용하지 않아야 합니다. 대량 넣기 및 꺼내기에서만 사용해야 합니다.

볼륨을 넣거나 제거하기 위해 AEM에 액세스할 때는 서비스 안전 도어를 아래로 내리고 AEM에 대한 액세스가 완료되면 올려야 하기 때문에 넣기 및 꺼내기 작업이 소량일 때 AEM을 사용하면 시간이 상당히 오래 걸립니다.

수량이 적은 경우에는 CAP 1-10을 사용하십시오. “CAP 번호 지정”을 참조하십시오.

AEM은 소규모 넣기 및 꺼내기 작업에 사용되지 않아야 하므로 최대 CAP 우선 순위는 1입니다. 이러한 설정은 CAP ID가 *audit*, *enter* 또는 *eject*에서 별표를 사용해서 와일드카드 문자로 표시된 경우 AEM이 선택되지 않도록 방지하는 데 도움이 됩니다.

무중단 유지 관리

AEM은 CSE가 라이브러리를 오프라인으로 전환할 필요 없이 안전 도어(또는 "창고" 도어)를 사용해서 TallBot 라이브러리를 액세스 및 서비스하도록 허용합니다. TallBot에 결함이 있으면 라이브러리가 온라인인 상태로 AEM에서 자동으로 장착됩니다. 중복 TallBot가 설치된 경우에는 작동하는 남은 TallBot를 통해 정상 작업을 계속 수행할 수 있습니다.

CSE는 특수 키를 사용해서 안전 도어에서 서비스 액세스 잠금을 해제합니다. 따라서 AEM 안전 도어가 내려가서 AEM이 나머지 라이브러리에서 분리됩니다. 안전 도어가 완전히 내려간 다음에는 라이브러리가 온라인인 상태에서 CSE가 AEM 안전 도어를 열고 결함이 있는 TallBot 또는 다른 AEM 구성 요소에서 유지 관리를 안전하게 수행합니다. 유지 관리 기간 중에는 AEM 자체가 오프라인으로 전환되어 카트리지 로드/언로드 기능이 일시 중지됩니다.

유지 관리가 완료되면 CSE에서 안전 도어를 닫고, 안전 도어가 올라가고, TallBot가 다시 초기화되고 AEM CAP 카트리지 슬롯에 대해 *audit*가 수행되고, AEM이 다시 온라인으로 전환됩니다.

SL3000 CAP ID 별칭 지정

SL3000은 최대 12개의 CAP를 포함하지만, 일부 백업 응용 프로그램에서는 LSM당 최대 3개의 CAP만 지원됩니다. CAP 별칭을 통해 3-10번의 CAP를 CAP 1 또는 2로 별칭을 지정할 수 있습니다.

Cap 0은 AEM(액세스 확장 모듈) 대량 로드 CAP용으로 예약되어 있습니다. SL3000에서 CAP 0은 특수한 속성을 갖기 때문에 별칭 CAP ID가 될 수 없습니다.

다음 템플리트를 사용해서 라이브러리 CAP ID를 별칭 CAP ID에 매핑합니다.

$ACS_HOME/data/external/SL3000/SL3000_CAP_Aliases.SAMPLE

주:

CAP ID를 설정할 때:
라이브러리 CAP ID의 경우:
- LSM은 0이어야 합니다.
- ID는 3-10이어야 합니다.
- 중복 라이브러리 CAP ID는 있을 수 없습니다.
별칭 CAP ID의 경우
- CAP 번호는 1 또는 2여야 합니다.
- 중복 별칭 CAP ID는 있을 수 없습니다.

```
# SL3000 CAP Aliases (SL3000_CAP_Aliases)
#
# This file maps real library SL3000 CAP IDs to CAP ID aliases.
# It is used when ACSAPI clients do not support SL3000
# CAP numbers greater than 2.
# (The SL3000 can have 12 CAPs, with a maximum CAP number of 11.)
# Alias a SL3000 CAP to an alias CAP number by providing the full
# SL3000 CAP ID and the aliased CAP number on the same line.
# For example:
#  SL3000 CAP ID     Alias CAP Number
#          0,0,6                   1
#
#    The LSM must be zero.
#    The CAP ID must be between 3 and 10
#    No duplicate SL3000 CAP IDs.
# Rules for SL3000 CAP IDs:
# Rules for Alias CAP numbers:
#    The CAP number must be 1 or 2.
#    No duplicate alias CAP numbers.
#
 # Using the CAP Aliases file:
#   (1) A # in the first column comments out a line.
#       The above CAP alias example is commented out.
#   (2) Only specify CAP numbers as an alias when there
#       is no actual CAP at that location.  For example,
#       use CAP 2 as an alias when there is no CAP 2
#       installed in the SL3000 library.
#   (3) Only CAP numbers 1 and 2 can be used as aliases.
#       NOTE: CAP number 0 (zero) is reserved for the
#       Access Expansion Module (AEM) and thus cannot
#       be specified as an alias.
#   (4) After updating the CAP Aliases file, you must:
#       a) Shutdown ACSLS.
#       b) Reconfigure ACSLS (using acsss_config) to update
#          the CAPs defined in the ACSLS database.
#       c) Restart ACSLS.
# Save the CAP alias file as:
#   $ACS_HOME/data/external/SL3000/SL3000_CAP_Aliases
# by copying and modifying the template:
#   $ACS_HOME/data/external/SL3000/SL3000_CAP_Aliases.SAMPLE
#
#   SL3000 CAP ID    Alias CAP Number
```

```
          0,0,6                  1
          0,0,4                  2
# Alias a SL3000 CAP to an alias CAP number by providing the full
# SL3000 CAP ID and the aliased CAP number on the same line.
# For example:
#  SL3000 CAP ID     Alias CAP Number
#          0,0,6                   1
#
# Rules for SL3000 CAP IDs:
#    The LSM must be zero.
#    The CAP ID must be between 3 and 10
#    No duplicate SL3000 CAP IDs.
# Alias a SL3000 CAP to an alias CAP number by providing the full
# SL3000 CAP ID and the aliased CAP number on the same line.
# For example:
#  SL3000 CAP ID     Alias CAP Number
#          0,0,6                   1
#
# Rules for SL3000 CAP IDs:
#    The LSM must be zero.
#    The CAP ID must be between 3 and 10
#    No duplicate SL3000 CAP IDs.
```

위 예에서는 수정된 *SL3000_CAP_Aliases* 파일을 보여줍니다. 별칭 CAP 번호 지정이 사용으로 설정되도록 지정하기 위해 굵게 표시된 라인에서 주석 문자(#)가 제거되었습니다. 이 예에서는 다음과 같습니다.

- 물리적 CAP ID 0,0,4의 별칭이 CAP 번호 1로 지정됩니다.
- 물리적 CAP ID 0,0,6의 별칭이 CAP 번호 2로 지정됩니다.

분할 영역에서 셀을 제거하기 전에 카트리지 이동

SL3000은 드라이브 및 셀 레벨까지 분할될 수 있습니다. 자세한 내용은 분할 영역에서 셀을 제거하기 전에 카트리지 이동 을 참조하십시오.

SL3000 CAP 동작

SL3000에서는 최대 12개의 CAP(카트리지 액세스 포트)가 지원됩니다. 각 SL3000 패널에는 CAP가 있을 수 있습니다.

분할된 라이브러리에서 CAP 동작은 “라이브러리 분할 또는 분할 영역 ID 변경 ”을 참조하십시오.

주의:

모든 ACS는 최소한 하나 이상의 CAP를 포함해야 합니다. 이 CAP는 **전용**으로 지정하거나 공유할 수 있습니다. SL3000의 모든 CAP는 다른 분할 영역에 전용으로 지정될 수 있으므로, SL3000 분할 영역을 구성할 때는 이러한 특성이 중요한 제한 사항이 될 수 있습니다.

누락된 SL3000 카트리지 찾기

카트리지가 배치되지 않았거나 ACSLS에서 고려되지 않은 경우:

1. SL 콘솔을 사용해서 SL3000에 대한 물리적 감사를 수행합니다.

   SL3000의 물리적 감사는 *mount* 및 기타 라이브러리 작업 요청을 처리하는 사이에 백그라운드 작업으로 수행됩니다.

   주의:

   SL3000 컨텐츠가 카트리지 직접 로드와 같은 수동 작업으로 인해 ACSLS와 동기화되지 않은 경우에는 계속 작업을 수행하지 않는 것이 좋습니다.

2. ACSLS *audit*를 실행해서 라이브러리 카트리지의 실제 인벤토리와 일치하도록 ACSLS 데이터베이스를 업데이트합니다.

SL3000 오프라인 전환

SL3000 구성 요소가 작동하지 않을 경우 전원을 켜기 전 그리고 SL3000 액세스 도어를 열기 전에 ACSLS에 대해 SL3000 구성 요소를 오프라인으로 전환(*vary*)합니다. 그러면 ACSLS에서 이러한 구성 요소의 사용 불가 상태가 인식됩니다. 사용 가능해지면 다시 온라인으로 전환(*vary*)합니다.

SL 콘솔이 아닌 ACSLS를 사용하여 SL3000 구성 요소를 오프라인으로 전환

SL 콘솔이 아닌 ACSLS에 대해 SL3000 구성 요소(ACS, LSM 및 CAP)를 오프라인으로 전환(*vary*)합니다.

ACSLS에서는 *vary offline*이 강제 적용되지 않는 한 구성 요소를 오프라인으로 전환하기 전에 대기 중인 요청을 완료할 수 있습니다. SL 콘솔에서는 ACSLS에 대해 대기 중인 요청이 인식되지 않습니다.

SL 콘솔을 사용해서 구성 요소를 오프라인으로 전환하면 진행 중인 요청이 실패할 수 있습니다.

ACSLS에 대해 SL3000 구성 요소를 오프라인으로 전환해야 하는 경우

이 절에서는 ACSLS에 대해 구성 요소를 오프라인으로 전환해야 하는 경우에 대해 설명합니다.

액세스 도어를 열기 전

SL3000 액세스 도어를 열기 전 다음 명령을 사용해서 ACS를 오프라인으로 전환(*vary*)합니다.

vary acs acs_id *offline*

주:

SL3000의 CAP가 자동 모드인 경우 다음을 수행해야 합니다.

1. 액세스 도어를 열기 전에 수동 모드로 설정합니다.

2. 액세스 도어를 닫고 SL3000을 온라인으로 전환한 다음 자동 모드로 다시 설정합니다.

CAP가 작동하지 않는 경우

CAP가 작동하지 않으면 다음 명령을 사용해서 오프라인으로 전환합니다.

vary cap cap_id offline

동적 구성(config) 유틸리티 사용

동적 구성(*config*) 유틸리티를 사용하면 ACSLS가 온라인 상태로 실행되는 동안 ACSLS 라이브러리(및 구성 요소)에 대한 구성 변경을 구현할 수 있습니다. 이러한 구성 변경사항은 *acsss_config.log* 파일에 기록됩니다.

다음과 같은 동적 구성 유틸리티가 지원됩니다.

- *config acs*
- *config drives*
- *config lsm*
- *config ports*

config 유틸리티를 사용하면 다음과 같은 이점이 있습니다.

- ACSLS가 실행을 계속하여 사용자가 영향을 받지 않는 라이브러리 구성 요소에 대해 *mount* 요청을 수행할 수 있습니다.
- 다른 모든 구성 정보가 변경되지 않은 상태에서 지정된 라이브러리 구성 요소를 다시 구성할 수 있습니다. 예를 들어, 다음과 같습니다.
  - 특정 ACS를 지정할 때는 다른 ACS의 구성 요소가 영향을 받지 않습니다.
  - 특정 LSM을 지정할 때는 다른 LSM의 구성 요소가 영향을 받지 않습니다.
  - 모든 기존 드라이브에 대한 드라이브 패널(패널에 있는 드라이브) 마운트 및 마운트 해제가 영향을 받지 않습니다.

새로운 카트리지 주소 감사

확장이 수행된 후에는 다음 절차를 완료합니다.

1. ACS 및 포트를 오프라인으로 전환(*vary*)합니다.
2. 다음 중 하나를 수행합니다.
   - 모듈을 추가하거나 제거합니다.
   - 용량 변경
   - 라이브러리 다시 분할
3. ACS 및 포트를 온라인으로 전환(*vary*)합니다.
4. 다음 절차 중 하나를 사용해서 ACSLS 데이터베이스에서 구성을 업데이트합니다.
   - 동적:

    - *config acs* acs_id 또는 config lsm lsm_id
    - 라이브러리를 감사(*audit*)합니다.
- 정적:
    - ACSLS 작동 중지: acsss disa;ble
    - ACSLS 구성 업데이트: *acsss_config*
    - ACSLS 작동: *acsss enable*
    - 라이브러리를 감사(*audit*)합니다.

ACSLS 이중 TCP/IP

ACSLS 서버와 라이브러리 사이의 이중 TCP/IP 연결은 SL3000에 대해 구입할 수 있는 옵션입니다.

SL3000 요구 사항

- 시스템 관리자 및 네트워크 관리자와 협력하여 현재 네트워크 환경을 이해하고 필요한 모든 IP 주소를 미리 식별합니다.
- 시스템 관리자와 협력하여 네트워크 인터페이스를 구성하거나 올바르게 구성되었는지 확인합니다.

이 기능에 대한 자세한 내용은 “개요 ” 및 *StreamLine Modular Library System Dual TCP/IP Feature* 문서를 참조하십시오.

주:

이중 TCP/IP 연결 구현 절차는 SL8500 및 SL3000 라이브러리 모두 동일합니다.

부록 E

부록 E. SL500의 ACSLS 지원

SL500 라이브러리는 SCSI에 연결된 단일 LSM 라이브러리입니다. 제어 경로 작업을 위해서는 ACSLS에 대해 광 섬유 채널 또는 LVD SCSI 연결이 필요합니다. 이 장에서는 SL500 라이브러리에 대한 ACSLS 지원에 대해 설명합니다.

ACSLS 연결

ACSLS 서버에는 SL500 라이브러리의 인터페이스 카드(LVD SCSI 또는 광 섬유)와 호환되는 HBA(호스트 버스 어댑터)가 필요합니다. ACSLS는 또한 인터페이스 카드 대신 SL500의 브리지 드라이브와 통신할 수 있습니다.

*$ACS_HOME/install/install_scsi_sol.sh*를 실행하면 시스템에서 JNI 카드 또는 LSILogic 카드와 같은 새로운 하드웨어가 있는지 프로브합니다. 이러한 HBA 카드가 시스템에 있지만, 해당 드라이버가 배치되지 않은 경우에는 적절한 드라이버 패키지를 찾아서 설치하기 위한 권한을 요청하는 프롬프트가 루틴에 표시됩니다.

주:

SL500은 분할 가능하지만, ACSLS에서는 분할된 SL500이 지원되지 않습니다.

주:

제어 경로용 SAS 브리지 드라이버는 지원되지 않습니다. 브리지 드라이브에 대한 광 섬유 채널 연결만 지원됩니다.

ACSLS 및 SL500 라이브러리 차이

이 절에서는 ACSLS와 SL500 라이브러리 차이에 대해 설명합니다.

라이브러리 구성

- ACS, LSM - SL500 라이브러리는 단일 LSM이 포함된 ACS입니다(1 ACS 및 1 LSM).
- SL500에서는 현재 전달 포트가 지원되지 않습니다.
- SL500에는 기본 모듈이 하나 이상 포함되며 최대 4개의 확장 모듈을 포함할 수 있습니다.
- ACSLS에서 각 SL500 모듈은 패널로 고려됩니다.
- ACSLS에서 관리되는 SL500 라이브러리는 ACSL 인터페이스를 통해 6자 *volser* 레이블을 보고합니다.

라이브러리 위치 식별

이 절에서는 라이브러리 위치에 대해 설명합니다.

주소 지정 체계

ACSLS 주소 지정은 ACS, LSM, 패널, 행, 열입니다.

- SL500 주소 지정은 라이브러리, 모듈, 행, 열 내의 LSM입니다.
- ACSLS 주소는 0부터 시작됩니다. 패널 0, 행 0 및 열 0부터 시작됩니다.
- SL500 라이브러리는 1부터 시작합니다. 모듈 1, 행 1 및 열 1부터 시작됩니다.
- SL500 라이브러리와 ACSLS 내부 주소 사이의 일관성을 보장하기 위해 ACSLS는 패널 0, 행 0 및 열 0에 대해 위치 표시자를 정의합니다.

패널

- 패널 0 = 설치되지 않음
- 패널 1 = 기본 모듈
- 패널 2-5 = 확장 모듈

행 번호

- 각 모듈(패널) 내에서 행은 1-12로 번호가 지정됩니다.
- 행은 위에서 아래로 연속해서 번호가 지정됩니다.
- 번호 지정은 SL500과 ACSLS 모두 동일합니다.

열 번호

- 1-11(최대)로 번호가 지정됩니다.
- 현재 SL500 구성은 9개 열로 제한됩니다.
- 열 1-4는 전면에서 후면으로 왼쪽에 있습니다.
- 열 5-8은 후면에서 전면으로 오른쪽에 있습니다.
- 열 9는 드라이브가 있는 후면 벽면입니다.
- 번호 지정은 SL500과 ACSLS 모두 동일합니다.

드라이브 주소

SL500 주소 지정은 LSM, 모듈, 행, 열입니다.

- 행은 드라이브 번호 또는 슬롯 번호와 동일합니다.
- 드라이브 번호는 기본 모듈의 경우 1-2이고 확장 모듈의 경우에는 1-4입니다.
- 열 번호는 드라이브에 대해 항상 9입니다.

ACSLS 주소 지정 = ACS, LSM, 패널, 드라이브

- 드라이브 번호 = 행 또는 슬롯

표 E.1. 드라이브 주소 지정 예

모듈	**ACSLS** 드라이브 식별자	**SL500** 드라이브 주소
기본 모듈	0, 0, 1, 1	0, 1, 1, 9

모듈	ACSLS 드라이브 식별자	SL500 드라이브 주소
	0, 0, 1, 2	0, 1, 2, 9
드라이브 확장 모듈	0, 0, 2, 1	0, 2, 1, 9
	0, 0, 2, 2	0, 2, 2, 9
	0, 0, 2, 3	0, 2, 3, 9
	0, 0, 2, 4	0, 2, 4, 9
다음 확장 모듈	0, 0, 3, 1	0, 3, 1, 9

ACSLS 제한 사항

- SL500 라이브러리의 최대 개수 = 31.
- LSM 최대 개수 = 127.
- SL500 라이브러리의 최대 패널 수 = 5.
- 확장 모듈은 각각 패널을 하나씩 추가합니다.
- 기본 모듈만 있는 SL500에는 패널이 하나만 있습니다.
- SL500 펌웨어에서는 레이블이 지정되지 않은 카트리지가 지원되지 않습니다. 즉, ”venter” 명령을 사용할 수 없습니다.
- 분할된 SL150 라이브러리는 지원되지 않습니다.
- 제어 경로용 SAS 브리지 드라이버는 지원되지 않습니다.

SL500 라이브러리 설정 구성

SL500에서는 ACSLS에 영향을 주는 새로운 구성 설정이 지원됩니다. 다음과 같습니다.

- CAP 스토리지 – CAP 셀
  - 확장 모듈에서만 허용됩니다.
  - 기본 모듈 CAP는 항상 CAP로 사용됩니다.
  - 모듈별 기준으로 구성됩니다.
  - 스토리지로 사용되는 모든 CAP는 넣기/꺼내기 작업에 사용할 수 없습니다.
- 예약된 셀은 라이브러리 전용(진단, 청소 카트리지)입니다.
  - 기본 모듈에만 영향을 줍니다.
  - 번호(n)는 0-9(기본 모듈의 최대 행)로 구성할 수 있습니다.
  - 열 1에 있는 처음(n) 셀이 예약됩니다.

    이러한 셀은 ACSLS에서 액세스할 수 없는 것으로 고려됩니다.

SL500 CAP 동작

기본 모듈의 CAP에는 1개의 5슬롯 매거진이 포함됩니다. 드라이브 확장 모듈이 추가된 경우 드라이브 확장 모듈의 CAP에는 2개의 5슬롯 매거진이 포함됩니다. 확장 모듈의 경우에는 확장 모듈별로 10개의 추가 스토리지 슬롯에 대해 CAP를 구성할 수 있습니다. 또한 다음 사항을 고려하십시오.

- 브리지 드라이브를 통해 연결된 SL500의 CAP는 다른 ACSLS 인스턴스가 라이브러리 관리 권한을 가질 때 잠긴 상태가 될 수 있습니다. 이 문제에 대한 자세한 내용 및 해결 방법은 SL500 부록의 “CAP(메일슬롯) 꺼내기 중 열지 않음 ”을 참조하십시오.
- 기본 모듈에 있는 것 이외의 CAP 셀은 CAP 셀 또는 스토리지 셀로 구성할 수 있습니다.
- 기본 모듈 CAP는 항상 CAP로 사용됩니다.
- 카트리지 넣기 및 꺼내기를 위해 SL500에는 하나의 활성 CAP가 포함되어야 합니다.
- CAP가 여러 개 있는 경우 SL500 라이브러리는 항상 CAP가 있는 ACSLS에 보고를 수행합니다.
- CAP가 스토리지 셀로 구성되지 않은 한 모든 CAP 섹션에 대한 도어는 하나의 장치로 잠금 및 잠금 해제됩니다.

  모듈이 CAP 셀을 스토리지 셀로 사용하도록 구성된 경우, CAP 섹션은 CAP에 액세스하는 라이브러리 작업의 영향을 받지 않습니다.
- ACSLS가 CAP를 감사할 때는 모든 셀을 조사합니다.

라이브러리 감사

다음과 같은 경우에는 항상 감사를 수행해야 합니다.

- 새 라이브러리인 경우
- 하나 이상의 모듈이 추가, 제거 또는 스왑된 경우
- 카트리지가 도어를 통해 수동으로 추가 또는 제거된 경우
- 라이브러리 구성 설정이 변경된 다음

라이브러리 자체 감사가 수행되는 경우:

- 라이브러리 전원을 켜거나 다시 초기화하는 경우
- 도어를 열고 닫는 경우

자체 감사가 완료된 다음에는 ACSLS를 사용해서 데이터베이스를 업데이트해야 합니다.

새 라이브러리인 경우

새 라이브러리는 라이브러리의 실제 내용을 데이터베이스와 동기화하기 위해 감사를 수행해야 합니다.

모듈을 추가, 제거 또는 스왑한 후

SL500 모듈이 추가, 제거, 교체된 다음 또는 라이브러리를 처음으로 감사할 경우에는 다음 단계를 수행합니다.

1. ACSLS에 대해 SL500 라이브러리를 오프라인으로 전환(*vary*)하고 라이브러리 전원을 끕니다.
2. 모듈 추가와 같은 라이브러리 변경 작업을 수행합니다.
3. SL500 라이브러리 전원을 켜거나 다시 초기화합니다.

4. 다음 명령을 사용해서 ACSLS를 종료합니다(실행 중인 경우).

   ```
   acsss disable
   ```
5. 다음 명령을 사용해서 *acsss_config*를 실행합니다.
6. 다음 명령을 실행하여 ACSLS를 시작합니다.

   ```
   acsss enable
   ```
7. 영향을 받는 SL500 패널에 대해 *audit*를 수행합니다.

도어를 통해 카트리지를 수동으로 추가 또는 제거한 후

라이브러리 도어를 통해 LSM에서 카트리지를 수동으로 추가 또는 제거한 경우에는 데이터베이스를 동기화해야 합니다. “query pool ”에 설명된 대로 *audit*를 수행합니다.

ACSLS *cmd_proc*를 사용해서 CAP를 통해 LSM에서 카트리지를 넣거나 꺼낸 경우에는 데이터베이스가 자동으로 업데이트됩니다.

라이브러리 구성 설정을 변경한 후

라이브러리 구성 설정을 변경한 경우에는 *acsss_config*를 사용해서 ACSLS를 다시 구성하고 라이브러리를 감사(*audit*)합니다. 다음 라이브러리 설정을 변경한 후 라이브러리를 다시 구성하고 감사(*audit*)합니다.

- 예약된 셀 수
- 스토리지 셀로 사용되는 CAP

다음 단계를 따릅니다.

1. ACSLS에 대해 SL500 라이브러리를 오프라인으로 전환(*vary*)하고 라이브러리 전원을 끕니다.
2. 라이브러리 변경을 수행합니다.
3. SL500 라이브러리를 다시 초기화합니다.
4. 다음 명령을 사용해서 ACSLS를 종료합니다(실행 중인 경우).

   ```
   acsss disable
   ```
5. *acsss_config*를 실행합니다.
6. 다음 명령을 실행하여 ACSLS를 시작합니다.

   ```
   acsss enable
   ```
7. 영향을 받는 SL500 패널에 대해 *audit*를 수행합니다.

부록 F

부록 F. SL150의 ACSLS 지원

SL150 라이브러리는 SCSI 연결 단일 LSM 라이브러리입니다. 이 장에서는 SL150 라이브러리에 대한 ACSLS 지원에 대해 설명합니다.

ACSLS 연결

ACSLS 서버에는 SL150 라이브러리에서 브리지 드라이브와 호환되는 FC HBA(호스트 버스 어댑터)가 필요합니다.

*`$ACS_HOME/install/install_scsi_sol.sh`*를 실행하면 시스템에서 JNI 카드 또는 LSILogic 카드와 같은 새로운 하드웨어가 있는지 프로브합니다. 이러한 HBA 카드가 시스템에 있지만, 해당 드라이버가 배치되지 않은 경우에는 적절한 드라이버 패키지를 찾아서 설치하기 위한 권한을 요청하는 프롬프트가 루틴에 표시됩니다.

주:

SL150은 분할 가능하지만, ACSLS에서는 분할된 SL150이 지원되지 않습니다.

주:

제어 경로용 SAS 브리지 드라이버는 지원되지 않습니다. 브리지 드라이브에 대한 광 섬유 채널 연결만 지원됩니다.

ACSLS 및 SL150 라이브러리 차이

이 절에서는 ACSLS와 SL150 라이브러리의 차이점에 대해 설명합니다.

라이브러리 구성

- ACS, LSM - SL150 라이브러리는 단일 LSM이 포함된 ACS입니다(1개의 ACS와 1개의 LSM).
- SL150에서는 현재 전달 포트가 지원되지 않습니다.
- SL150에는 하나 이상의 기본 모듈이 포함되며, 최대 9개의 확장 모듈을 포함할 수 있습니다.
- ACSLS는 각 SL150 모듈을 패널로 고려합니다.
- ACSLS에서 관리되는 SL150 라이브러리는 ACSLS 인터페이스를 통해 6자 *`volser`* 레이블을 보고합니다.

라이브러리 위치 식별

이 절에서는 라이브러리 위치에 대해 설명합니다.

주소 지정 체계

- ACSLS 주소 지정은 ACS, LSM, 패널, 행 및 열입니다.
- SL150 주소 지정은 스토리지 셀에 대한 모듈, 측면, 행, 열입니다.
- ACSLS 주소는 0부터 시작됩니다. 패널 0, 행 0 및 열 0부터 시작됩니다.
- SL150 라이브러리는 1부터 시작됩니다. 모듈 1, 행 1 및 열 1부터 시작됩니다.
- SL150 라이브러리와 ACSLS 내부 주소 사이의 일관성을 보장하기 위해 ACSLS는 패널 0, 행 0 및 열 0에 대해 위치 표시자를 정의합니다.

패널

- 패널 0 = 설치되지 않음
- 패널 1 = 기본 모듈
- 패널 2-10 = 확장 모듈

행 번호

- 각 모듈(패널) 내에서 행은 1-3까지 번호가 지정됩니다.
- 행은 위에서 아래로 연속해서 번호가 지정됩니다.
- 번호 지정은 SL150과 ACSLS 사이에 동일합니다.

열 번호

SL150 주소 지정:

- 모듈, 측면, 행, 열
- 열 1-5, 각 측면의 앞에서 뒤까지

ACSLS 주소 지정:

- 1~10까지 번호가 지정됩니다.
- 열 1-5는 왼쪽 측면의 앞에서 뒤까지입니다.
- 열 6-10은 오른쪽 측면의 앞에서 뒤까지입니다.

드라이브 주소

SL150 주소 지정:

- 모듈, 위치
- 모듈 1 = 기본
- 모듈 2-10 = 확장
- 위치 = 위에서 아래로

ACSLS 주소 지정:

- ACS, LSM, 패널, 드라이브
- 패널 = 모듈
- 드라이브 번호 = 1-20, 위에서 아래로 번호 지정

표 F.1. 드라이브 주소 지정 예

모듈	**ACSLS** 드라이브 식별자	**SL150** 드라이브 주소
기본 모듈	0, 0, 1, 1	1, T
	0, 0, 1, 2	1, B
확장 모듈	0, 0, 2, 3	2, T
	0, 0, 2, 4	2, B
확장 모듈	0, 0, 3, 5	3, T
	0, 0, 3, 6	3, B
확장 모듈	0, 0, 4, 7	4, T
	기타	기타

ACSLS 제한 사항

- SL150 라이브러리 최대 개수 = 31.
- LSM 최대 개수 = 127.
- SL150 라이브러리에 있는 모듈(패널) 최대 개수 = 10.
- 확장 모듈은 각각 패널을 하나씩 추가합니다.
- 기본 모듈만 있는 SL150에는 패널이 하나만 있습니다.
- SL150 펌웨어에서는 레이블이 지정되지 않은 카트리지가 지원되지 않습니다. 즉, *venter* 명령을 사용할 수 없습니다.
- 분할된 SL150 라이브러리는 지원되지 않습니다.
- 제어 경로용 SAS 브리지 드라이버는 지원되지 않습니다.

SL150 라이브러리 설정 구성

SL150에서는 ACSLS에 영향을 주는 구성 설정이 지원됩니다.

- 분할

  분할된 SL150 라이브러리는 ACSLS에서 지원되지 않습니다.
- 브리지 드라이브

  ACSLS에서는 FC 제어 경로만 지원됩니다. 제어 경로용 SAS 브리지 드라이버는 지원되지 않습니다.
- 메일슬롯 구성

  기본 모듈의 한 측면에서 스토리지 슬롯을 표준 메일슬롯의 확장으로 사용할 수 있습니다.
  - 필요한 ACSLS 설정: 표준 메일슬롯(4개 슬롯)
- 드라이브 요소 주소 지정 모드

라이브러리에 빈 드라이브 슬롯이 있을 때 SCSI 요소 주소가 보고되는 방법을 제어할 수 있습니다.
  - 필요한 ACSLS 설정: Address Only Installed Drives
- 예약된 셀

  예약된 셀은 라이브러리 전용입니다(진단 및 청소 카트리지).
  - 번호(n)는 0부터 3까지 구성할 수 있습니다(기본 모듈의 최대 행 수).
  - 기본 모듈에만 영향 - 열 1에 있는 처음 (n)개 셀이 예약됩니다.
  - 이러한 셀은 ACSLS에서 액세스할 수 없는 것으로 고려됩니다.

SL150 CAP 동작

SL150은 CAP에 대해 "메일슬롯"이라는 용어를 사용합니다. 기본 모듈의 메일슬롯에는 1개의 4슬롯 매거진이 포함됩니다.

모든 메일슬롯 작업은 라이브러리 터치 스크린 또는 BUI를 사용해서 수행됩니다.

- SL150은 카트리지 넣기 및 꺼내기를 위해 하나의 활성 메일슬롯을 포함해야 합니다.
- ACSLS가 메일슬롯을 감사할 때는 모든 셀을 검사합니다.

CAP(메일슬롯) 꺼내기 중 열지 않음

이전에 라이브러리를 관리하는 ACSLS 인스턴스에 의해 SL150 CAP(메일슬롯)가 잠긴 상태로 유지될 경우 *eject*(또는 *enter*) 작업 중 문제가 발생할 수 있습니다. 이러한 문제가 발생할 수 있는 시나리오에는 새로운 ACSLS 서버로의 마이그레이션 또는 HA 페일오버 이벤트가 포함됩니다.

메일슬롯이 호스트에 의해 잠긴 경우, 라이브러리 터치 스크린에 다음 항목이 표시됩니다.

State: Locked by SCSI Prevent Media

ACSLS는 일반적으로 자동 CAP를 잠김 상태로 유지하지 않기 때문에 이 문제는 자동 모드의 CAP에서 발생할 가능성이 더 적습니다. 수동 모드 CAP 또는 오프라인으로 전환된 라이브러리에 있는 CAP(모든 모드)에서는 이 문제가 발생할 가능성이 높습니다.

문제 완화:

이 문제가 발생하지 않도록 하려면 다음과 같은 절차가 권장됩니다.

- SL150 라이브러리의 제어를 새로운 ACSLS 서버로 마이그레이션하는 경우(비HA 구성):
  - 자동 모드 CAP(권장):

    원본 서버에서 ACSLS를 작동 중지하기 전에 라이브러리를 오프라인으로 전환하지 않습니다. 라이브러리가 온라인일 때 ACSLS는 종료 시 CAP를 잠금 해제된 상태로 둡니다.

- 수동 모드 CAP:

  원본 서버에서 ACSLS를 종료하기 전에 CAP가 자동 모드로 작동하도록 설정합니다. 라이브러리를 오프라인으로 전환하지 않습니다.

- ACSLS HA 설치에서 라이브러리를 관리할 때는 자동 모드로 CAP를 작동하는 것이 가장 좋습니다.
  - 자동 모드 CAP(권장):

    온라인 라이브러리에는 필요한 작업이 없습니다(CAP가 일반적으로 잠금 해제됨). 페일오버 이벤트 중 라이브러리가 오프라인이면 아래 단계를 수행해서 브리지 드라이브를 다시 시작하여 CAP를 잠금 해제합니다.
  - 수동 모드 CAP:

    아래 단계에 따라 브리지 드라이브를 다시 시작하여 CAP를 잠금 해제합니다.

해결 방법:

이전 호스트에서 남겨진 잠금 상태는 SL150 BUI를 사용해서 브리지 드라이브를 다시 시작하여 해제할 수 있습니다. 브리지 드라이브 다시 시작에 대한 자세한 내용은 SL150 제품 설명서를 참조하십시오.

경고:

이 작업은 드라이브의 현재 데이터 경로 작업을 중단시킵니다. 브리지 드라이브 다시 시작은 데이터 경로 작업을 중단하지 않을 시간에 수행되도록 일정을 잡을 수 있습니다.

이 문제가 발생한 경우에는 잠금을 해제해야 합니다.

1. 브리지 드라이브에서 데이터 경로(읽기/쓰기) 작업이 수행되고 있지 않은지 확인합니다.
2. SL150 BUI를 사용해서 브리지 드라이브를 다시 시작합니다.

라이브러리 감사

감사를 수행하는 경우:

- 새 라이브러리인 경우
- 하나 이상의 모듈이 추가, 제거 또는 스왑된 경우
- 메일슬롯을 통해 카트리지가 수동으로 추가 또는 제거된 경우
- 라이브러리 구성 설정이 변경된 다음

라이브러리 자체 감사가 수행되는 경우:

- 라이브러리 전원을 켜거나 다시 초기화하는 경우
- 메일슬롯을 열고 닫은 경우

자체 감사가 완료된 다음에는 ACSLS를 사용해서 데이터베이스를 업데이트해야 합니다.

새 라이브러리인 경우

새 라이브러리는 라이브러리의 실제 내용을 데이터베이스와 동기화하기 위해 감사를 수행해야 합니다.

모듈을 추가, 제거 또는 스왑한 후

SL150 모듈이 추가, 제거 또는 스왑된 다음 또는 라이브러리를 처음으로 감사하는 경우에는 다음 단계를 수행합니다.

1. SL150 라이브러리를 ACSLS에 대해 오프라인으로 전환(*vary*)하고 라이브러리 전원을 끕니다.
2. 모듈 추가와 같은 라이브러리 변경 작업을 수행합니다.
3. SL150 라이브러리 전원을 켜거나 다시 초기화합니다.
4. 다음 명령을 사용해서 ACSLS를 종료합니다(실행 중인 경우).

   ```
   acsss disable
   ```
5. *acsss_config*를 실행합니다.
6. 다음 명령을 실행하여 ACSLS를 시작합니다.

   ```
   acsss enable
   ```
7. 영향을 받는 SL150 패널에 대해 *audit*를 수행합니다.

메일슬롯을 통해 카트리지를 수동으로 추가 또는 제거한 후

라이브러리 메일슬롯을 통해 LSM에서 카트리지를 수동으로 추가 또는 제거한 경우에는 데이터베이스를 동기화해야 합니다. “query pool ”에 설명된 대로 *audit*를 수행합니다.

ACSLS *cmd_proc*를 사용해서 메일슬롯을 통해 LSM에서 카트리지를 넣거나 꺼낸 경우에는 데이터베이스가 자동으로 업데이트됩니다.

라이브러리 구성 설정을 변경한 후

라이브러리 구성 설정을 변경한 경우에는 *acsss_config*를 사용해서 ACSLS를 다시 구성하고 라이브러리를 *audit*해야 합니다. 또한 예약된 셀 수를 변경한 후 라이브러리를 다시 구성하고 *audit*해야 합니다.

다음 단계를 따릅니다.

1. SL150 라이브러리를 ACSLS에 대해 오프라인으로 전환(*vary*)하고 라이브러리 전원을 끕니다.
2. 라이브러리 변경을 수행합니다.
3. SL150 라이브러리를 다시 초기화합니다.
4. 다음 명령을 사용해서 ACSLS를 종료합니다(실행 중인 경우).

   ```
   acsss disable
   ```

5. *acsss_config*를 실행합니다.
6. 다음 명령을 실행하여 ACSLS를 시작합니다.

   ```
   acsss enable
   ```
7. 영향을 받는 SL150 패널에 대해 *audit*를 수행합니다.

부록 G

부록 G. StorageTek 가상 테이프 라이브러리의 ACSLS 지원

ACSLS는 FalconStor 기반 VTL 제품 라인에 구현된 대로 일반 가상 라이브러리를 식별하는 새 LSM 유형(StorageTek VTL)을 지원합니다. StorageTek VTL 라이브러리는 다른 모든 단일 LSM, SCSI 연결 라이브러리와 같이 구성, 관리 및 작동될 수 있습니다.

ACSLS는 사용자에게 친근한 PRC 스타일 식별자(*panel, row, column*)를 사용하여 StorageTek VTL 구성을 나타냅니다. StorageTek VTL 라이브러리에 대한 ACSLS 또는 ACSAPI 요청을 실행할 때 이러한 PRC 스타일 식별자를 사용합니다. 여기에는 셀 식별자(*acs, lsm, panel, row, column*), 드라이브 식별자(*acs, lsm, panel, transport*) 및 CAP 식별자(*acs, lsm, cap*)가 포함됩니다.

지원되는 구성

StorageTek VTL 구성은 ACSLS에 대해 라이브러리가 구성될 때 *acsss_config.log* 파일에 기록됩니다. 이 파일은 일반적으로 진단 목적으로만 사용됩니다. *config* 유틸리티를 사용하면 동적 구성 시 패널 설명도 표시됩니다.

다음 표에서는 ACSLS에서 지원되는 VTL 구성에 대해 설명합니다.

스토리지 셀 수	시작 요소 주소	드라이브 수	시작 요소 주소	가져오기/내보내기 요소 수	시작 요소 주소
1-10,000	1000	1-100	500	1-20	10

- 스토리지 셀 수: 1-10,000
- 시작 요소 주소: 1000
- 드라이브 수: 1-100
- 시작 요소 주소: 500
- 가져오기/내보내기 요소 수: 1-20
- 시작 요소 주소: 10

다음에서는 라이브러리 구성 요소 지원에 대해 설명합니다.

- 패널

  지원되는 각 구성은 다음과 같이 일련의 패널로 표시됩니다.
  - 패널 0 - CAP만 포함함
  - 패널 1-10 - 스토리지 셀 및 드라이브를 포함함
  - 패널 11-50 - 스토리지 셀을 포함함
- CAP

단일 CAP가 지원됩니다. 이는 CAP 0으로 식별되고 최대 20개의 CAP 셀을 포함할 수 있습니다. CAP는 StorageTek VTL의 패널 0에 정의됩니다.

- 스토리지 셀

  스토리지 셀은 패널당 200개의 셀을 포함하는 패널로 구성됩니다. 가장 큰 구성인 총 10,000개 셀의 경우 200개의 셀이 포함된 스토리지 패널이 50개가 있는 것입니다.

  스토리지 셀은 StorageTek VTL의 패널 1-50에 정의됩니다.
- 드라이브

  드라이브는 패널당 최대 10개의 드라이브를 포함하는 패널로 구성됩니다. 가장 큰 드라이브 구성인 총 100개 드라이브의 경우 10개의 드라이브 패널이 있는 것입니다.

  드라이브는 StorageTek VTL의 패널 1-10에 정의됩니다.
- PTP

  전달 포트는 지원되지 않습니다.

VTL 동작

다음 동작은 특히 VTL에 적용됩니다.

- 넣기 작업

  ACSLS의 *Enter* 작업은 StorageTek VTL 라이브러리에 테이프 볼륨을 추가하는 데 유용하지 않습니다. VTL 라이브러리에서 새 볼륨을 검색하려면 해당 라이브러리에 대해 *audit*을 수행해야 합니다. 여기에는 VTL 가상 저장소에서 가져왔을 수도 있는 볼륨 검색이 포함됩니다.

  Enter 작업은 StorageTek VTL에 대해 명시적으로 거부되지 않습니다. *Enter* 작업을 시작하는 경우 이 작업을 간단하게 취소할 수 있습니다.
- 꺼내기 작업

  ACSLS에서 *Eject* 작업이 지원됩니다. 꺼낸 모든 볼륨은 이제 가상 저장소에 있으므로 VTL 제품에서 볼 수 있지만 이는 VTL 기능일 뿐입니다. ACSLS는 가상 저장소를 인식하지 않습니다.
- 마운트 및 마운트 해제 작업

  ACSLS *mount* 옵션(*read-only, bypass*) 및 *dismount* 옵션(*force*)은 수락되지만 StorageTek VTL 라이브러리에서는 무시됩니다. 또한 force 옵션을 사용하지 않는 일반 마운트 해제는 force 옵션이 지정된 것처럼 작동됩니다.

  주의:

  마운트 해제 작업을 수행할 때 데이터 경로 작업이 진행 중인 경우 가상 테이프 볼륨을 마운트 해제할 수 있으므로 주의하십시오.
- 가상 테이프 드라이브

VTL에서 가상 드라이브를 추가하거나 제거한 후 `config` 드라이브 유틸리티를 사용하여 ACSLS에서 드라이브 구성을 다시 검사합니다. VTL에서 가상 테이프 드라이브를 제거할 때 VTL 콘솔을 사용하여 드라이브를 지정 해제하는 것이 아니라 삭제해야 합니다. 가상 드라이브 지정 또는 지정 해제는 데이터 경로에 영향을 주며 ACSLS에는 영향을 주지 않습니다.

ACSLS에 대한 VTL 구성

이 절에서는 ACSLS에 대한 VTL을 구성할 때 필수 조건 및 설치 절차에 대해 설명합니다.

필수 조건

ACSLS에 대한 VTL을 구성하기 전에 다음 필수 조건을 충족해야 합니다.

- ACSLS가 설치되어 있고 광 섬유 채널 HBA를 포함합니다.
- VTL 시스템이 설치되어 있습니다.
- 시스템 간의 광 섬유 채널 연결이 있습니다. ACSLS 개시자 포트가 VTL 대상 포트에 연결되어 있어야 합니다.

설치

ACSLS에 대한 VTL을 구성하려면 다음 단계를 수행합니다.

1. VTL 콘솔을 사용하여 가상 라이브러리(StorageTek-VTL)를 만듭니다.

   주:
   - ACSLS에서 사용할 가상 테이프 라이브러리를 만들 때 해당 라이브러리에 대한 테이프 식별자는 ACSLS에서 지원되는 일반 6자 볼륨 레이블 형식과 일치해야 합니다.
   - VTL 콘솔에서는 ACSLS에서 지원하는 제한(슬롯 10,000개, 드라이브 100개)을 초과하는 StorageTek VTL 라이브러리를 만들 수 있습니다. 그러나 ACSLS에 대한 지원되지 않는 구성을 구성하려는 경우 구성 요청이 실패합니다.
   - 새 StorageTek VTL 라이브러리를 만들 때 VTL 콘솔은 678보다 큰 슬롯 개수 설정에 대한 경고를 표시합니다. 이 경고 제한은 L700 용량을 기반으로 하며 StorageTek VTL 라이브러리 모델에서 무시할 수 있습니다.

   절차는 *StorageTek Virtual Tape VTL Plus 2.0 (Update 2) Library Software Configuration Guide*를 참조하십시오.
2. ACSLS에 VTL 라이브러리를 지정합니다.

   절차는 *StorageTek Virtual Tape VTL (Plus 2.0 Update 2) Library Software Configuration Guide*를 참조하십시오.
3. SCSI mchanger 장치 드라이버를 만듭니다.

   ACSLS는 자동으로 StorageTek VTL 라이브러리를 검색하고 해당 `/dev/mchanger` 항목을 만들 수 있습니다. 이러한 검색 작업은 설치 시 `install_scsi_sol.sh` 스크립트를 실행할 때 수행됩니다.

"라이브러리 하드웨어에 대한 연결 설치 "를 참조하십시오.

4. ACSLS에 대한 VTL을 구성합니다.
   - VTL 구성 요소가 작동되면 ACSLS에 대한 VTL을 구성합니다.
   - 다음 명령을 사용합니다.

     ```
     acsss_config
     ```

   자세한 내용은 "CSI 조정 변수 설정" 또는 `config acs new`("새 ACS 추가 " 참조)를 참조하십시오.
5. VTL 라이브러리를 감사합니다.

   ACSLS에 대한 StorageTek VTL 라이브러리를 구성한 후 가상 테이프 볼륨을 검색하고 액세스할 수 없는 셀을 식별(패널에는 부분적으로만 액세스할 수 있음)하려면 `audit`를 수행해야 합니다.
6. `display lsm * -f type serial_num` 명령을 사용하여 VTL 구성을 확인합니다.

부록 H

부록 H. 논리적 라이브러리 지원

이 부록에서는 논리적 라이브러리 지원에 대해 설명합니다.

논리적 라이브러리 정보

ACSLS 물리적 라이브러리 구성의 일부는 SAN에서 대상 장치로 작동할 수 있는 논리적 라이브러리로 SCSI 클라이언트에 표시될 수 있습니다. 실제로 논리적 라이브러리는 테이프 드라이브와 볼륨을 포함하여 특정 사용자 지정 리소스가 있는 물리적 라이브러리의 일부 정의된 부분입니다. 논리적 라이브러리는 SCSI(광 섬유 연결) 인터페이스를 통해 클라이언트 응용 프로그램 소프트웨어에 제공됩니다.

광 섬유 채널 HBA 포트는 일반적으로 개시자 모드로 작동하며, 디스크 드라이브, 테이프 드라이브 또는 매체 교환기 장치와 같은 원격 대상 장치에 대해 SCSI 요청을 실행할 수 있습니다. 논리적 라이브러리를 사용하려면 대신 대상 모드에서 작동할 하나 이상의 광 섬유 포트를 구성해야 합니다.

install.sh 또는 getHba.sh 유틸리티를 실행할 때 대상 모드 작업에 대해 하나 이상의 FC 포트를 선택하거나 대상 모드 포트를 다시 개시자 모드로 설정할 수 있습니다. 변경사항을 적용하려면 재부트해야 합니다.

논리적 라이브러리는 적격한 모든 물리적 라이브러리를 원형 ACS로 사용해서 만들 수 있습니다. 원형 ACS는 논리적 라이브러리가 생성되는 물리적 라이브러리를 식별합니다. 원형 ACS:

- ACSLS 라이브러리 서버에 구성되어야 하지만, 논리적 라이브러리를 만들기 위해 온라인일 필요는 없습니다.
- 전체 물리적 ACS일 수도 있고, 라이브러리 자체에서 물리적 분할이 지원될 경우 ACS의 물리적 분할 영역일 수 있습니다.

논리적 라이브러리를 만들거나, 관리 또는 삭제하려면 ACSLS GUI(그래픽 사용자 인터페이스) 또는 lib_cmd CLI(명령줄 인터페이스)를 사용할 수 있습니다.

이점

논리적 라이브러리는 다음을 수행할 수 있게 해줍니다.

- 물리적 라이브러리를 논리적 라이브러리로 분할합니다.

  논리적 라이브러리는 구분된 라이브러리인 것처럼 클라이언트 응용 프로그램에서 관리 및 사용할 수 있습니다.

ACSLS는 특정 물리적 스토리지 위치에 연결되지 않은 유연한 분할 메커니즘을 제공합니다. 논리적 라이브러리는 볼륨 및 드라이브로 정의되며 ACSLS는 논리적 위치를 사용해서 이를 클라이언트에 제공합니다. 논리적 라이브러리는 패널 또는 레일 경계와 같은 ACS 내부의 물리적 부분으로 제한되지 않습니다.

논리적 라이브러리는 또한 중단 가능성이 있는 변경사항으로부터 클라이언트를 보호합니다. ACSLS가 볼륨에 대해 새로운 물리적 홈 셀을 선택하여, 호환되는 드라이브와 가깝게 유지하거나, 기존 논리적 라이브러리가 확장될 때, 클라이언트 응용 프로그램은 영향을 받지 않습니다.

- 전체 8자 바코드를 *volser*로 보고합니다.

  논리적 라이브러리를 만들 때 클라이언트에 대한 볼륨 레이블 형식을 지정할 수 있습니다(예: 6자, 8자 접두어, 8자 접미어 또는 모두). 8자 바코드에는 매체 도메인과 *volser* 뒤 또는 앞의 유형이 포함됩니다.

  주:

  6자 이상의 볼륨 레이블은 ACSAPI 클라이언트에서 액세스할 수 없습니다.

- 드라이브 및 볼륨에 대한 클라이언트 액세스를 관리합니다.

  논리적 라이브러리에 지정된 드라이브 및 볼륨은 라이브러리를 사용 중인 클라이언트에서만 액세스할 수 있으며, 다른 FC 또는 ACSAPI 클라이언트에 표시되지 않습니다. 하지만 ACSLS GUI 및 *cmd_proc*는 이러한 인터페이스가 시스템 관리자 용량으로 작동하기 때문에 이를 볼 수 있습니다.

- 다중 개시자 지원을 제공합니다.

  논리적 라이브러리는 여러 클라이언트(특히 여러 개시자 포트)에서 액세스하도록 설정할 수 있습니다. 이러한 목적은 여러 클라이언트의 논리적 라이브러리 동시 작업을 허용하기 위한 것이 아니라 클라이언트측에서 중복성을 지원하기 위한 것입니다. 언제라도 하나의 클라이언트 시스템이 하나의 논리적 라이브러리를 작동해야 합니다.

  여러 개시자를 허용할 경우, 단일 클라이언트 시스템은 다중 FC HBA 또는 포트를 통해 지정된 논리적 라이브러리에 액세스할 수 있습니다. 클라이언트 환경에서 페일오버 기능이 지원될 경우, 논리적 라이브러리는 새로운 활성 클라이언트 시스템에서 즉시 액세스할 수 있습니다.

  논리적 라이브러리에 대한 모든 Unit Attention 또는 Check Condition이 발생할 경우 구성된 각 연결에 제공됩니다.

제한 사항

논리적 라이브러리:

- ACSLS SCSI 매체 교환기 클라이언트 인터페이스를 사용해서 클라이언트에서만 액세스할 수 있습니다. 레거시 ACSAPI를 사용하는 클라이언트에는 제공되지 않습니다.
- 두 개 이상의 물리적 ACS(또는 물리적 분할 영역)로 확장될 수 없습니다.

- 물리적 ACS에서 특정 스토리지 셀을 보존할 수 없습니다. 논리적 및 물리적 분할을 결합해도 비슷한 결과를 얻을 수 있습니다.
- 대상 모드 FC 포트는 fcinfo와 같은 특정 Solaris 명령의 출력에 더 이상 포함되지 않을 수 있습니다.

논리적 라이브러리 만들기

논리적 라이브러리를 만들기 위해 ACSLS GUI 또는 *lib_cmd*를 사용하기 전에 *acsss_config* 또는 *config acs* 유틸리티를 사용해서 물리적 라이브러리를 만들어야 합니다. 또한 논리적 라이브러리를 만들기 전에 *audit*를 수행해야 합니다.

논리적 라이브러리를 만들 때는 다음을 수행합니다.

- 물리적 ACS 지정
- 논리적 라이브러리 속성 지정
- 하나 이상의 물리적 드라이브 지정
- 하나 이상의 물리적 볼륨 지정
- 논리적 라이브러리에 대한 클라이언트 액세스 지정

물리적 ACS 지정

논리적 라이브러리에 대한 원형 ACS를 지정할 때는 이후 단계에서 논리적 라이브러리에 지정할 수 있는 드라이브 및 볼륨도 자동으로 제한됩니다. 해당 ACS에 있고 다른 논리적 라이브러리에 아직 지정되지 않은 드라이브 및 볼륨만 사용할 수 있습니다.

논리적 ACS 번호는 논리적 라이브러리가 생성될 때 ACSLS에 의해 지정됩니다. 논리적 라이브러리의 ACS 번호 범위는 n001-n999입니다. 설명:
n = (1 + 원형 물리적 ACS ID).

예:

- 물리적 ACS 0에서 지원되는 논리적 라이브러리가 1001인 경우
- ACS 4에서 지원되는 논리적 라이브러리가 5001인 경우

논리적 라이브러리의 속성 지정

논리적 라이브러리에는 다음 속성을 지정할 수 있습니다.

- 논리적 라이브러리의 고유 이름
- 용량

  용량은 언제라도 논리적 라이브러리에서 액세스할 수 있는 최대 볼륨 수입니다(인벤토리에 포함할 수 있는 스토리지 요소 수). 최소값은 0이고 최대값은 64,536입니다. 기본값은 없습니다.

용량을 사용하면 논리적 라이브러리의 크기를 제한하거나(실제 물리적 용량보다 적은 한도로 설정), 물리적 용량을 초과 할당(실제 사용 가능한 것보다 많은 한도로 설정)할 수 있습니다.

◦ 가져오기/내보내기 셀 수

최소값은 2이고 최대값은 400입니다. 기본값은 2입니다.

◦ 드라이브 슬롯 수

이 개수는 지정할 수 있는 최대 드라이브 수입니다. 최소값은 0이고 최대값은 500입니다. 기본값은 없습니다.

◦ 레이블 형식

현재 물리적 라이브러리에서 긴 볼륨 레이블 지원은 라이브러리 펌웨어 및 구성에 따라 달라집니다.

하지만 논리적 라이브러리는 전체 바코드(매체 도메인 및 유형이 접미어로 지정된 6자의 volser 문자)를 보고할 수 있습니다. 이 기능은 8자 volser 문자(예: xxxxxxL4)를 제공합니다. 논리적 라이브러리는 매체 도메인 및 유형을 volser 끝에 추가하거나 volser 앞에 접두어로 지정하도록(예: L4xxxxxx) 구성할 수 있습니다.

지원되는 레이블 형식은 6자, 8자 접미어(기본값) 및 8자 접두어입니다.

하나 이상의 물리적 드라이브 지정

원형 ACS에 설치 및 구성된 모든 물리적 드라이브는 논리적 드라이브에 추가할 수 있습니다. 논리적 라이브러리에 대한 드라이브 유형은 제한이 없습니다.

논리적 라이브러리에 할당할 수 있는 물리적 드라이브:

- ACSAPI 클라이언트에서 액세스할 수 없습니다.

  물리적 라이브러리는 논리적 라이브러리에 할당되지 않은 모든 드라이브 및 볼륨과 함께 ACSAPI 클라이언트에서 액세스할 수 없는 상태로 유지됩니다.
- 배타적으로 할당됩니다.

  논리적 라이브러리 간에 공유될 수 없습니다.

논리적 라이브러리를 만들 때 지정된 드라이브 슬롯 수보다 많은 물리적 드라이브를 논리적 라이브러리에 지정할 수 없습니다. 하지만 논리적 라이브러리가 비어 있는 드라이브 슬롯을 포함할 수 있으므로, 더 적은 수의 드라이브를 지정하는 것은 가능합니다.

주:

잠긴 드라이브는 논리적 라이브러리에 지정할 수 없습니다. (잠긴 드라이브는 이미 다른 ACSAPI 클라이언트에서 사용 중입니다.)

언제라도 드라이브 슬롯 수를 늘리거나 줄일 수 있습니다. 슬롯 수를 현재 지정된 물리적 드라이브 수보다 낮은 값으로 줄이려면 먼저 일부 드라이브를 지정 해제해야 합니다.

하나 이상의 물리적 볼륨 지정

논리적 라이브러리를 만들 때는 언제라도 액세스할 수 있는 최대 볼륨 수인 해당 용량을 지정합니다. 볼륨을 논리적 라이브러리에 지정하여 액세스 가능하도록 만듭니다.

지정은 배타적으로 수행됩니다. 카트리지는 하나의 라이브러리에만 지정할 수 있습니다. 원형 ACS에 있고 아직 다른 논리적 라이브러리에 지정되지 않은 대부분의 카트리지는 지정 대상으로 선택할 수 있습니다.

다음과 같은 볼륨은 지정할 수 없습니다.

- 청소 카트리지(ACSLS 및 라이브러리 핸들 청소).
- 잠긴 볼륨(잠금은 ACSAPI 클라이언트에서 사용 중임을 나타냅니다.)
- 소유된 볼륨(소유자는 ACSAPI 클라이언트에서 사용 중임을 나타냅니다.)

논리적 라이브러리에서 액세스 가능한 볼륨 수가 최대 용량에 도달한 다음에는 더 이상 카트리지를 추가할 수 없습니다. 라이브러리 용량을 늘리거나 일부 사용된 공간을 비워야 합니다.

지정된 볼륨이 액세스할 수 없게 되면 논리적 라이브러리에서 공간이 해제됩니다. 이러한 경우는 지정된 볼륨이 다음과 같이 될 때 발생할 수 있습니다.

- FC 클라이언트에서 *eject*를 수행하도록 표시된 경우
- 라이브러리에서 꺼내진 경우
- ACSLS에서 없는 것으로 표시된 경우

지정되었지만 액세스할 수 없는 볼륨은, 다시 액세스 가능하게 된 경우(예: 꺼낸 볼륨을 원형 ACS에 다시 넣을 수 있음) 및 논리적 라이브러리에 사용 가능한 공간이 포함된 경우 자동으로 다시 활성화됩니다.

마지막으로, 볼륨을 지정 해제하여 논리적 라이브러리에서 공간을 비울 수 있습니다. 카트리지가 사용 중이 아닌 경우에는(예: 논리적 드라이브에 마운트할 수 없거나, *eject* 작업을 위해 선택할 수 없는 경우 등) 논리적 라이브러리에서 지정 해제할 수 있습니다.

주:

- 논리적 라이브러리에서 볼륨을 지정 해제해도 삭제 또는 꺼내기와 같은 유형의 작업을 암시하지는 않습니다. 볼륨 꺼내기도 지정 해제 작업을 암시하지 않습니다.
- 물리적 스토리지 셀은 미리 할당되거나 논리적 라이브러리에 지정되지 않습니다. 논리적 라이브러리에는 특정 셀이 아닌 특정 볼륨이 포함됩니다. 셀은 지정된 볼륨을 포함하는 경우 단지 논리적 라이브러리에 "속할" 뿐입니다.
- 원형 ACS를 공유하는 논리적 라이브러리 사이에 액세스 가능한 볼륨의 총 개수는 원형 ACS의 물리적 용량보다 클 수 없습니다.

논리적 라이브러리에 지정된 볼륨 수는 *lib_cmd* 또는 GUI의 Logical Library Detail Information 페이지에서 표시할 수 있습니다. 지정된 볼륨은 두 개의 개별 개수로 표시됩니다.

- 액세스 가능한 볼륨 - 지정되었고, 라이브러리에 있으며, 일반적으로 클라이언트용으로 사용할 수 있는 볼륨을 나타냅니다.
- 액세스 불가능한 볼륨 - 이 수치는 지정되었지만, 다음과 같은 이유로 인해 클라이언트용으로 사용할 수 없는 볼륨을 나타냅니다. 1) 볼륨이 비어 있거나 꺼내진 것으로 표시됨, 2) 볼륨이 FC 클라이언트에서 꺼내기 용으로 표시됨, 3) 볼륨을 다시 넣었지만 논리적 라이브러리가 가득 찼음, 4) 볼륨을 잘못된 물리적 라이브러리에 다시 넣음.

언제라도 스토리지 슬롯 수를 늘리거나 줄일 수 있습니다. 슬롯 수를 현재 지정된 물리적 볼륨 수보다 낮은 값으로 줄이려면 먼저 일부 볼륨을 지정 해제해야 합니다.

클라이언트 연결 지정

ACSLS SCSI 매체 교환기 인터페이스는 FC(광 섬유 채널) 연결을 통해 클라이언트가 논리적 라이브러리를 사용할 수 있도록 설정합니다. FC 클라이언트가 논리적 라이브러리에 액세스하도록 허용하려면 대상 모드에서 작동하도록 구성된 FC 포트가 하나 이상 필요합니다.

ACSLS 설치 프로세스 중 *`install.sh`*를 실행하는 동안 대상 포트를 구성한 경우, 논리적 라이브러리 만들기를 계속 수행합니다. 대상 포트를 아직 구성하지 않은 경우에는 *`getHba.sh`*를 사용해서 대상 포트를 구성합니다. 그런 다음 ACSLS를 종료하고 ACSLS 서버를 재부트합니다. 자세한 내용은 “대상 모드에서 FC 포트를 구성하는 방법”을 참조하십시오.

대상 포트를 구성한 다음에는 GUI를 통해 논리적 라이브러리를 만들거나 업데이트할 때 클라이언트 연결을 쉽게 관리할 수 있습니다. GUI 화면은 ACSLS 시스템에서 사용 가능한 대상 포트 목록을 제공하며, FC 연결을 통해 이러한 포트를 문의한 모든 클라이언트 목록을 제공합니다. ACSLS는 이러한 문의를 자동으로 감지하고 각 클라이언트의 WWN을 기억합니다.

논리적 라이브러리를 클라이언트에 지정하려면 GUI에서 적합한 상자를 선택합니다.

대상 포트를 구성한 다음에는 논리적 라이브러리를 만들거나 업데이트할 때 *`lib_cmd`* 또는 GUI를 사용해서 클라이언트 연결을 쉽게 관리할 수 있습니다. GUI 및 대화식 모드의 *`lib_cmd`* 모두 ACSLS 시스템에서 사용할 수 있는 대상 포트 목록을 제공합니다. 또한 FC 연결을 통해 이러한 포트를 문의한 모든 클라이언트 목록을 제공합니다. ACSLS는 이러한 문의를 자동으로 감지하고 각 클라이언트의 WWN을 기억합니다.

클라이언트에 논리적 라이브러리를 지정하려면 *`lib_cmd`*에서 *`create mapping`*을 사용하거나 GUI의 Manage Connections 페이지에서 적합한 상자를 선택합니다.

WWN 대신 친숙한 '별칭' 이름을 사용하여 각 클라이언트 시스템을 식별할 수 있습니다. 별칭을 지정하려면 *`lib_cmd`*에서 *`edit initiator`*를 사용하거나 GUI의 Manage Connections 페이지에서 '*`Edit Initiator Alias`*' 작업을 선택합니다.

빠른 로드

ACSLS 빠른 로드 설정은 대상 요소가 테이프 드라이브일 때 *`move`* 명령의 완료가 보고되는 방법을 제어합니다.

빠른 로드가 사용으로 설정되었으면 작업이 검증되고 ACSLS에서 수락된 다음, 카트리지 이동이 시작되기 전에 성공 상태가 반환됩니다. 이동 중에 일부 오류가 발생할 경우, ACSLS는 해당 정보를 보고하지 않습니다. 클라이언트는 볼륨이 로드되어 사용 가능한지 확인하고, 오류 발생 시 요청을 시간 초과시켜야 합니다.

빠른 로드가 사용 안함으로 설정된 경우(기본 설정), 이동이 완료된 것으로 물리적 라이브러리에서 보고될 때까지 성공 상태가 반환되지 않습니다. 하지만 이동 중 오류가 발생하면, ACSLS가 해당 정보를 클라이언트에 보고합니다.

물리적 라이브러리는 ACSLS가 성공 상태를 반환하기 전에 필요한 시간에 영향을 줄 수 있는 고유한 빠른 로드 옵션을 제공합니다(ACSLS 빠른 로드가 사용 안함으로 설정된 경우만 해당됨). ACSLS 빠른 로드가 사용으로 설정되었으면, 라이브러리 설정이 클라이언트 알림에 영향을 주지 않습니다.

논리적 라이브러리 삭제

논리적 라이브러리는 더 이상 필요하지 않은 경우 ACSLS 구성에서 제거할 수 있습니다. 논리적 라이브러리를 제거하면 모든 논리적 드라이브, 볼륨 지정 및 클라이언트 매핑도 제거됩니다. 그러면 이 논리적 라이브러리에 지정된 번호를 새 논리적 라이브러리를 만들 때 사용할 수 있습니다. ACSLS는 논리적 라이브러리 번호 지정 시 간격을 허용하지만 사용 가능한 번호를 재사용합니다.

주:

라이브러리 구성에서 ACS를 제거하기 전에 물리적 ACS에 연관된 모든 논리적 라이브러리를 제거하십시오. 이렇게 하면 연관된 광 섬유 채널 연결도 올바르게 제거됩니다.

논리적 라이브러리를 제거하기 전에 다음 사항에 주의해야 합니다.

- 논리적 라이브러리가 오프라인 상태여야 합니다.
- 모든 논리적 드라이브가 오프라인 상태여야 합니다.

이렇게 하면 라이브러리를 삭제할 때 진행 중인 클라이언트 작업이 없도록 보장할 수 있습니다. 사용 중인 논리적 드라이브 또는 볼륨이 발견될 경우, 논리적 라이브러리가 삭제되지 않습니다.

문제 해결

이 절에서는 몇 가지 발생 가능한 문제 시나리오에 대해 설명하고 문제를 진단하고 해결하는 방법에 대한 의견을 제공합니다.

내가 지정한 논리적 라이브러리가 클라이언트에 표시되지 않는 경우에는 어떻게 합니까?

1. *acsss status*를 실행하여 ACSLS가 실행 중인지 확인합니다.

   이 유틸리티는 ACSLS가 일반적으로 사용으로 설정되었는지 여부를 보여줍니다.

FC 문제의 경우 smce 서비스를 조사합니다. 이 서비스는 논리적 라이브러리에 대해 SCSI 매체 교환기 인터페이스를 제공합니다.

2. ACSLS가 실행 중이면 클라이언트 연결이 올바르게 정의되었는지 확인합니다.
   - 실제로 FC를 통해 연결되지 않은 대상-개시자 쌍을 지정했을 수 있습니다.
   - 논리적 라이브러리를 잘못된 클라이언트에 지정했을 수 있습니다(잘못된 대상-개시자 쌍).
   - *getHba.sh* 유틸리티는 검색된 개시자(클라이언트) 및 특정 대상 포트에 연결된 항목을 식별하는 데 유용할 수 있습니다.

클라이언트가 올바르게 연결되었지만, 계속 논리적 라이브러리가 표시되지 않는 경우에는 어떻게 합니까?

클라이언트가 대상 장치를 다시 검색하도록 설정해야 할 수 있습니다. 실제 메커니즘은 클라이언트 시스템의 운영체제에 따라 달라집니다. 클라이언트 응용 프로그램이 장치를 사용할 수 있으려면 먼저 클라이언트 OS가 해당 장치를 볼 수 있어야 합니다.

예를 들어, Solaris 클라이언트의 경우 *cfgadm* 및 *devfsadm* 명령을 사용해서 장치 파일 시스템 및 구성을 업데이트할 수 있습니다.

논리적 라이브러리에 대해 클라이언트를 구성하는 중 문제가 발생하면 어떻게 합니까?

클라이언트 백업 응용 프로그램은 지원되는 라이브러리 유형으로 ACSLS 논리적 라이브러리를 인식하지 못할 수 있습니다. 논리적 라이브러리는 소프트웨어 공급업체로부터 업데이트가 필요할 수 있는 고유한 질문을 반환합니다. 또한 라이브러리 및 해당 테이프 드라이브를 올바르게 구성하려면 응용 프로그램 특정 매핑 파일이 필요할 수도 있습니다.

일부 경우, 클라이언트 소프트웨어는 테이프 라이브러리가 SCSI 대상에서 항상 LUN 0으로 표시되어야 합니다. 일반적으로 클라이언트의 대상 포트에서 첫번째 논리적 라이브러리는 실제로 LUN 0으로 표시됩니다. 하지만, 특정 대상 포트에서 클라이언트에 여러 라이브러리가 지정된 경우 항상 LUN 0으로 라이브러리가 표시되지 않을 수 있습니다. 이 문제는 클라이언트 문제입니다.

대상 모드에서 FC 포트를 구성하는 방법

설치 중 대상 포트로 사용할 HBA를 선택하라는 메시지가 표시됩니다. 이 작업은 SCSI(FC) 클라이언트 응용 프로그램에 하나 이상의 논리적 라이브러리를 제공하려는 경우에만 적용됩니다.

설치 후 *getHba.sh* 유틸리티를 사용해서 대상 모드에서 광 섬유 채널 포트를 구성합니다. *getHBA.sh* 유틸리티는 시스템에 새 HBA를 추가할 때마다 또는 HBA 포트를 다시 정렬할 때마다 언제라도 실행할 수 있습니다. 이 유틸리티는 ACSLS 논리적 라이브러리에 대해 클라이언트 액세스 지점을 노출하기 위해 개시자에서 대상 모드로 변경할 적합한 HBA 포트를 식별합니다.

SCSI 직접 연결 클라이언트에 대해 하나 이상의 논리적 라이브러리를 구성하려는 경우, 개시자 모드에서 대상 모드로 변환할 HBA를 선택합니다. 이상적으로, 사용자가 선택하는 HBA는 패브릭에 연결되며, SCSI 클라이언트 개시자로 사용할 원격 HBA에 표시됩니다.

대상 모드 변경이 특정 HBA에 대해 설정된 다음에는 HBA 변환을 완료하기 위해 재부트해야 합니다.

대상 모드에서 광 섬유 채널 포트를 구성하려면 다음 절차를 완료합니다.

1. *root*로 로그인합니다.
2. *$ACS_HOME/install* 디렉토리로 이동하거나 이 디렉토리를 경로에 포함시킵니다.
3. *getHba.sh*를 실행합니다.

예:

```
# ./getHba.sh
Please select the HBE port you intend for Target-mode operation:
Select from the following list:
1)HBA Port WWN 210000lb3213ble2   Not connected.
2)HBA Port WWN 210100lb3233ble2   Not connected.
3)None of these.
2
HBA Port WWN 210100lb3233ble2/pci@1,0/pci1023,7450@1/pci1077,141@3,1
Is this correct? (y or n):y
Are there additional ports you wish to reconfigure? (y or n):n
A reboot will be necessary for these changes to take effect.
```

부록 I

부록 I. 라이브러리 분할

SL8500 및 SL3000에서는 해당 라이브러리에 대한 분할이 지원됩니다.

하나의 ACSLS 서버는 동일 라이브러리에 있는 여러 분할 영역을 관리할 수 있습니다.

- SL8500 분할
  - SL8500 컴플렉스 분할

    SL8500 8.3 및 이후 펌웨어에서는 라이브러리 컴플렉스 간 분할이 지원됩니다. 전달 포트로 연결된 SL8500의 라이브러리 컴플렉스에서 최대 16개의 분할 영역을 만들 수 있습니다.

    ACSLS 8.3 및 이후 릴리스에서는 최대 16개까지 분할 영역이 지원됩니다. ACSLS 8 .2 및 이전 릴리스에서는 최대 8개까지 분할 영역이 지원됩니다.
  - 향상된 SL8500 분할(SL8500 7.0x 펌웨어에서 제공)
    - 향상된 SL8500 분할을 사용하면 단일 독립형 SL8500만 분할할 수 있습니다.
    - 향상된 SL8500 분할은 최대 8개까지 물리적 분할 영역을 지원할 수 있습니다.
    - 향상된 SL8500 분할은 드라이브 및 셀 배열 레벨에서 수행됩니다.
  - 레거시 SL8500 분할
    - 레거시 SL8500 분할을 사용하면 단일 독립형 SL8500만 분할할 수 있습니다.
    - 레거시 SL8500 분할은 최대 4개까지 물리적 분할 영역을 지원할 수 있습니다.
    - 레거시 SL8500 분할은 레일(LSM) 레벨에서 수행되며, 레일에는 해당 레일에 있는 모든 사용 허가된 테이프 드라이브 및 카트리지가 포함됩니다.
    - 레거시 SL8500 분할 영역은 1~4개의 레일을 포함할 수 있습니다. 레거시 SL8500 분할 영역에 있는 레일은 연속적이어야 합니다. 예를 들어, 분할 영역에 2, 3, 4 레일은 포함될 수 있지만, 2 및 4 레일은 분할 영역이 될 수 없습니다.

분할에 대한 자세한 내용은 SL8500 설명서 세트를 참조하십시오.

- SL3000 분할
  - SL3000은 최대 8개까지 물리적 분할 영역을 지원할 수 있습니다.
  - SL3000은 드라이브 및 셀 레벨까지 분할될 수 있습니다.

자세한 내용은 *SL3000 User's Guide*를 참조하십시오.

SL8500 및 SL3000 분할의 공통 사항

SL8500 또는 SL3000 라이브러리 분할은 다음과 같은 기능을 제공합니다.

- ACSLS가 열려 있는 시스템 드라이브 및 카트리지를 관리할 수 있고, HSC가 라이브러리를 공유하는 메인프레임 드라이브 및 카트리지를 관리할 수 있습니다.
- 두 개 이상의 ACSLS 인스턴스가 라이브러리에 액세스할 수 있습니다.
- 하나의 ACSLS 인스턴스가 동일 라이브러리에 있는 여러 분할 영역을 관리할 수 있습니다.
- 개별 분할 영역에 있는 데이터 카트리지의 보호 및 격리를 제공합니다.
- 더 높은 수준의 데이터 구성이 가능합니다.
- 사용자 효율성이 증가합니다.

주의:

분할된 라이브러리는 구성되거나 ACSLS로 다시 구성된 후 반드시 감사(*audit*)를 수행해야 합니다. 라이브러리는 ACSLS가 이를 감사할 때 ACSLS에서 액세스할 수 없는 잠재적인 셀 위치를 보고합니다. 액세스할 수 없는 셀 위치에는 CAP, 드라이브 및 운영자 패널이 설치된 위치, 로봇이 액세스할 수 없는 셀 위치, 활성화되지 않은 셀 위치, 이 분할 영역에 없는 셀 위치가 포함됩니다.

주의:

셀이 하나의 분할 영역에서 다른 분할 영역으로 다시 지정된 경우, 이러한 셀에 있는 카트리지가 고립되며, 이전에 있던 분할 영역에서 더 이상 액세스할 수 없게 됩니다. 다른 분할 영역을 관리하는 호스트가 이 카트리지에 있는 데이터를 겹쳐 쓸 수 있습니다.

주:

SL500은 분할 가능하지만, ACSLS에서는 분할된 SL500이 지원되지 않습니다.

분할 지침

라이브러리를 분할할 때는 따라야 하는 몇 가지 단계가 있습니다. 다음과 같습니다.

1. 새 구성을 계획합니다.
2. 라이브러리 분할 또는 재분할을 위한 중단 일정을 잡습니다.

   라이브러리가 재분할된 경우, 해당 분할 영역이 이러한 변경의 영향을 받지 않는 경우라도 라이브러리는 모든 분할 영역을 관리하는 모든 호스트에 대한 통신이 끊어집니다.

   주:

   ACSAPI 클라이언트는 ACS에 지정할 수 없기 때문에 라이브러리의 물리적 분할 영역에 지정할 수 없습니다.
3. 서비스 대표 라이센스를 준비하고 SL8500 또는 SL3000에서 분할을 사용으로 설정합니다.
4. 액세스할 수 없는 위치로부터 ACSLS에서 관리되는 분할 영역에 계속 존재하게 될 위치로 카트리지를 이동합니다.
5. SL 콘솔을 사용해서 분할 영역을 만듭니다.

   자세한 내용은 SL 콘솔 도움말을 참조하십시오.
6. “라이브러리 분할 또는 분할 영역 ID 변경 ”에 설명된 대로 ACSLS를 사용해서 분할을 활성화합니다.

7. *acsss_config*를 사용해서 라이브러리가 분할되도록 지정하고 분할 영역 번호를 입력하여 분할된 라이브러리에 대해 ACSLS를 구성해야 합니다.

새 구성 계획

새로운 분할 구성을 미리 계획합니다.

- 성능을 위해 카트리지 및 드라이브를 구성합니다.
- 분할 영역에서 넣기 및 꺼내기를 많이 수행할 경우, SL8500 분할 영역에는 아래쪽 세 개 레일 중 하나 이상이 포함되어야 합니다. 넣기 및 꺼내기가 적은 분할 영역은 위쪽 레일에 배치할 수 있습니다.

SL8500 또는 SL3000 분할 시 중단 최소화

SL8500 또는 SL3000을 분할하거나 SL8500 또는 SL3000의 분할 영역이 수정될 경우에는 데이터에 대한 액세스가 영향을 받습니다. 하나의 호스트 서버에서 관리되던 테이프 드라이브 및 카트리지가 다른 호스트에서 관리될 수 있습니다.

구성 변경사항

다음과 같은 경우에는 구성이 변경됩니다.

- 라이브러리 구성이 변경된 경우
- 라이브러리가 다시 분할되고 ACSLS에서 관리되던 분할 영역이 변경된 경우(메시지가 표시됨)
- 라이브러리 용량이 변경되고 ACSLS에 사용 가능으로 보고된 셀이 변경된 경우
- LSM 구성 또는 용량이 변경된 경우

구성이 변경되었을 수 있는 경우에는 ACSLS 이벤트 로그 및 이벤트 알림 메시지를 통해 확인할 수 있습니다.

중단을 최소 상태로 유지

아래 제공된 다음 절차는 중단을 최소화하고 데이터를 계속 액세스할 수 있도록 보장합니다. 이러한 절차에서는 다음과 같은 방법을 설명합니다.

- 분할되지 않은 SL8500 또는 SL3000을 두 개 이상의 분할 영역으로 분할합니다.
- 이미 분할된 SL8500 또는 SL3000의 분할 영역을 수정합니다.
- 분할된 SL8500 또는 SL3000을 병합해서 분할되지 않은 단일 ACS로 만듭니다.

과제

LSM(SL8500 레일)이 레거시 SL8500 분할 영역에서 제거된 경우 또는 셀 및 드라이브가 SL3000 또는 향상된 SL8500 분할 영역에서 제거된 경우에는 가장 어려운 과제가 발생할 수 있습니다.

자세한 내용은 “분할 영역에서 셀을 제거하기 전에 카트리지 이동” 절차를 참조하십시오.

라이브러리 분할 또는 분할 영역 ID 변경

다음 절차에서는 분할되지 않은 라이브러리를 분할하는 방법 또는 분할된 기존 라이브러리에서 분할 영역 ID를 변경하는 방법에 대해 설명합니다.

1. ACS에서 SL8500 LSM 또는 SL3000 또는 SL8500 셀을 제거하기 전에 카트리지를 이동합니다.

   ACSLS가 새 라이브러리 구성에서 분할 영역 중 하나를 관리하는 경우, 볼륨 목록 파일 만들기 또는 "분할 영역에서 셀을 제거하기 전에 카트리지 이동" 절차를 수행하여 다른 분할 영역에 지정할 LSM으로 카트리지를 이동합니다.
2. 다시 분할하기 전에 라이브러리를 정지하고 ACS 및 포트를 오프라인으로 전환(*vary*)합니다.
3. SL 콘솔을 사용하여 라이브러리를 분할합니다.
4. <*Apply*>를 누른 후에는 다음 작업이 수행됩니다. 라이브러리:
   - ACSLS에서 모든 새 요청을 거부합니다.
   - 모든 호스트 연결을 삭제합니다.
   - 분할 변경사항을 적용합니다.
   - 새 호스트 연결을 수락합니다.
5. 분할 영역 변경사항을 활성화하도록 ACSLS를 재구성합니다.

   라이브러리가 처음으로 분할되었거나 다시 분할되고 분할 영역 ID가 변경되었으므로, ACSLS가 이 라이브러리에 다시 연결할 수 없습니다. 다시 연결하려면 아래 절차를 따릅니다.

   a. ACSLS 작동 중지: *acsss disable*.

   b. *acsss_config*를 실행하여 ACS를 구성하고 옵션 8을 선택합니다.

      ACS가 분할된 라이브러리인지 묻는 메시지에 대해 y(예)를 입력합니다. y(예)를 다시 입력한 후 분할 영역 번호를 입력하고 *enter*를 누릅니다. 자세한 내용은 "CSI 조정 변수 설정"을 참조하십시오.
6. 새 구성을 사용해서 ACSLS 시작: *acsss enable*
7. 포트를 온라인으로 전환(*vary*)하고 ACS를 진단 모드로 전환합니다.
8. ACS에 대해 *Audit*를 수행합니다.
9. ACS를 온라인으로 전환(*vary*)합니다.

라이브러리 다시 분할

다음 절차에서는 기존에 분할된 라이브러리를 다시 분할하는 방법에 대해 설명합니다.

1. ACS에서 LSM 또는 셀을 제거하기 전에 카트리지를 이동합니다.

   ACSLS가 새 라이브러리 구성에서 분할 영역 중 하나를 관리하는 경우, 볼륨 목록 파일 만들기 또는 "분할 영역에서 셀을 제거하기 전에 카트리지 이동" 절차를 수행하여 다른 분할 영역에 지정할 LSM으로 카트리지를 이동합니다.

2. 다시 분할하기 전에 라이브러리를 정지하고 ACS 및 포트를 오프라인으로 전환(*vary*)합니다.

   그러면 ACS가 동적 재구성에 대해 사용 가능한 상태에서(다시 분할 후) ACSLS가 클라이언트로부터의 새 요청을 거부합니다.
3. SL 콘솔을 사용하여 라이브러리를 다시 분할합니다.

   그런 다음 **Apply**를 누르면 다음 작업이 수행됩니다. 라이브러리:
   - ACSLS에서 모든 새 요청을 거부합니다.
   - 모든 호스트 연결을 삭제합니다.
   - 분할 변경사항을 적용합니다.
   - 새 호스트 연결을 수락합니다.
4. 분할 영역 변경사항을 활성화하도록 ACSLS를 재구성합니다.

   주:

   특정 분할 영역에 대해서는 변경사항이 없을 수 있습니다. 변경사항은 다른 분할 영역에만 영향을 줄 수 있습니다. 이 경우에는 ACSLS 구성 변경이 필요하지 않습니다.

   주:

   라이브러리가 처음으로 분할되지 않았거나 다시 분할되었지만 분할 영역 ID가 변경되지 않았으므로, ACSLS가 라이브러리에 다시 연결할 수 있습니다.

   다음 작업 중 하나를 선택합니다.
   - ACSLS가 실행되는 동안 동적 구성(*config acs* `acs_id`)을 실행합니다. 그러면 새로운 라이브러리 구성과 일치하도록 ACSLS 구성이 업데이트됩니다. 다음 절차를 따릅니다.
     - 포트를 온라인으로 전환(*vary*)합니다.
     - *config acs* `acs_id`를 사용해서 ACS 구성을 업데이트합니다.
   - 또는 ACSLS를 중지하고 *acsss_config*(옵션 8)를 실행하여 ACSLS에서 분할 영역 구성을 업데이트합니다. 다음 절차를 따릅니다.
     - ACSLS 작동 중지: *acsss disable*
     - *acsss_config*를 사용해서 구성을 업데이트합니다.
     - ACSLS 다시 시작: *acsss enable*.
     - 포트를 온라인으로 전환(*vary*)합니다.
5. ACS를 진단 모드로 전환(*vary*)합니다.
6. ACS에 대해 *Audit*를 수행합니다.
7. ACS를 온라인으로 전환(*vary*)합니다.

분할된 ACS를 분할되지 않은 ACS로 변경

1. 선택적으로 ACSLS에서 관리되는 분할 영역에서 카트리지를 꺼냅니다(*eject*).

ACSLS가 분할되지 않은 라이브러리를 관리하지 않을 경우, ACSLS에서 관리되는 분할 영역에서 카트리지를 꺼내야 할 수 있습니다. 이렇게 하면 ACSLS에서 관리하는 다른 ACS에 넣을 수 있습니다.

여러 카트리지를 꺼내려면 *ejecting.sh* 유틸리티를 사용할 수 있습니다.

2. 다시 분할하기 전에 라이브러리를 정지하고 ACS 및 포트를 오프라인으로 전환(*vary*)합니다.
3. SL 콘솔을 사용하여 라이브러리를 다시 분할합니다.

   그런 다음 **Apply**를 누르면 다음 작업이 수행됩니다. 라이브러리:
   - ACSLS에서 모든 새 요청을 거부합니다.
   - 모든 호스트 연결을 삭제합니다.
   - 분할 변경사항을 적용합니다.
   - 새 호스트 연결을 수락합니다.
4. ACS를 분할되지 않은 라이브러리로 변경하거나 해당 구성에서 이 ACS(분할 영역)를 제거하도록 ACSLS를 재구성합니다.

   라이브러리가 분할된 상태에서 분할되지 않은 상태로 변경되었기 때문에 ACSLS가 라이브러리에 다시 연결할 수 없습니다. 다시 연결하려면 아래 절차를 따릅니다.
   a. ACSLS 작동 중지: *acsss disable*
   b. *acsss_config*를 실행하여 ACS를 구성합니다.

      ACS가 분할된 라이브러리인지 묻는 메시지에 대해 n(아니오)을 입력합니다. 자세한 내용은 "CSI 조정 변수 설정"을 참조하십시오.
5. 새 구성을 사용해서 ACSLS 시작: *acsss enable*.
6. 포트 및 ACS를 온라인으로 전환(*vary*)합니다.

ACS 분할 영역 ID 보기

하나 이상의 ACS에 대해 분할 영역 ID를 보려면 다음 명령 중 하나를 사용합니다.

- *query lmu all* - 모든 *ACS*의 경우
- query lmu acs_id - 단일 ACS의 경우

출력 예:

```
ACSSA> q lmu 0
2008-02-27 06:08:02

ACS:   0      Mode: Single LMU         Active Status:  Not Comm
Not Partitioned                        Standby Status:  -

      ACS State     Desired State
      offline           online

Port    Port State  Desired State     Role  CL  Port Name0, 0       offline
   online            -     13      172.27.2.6
```

CAP 동작

이 절에서는 CAP 동작에 대해 설명합니다.

분할된 라이브러리

경고:

모든 ACS는 전용이거나 공유되는 하나의 CAP를 포함해야 합니다. SL3000의 모든 CAP는 다른 분할 영역에 전용으로 지정될 수 있으므로, SL3000 분할 영역을 구성할 때는 이러한 특성이 중요한 제한 사항이 될 수 있습니다.

분할된 라이브러리에서 CAP 전용 지정

SL3000 라이브러리에서 CAP는 하나의 분할 영역에 대해 전용으로 지정될 수 있습니다. CAP가 하나의 분할 영역에 전용으로 지정된 경우에는 ACSLS가 이러한 CAP를 자동 모드로 설정할 수 있습니다.

분할된 라이브러리에서 CAP 공유

레거시 분할을 사용하는 SL8500 라이브러리에서는 CAP가 항상 모든 분할 영역 사이에 공유됩니다.

향상된 분할이 사용된 분할된 SL3000 또는 SL8500에서는 라이브러리 CAP를 분할 영역 간에 공유할 수 있습니다. 프로세스는 다음과 같습니다.

1. 호스트가 카트리지를 넣거나 꺼내기 위해 CAP를 사용할 때 호스트는 CAP를 배타적으로 사용할 수 있도록 예약합니다.
2. 호스트는 넣기 또는 꺼내기를 완료한 후 CAP를 비우고 닫아야 합니다.

   다음 중 하나에서 넣기를 종료해야 할 경우:
   - ACSLS - 취소된 *enter* 명령을 실행해야 합니다.
   - HSC - *drain* 명령을 실행해야 합니다.
3. 이제 CAP를 라이브러리에 있는 모든 분할 영역에서 사용할 수 있습니다.

다른 호스트에 CAP가 필요한 경우

호스트가 넣기 또는 꺼내기를 완료하지 않은 경우, CAP는 해당 분할 영역에 대해 카트리지를 계속 포함할 수 있습니다. CAP는 계속 예약된 상태입니다. 그러면 다른 분할 영역이 이 CAP를 사용해야 할 경우에 문제가 발생합니다. 이를 해결하기 위해서는 라이브러리가 해당 예약 상태를 보유 중인 호스트를 식별하고 이러한 호스트로부터 넣기 또는 꺼내기를 완료해야 합니다.

드문 경우지만, 호스트가 중단되고 호스트를 다시 시작할 수 없거나 필요한 명령을 실행하기 위한 물리적 액세스 또는 보안 권한이 없는 경우 예약을 보유 중인 호스트에 액세스하지 못할 수 있습니다. 이러한 경우에는 SL 콘솔을 사용해서 CAP 예약을 대체해야 합니다.

주의:

SL 콘솔을 사용한 CAP 예약 대체는 마지막 방법으로만 수행해야 합니다.

공유되는 CAP에 대한 CAP 우선순위 지정

ACSLS는 동일 라이브러리에서 여러 분할 영역을 관리할 수 있으며 이러한 분할 영역은 동일한 CAP를 공유할 수 있습니다. 감사를 수행하면 중복되고 읽기 불가능한 *vol_id*를 가진 모든 카트리지를 꺼냅니다. 여러 분할 영역(ACS)을 동시에 감사할 때는 감사 작업 중 카트리지를 꺼내기 위해 동일한 공유 CAP를 사용하려고 시도할 수 있습니다. 첫번째 감사가 CAP를 예약하고, 다른 감사는 CAP를 예약할 수 없게 됩니다. 그러면 나중 감사에서 카트리지를 꺼낼 수 없습니다.

이러한 경우를 방지하기 위해서는 다른 공유 CAP가 다른 분할 영역에서 가장 높은 우선순위 CAP가 되도록 CAP 우선순위를 설정합니다. 예를 들어, SL3000의 분할 영역 2와 3이 CAP 5와 6을 공유할 경우, CAP 5를 분할 영역 2에서 가장 높은 우선순위 CAP로 설정하고, CAP 6을 분할 영역 3에서 가장 높은 우선순위 CAP로 설정합니다.

CAP 예약

CAP를 예약할 때는 다음을 고려해야 합니다.

- CAP가 넣기 또는 꺼내기 작업을 위해 사용 중인 경우에는 모든 39개 슬롯(SL8500) 및 26개 슬롯(SL3000)이 해당 작업을 위해 예약됩니다. CAP는 나눌 수 없습니다.
- 주소 지정 목적으로 CAP에는 위치(ACS,LSM,CAP#)가 필요합니다. LSM 번호는 각 라이브러리에 있는 두번째 레일과 연관됩니다. 예: CAP A의 경우 ACS#,1,0 및 CAP B의 경우 ACS#,1,1
- CAP 예약

  CAP는 분할 영역 간 공통 구성 요소입니다. 각 호스트는 다음을 수행합니다.
  - *enter* 또는 *eject*에 사용할 수 있도록 CAP를 예약합니다.
  - 넣기 또는 꺼내기가 완료되면 CAP를 해제합니다. 그러면 또 다른 호스트가 이를 사용할 수 있습니다.

CAP 예약 종료

1. ACSLS에서 CAP가 다른 분할 영역에 의해 예약된 것으로 확인되면 해당 예약을 보유하고 있는 분할 영역 ID와 호스트 ID가 보고됩니다.

   예약된 CAP를 소유 중인 호스트에 연결해서 *enter* 또는 *eject*를 완료하도록 요청합니다.

   이 작업을 수행한 다음에는 비어 있는 CAP가 예약 해제되어 사용할 수 있게 됩니다.
2. CAP 예약을 보유 중인 호스트가 이를 해제할 수 없는 경우, *SL Console*(SLC) 명령이 CAP 예약을 대체합니다.
   a. SLC 명령은 특정 분할 영역 소유자의 예약이 제거됨을 알리는 경고를 호스트에 표시합니다.

b. SL 콘솔은 CAP를 질의해서 분할 영역 소유자/요청자/예약자를 찾습니다.
c. SLC는 경고에 분할 영역 이름(HLI1, HLI2, HLI3, HLI4 또는 기본값)을 표시합니다.

3. 라이브러리는 예약을 보유하는 호스트에 CAP 대체 메시지를 전송합니다.

   이 호스트는 진행 중이던 *enter* 또는 *eject* 작업을 종료해야 합니다.
4. 라이브러리가 이제 CAP를 소유합니다.

   CAP를 다른 호스트에서 사용할 수 있으려면 먼저 CAP를 비우고 닫아야 합니다. 다음을 참조하십시오.
   - CAP에 카트리지가 포함된 경우에는 SL 콘솔을 사용해서 분할 영역이 CAP를 사용할 수 있도록 먼저 CAP를 비워야 합니다.
   - CAP가 열려 있는 경우에는 CAP를 비우고 닫아야 합니다. 그런 다음 라이브러리가 이에 대해 *audit*를 수행하고 비어 있는지 확인할 수 있습니다. CAP가 닫히기 전까지는 분할 영역에 제공될 수 없습니다.
   - CAP가 비어 있는 경우에는 잠금이 설정되고 모든 분할 영역에서 예약할 수 있도록 제공됩니다.
5. CAP가 이제 예약되었고 모든 분할 영역에서 사용할 수 있습니다.

카트리지를 특정 셀로 이동

SL3000은 드라이브 및 셀 레벨로 분할될 수 있으며, 향상된 분할 기능이 있는 SL8500은 드라이브 및 셀 배열 레벨로 분할될 수 있습니다. 셀이 하나의 분할 영역에서 다른 분할 영역으로 다시 지정된 경우, 이러한 셀에 있는 카트리지가 고립되며, 이전에 있던 분할 영역에서 더 이상 액세스할 수 없게 됩니다. 그런 다음 다른 분할 영역을 관리하는 호스트가 이 카트리지에 있는 데이터를 겹쳐 쓸 수 있습니다.

분할 영역 경계가 변경될 때 그리고 라이브러리를 다시 분할하기 전에 카트리지가 고립되지 않도록 방지하려면 분할 영역에 유지되는 셀로 카트리지를 이동합니다.

SL3000은 단일 LSM이기 때문에 기존 ACSLS *move* 명령이 작동하지 않습니다. 카트리지가 라이브러리의 다른 위치로 이동되고 분할 영역에서 제거될 다른 셀로 이동될 수 있습니다. 이를 처리하기 위해 move 명령은 카트리지를 특정 셀로 이동하는 기능을 제공합니다.

카트리지를 셀로 이동하는 구문은 LSM으로 이동하는 것과 비슷합니다. 하지만 LSM ID를 지정하는 대신 다음과 같이 셀 ID를 지정합니다.

a=acs, l=*lsm*, p=*panel*, r=*row* 및 c=*column*

move AAAAAA a,l,p,r,c (셀 이동의 경우)

move AAAAAA a,l (표준 LSM 이동의 경우)

셀 이동 예:

move EDU010 0,1,0,5,1

주:

display 명령을 사용해서 지정된 패널에 사용 가능한(비어 있는) 셀 목록을 표시할 수 있습니다. 예:
display cell a,l,p,,* -status empty -f status*

부록 J

부록 J. 문제 해결

이 부록은 ACSLS에서 발생한 문제 해결을 위한 도구, 팁 및 기술을 요약합니다. 문제 해결 리소스 범위에는 로그, 주요 관찰 포인트 및 진단 프로브가 포함됩니다.

ACSLS 이벤트 로그

ACSLS 이벤트 로그는 라이브러리 작동에 문제가 발생하는 경우 유용한 정보를 찾을 수 있는 첫번째 지점입니다. 이 로그에는 라이브러리 이벤트, 상태 변경 및 오류에 대한 정보가 포함되어 있습니다. ACSLS 내 모든 하위 구성 요소는 이벤트 로거라고 하는 프로세스에 메시지를 보내 *acsss_event.log*에 이벤트를 보고합니다. ACSLS가 설치될 때 자동으로 생성되는 표준 이벤트 로그는 *$ACS_HOME/log/acsss_event.log* 파일에 포함됩니다. 여기서 *$ACS_HOME*은 대개 */export/home/ACSSS/*입니다.

기록된 이벤트는 다음과 같습니다.

- 중요한 이벤트

  중요한 이벤트는 라이브러리 관리를 지원할 수 있는 일반 이벤트입니다. 예를 들어 감사가 시작되거나 종료될 때, 장치의 상태가 변경될 때 또는 CAP가 열리거나 닫힐 때 이벤트가 기록됩니다.
- 라이브러리 오류

  라이브러리 오류는 치명적이거나 치명적이지 않은 하드웨어 및 소프트웨어 오류가 모두 기록되는 이벤트입니다. 예를 들어 LSM 실패, 카트리지 문제, 데이터베이스 오류, 프로세스 실패 및 라이브러리 통신 실패가 있습니다.

이벤트 로그의 각 메시지에는 시간 기록, 메시지를 보고하는 구성 요소의 이름 및 이벤트에 대한 설명이 포함되어 있습니다. 각 메시지의 전체 설명은 *ACSLS Messages* 설명서를 참조하십시오.

ACSLS 콘솔 창에는 실행 중인 이벤트 로그의 끝 부분이 표시됩니다. 모든 셸 창에서 유사한 표시를 생성할 수 있습니다.

1. *acsss* 사용자로 다음 명령을 실행합니다.

   ```
   acs_tail $ACS_HOME/log/acsss_event.log
   ```
2. 전체 이벤트 로그를 보려면 로그를 탐색하거나, 특정 오류를 검색하거나, 이벤트의 특정 시퀀스를 따를 수 있는 vi와 같은 텍스트 편집기를 사용합니다.

이벤트 로그 관리

ACSLS는 계속해서 *acsss_event.log*에 메시지를 보냅니다.

- 이 파일이 임계값 크기(기본값 500KB)에 도달하면 파일의 이름이 *event0.log*로 바뀐 다음 로그 디렉토리에 저장됩니다. 이에 따라 *acsss_event.log*는 새 파일로 계속 존재하게 됩니다.
- *acsss_event.log*가 다시 임계값 크기에 도달하면 *event0.log*의 이름은 *event1.log*로 변경되고 *acsss_event.log*의 이름은 *event0.log*로 변경됩니다.
- 이 프로세스는 보존을 위해 구성된 로그 파일 개수만큼 진행됩니다.

기본적으로 9개의 이벤트 로그 파일이 로그 디렉토리에 보존됩니다. 각 후속 임계값에 따라 가장 오래된 파일은 제거되고 남은 모든 파일의 이름이 순차적으로 바뀝니다.

옵션 2인 *acsss_config*를 사용하여 보존할 로그 파일 수 및 *acsss_event.log*의 최대 크기를 구성할 수 있습니다. "CSI 조정 변수 설정"을 참조하십시오.

*greplog*를 사용하여 이벤트 로그 검색

진단 도구인 *greplog*를 사용하면 모든 이벤트 로그 파일을 통해 키워드 검색을 수행할 수 있습니다. UNIX *grep* 유틸리티와 같이 가장 많이 사용되는 *greplog*는 지정된 키워드 표현식과 연관된 전체 로그 메시지를 반환합니다. 이를 통해 해당 표현식을 포함하는 모든 메시지와 관련된 메시지의 날짜 및 시간 기록, 메시지 번호 및 함수 텍스트를 볼 수 있습니다.

형식

```
greplog [-iv] pattern file_1 file_2 ... feline
```

옵션

- *-i*는 *greplog*가 검색 패턴 표현식의 대소문자를 무시하도록 지시합니다.
- *-v*는 *greplog*가 표현식을 포함하는 모든 메시지를 제외하고 로그 파일의 모든 항목을 표시하도록 지시합니다. 패턴 표현식과 일치하는 항목은 예외입니다.

  패턴: 패턴은 사용될 검색 조건입니다.

  ```
  file_1 file_2 ... file_n
  ```

greplog 도구는 파일 목록에 있는 여러 파일 매개변수 및 와일드카드 표현식을 수락합니다.

예

- 이벤트 시퀀스 내 모든 발생을 표시하려면 시퀀스 번호를 사용합니다.

  ```
  greplog 1392 acsss_event.log
  ```
- 이벤트 로그에서 볼륨 CART89에 대한 모든 메시지를 검색하려면 다음을 수행합니다.

  ```
  greplog CART89 acsss_event.log
  ```

- 이벤트 로그의 아카이브된 모든 복사본에서 테이프 마운트에 대한 메시지를 검색하려면 다음을 수행합니다.

```
greplog -i mount event*.log
```

추가 로그

*acsss_event.log*에는 ACSLS의 실행 중인 프로세스에 대한 모든 측면과 관련된 모든 메시지가 포함됩니다. 이외에도 백업, 복원 및 설치 유틸리티와 같은 외부 유틸리티에 대한 상태 정보가 포함된 추가 파일이 로그 디렉토리에 있습니다.

- *acsss.pid* - 현재 실행 중인 *acsss_daemon*의 프로세스 ID를 저장합니다.
- *acsss_config.log* - 각 라이브러리 구성의 요약을 포함합니다.
- *acsss_config_event.log* - *acsss_config* 루틴에 의해 게시된 이벤트 메시지를 포함합니다.
- *bdb_event.log* - *bdb.acsss* 데이터베이스 백업 유틸리티에 의해 게시된 이벤트 메시지를 포함합니다.
- *cron_event.log* - *cron* 유틸리티에 의해 게시된 메시지를 포함합니다. cron 일정을 보려면 *crontab -l* 명령을 실행합니다.
- *acsls_start.log* - *acsls* 서비스와 관련된 시작 또는 종료 메시지를 포함합니다.
- *di_trace.log* - 데이터베이스 인터페이스와 관련된 추적 정보를 포함합니다.
- *ejectingLogs* - 지난 10일 동안 수행된 *ejecting.sh* 작업의 요약 정보를 포함하는 디렉토리입니다.
- *install.log* - *install.sh* 설치 스크립트를 실행하는 동안 게시된 이벤트 메시지를 포함합니다.
- *ipc_trace.log* - ACSLS 프로세스 간 통신과 관련된 추적 정보를 포함합니다.
- *rdb_event.log* - *rdb.acsss* 데이터베이스 복원 유틸리티에 의해 게시된 이벤트 메시지를 포함합니다.
- *timed_bkup.sh.log* - 자동 데이터베이스 백업 유틸리티와 관련된 이벤트 메시지를 포함합니다.

시스템에서 사용으로 설정한 특정 추적에 따라 로그 디렉토리에서 추가 추적 로그가 발견될 수도 있습니다. 로그에는 다음이 포함됩니다.

- *acsss_stats.log* - *acsss_config*에 의해 볼륨 통계 추적이 사용으로 설정됩니다.
- *acsss_trace.log* - 소프트웨어 지원 센터 담당자의 요청에 따라 클라이언트-서버 추적이 사용으로 설정됩니다.
- *acslh.log* - 소프트웨어 지원 센터 담당자의 요청에 따라 호스트-LMU 추적이 사용으로 설정됩니다.
- *scsilh.log, mchangerX.log, scsipkt.log* - 이 세 로그 모두에는 SCSI 연결 라이브러리에 대한 SCSI 통신의 추적이 포함되고, 소프트웨어 지원 센터 담당자의 요청에 따라 사용으로 설정됩니다.

추적 로그 관리

소프트웨어 지원 센터의 요청에 따라 사용으로 설정된 추적 로그가 매우 빠르게 증가할 수 있습니다. 디스크가 가득 차는 문제를 완화하기 위해 이러한 로그를 모니터링하고 관리해야 합니다.

monitor.sh 유틸리티는 자동 로그 관리 및 아카이브 서비스를 수행하기 위해 제공됩니다. 구문은 다음과 같습니다.

monitor.sh <로그 이름>

특정 로그를 모니터링하도록 이 유틸리티가 사용으로 설정된 경우 로그가 1MB 크기(기본값)까지 커질 수 있도록 지정한 다음 *gzip*을 사용하여 로그를 압축합니다. 이 압축된 로그 파일(시간 기록된 이름 포함)을 *ACSSS/log/log_archives* 하위 디렉토리에 배치합니다. 이 작업은 추적이 사용으로 설정되어 있는 경우 계속됩니다.

Java 구성 요소 로그

ACSLS GUI 및 논리적 라이브러리 소프트웨어 구성 요소를 포함하는 ACSLS의 Java 구성 요소에서 다양한 로그가 유지 관리됩니다. 이러한 로그는 *$ACS_HOME/log/sslm* 디렉토리에서 찾을 수 있습니다.

WebLogic 설치 프로시저는 *weblogic.log*에 기록됩니다. WebLogic 및 ACSLS GUI 작업은 *AcslsDomain.log* 및 *AdminServer.log*에 기록됩니다.

웹 기반 GUI의 사용자 작업에 대한 감사 추적은 *guiAccess.log*에서 찾을 수 있습니다.

Java 구성 요소와 레거시 ACSLS 구성 요소 간의 트랜잭션은 *surrogate_trace.0.log*에 기록됩니다.

Java 클라이언트 구성 요소와 ACSLS 서버 간의 IPC 패킷은 *acslm_ipc_trace.0.log*에서 추적됩니다.

ACSLS GUI에서 발생하는 오류는 *gui_trace.0.log*에 기록됩니다.

SMCE 클라이언트와 SCSI(광 섬유) 클라이언트 간 하위 레벨의 통신은 *smce _trace.0.log*에 기록됩니다.

이러한 로그는 *$ACS_HOME/log/sslm* 디렉토리에서 찾을 수 있습니다.

주요 관찰 포인트

ACSLS의 다양한 측면 상태를 확인할 수 있는 여러 유틸리티가 있습니다.

- *psacs* - 모든 ACSLS의 실행 중인 프로세스에 대한 요약을 표시합니다. ACSLS가 실행 중인지 여부를 가장 잘 나타냅니다. 일반적인 출력은 최소한 12개의 서로 다른 프로세스(모두 공통 상위 프로세스의 하위 프로세스임)를 표시해야 합니다.
- *acsss status* - acsdb 데이터베이스 서비스가 실행 중인지 여부를 확인합니다.

- ACSLS 릴리스 및 유지 관리 레벨을 표시하려면 다음을 수행합니다.
  - Solaris의 경우:

    ```
    pkginfo -l STKacsls
    ```
  - Linux의 경우:

    ```
    rpm -q ACSLS
    ```
  - Solaris 또는 Linux의 경우:

    ```
    in_get_version
    ```

ACSLS 시작 문제 진단

- *acsss_event.log*를 확인합니다.
- *acsls_start.log*를 확인합니다.
- *acsss_event.log*의 끝에서 문제를 설명하는 메시지를 확인합니다.
- 메시지에 대한 설명 및 해당 메시지를 해결하기 위해 수행할 수 있는 작업에 대해서는 ACSLS Messages 설명서를 참조하십시오.
- *acsss l-status*로 ACSLS 서비스의 상태를 표시합니다.

  ACSLS 서비스의 상태 요약을 표시하려면 *acsss l-status*를 사용합니다. 각 서비스의 *logfile* 항목은 ACSLS의 시작을 방해했던 조건을 설명하는 자세한 메시지가 포함될 수 있는 로그 데이터를 가리킵니다.
- ACSLS가 시작하는 동안 시간을 초과합니다.
- Solaris의 경우 구성에 따라 계산된 ACSLS 시작 시간 초과 기간을 표시하려면 *acsss timeout*을 사용합니다.

라이브러리 연결 테스트

ACSLS는 라이브러리에 대한 양호한 물리적 연결을 확인하는 유틸리티를 제공합니다. 작동 컨텍스트에 따라 선택해야 할 최선의 도구가 결정됩니다.

testports

이 유틸리티는 StorageTek ACSLS에 구성된 각 라이브러리에 대한 연결을 테스트합니다. 또한 사용이 가장 간단하고 가장 광범위하게 사용할 수도 있습니다. 이 테스트는 눈에 띄지 않으며 일반 라이브러리 작동에 영향을 주지 않습니다. *testports*는 StorageTek ACSLS 데이터베이스를 사용하여 라이브러리 포트 이름 및 라이브러리 유형을 확인하므로 *testports*의 작동을 위해서는 라이브러리가 StorageTek ACSLS에 대해 이미 구성되어 있어야 합니다.

- TCP/IP 라이브러리의 경우 *testports*가 연결되고, 라이브러리가 온라인 상태인지 여부와 StorageTek ACSLS에서 사용 중인지 여부를 확인합니다.
- SCSI 및 직렬 연결 라이브러리의 경우 *testports*에서 테스트 연결을 열려면 'acs' 및 'port'가 오프라인이어야 합니다.

이 유틸리티를 실행하려면 다음과 같은 명령 구문을 사용합니다.

testports

라이브러리 호환성 레벨 또는 마이크로코드 레벨이 표시됩니다.

testlmutcp

이 유틸리티는 TCP/IP 패킷을 네트워크 연결 라이브러리에 제출합니다.

라이브러리 연결을 테스트하려면 명령줄에 라이브러리 호스트 이름 또는 IP 주소를 포함합니다.

testlmutcp <ip_address> **또는**

testlmutcp <hostname>

라이브러리가 ACSLS에 대해 온라인 상태인 동안 연결을 테스트하려면 50002에서 50016 사이에서 사용되지 않은 소켓 번호를 지정합니다. 예:

testlmutcp <ip_address>:50002

성공적인 응답에는 연결 라이브러리의 호환성 레벨이 포함됩니다.

testlmu

이 유틸리티는 ACSLS와 레거시 StorageTek 직렬 연결 라이브러리 간의 연결을 테스트하는 데 사용할 수 있습니다. 이 유틸리티를 실행하려면 devlink 경로를 직렬 포트 장치 노드에 제출합니다.

testlmu /dev/term/0

*testlmu*에서 직렬 연결을 열려면 라이브러리가 ACSLS에 대해 오프라인 상태여야 합니다.

pinglmu.sh

이 유틸리티를 사용하면 라이브러리가 ACSLS에 대해 온라인 상태인 동안 ACSLS와 직렬 연결 라이브러리 간의 통신을 확인할 수 있습니다. 성공적인 응답에는 라이브러리 호환성 레벨이 포함됩니다.

probescsi.sh

이 유틸리티는 ACSLS 서버와 SCSI 또는 광 섬유 연결 라이브러리 간의 연결을 실행합니다. 이 유틸리티를 실행하려면 mchanger 장치에 대한 devlink 경로를 지정합니다. 구문은 다음과 같습니다.

*probescsi.sh /dev/*mchangerX

여기서 *X*는 테스트 중인 라이브러리의 특정 mchanger 인스턴스입니다.

*probescsi*에서 SCSI 연결을 열려면 ACSLS에 대해 라이브러리가 오프라인 상태여야 합니다. 성공적인 응답에는 연결 라이브러리의 마이크로코드 레벨이 포함됩니다.

probeFibre.sh

이 유틸리티는 ACSLS 서버에서 연결할 수 있는 모든 광 섬유 연결 라이브러리를 검색합니다. 구문은 다음과 같습니다.

```
probeFibre.sh
```

성공적인 응답은 각 광 섬유 연결 라이브러리의 모델 번호를 해당 대상, LUN ID 및 WWPN(World Wide Port Name)과 함께 표시합니다.

-v 옵션을 사용하면 호스트 버스 어댑터의 모델 번호도 표시할 수 있습니다.

```
probeFibre.sh -v
```

showDevs.sh

이 유틸리티는 mchanger 링크가 생성된 모든 mchanger 장치에 대한 세부정보를 표시합니다.

- *showDevs.sh*

  연결된 각 mchanger 라이브러리의 라이브러리 모델, 개정 레벨 및 용량을 표시합니다.
- *showDevs.sh -w*

  또한 이 옵션은 각 라이브러리의 WWPN을 포함합니다.
- *showDevs.sh -s*

  또한 이 옵션은 각 라이브러리의 일련 번호를 포함합니다.

클라이언트 연결 테스트

클라이언트 응응 프로그램은 RPC(원격 프로시저 호출) 프로토콜을 사용하여 TCP/IP를 통해 ACSLS와 통신합니다. 클라이언트 시스템이 ACSLS와 통신할 수 없는 경우 *rpcinfo*를 사용하여 클라이언트 시스템에서 ACSLS에 연결할 수 있는지 여부를 테스트할 수 있습니다.

1. ACSLS 서버에서 ACSLS가 실행 중인지 확인합니다.

   ```
   psacs
   ```
2. ACSLS 서버에서 RPC 데몬이 실행 중인지 확인합니다.

   ```
   ps -ef | grep rpc
   ```
3. ACSLS 서버에서 프로그램 번호 300031이 TCP 및 IDP에 등록되어 있는지 확인합니다.

   ```
   rpcinfo | grep 300031
   ```

이 프로그램 번호는 ACSLS가 실행 중이며 ACSLS가 RPC에 등록되었는지 확인합니다.

4. 클라이언트 시스템 또는 네트워크의 모든 UNIX 시스템에서 `rpcinfo`를 사용하여 ACSLS 서버의 프로그램 번호 300031과 패킷을 교환합니다.

   ACSLS 서버의 IP 주소를 프로그램 번호와 함께 지정합니다.

   ```
   rpcinfo -t <ip address> 300031
   ```

   통신 교환에 성공하는 경우 `rpcinfo` 유틸리티는 다음 메시지를 표시합니다.

   ```
   program 300031 version 1 ready and waiting
   ```

   ```
   program 300031 version 2 ready and waiting
   ```

   이를 통해 네트워크 간의 클라이언트 연결에 ACSLS를 사용할 수 있음을 확인합니다.

브리지 드라이브를 통해 연결된 광 섬유 라이브러리의 CAP가 잠김

다른 ACSLS 인스턴스가 라이브러리 관리를 담당할 때 브리지 드라이브를 통해 연결된 광 섬유 연결 라이브러리의 CAP가 잠길 수도 있습니다. 이 문제에 대한 자세한 내용 및 해결 방법은 SL150 부록의 “CAP(메일슬롯) 꺼내기 중 열지 않음 ”을 참조하십시오.

오라클 고객지원센터를 위해 진단 정보 수집

서비스 통화의 일부로 오라클 고객지원센터에서 분석을 위해 진단 로그 전체와 다른 진단 정보를 보내달라고 요청할 수도 있습니다. 이러한 모든 데이터는 다음 단일 명령으로 수집할 수 있습니다.

```
get_diags
```

이 유틸리티가 모든 정보를 수집하면 데이터를 전자 메일로 보낼지 아니면 수동 전송에 사용할지 묻는 메시지가 표시됩니다.

ACSLS 시스템에서 직접 데이터를 전자 메일로 보내기로 결정할 경우 ACSLS 시스템과 인터넷 간에 전자 메일 통신이 가능한지 확인하십시오. 회사에 대상 시스템에서 직접 전자 메일을 보내지 못하도록 방화벽이 있을 수 있습니다. 이 경우 회사 내부의 자신에게 전자 메일로 정보를 보낸 다음 진단 데이터를 오라클 고객지원센터에 전달할 수 있습니다.

아니면 정보를 수동으로 전송하도록 선택할 수 있습니다. *`get_diags`* 유틸리티에서 전송 대기 중인 tar 패키지가 있는 위치를 알려줍니다. 일반적으로 진단 데이터에 대한 준비 영역은 */export/backup/diag/acsss*입니다.

ACSLS 및 SELinux(Security-Enhanced Linux)

SELinux는 Oracle Linux에서 기본적으로 사용으로 설정됩니다. SELinux는 표준 Unix 레벨 액세스 제어를 벗어나 사용자 역할 및 즉시 컨텍스트 도메인에 따라 시스템 리소스에 대한

액세스를 적용합니다. SELinux 시행이 사용으로 설정되면 ACSLS가 자체 PostgreSQL 데이터베이스에 액세스하는 기능은 해당 액세스에 대한 역할 및 컨텍스트를 설정하는 특별한 정책 없이 금지됩니다.

ACSLS에 대한 SELinux 정책 모듈 설치 해제

ACSLS를 설치할 때 세 가지 SELinux 정책 모듈(*allowPostgr*, *acsdb*, *acsdb1*)이 커널에 로드됩니다. 이러한 모듈은 SELinux 시행이 활성 상태인 동안 ACSLS가 고유의 데이터베이스 및 기타 시스템 리소스에 액세스하는 데 필요한 정의 및 적용 예외사항을 제공합니다. 이러한 모듈이 설치된 상태에서 SELinux 시행을 사용 안함으로 설정할 필요 없이 *bdb.acsss*, *rdb.acsss*, *db_export.sh*, *db_import.sh*와 같은 데이터베이스 작업을 포함하는 일반적인 ACSLS 작업을 실행할 수 있어야 합니다.

신속한 소프트웨어 업그레이드를 위해 ACSLS에 의해 로드된 SELinux 정책 모듈은 ACSLS 패키지를 설치 해제할 때 자동으로 제거되지 않습니다. 이러한 모듈을 수동으로 제거하려면 ACSLS 모듈 목록을 가져옵니다.

```
# semodule -l | grep acsdb

# semodule -l | grep allowPostgr
```

각 모듈의 경우 다음 방식으로 모듈을 제거합니다.

```
# semodule -r <module name>
```

SELinux 시행 관리

ACSLS를 설치한 후 기존 파일 권한 설정이 유효한데도 시스템이 'permission denied'로 응답하는 액세스 관련 문제가 발생하는 경우 액세스 거부의 원인은 SELinux일 수 있습니다.

SELinux 시행이 사용으로 설정되었는지 여부를 확인하려면 *sestatus* 명령을 실행합니다.

```
# sestatus
SELinux status:   enabled
Current mode:     enforcing
```

setenforce 명령을 사용하여 SELinux 시행을 일시적으로 사용 안함으로 설정할 수 있습니다.

```
# setenforce Permissive
```

허가 모드에서 SELinux 시행 시 이제 실패한 리소스에 대한 액세스를 복원할 수 있는지 여부를 확인할 수 있습니다. 적용 모드가 아닌 허가 모드에서 권한이 부여된 사용자가 필요한 리소스를 사용할 수 있는 경우 업데이트된 SELinux 정책이 필요한 것입니다.

부트 간에 SELinux 보안을 영구적으로 사용 안함으로 설정하려면 다음을 수행합니다.

1. */etc/selinux/config* 파일을 편집합니다.
2. *SELINUX=enforcing*을 *SELINUX=permissive*로 변경합니다.

SELinux 시행을 다시 사용으로 설정하려면 *root*에 *sysadm_r* 역할이 있어야 합니다.

```
# newrole -r sysadm_r
# setenforce enforcing
```

SELinux가 명백한 제한의 원인임을 확인한 후에는 SELinux 감사 로그를 확인하여 필요한 리소스에 대한 액세스를 허용하지 않은 실제 규칙을 확인할 수 있습니다.

```
# vi /var/log/audit/audit.log
```

*audit.log*는 SE 시행을 성공하거나 실패한 각 액세스 시도에 대한 요약을 제공합니다. 실패한 이벤트를 찾아야 합니다. ACSLS의 경우, 특히 *acsss* 및 *acsdb* 사용자와 관련된 이벤트를 확인하십시오.

지정된 파일 또는 디렉토리와 연관된 SELinux 컨텍스트 속성을 볼 수 있습니다.

```
# ls -Z <file name>
```

secon 명령을 사용하여 현재 셸의 컨텍스트 속성 또는 지정된 프로세스의 컨텍스트 속성을 볼 수 있습니다. *chcon* 명령을 사용하여 파일 또는 디렉토리의 컨텍스트 속성을 변경할 수 있습니다. 이러한 작업에 대한 매뉴얼 페이지를 참조하십시오.

audit.log에서 찾은 실패한 작업에 대한 응답으로 정책 모듈을 만들 수 있습니다.

```
# cd /var/log/audit
# audit2allow -a -M <ModuleName>
```

이를 통해 SELinux에 의해 기록된 실패를 평가하고 정책 모듈 파일 <*ModuleName*>*.pp*를 만듭니다. 이제 이 파일을 Linux 커널에 로드하여 차단되었던 작업을 허용할 수 있습니다.

```
# semodule -i <ModuleName>.pp
```

*audit2allow*는 audit.log에서 식별된 모든 제한을 사용으로 설정하는 정책을 만들기 때문에 audit.log에는 특별히 허용하려는 해당 작업만 포함하는 것이 좋습니다. 원본 audit.log를 저장하고 새 audit.log를 만들 수 있습니다.

```
# mv audit.log audit1.log
# touch audit.log
```

audit.log에 대한 정책 모듈을 만들기 전에 캡처할 작업을 계속 진행합니다.

SELinux에 대한 자세한 내용은 매뉴얼 페이지를 참조하십시오.

```
# man selinux
```

GUI가 작동 중인지 확인

checkGui.sh 유틸리티는 공통 요소를 확인하여 GUI가 작동 중인지 여부를 평가합니다. GUI가 작동하지 않는 경우 이 유틸리티는 사용자에게 문제가 발생한 가능한 원인을 알려줄 수 있습니다.

이 유틸리티는 다음을 확인합니다.

- WebLogic이 시스템에서 사용으로 설정되어 있는지 여부
- WebLogic 작업을 방해하는 팬텀 또는 사용되지 않는 프로세스가 있는지 여부
- SlimGUI 응용 프로그램이 성공적으로 배치되었는지 여부
- WebLogic 및 GUI가 localhost에 전송된 http 요청에 응답할 수 있는지 여부
- WebLogic이 호스트의 인터넷 주소에 전송된 http 요청에 응답할 수 있는지 여부
- 서버에 방화벽 서비스가 사용으로 설정되어 있는지 여부 설정되어 있다면 WebLogic 포트 7001 및 7002에 대한 수신 요청을 수락하는 정책이 있는지 여부

Linux 시스템에서 *iptables*라고 하는 방화벽이 기본적으로 사용으로 설정되어 있는지 확인할 수 있습니다. *iptables*를 완전히 사용 안함으로 설정하거나 수신 트래픽을 수락하는 정책을 포트 7001 및 7002에 추가할 수 있습니다.

1. *root*로 이러한 포트를 사용으로 설정하려면 */etc/sysconfig/iptables* 파일을 편집합니다. 다음 두 행을 추가합니다.

   ```
   -A INPUT -p tco --dport 7001 -j ACCEPT
   -A INPUT -p tco --dport 7002 -j ACCEPT
   ```

   이러한 규칙이 검사되기 전에 해당 규칙을 수신 패킷과 일치할 다른 규칙 다음에 삽입하지 않아야 합니다. 예를 들어 이러한 규칙을 *REJECT all* 규칙 다음의 iptables 체인 끝에 추가하지 마십시오.

   iptables 명령을 사용하여 이러한 규칙을 추가하는 경우 다음을 수행합니다.
   - 테이블을 나열(*iptables -L*)하거나 인쇄(*iptables -S*)합니다.
   - 규칙을 추가합니다.

     이전 규칙으로 인해 새 규칙이 입력과 일치하지 않을 수도 있으므로 규칙을 체인 끝에 추가(*iptables –A*)하는 것만으로는 원하는 결과가 생성되지 않을 수 있습니다.

     *rulenum*별로 규칙을 삽입(*iptables -I)*합니다.
   - 변경 후 테이블을 나열(*iptables -L*)하거나 인쇄(*iptables -S*)하고, 기존 규칙으로 인해 포트 7001 및 7002에 대한 새 규칙이 검사되지 않는 일은 없도록 해야 합니다.

     이렇게 하면 새 규칙이 수신 패킷과 일치할 수 있습니다.

     checkGui.sh 유틸리티는 포트 7001 및 7002의 ACCEPT 입력에 대한 규칙이 존재하는지 여부를 검사합니다. 이 유틸리티는 해당 규칙이 올바른 iptables 체인에 있는지 또는 새 규칙이 실제로 처리될지는 확인하지 않습니다. 즉, *checkGui.sh*는 새 규칙을 검사하지 못하도록 막을 수 있는 이전 규칙이 없는지 확인하지 않습니다.
2. *iptables*를 다시 시작합니다.

   ```
   service iptables restart
   ```

Solaris에서 유사한 서비스는 *ipfilter*이며, 대개 기본적으로 사용으로 설정되지 않습니다.

GUI 문제 해결 팁

다음 표에서는 일부 GUI 문제 해결 팁에 대해 설명합니다.

표 J.1. GUI 문제 해결 팁

문제	해결 방법
https://<hostname>을 브라우저에 입력했는데 응답 페이지가 "Unable to connect(연결할 수 없음)"를 선언합니다.	올바른 URL은 https://hostname.domain:7002/SlimGUI/faces/Slim.jsp입니다.
ACSLS GUI 페이지가 불완전합니다. 일부 프레임이 불완전하거나 전체 섹션이 누락되었습니다.	브라우저에서 Refresh(새로 고침) 버튼을 누릅니다.
유효한 사용자 ID 및 암호를 Java WebLogic에서 거부합니다. 로그인할 수 없습니다.	해당 지역의 ACSLS 관리자와 상의하십시오. 관리자인 경우 userAdmin.sh 유틸리티를 사용하여 사용자를 나열 또는 추가하거나 사용자 암호를 변경합니다. 사용자가 로그인하는 데 계속 문제가 있는 경우 시스템에 메모리가 충분한지 확인한 다음 userAdmin.sh의 옵션 5로 ACSLS GUI를 다시 시작합니다. 또는 svcadm disable weblogic 및 svcadm enable weblogic을 사용하여 WebLogic을 다시 시작할 수 있습니다.
Java 오류 스택 추적이 GUI 창 하나 이상에 표시됩니다.	브라우저에서 Refresh(새로 고침) 버튼을 누릅니다. 문제가 지속되면 *acsss status*를 사용하여 ACSLS 서비스가 실행 중인지 확인합니다. 서비스가 실행되지 않는 경우 *acsss enable*을 사용하여 서비스를 작동합니다. ACSLS 서비스가 실행 중인 경우 userAdmin.sh를 사용하여 GUI를 다시 시작합니다. 또는 svcadm disable weblogic 및 svcadm enable weblogic을 사용하여 WebLogic을 다시 시작할 수 있습니다. 시스템에 *root* 액세스를 시도할 수 없는 경우 *acsss shutdown*으로 모든 서비스를 종료한 다음 *acsss enable*을 사용하여 해당 서비스를 다시 시작할 수 있습니다. 이 프로세스로 GUI가 다시 시작됩니다.
인덱스 트리 프레임에서 "Logical Libraries" 선택이 누락되었습니다.	먼저 논리적 라이브러리를 만들어야 합니다. Configuration and Administration -> Logical Library Configuration -> Create Logical Library를 선택합니다.
Volumes 페이지의 Tape Library Operations 또는 Tape Libraries & Drives 아래에 나열된 볼륨이 없습니다.	이는 라이브러리에 대한 초기 감사가 수행되지 않았음을 나타냅니다. Tape Library Operations ->Audit을 선택합니다.
Volumes 페이지의 Logical Libraries 아래에 나열된 볼륨이 없습니다.	이는 볼륨이 아직 논리적 라이브러리에 지정되지 않았음을 나타냅니다. Logical Library Configuration -> Assign Volumes를 선택합니다.
GUI 응답 시간이 느립니다.	GUI 마스트헤드의 Preferences 버튼 아래에서 Alert Update Interval을 늘립니다.
GUI 사용자를 추가하거나, GUI 사용자의 암호를 변경하거나, *acsls_admin* 암호를 설정해야 합니다.	"userAdmin.sh "를 참조하십시오. 이 유틸리티를 통해 사용자를 추가하고, 사용자의 암호를 변경하고, *acsls_admin* 암호를 재설정하는 방법을 알 수 있습니다.
브라우저를 사용하려면 보안 인증서가 필요합니다.	"HTTPS에 자체 지정 디지털 인증서 구성 "을 참조하십시오.

부록 K. ACSLS 클라이언트 응용 프로그램 설치 문제

이 부록에서는 ACSLS와 통신하는 클라이언트 응용 프로그램을 설치할 때의 알려진 문제 및 이러한 문제의 성공적인 해결 방법을 요약해서 보여줍니다.

Solaris 11에 ACSAPI 클라이언트 설치

Solaris 11은 클라이언트 응용 프로그램이 RPC를 사용해서 ACSLS 서버와 통신하는 데 필요한 모든 패키지를 포함하지 않습니다.

문제:

이 예제에서 클라이언트는 ACSLS 라이브러리를 감지할 수 없었습니다.

해결 방법:

1. /etc/hosts 파일을 수정하고 ACSAPI 클라이언트 서버를 재부트합니다.

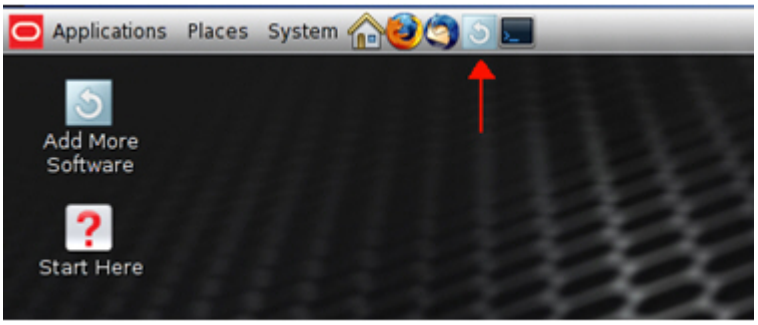
   Solaris 11의 기본값: (루프백 IP 주소만 포함)

   ```
   ::1 <client_hostname> localhost
   127.0.0.1 <client_hostname> localhost loghost
   ```

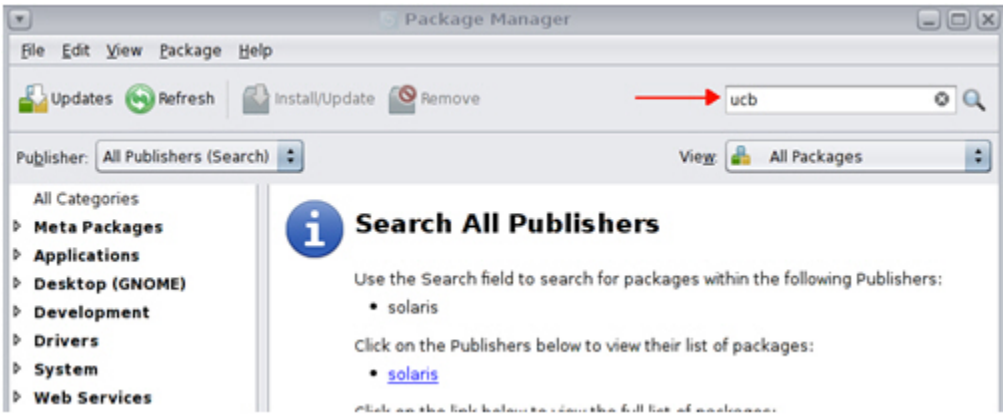
   Solaris 11의 업데이트된 */etc/hosts* 파일: (ACSAPI 클라이언트에 대한 실제 IP 주소 포함)

```
::1 localhost
127.0.0.1 localhost
<client_IP_adddr> <client_hostname> loghost
```

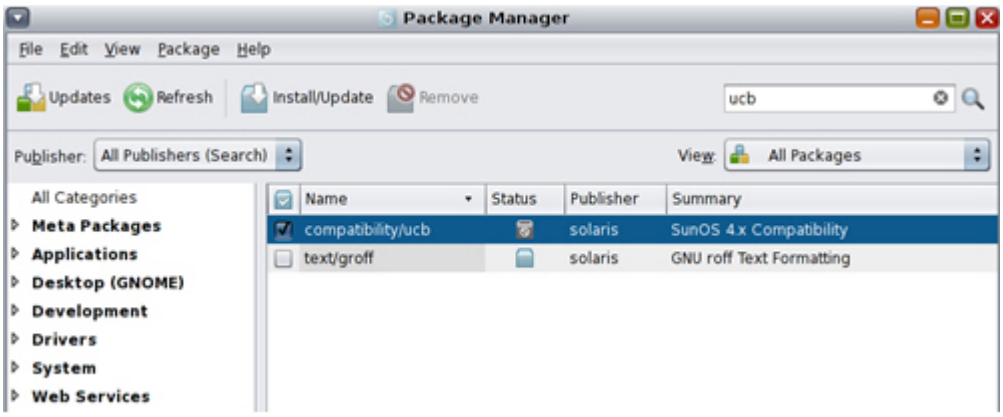
2. Package Manager를 사용해서 ACSLS 통신에 필요한 compatibility/ucb 패키지를 설치합니다.
   a. Package Manager 아이콘을 누릅니다.



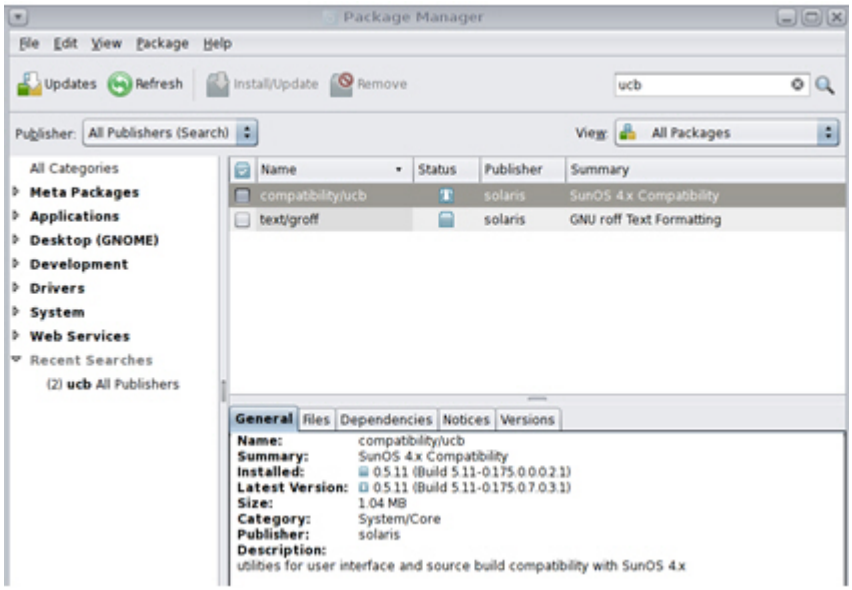
   b. 'ucb'를 찾습니다.



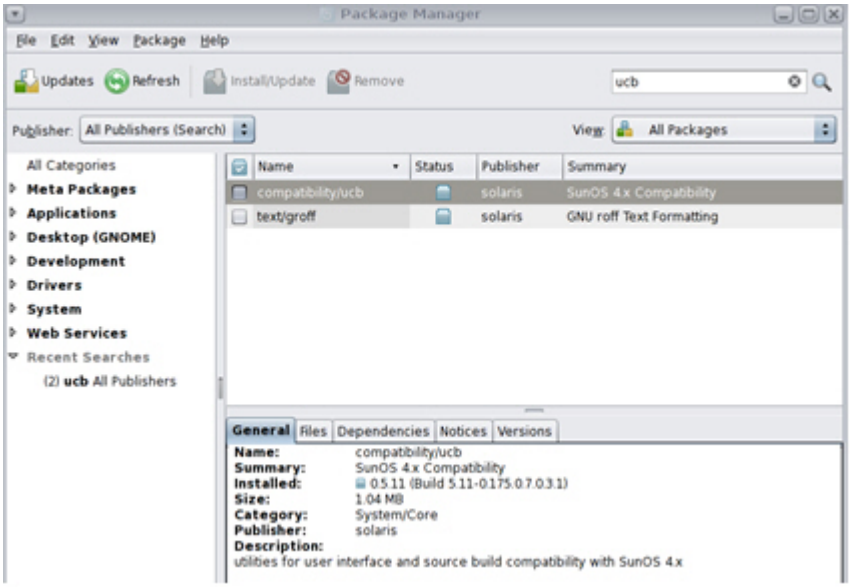
   c. compatibility/ucb 패키지를 찾고 Install/Update를 누릅니다.



   d. 최신 버전이 설치되어 있는지 확인합니다.



e. 새 버전을 사용할 수 있으면 Install/Update를 다시 누릅니다.



3. 클라이언트를 다시 구성합니다.


Device Configuration Wizard
Scanning Hosts
Wait while devices are auto-discovered.
Auto-detection has completed for all hosts checked.
Progress:
Host
Operation
Status
packer
Scanning for devices.
Detected 1 tape drive paths and 1 robots.

부록 L

부록 L. 라이브러리 성능

이 장에서는 LSM(라이브러리 스토리지 모듈) 간 전달 작업 중 라이브러리 성능을 극대화하는 방법에 대해 설명합니다. ACSLS는 여러 방식으로 전달 작업을 최소화하기 위한 작업을 수행합니다. 이러한 ACSLS 기능을 사용해서 전달 작업을 최소화하면 성능을 향상시킬 수 있습니다. 이러한 기능은 다음과 같습니다.

- 동시 마운트 및 마운트 해제 요청을 충분히 전송
- 여러 SL8500에 연결
- LSM 간 전달 작업 최소화
- 마운트 해제 중 카트리지 Float 작업
- 라이브러리에 카트리지 넣기
- 라이브러리에서 카트리지 꺼내기
- 라이브러리에서 비어 있는 스토리지 셀 유지 관리
- ACSAPI 요청 및 ACSLS 명령으로 전달 작업 최소화
- ACSAPI 요청 및 ACSLS 명령 사용

동시 마운트 및 마운트 해제 요청을 충분히 전송

일부 클라이언트 응용 프로그램은 한 번에 하나의 단일 요청만 처리하는 단순 SCSI 매체 교환기 라이브러리용으로 설계되어 있습니다. 하지만 ACSLS와 SL8500 및 SL3000 라이브러리는 동시에 여러 요청을 처리하도록 설계되었습니다. 각 라이브러리는 해당 대기열에 최소 40-50개의 동시 요청을 포함할 수 있으며, ACSLS 대기열은 기본적으로 제한이 없습니다.

클라이언트는 동시에 수백 개의 마운트 및 마운트 해제 요청을 ACSLS에 전송할 수 있습니다. 라이브러리 성능을 극대화하기 위해 클라이언트는 라이브러리에 있는 모든 로봇 등을 작동 상태로 유지하기 위해 충분히 많은 요청을 ACSLS에 전송해야 합니다.

클라이언트가 제한된 개수의 동시 요청 수만 ACSLS에 전송할 수 있는 경우 여러 클라이언트를 ACSLS에 연결할 수 있습니다. 예를 들어, 특정 클라이언트 응용 프로그램이 한 번에 하나의 단일 마운트 또는 마운트 해제 요청만 ACSLS에 전송할 수 있는 경우, SL8500당 12-16개 클라이언트 응용 프로그램을 ACSLS에 연결함으로써, 동시 마운트 및 마운트 해제 요청을 충분히 전송하여 SL8500의 모든 로봇을 작동 상태로 유지할 수 있습니다.

라이브러리 로봇을 작동 상태로 유지하기 위해 동시 요청이 얼마나 많이 필요한지를 이해하려면 다음 단락을 참조하십시오.

- 라이브러리는 장치가 로드되어 준비될 때까지 ACSLS에 마운트 응답을 반환하지 않습니다. 이 방식에서 ACSLS는 마운트가 성공했는지 여부를 알 수 있습니다.

- 스토리지 셀이 약 3000개 정도인 소규모 SL8500에서는 로봇이 카트리지를 드라이브로 이동하는 데 약 10-15초 정도가 걸립니다.
- 하지만, LTO 드라이브가 카트리지를 로드하고 준비 상태가 되려면 약 19초가 걸립니다.
- 한 레일에서 하나의 로봇이라도 사용 중인 상태로 유지하기 위해서는 이 레일에서 최소한 3개의 마운트 요청이 겹쳐서 수행되어야 합니다.

  이렇게 하면 카트리지를 수신 중인 첫번째 드라이브가 로드되어 준비되는 동안 로봇이 다른 마운트를 처리할 수 있습니다. 레일당 로봇이 2개인 경우, 레일당 최소 4개의 동시 마운트 또는 마운트 해제가 필요합니다.
- SL8500에는 레일이 4개 있으므로, SL8500에 있는 모든 로봇을 작동 상태로 유지하려면 SL8500당 12-16개의 동시 마운트 및 마운트 해제 요청이 필요합니다.

  이러한 모든 동시 요청은 단일 ACSAPI 클라이언트로부터 시작될 수도 있고, 12-16개의 서로 다른 클라이언트에서 시작될 수도 있습니다. 이러한 서로 다른 클라이언트는 각각 한 번에 하나의 단일 요청만 실행하고 다음 요청을 실행하기 전 응답을 대기합니다.

여러 SL8500에 연결

ACSLS 서버가 전달 포트를 통해 연결된 SL8500 스트링을 관리하는 경우 해당 스트링의 각 SL8500에 연결합니다. 이렇게 하면 상태를 응답하거나 카트리지 이동을 시작하는 요청을 라이브러리로 경로 지정하여 ACSLS가 라이브러리 성능을 최적화할 수 있습니다.

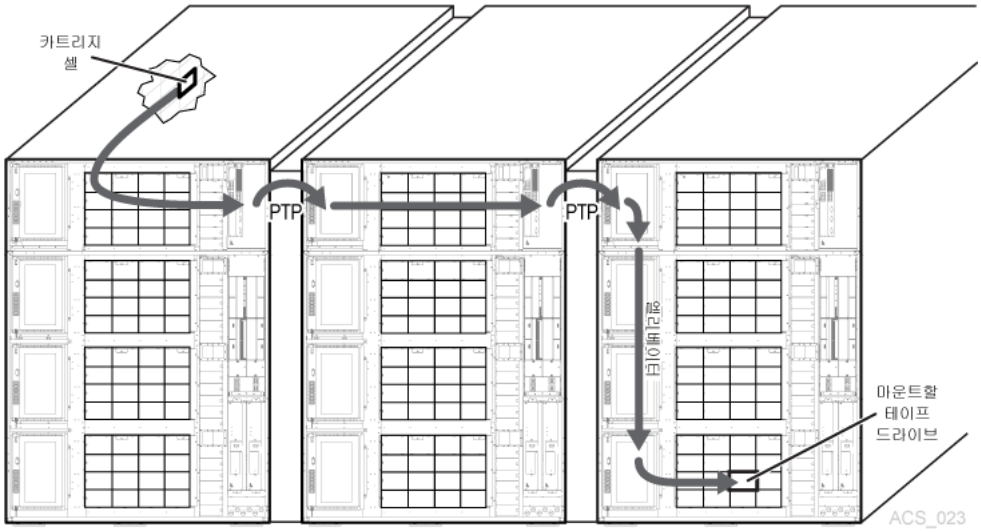
한 스트링에 있는 여러 SL8500에 연결하는 방법에 대한 자세한 내용은 “다중 TCP/IP 지원 ”을 참조하십시오.

LSM 간 전달 작업 최소화

테이프 라이브러리 성능을 향상시키기 위해서는 한 라이브러리에서 다른 라이브러리로의 전달 작업을 최소화하는 것이 중요합니다. 카트리지 및 드라이브가 연결된 라이브러리에 있는 경우, 드라이브 및 카트리지가 동일 LSM(라이브러리 내 레일)에 있을 때 테이프 마운트를 수행하는 것이 항상 좋습니다. 마운트 시 필요한 전달 작업이 적을수록 성능이 더욱 향상됩니다.

그림 L.1. “LSM 간 전달 작업 최소화 ”에서는 카트리지 및 드라이브가 다른 LSM에 있을 때 필요할 수 있는 전달 작업을 보여줍니다.

그림 L.1. LSM 간 전달 작업 최소화



자세한 내용은 “엘리베이터 및 PTP 작업 최소화 ”,“작업 로드를 지원하도록 테이프 드라이브 구성 ” 및 “카트리지 위치 관리 ”를 참조하십시오.

마운트 해제 중 카트리지 Float 작업

카트리지가 마운트 해제될 때 ACSLS는 카트리지의 이전 홈 셀이 드라이브와 다른 LSM에 있을 때마다 새 홈 셀을 지정함으로써 LSM 간의 전달 작업을 방지하려고 노력합니다. 이 동작은 ACSLS의 기본 동작입니다. 이 기능을 사용 안함으로 설정하려면 LSM에 대해 "Extended Store" 기능을 사용합니다. 마운트 해제 중인 카트리지는 LSM 간 이동을 방지하기 위해 새 홈 셀로 "Float"됩니다.

LSM에서 비어 있는 스토리지 셀 유지 관리

ACSLS는 모든 스토리지 셀이 가득 찬 경우 마운트 해제 시에 카트리지를 LSM으로 "Float"할 수 없습니다. 마찬가지로, 카트리지를 꽉 찬 LSM에 넣을 때는 카트리지를 셀이 비어 있는 LSM으로 전달해야 합니다.

성능을 최적화하기 위해서는 StreamLine 액세스를 위해 라이브러리에 유지할 필요가 없는 카트리지를 식별하고 라이브러리에서 그러한 카트리지를 꺼내야 합니다. 또한 모든 LSM에 빈 스토리지 셀이 포함되도록 가득 찬 LSM에서 빈 셀이 충분히 많은 다른 LSM으로 카트리지를 이동할 수도 있습니다. “활성 LSM에서 가장 오래 전에 액세스한 카트리지 이동 ”을 참조하십시오.

주:

라이브러리에서 카트리지 관리는 라이브러리에서 마운트 요청 작업이 없을 때 수행되어야 합니다.

마운트 및 마운트 해제 시간 초과

ACSLS와 클라이언트 응용 프로그램 사이에는 마운트 및 마운트 해제에 대한 시간 초과를 조정해야 합니다. 클라이언트의 시간 초과는 SL8500 레일과 라이브러리 간 전달이 필요한 마운트 및 마운트 해제가 완료될 수 있도록 충분히 길어야 합니다. 특히 여러 요청이 전달 포트를 기다릴 수 있는 경우에는 이러한 여러 동시 요청이 완료될 수 있도록 충분히 길어야 합니다.

ACSLS는 또한 임시 라이브러리 또는 테이프 드라이브가 중단된 동안 요청을 자동으로 대기열에 넣고 라이브러리 또는 드라이브를 사용할 수 있을 때 이를 다시 시도합니다. 이러한 방식은 중복 전자 부품 전환 중에, 라이브러리 또는 드라이브를 재부트할 때 또는 라이브러리의 액세스 도어가 열려 있을 때 요청이 실패하지 않도록 방지합니다.

마운트 및 마운트 해제를 대기열에 넣는 기본 시간(*MOUNT_RETRY_TIME_LIMIT*)은 20분입니다. 고객은 언제라도 이 값을 5에서 80분 사이의 값으로 설정할 수 있습니다. ACSLS는 또한 모든 *MOUNT_RETRY_DELAY*를 검사해서 대기열에 넣은 요청을 처리하기 위해 라이브러리를 사용할 수 있는지 여부를 확인합니다. 이러한 모든 변수는 동적 변수입니다. 즉, ACSLS가 실행되는 동안 변경할 수 있으며, 변경사항은 즉시 적용됩니다.

대규모 라이브러리 컴플렉스의 요청이 완료될 수 있도록 ACSAPI 클라이언트의 요청을 조정하십시오. 또한 ACSAPI 클라이언트의 마운트 및 마운트 해제 시간 초과를 사용해서 *MOUNT_RETRY_TIME_LIMIT*를 조정합니다.

라이브러리에 카트리지 넣기

카트리지를 라이브러리에 넣으면 CAP에서 가장 가까운 LSM에서 각 카트리지에 홈 셀이 지정됩니다. CAP가 있는 LSM에 빈 셀이 있으면 이 LSM에서 홈 셀이 지정됩니다. CAP가 있는 LSM이 가득 찼으면 빈 셀이 있는 가장 가까운 LSM에서 홈 셀이 지정됩니다.

성능을 최적화하기 위해서는 카트리지와 호환되는 테이프 드라이브가 있는 LSM에 카트리지를 넣으십시오. 예를 들어, 9940 카트리지는 T9840B 테이프 드라이브만 있는 LSM이 아니라 T9940B 테이프 드라이브가 있는 LSM에 넣습니다. 그렇지 않으면 9940 카트리지를 T9940B 테이프 드라이브가 있는 LSM으로 전달해야 합니다.

라이브러리에서 카트리지 꺼내기

라이브러리에서 카트리지를 꺼낼 때는 해당 카트리지를 지정된 CAP로 이동해야 합니다.

성능을 최적화하기 위해서는 카트리지가 있는 LSM의 CAP로 카트리지를 꺼냅니다. 카트리지가 여러 LSM에 배치된 경우, 대부분의 카트리지에서 가장 가까운 CAP를 선택합니다.

ACSAPI 요청 및 ACSLS 명령을 사용해서 전달 작업 최소화

이러한 ACSAPI 요청 및 ACSLS *cmd_proc* 명령은 LSM 간의 불필요한 전달 작업을 방지하는 데 도움이 될 수 있습니다.

다음은 클라이언트에서 ACSLS로의 ACSAPI 요청 및 *cmd_proc*를 사용해서 입력된 ACSLS 명령에 모두 적용됩니다. "요청"이라는 용어는 ACSAPI 요청 및 *cmd_proc* 명령 모두에 사용됩니다.

특정 카트리지 마운트

이 항목은 사용자가 선택한 데이터 카트리지 또는 특정 스크래치(비어 있는) 카트리지를 마운트하는 데 적용됩니다. 다음 명령을 사용합니다.

- query mount
- mount

query mount

- 지정된 카트리지(vol_id)에 대해 카트리지와 가까운 순서로 카트리지의 매체 유형과 호환되는 드라이브 목록을 반환합니다. (가장 가까운 LSM에 있는 호환되는 드라이브가 먼저 나열됩니다.)
- ACSLS 7.3 이상 릴리스에서는 카트리지에서 동일한 전달 거리에 있는 드라이브는 가장 오래 전에 사용된 순서로 나열됩니다.

  예제: 카트리지의 LSM 내에서 카트리지가 마운트 해제된 이후 가장 오래된 호환되는 드라이브가 먼저 표시되고, 그 다음으로 오래된 드라이브가 두번째로 표시됩니다.

명령: *query mount* vol_id

mount

가능한 한 목록 맨 위에서 가까운 사용 가능한(사용 중이 아닌) 드라이브를 선택해서 특정 카트리지를 마운트합니다.

ACSLS에서 선택된 스크래치 카트리지 마운트

다음 절에서는 ACSLS에서 선택된 스크래치 카트리지 마운트에 대해 설명합니다.

query mount scratch(query mount *라고도 부름)

지정된 스크래치 풀에 대해 지정된 스크래치 풀에 있는 카트리지의 매체 유형과 호환되는 모든 드라이브 목록이 반환됩니다. 특정 매체 유형을 지정하여 지정된 매체 유형과 호환되는 항목으로만 드라이브를 제한할 수 있습니다.

반환된 드라이브 목록은 순서에 따라 정렬되며, 가장 밀도가 높은 스크래치 풀에 가장 가까운 드라이브가 먼저 표시됩니다.

명령:

- *ACSAPI*
- *cmd_proc*
- *query mount scratch*

*query mount * pool_id [media media_type]*

특정 매체 유형 또는 *ALL_MEDIA_TYPE*(*cmd_proc* 명령의 *media **)을 지정할 수 있습니다. *ANY_MEDIA_TYPE*은 지원되지 **않습니다**.

주:

*ALL_MEDIA_TYPE*이 지정된 경우 가장 밀도가 높은 스크래치 풀을 확인하기 위해 드라이브와 호환되는 매체가 있는 카트리지가 선택됩니다.

mount scratch(query mount *라고도 부름)

지정된 드라이브에 대해 스크래치(비어 있는) 카트리지를 선택하고 이를 마운트합니다. 선택적으로, 지정된 스크래치 풀에서 카트리지를 선택하거나 및/또는 지정된 매체 유형의 카트리지를 선택합니다. ACSLS는 드라이브에 가장 가까운 LSM에 있는 호환되는 스크래치 카트리지를 선택합니다. 모든 카트리지를 돌아가면서 사용하기 위해, 선택한 LSM 내에서 액세스 날짜가 가장 오래된 호환되는 카트리지가 선택됩니다.

명령:

- *ACSAPI*

  cmd_proc

  mount scratch

  *mount * drive_id [pool_id] [media media_type]*
- ACSAPI 요청의 경우에는 drive_id, pool_id 및 *media_type*을 지정해야 합니다. (media_type에는 특정 매체 유형, *ALL_MEDIA_TYPE* 또는 *ANY_MEDIA_TYPE*을 지정할 수 있습니다.)
- *cmd_proc* mount * 명령의 경우 pool_id가 지정되지 않았으면 기본적으로 일반 풀(풀 0)로 지정됩니다.

  특정 매체 유형을 식별할 수 있습니다. *media **를 지정하면 *ANY_MEDIA_TYPE*이 선택됩니다. 매체를 지정하지 않으면 *ALL_MEDIA_TYPE*이 선택됩니다.

특수 media_type 값 ANY_MEDIA_TYPE 및 ALL_MEDIA_TYPE은 다음과 같이 처리됩니다.

- *ALL_MEDIA_TYPE*이 지정되었으면 매체 유형이 드라이브와 호환되는 카트리지가 선택됩니다. (이 동작은 *media_compatibility* 파일을 기반으로 합니다.)
- *ANY_MEDIA_TYPE*이 지정되었으면 *scratch_preferences* 파일이 드라이브에 마운트할 선호 매체 목록을 식별합니다.

ACSAPI 요청 및 ACSLS 명령 사용

이 절에서는 테이프 라이브러리 성능 향상을 위해 ACSAPI 요청 및 명령을 사용하는 방법에 대해 설명합니다.

다음 설명 중에서 "요청"이라는 용어는 ACSAPI 요청 및 *cmd_proc* 명령 모두에 사용됩니다.

특정 카트리지를 마운트하는 경우(*vol_id*가 알려진 경우):

- *mount* 요청 앞에 *query mount* 요청을 둡니다.
- 첫번째 "사용 가능한" 드라이브를 선택하고, *mount* 요청에서 이 드라이브를 지정합니다.

특정 드라이브에 스크래치 카트리지를 마운트할 때는 다음 두 옵션을 선택할 수 있습니다.

- 특정 드라이브에 가장 가까운 스크래치 카트리지를 선택하려는 경우:

  mount scratch 요청을 사용해서 드라이브와 스크래치 풀(선택사항)을 지정합니다.

  ACSAPI 요청의 경우에는 다음 중 하나를 지정합니다.
  - *ALL_MEDIA_TYPE*(호환되는 매체가 선택됨)
  - *ANY_MEDIA_TYPE*(스크래치 선호 목록 사용).

  cmd_proc 명령의 경우에는 다음 중 하나를 선택합니다.
  - 매체 유형 지정 안함(호환되는 매체가 선택됨)
  - *media ** 지정(스크래치 선호 목록 사용)
- 드라이브 목록에서 특정 드라이브를 선택하고 가장 가까운 스크래치 카트리지를 선택하려는 경우:

  드라이브를 선택한 후 스크래치 테이프 선택: 이렇게 하면 스크래치 카트리지에 가까운 드라이브가 선택됩니다.
  - *query mount scratch* 요청을 입력하여 원하는 스크래치 풀에서 대부분의 스크래치 매체에 사용 가능한 가장 가까운 드라이브를 식별합니다.
  - *mount scratch* 요청을 입력하여 선택한 드라이브에서 스크래치 카트리지를 마운트합니다. 선택적으로, 스크래치 풀을 지정합니다.

  ACSAPI 요청의 경우 다음 중 하나를 지정합니다.
  - *ALL_MEDIA_TYPE*(호환되는 매체가 선택됨) 또는
  - *ANY_MEDIA_TYPE*(스크래치 선호 목록 사용)

  cmd_proc 명령의 경우에는 다음 중 하나를 수행합니다.
  - 매체 유형 지정 안함(호환되는 매체가 선택됨) 또는
  - *media ** 지정(스크래치 선호 목록 사용)

부록 M

부록 M. 방화벽 보안 옵션

방화벽 보안 옵션은 클라이언트 소프트웨어가 방화벽을 통해 요청을 수행하는 동안 방화벽 뒤에서 ACSLS가 실행되도록 지원합니다.

방화벽 보안은 또한 ACSLS 클라이언트에도 제공되어, 클라이언트가 해당 방화벽 뒤에서 작동할 수 있도록 허용합니다. 이러한 기능은 오라클에서 해당 ISV(독립 소프트웨어 공급업체) 파트너에게 제공합니다. 각 특정 클라이언트의 최신 상태를 확인하려면 해당 클라이언트 소프트웨어 구성 요소의 ISV에 문의하십시오.

방화벽 뒤에서 ACSLS 실행

이 방화벽 보안 솔루션은 다음과 같은 이점을 제공합니다.

- ACSLS가 방화벽 뒤에서 실행되도록 지원합니다(방화벽의 보안 측면에서 ACSLS를 실행하고, 반대쪽에서는 클라이언트 실행).
- ACSLS 클라이언트가 고유 방화벽 뒤에서 실행되도록 지원합니다(보안 측면에서 클라이언트를 실행하고, 방화벽의 반대쪽에서는 ACSLS 실행).

  주:

  ISV는 자신의 클라이언트측 소프트웨어 구성 요소 내에서 제공된 변경사항을 구현해야 합니다.

- 현재 ACSLS 클라이언트 구현과의 호환성을 보존하여, 이러한 클라이언트가 방화벽 솔루션에서 ACSLS와 함께 계속 실행되도록 허용합니다.
- 현재 ACSAPI/클라이언트 기능 및 성능을 보존합니다. 여기에는 비방화벽 환경에서 제공되는 모든 기능이 포함됩니다.

전체 솔루션에는 위에 설명한 처음 2개의 기능이 결합되어 있습니다. 이를 통해 ACSLS 및 ACSLS 클라이언트는 서로 고유의 방화벽 뒤에서 실행될 수 있으며(ACSLS와 클라이언트 사이에 2개의 방화벽이 있음), 비방화벽 환경과 동일한 통신 성능을 보유할 수 있습니다.

보안 영역 문제 해결

ACSLS는 다음과 같은 보안 문제를 해결했습니다.

RPC

ACSLS 내에서의 RPC 사용은 방화벽 환경 내에서 실행을 시도할 때 일부 사이트에서 문제가 될 수 있습니다. 현재 설치된 클라이언트 기반과의 호환성을 보존하기 위해서는 ACSLS에서 완전한 RPC 제거가 불가능합니다.

ACSLS 방화벽 보안 기능은 다음과 같이 RPC에 내재된 문제들을 해결합니다.

- 외부(신뢰할 수 없는) 상대가 제한되지 않는 포트 범위(1024-65535)에서 신뢰할 수 있는 호스트에 대해 연결을 시작할 수 있도록 허용해야 합니다.
- 잘 알려진 포트 111에서 실행되는 portmap(또는 *rpcbind*) 데몬을 통해 플랫폼에서 사용 가능한 서비스 매핑을 노출합니다.

보안

방화벽 솔루션에서 기본적인 보안은 비보안 영역에서 신뢰할 수 있는(보안) 영역으로의 액세스를 제한하는 것부터 시작됩니다. 어떤 경우라도 통신을 수행하고 데이터 교환을 허용하기 위해서는 일부 제한적이고 제어되는 액세스를 허용해야 합니다. 이를 위해서는 잘 정의되고 제한된 일련의 시작점 내에서 데이터 교환을 허용하여 액세스 지점 및 해당 통신을 제어할 수 있어야 합니다. 이 솔루션을 통해 이러한 목표를 충족시킬 수 있습니다.

주:

IPv4 기반의 경계면 방화벽을 사용하는 경우, IPv6-over-IPv4 터널링 트래픽이 내부 호스트에 도달할 수 없도록 모든 아웃바운드 IPv4 프로토콜 41 패킷과 UDP 포트 3544 패킷을 드롭하도록 구성되어야 합니다.

통신 구성 요소

ACSLS/클라이언트 통신은 2개의 네트워크 인터페이스 구성 요소를 사용해서 클라이언트 플랫폼과 ACSLS 플랫폼 사이의 네트워크 통신을 처리합니다. ACSLS에 대해 클라이언트 또는 프록시 서버로 작동하는 소프트웨어는 ACSLS 플랫폼 및 기존 클라이언트와 호환되도록 이러한 두 가지 구성 요소 중 하나를 구현합니다. 클라이언트 플랫폼에 상주하는 구성 요소를 SSI라고 부르고, ACSLS 플랫폼에 상주하는 구성 요소를 CSI라고 부릅니다. 클라이언트 호환성을 유지 관리하고 모든 방화벽 보안 기능을 제공하기 위해서는 하나의 위치(예: ACSLS 플랫폼) 내에서 모든 변경사항을 구현하는 것이 바람직하지만, 효과적인 사용을 위해서는 해당 변경사항을 각 위치에 모두 적용해야 합니다. 이러한 방식의 한 가지 장점은 각 위치가 기능을 독립적으로 구현하고 각 고유 위치에서 방화벽 보안 이점을 얻을 수 있다는 것입니다(예: ACSLS에 대한 변경으로 ACSLS 플랫폼을 보안 방화벽 뒤에서 실행 가능).

방화벽 보안 옵션의 이점

이 절에서는 방화벽 보안 옵션의 이점에 대해 설명합니다.

ACSLS 서버측

이 방화벽 보안 솔루션 내에 제공된 대로 서버측 구성 요소만 변경된 경우, 이점은 다음과 같습니다.

- 등록된 모든 프로그램 번호에 대해 ACSLS 통신을 위한 수신 연결을 단일 TCP 포트로 제한합니다. ACSLS CSI에 대해 등록된 프로그램 번호가 2개이고, 두 번호 모두 단일 포트에서 서비스됩니다.
- 사용자가 해당 포트의 ID를 지정하고 해당 방식으로 방화벽을 구성할 수 있습니다.
- 사용자가 UDP 포트에 대한 ACSLS 통신을 해제할 수 있습니다.

- 사용자가 **클라이언트측** portmapper*에 대한 ACSLS 서버의 통신을 모두 사용 안함으로 설정할 수 있습니다(*UDP/TCP port 111*). 클라이언트측 코드와의 호환성을 보존하기 위해서는 portmapper가 클라이언트 플랫폼에서 계속 실행 중인 상태로 유지되어야 합니다. 하지만, 서버에서 시작되는 네트워크 통신에는 사용되지 않으므로, 이에 대한 액세스를 허용하지 않도록 클라이언트 방화벽을 구성할 수 있습니다.
- ACSLS 서버측에서 클라이언트로의 송신 연결이 제한되지 않으므로 서버측 포트를 사용하여 현재 성능을 보존할 수 있습니다. 이 방식은 보안 커뮤니티에서 사용되는 일반적인 방식을 따릅니다.

ACSLS 서버 포트 제한

이 방화벽 솔루션은 외부 상대가 네트워크 통신을 시작할 수 있는 수신 포트 수를 제한합니다. 포트는 1개 또는 3개로 제한됩니다. ACSLS 수신 요청에 대한 단일 고객 지정 포트가 하나 있고, 여기에 더해 2개의 portmapper 포트(TCP 및 UDP 포트 111)를 지정할 수 있습니다.

주:

ACSLS 서버 portmapper에 대한 클라이언트 액세스를 금지하기 위해 UDP 및 TCP 포트 111에 대한 액세스를 금지하려면 **클라이언트** 소프트웨어 구성 요소를 변경해야 합니다. 아래에서 클라이언트측 설명을 참조하십시오.

위에서 이 솔루션의 서버측은 ACSLS 내에서 완전하게 구현됩니다.

클라이언트측(CSC)

CSC에 대한 변경은 위에서 설명한 것과 동일한 제한 사항을 클라이언트측 플랫폼에 적용합니다. 그 결과 CSC도 동일하게 고유 보안 방화벽 뒤에 상주할 수 있게 됩니다. 이 솔루션은 다음과 같은 이점을 제공합니다.

- 등록된 각 프로그램 번호에 대해 CSC 통신을 위한 수신 연결(응답)을 단일 TCP 포트로 제한합니다. ACSLS SSI에 대해서는 등록된 프로그램 번호가 하나 있습니다.
- 최종 사용자는 TCP 포트의 ID를 지정하고 비슷한 방식으로 해당 방화벽을 구성할 수 있습니다.
- UDP 포트에 대한 클라이언트측 통신을 해제합니다.
- **ACSLS 서버** portmapper(UDP/TCP 포트 111)에 대한 클라이언트의 모든 통신을 사용 안함으로 설정합니다. ACSLS 코드와의 호환성을 보존하기 위해 portmapper는 ACSLS 플랫폼에서 계속 실행 중인 상태로 유지되어야 합니다. 하지만, 클라이언트 네트워크 통신은 portmapper를 통해 시작되지 않습니다. 따라서 이에 대한 액세스를 허용하지 않도록 ACSLS 서버 방화벽을 구성할 수 있습니다.
- 클라이언트측에서 ACSLS 서버로의 송신 연결이 제한되지 않으므로 클라이언트측 포트를 사용하여 현재 성능을 보존할 수 있습니다.

클라이언트 포트 제한

이 솔루션은 외부 요소가 네트워크 통신을 시작할 수 있는 수신 포트 수를 제한합니다. 포트는 1개 또는 3개로 제한됩니다. 클라이언트 수신 응답에 대한 단일 고객 지정 포트가 하나 있고, 여기에 더해 2개의 portmapper 포트(TCP 및 UDP 포트 111)를 지정할 수 있습니다.

주:

클라이언트 portmapper에 대한 ACSLS 서버 액세스를 금지하기 위해 UDP 및 TCP 포트 111에 대한 액세스를 금지하려면 **ACSLS 서버** 소프트웨어 구성 요소를 변경해야 합니다(위의 ACSLS 서버측 설명 참조).

이 솔루션은 2단계로 구현됩니다.

- Oracle StorageTek에서는 CSC Developer's Toolkit 2.3(이상) 소스 코드에서 필요한 코드가 변경되었습니다.
- 클라이언트 플랫폼에 대해 이 보안을 제공해야 하는 ACSLS의 클라이언트는 클라이언트 측 SSI 코드에 이러한 변경사항을 통합하고, 제품을 재구축하고, 다시 ACSLS에서 해당 CSC(클라이언트 시스템 구성 요소)를 인증해야 합니다.

이점

솔루션의 클라이언트측 및 서버측 부분은 서로 독립적입니다. 따라서 둘 중 하나만 방화벽 뒤에 있을 경우, 나머지 한쪽에서만 소프트웨어 변경을 구현하면 됩니다. 또한 한쪽만 변경하기 때문에 기존에 존재하는 모든 현재 클라이언트 및 서버 구현과의 호환성이 유지되고, CSI/SSI 인터페이스를 사용하는 다른 소프트웨어 구성 요소와의 호환성도 유지됩니다.

주:

여기에는 현재 Oracle StorageTek 제품과의 호환성이 포함됩니다.

이 솔루션은 클라이언트/서버 통신의 현재 성능에 영향을 주지 않습니다.

방화벽 보안 기능 켜기 및 변수 설정

방화벽 뒤에서 ACSLS 서버를 실행하고, 선택적으로 방화벽 뒤에서 ACSLS 클라이언트를 실행하려면 ACSLS 서버 및 클라이언트 시스템이 방화벽 뒤에 있을 경우 두 시스템 모두에 변수를 설정합니다. 이러한 변수를 사용하면 수신 통신을 단일 포트로 제한하고, 선택적으로 portmapper를 사용 안함으로 설정할 수 있습니다.

ACSLS 변수

`CSI_TCP_RPCSERVICE` - TCP 프로토콜을 사용한 RPC에 대한 CSI 지원을 사용으로 설정합니다.

- **기능:** CSI가 TCP RPC 서버로 작동하도록 설정합니다. 클라이언트가 TCP를 통해 ACSLS와 통신해야 할 경우 이 옵션을 TRUE로 설정합니다.
- **유효한 옵션:** TRUE 또는 FALSE(TRUE가 기본값)
  - TRUE는 CSI에 대한 클라이언트의 TCP 액세스를 사용으로 설정합니다.
  - FALSE는 CSI에 대한 클라이언트의 TCP 액세스를 사용 안함으로 설정합니다.
- **기타 세부정보:** 이 옵션을 적용하려면 ACSLS 제품을 다시 시작해야 합니다.

`CSI_UDP_RPCSERVICE` - UDP 프로토콜을 사용한 RPC에 대한 CSI 지원을 사용으로 설정합니다.

- **기능:** 이 옵션을 선택하면 CSI가 UDP RPC 서버로 작동하도록 설정됩니다. 클라이언트가 UDP를 통해 ACSLS와 통신해야 할 경우 이 옵션을 TRUE로 설정합니다.
- **유효한 옵션:** TRUE 또는 FALSE(FALSE가 권장됨)
  - TRUE는 CSI에 대한 클라이언트의 UDP 액세스를 사용으로 설정합니다.
  - FALSE는 CSI에 대한 클라이언트의 UDP 액세스를 사용 안함으로 설정합니다.
- **기타 세부정보:** 이 옵션을 적용하려면 ACSLS 제품을 다시 시작해야 합니다. 방화벽 보안 CSI는 TCP 통신에 대해서만 지원됩니다. ACSLS 서버에서 방화벽 내에 레거시 클라이언트 응용 프로그램이 있지 않은 한 *CSI_UDP_RPCSERVICE*를 FALSE로 설정합니다.

CSI_USE_PORTMAPPER – portmapper를 사용으로 설정합니다.

- **기능:** 이 옵션을 선택하면 클라이언트에 응답을 보낼 수 없을 때 CSI가 portmapper를 조사합니다. 클라이언트에서 portmapper에 대한 액세스를 허용하지 않으려는 경우 이 옵션을 ALWAYS로 설정합니다.
- **유효한 옵션:** ALWAYS, NEVER 또는 IF_DUAL_LAN_NOT_ENABLED
  - ALWAYS - CSI가 클라이언트에 응답을 전송할 수 없을 때 항상 portmapper를 조사합니다.
  - NEVER - CSI가 클라이언트에 응답을 전송할 수 없을 때 portmapper를 조사하지 않습니다. 이 옵션은 클라이언트가 포트 매퍼를 지원하지 않을 때 선택해야 합니다.
  - *IF_DUAL_LAN_NOT_ENABLED* - 이중 LAN 지원이 사용 안함으로 설정된 경우에만 portmapper를 조사하도록 지정합니다. 이중 LAN 지원이 사용으로 설정된 경우 클라이언트가 포트 매퍼를 지원하지 않는 것으로 가정합니다. *IF_DUAL_LAN_NOT_ENABLED*는 이전 버전과의 호환성을 위한 기본값입니다.
- **기타 세부정보:** 이 옵션을 적용하려면 ACSLS 제품을 다시 시작해야 합니다.

CSI_FIREWALL_SECURE - 방화벽 뒤에서 CSI를 사용하도록 설정합니다(사용자 정의된 인바운드 포트 사용).

- **기능:** 이 옵션은 ACSLS 서버가 보안 방화벽 뒤에서 작동하도록 설정합니다. ACSLS에서 사용되는 인바운드 포트를 지정하고 액세스를 단일 포트로 제한합니다. 이 포트를 제외한 모든 포트에서 수신되는 ACSLS 트래픽을 거부하도록 방화벽을 구성합니다. 이렇게 하면 ACSLS와 통신을 시작하려는 외부 클라이언트가 해당 포트만 사용할 수 있습니다.

  포트 액세스를 제한하려면 다음 단계를 완료해서 지정된 포트에 대한 보안 방화벽을 설정합니다.
  - 이 옵션을 TRUE로 설정합니다.
  - 수신되는 ACSLS 요청을 허용할 CSI에서 사용되는 포트를 지정합니다. (*CSI_INET_PORT*로 지정됩니다.)
  - 고정된 포트 RPC를 지원하지 않는 일부 레거시 클라이언트 응용 프로그램의 경우, 클라이언트에서 portmapper 질의 요청을 지원하려면 방화벽에서 UDP/TCP 포트 111을 열어야 할 수 있습니다.
  - 방화벽 보안 CSI는 TCP 통신에 대해서만 지원됩니다.

    ACSLS 서버에서 방화벽 내에 레거시 클라이언트 응용 프로그램이 있지 않은 한 *CSI_UDP_RPCSERVICE*를 FALSE로 설정합니다.

  - ACSLS 서버가 상주하는 방화벽을 구성해서 외부 클라이언트가 이전에 지정된 포트에서 통신을 시작 및 수신하도록 허용합니다. 열린 방화벽 포트를 최소화하기 위해 클라이언트 응용 프로그램에서 동일한 포트로 고정 포트를 설정하는 것도 잊지 않아야 합니다.
  - ACSLS를 다시 시작합니다.
- **유효한 옵션:** TRUE 또는 FALSE(기본값 TRUE)
  - TRUE – 클라이언트의 수신 요청에 대해 단일 포트만 사용하도록 ACSLS 서버에 대한 액세스를 제한합니다.
  - FALSE – ACSLS 서버에 대한 클라이언트 요청에 사용되는 포트를 제한하지 않습니다.
- **기타 세부정보:** 이 옵션을 적용하려면 ACSLS 제품을 다시 시작해야 합니다.

CSI_INET_PORT - 수신되는 ACSLS 요청을 수신하기 위해 CSI에서 사용되는 포트 번호입니다.

- **기능:** 이 옵션은 클라이언트에서 수신되는 TCP 요청에 대해 CSI가 사용하는 포트를 지정합니다.
- **유효한 옵션:** 50003을 제외하고 1024~65535 사이의 번호입니다. (기본값 30031)
- **기타 세부정보:** 이 옵션을 적용하려면 ACSLS 제품을 다시 시작해야 합니다. 이 변수는 방화벽 보안 CSI가 *CSI_FIREWALL_SECURE*로 사용으로 설정되었고 TRUE로 설정된 경우에만 사용됩니다.

ACSLS 변수 표시 및 설정

ACSLS *acsss_config* 유틸리티 또는 *dv_config* 유틸리티를 사용해서 ACSLS 정적 및 동적 변수를 표시하고 설정합니다.

- *dv_config –d*

  모든 ACSLS 정적 및 동적 변수와 해당 설정을 표시합니다.
- *dv_config -p* <*variable_name*> *-u*

  변수 변경에 대한 프롬프트가 표시되거나, 동적 변수인 경우 ACSLS 전역 공유 메모리 업데이트에 대한 프롬프트가 표시됩니다. 변수에 대한 자세한 설명을 보려면 프롬프트에 ? 를 입력합니다. 자세한 설명이 표시된 다음 다시 프롬프트가 표시됩니다.

ACSAPI 클라이언트 시스템 변수

클라이언트 시스템에 대해 방화벽 보안 작업을 사용으로 설정하려면 ACSLS CSC Toolkit 2.3(이상) 또는 이후 릴리스를 사용해서 ACSLS 클라이언트 시스템을 구축해야 합니다.

ACSLS 클라이언트에서 방화벽 보안 작업을 위해 사용으로 설정할 환경 변수는 4개입니다. 이러한 변수는 특정 값으로 설정해야 합니다. SSI 프로세스를 시작하기 전에 이러한 각 변수를 설정하고 SSI 환경으로 내보내야 합니다. 그런 다음 아래에 표시된 것처럼 SSI에서 변수가 해석되고 사용됩니다.

CSC가 SSI 시작을 위해 스크립트를 사용하는 경우에는 해당 스크립트 내에서 이러한 변수를 설정하고 내보내는 것이 좋습니다. 또한 클라이언트 개발자가 CSC 및 그 실행 환경에 따라 최종 고객이 이를 올바르게 구성할 수 있는 방법을 제공했을 수도 있습니다.

CSI_UDP_RPCSERVICE – 네트워크 통신에 UDP를 사용할지 여부를 결정합니다.

- **기능:** SSI 네트워크 통신을 위한 기본 네트워크 통신 계층으로 UDP를 사용/사용 안함으로 설정합니다.
- **유효한 옵션:** TRUE 또는 FALSE
- **기타 세부정보:** 방화벽 보안 CSC의 경우 이 환경 변수를 FALSE로 설정해야 합니다. 방화벽 보안 ACSLS 응용 프로그램 패킷은 모두 TCP 네트워크 전송을 사용해서 전송됩니다.

CSI_TCP_RPCSERVICE – 네트워크 통신에 TCP를 사용할지 여부를 결정합니다.

- **기능:** SSI 네트워크 통신을 위한 기본 네트워크 통신 계층으로 TCP를 사용/사용 안함으로 설정합니다.
- **유효한 옵션:** TRUE 또는 FALSE
- **기타 세부정보:** 방화벽 보안 CSC의 경우 이 환경 변수를 TRUE로 설정해야 합니다. 방화벽 보안 ACSLS 응용 프로그램 패킷은 모두 TCP 네트워크 전송을 사용해서 전송됩니다.

CSC Toolkit 2.3의 새 변수

SSI_INET_PORT – 수신 응답을 위한 고정 포트 번호입니다.

- **기능:** SSI가 수신 ACSLS 응답을 위해 사용할 포트를 지정합니다.
- **유효한 옵션:** 0 또는 1024–65535(50001 및 50004 제외)
  - 0은 동적 포트 할당을 허용하는 이전 동작이 계속 적용됨을 나타냅니다.
  - 1024 –65535는 SSI가 ACSLS 응답을 수락하는 TCP 포트로 해당 번호가 사용됨을 의미합니다.
  - 50001 또는 50004는 *mini_el* 및 SSI에서 사용되므로 지정하지 않습니다.
- **기타 세부정보:** 이 환경 변수를 0이 아닌 값으로 설정하면 SSI가 이 포트를 수신 ACSLS 응답에 대해 사용합니다. 즉, ACSLS 응답을 SSI가 수신할 수 있으려면 방화벽이 해당 포트에서 수신 요청을 허용해야 합니다. 이 포트는 ACSLS가 CSC의 SSI와 연결을 시작하는 유일한 포트입니다.

  주:

  이 값은 이 포트에서 수신되는 연결 요청을 허용하도록 CSC 플랫폼을 보호하기 위해 방화벽에 구성된 값과 일치해야 합니다.

CSI_HOSTPORT – ACSLS 서버에서 portmapper에 대한 질의를 제거합니다. 대신 ACSLS 서버에서 요청을 이 포트로 전송합니다.

- **기능:** ACSLS 서버에서 SSI가 해당 ACSLS 요청을 전송하는 포트를 지정합니다. ACSLS CSI는 CSC에서 인바운드 ACSLS 요청을 수락하기 위해 이 포트를 사용해야 합니다. 즉, 방화벽 보안 고정 포트를 이 값과 동일하게 설정해야 합니다.

- **유효한 옵션:** 1024 –65535(50003, 및 0 제외)(이 값은 인바운드 패킷용으로 CSI에서 사용되는 포트에 대해 ACSLS 서버에 설정된 값과 일치해야 함)
  - 0은 ACSLS 서버에서 portmapper를 질의하는 이전 동작이 계속 사용됨을 의미합니다.
  - 1024 –65535는 수신 요청에 대해 CSI에서 사용되는 값을 의미합니다.
  - 50003은 acslm에서 사용되므로 지정하지 않습니다.
- **기타 세부정보:** 이 환경 변수를 설정하면 SSI에서 ACSLS 서버 portmapper로의 질의가 제거됩니다. 이 변수 값은 SSI가 송신 ACSLS 요청을 전송할 ACSLS 서버의 포트 번호를 지정합니다. 그 결과 방화벽 보호 ACSLS 서버가 해당 방화벽에서 portmapper에 대한 액세스를 금지할 수 있습니다. portmapper 질의는 이전에 SSI가 ACSLS 요청을 전송해야 하는 포트 번호를 제공했습니다.

  주:

  이 값은 수신 요청을 수락 및 서비스하기 위해 CSI가 사용하는 포트 값과 일치해야 합니다. 이 포트가 지정된 값에서 안정적으로 고정된 상태로 유지될 수 있도록 방화벽 보안 기능을 ACSLS에 적용해야 합니다. 항목이 일치하지 않으면, CSC와 ACSLS 사이에 통신이 수행되지 않습니다.

클라이언트에서 환경 변수 표시 및 설정

클라이언트에서 환경 변수 설정에 사용되는 명령은 사용자의 셸 및 OS에 따라 달라집니다.

- UNIX 및 Linux의 경우에는 다음 명령을 사용해서 환경 변수를 표시합니다.

  echo $<variable-name>
- ksh 및 bash 셸의 경우에는 다음 명령을 사용해서 환경 변수를 설정할 수 있습니다.

  <environment_variable> = <value>

  export <environment_variable>

방화벽 보안 솔루션 시나리오

다음 다이어그램은 ACSLS 구성 요소의 작업, 포트 사용 및 관계가 방화벽에서 사용될 때 발생 가능한 시나리오를 보여줍니다. 이러한 시나리오는 위에 설명한 내용을 이해하기 위한 목적으로 제공됩니다. 다음 다이어그램에서 "SSI"는 통신의 클라이언트측에서 실행되는 ACSLS의 네트워크 인터페이스 구성 요소입니다. CSI는 ACSLS 플랫폼에서 실행되는 ACSLS의 네트워크 인터페이스 구성 요소입니다.

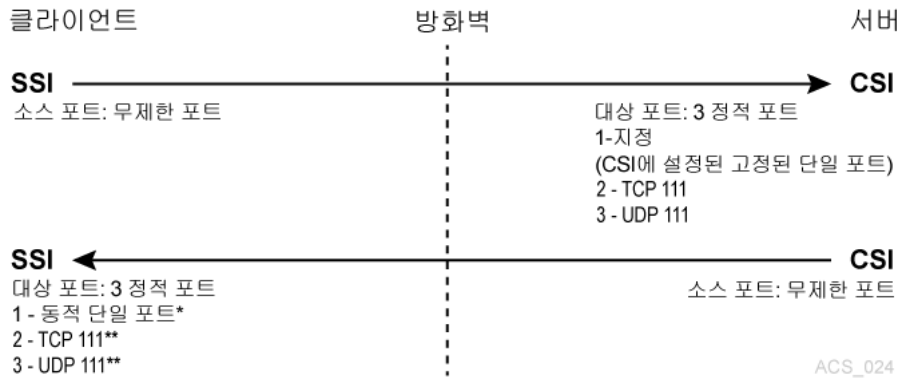
주:

이러한 시나리오를 지원하기 위해서는 ACSLS CSC Developer's Toolkit 2.3(이상) 및 새로운 환경 변수가 필요합니다.

ACSLS 서버측만의 방화벽 보안

이 예에서 방화벽 보안은 ACSLS 서버측(CSI)에서만 구현됩니다. CSC Toolkit 2.3(이상) 및 새로운 환경 변수는 이 시나리오를 지원하는 데 필요하지 **않습니다**.

그림 M.1. ACSLS 서버측만의 방화벽 보안



이 예에서 "동적"은 시작 시 1024-65535 범위로 SSI에서 포트가 선택됨을 의미합니다. 사용자가 포트를 지정하지 않으며, SSI를 새로 실행할 때마다 즉, SSI 실행 프로세스의 한 인스턴스가 다음 인스턴스로 바뀌어도 동일한 포트로 지정되지도 않습니다.

SSI측의 portmapper 111 포트는 CSI에서 드문 경우에만 질의됩니다. 이 포트는 응답 패킷을 다시 SSI로 전송할 때 해당 요청 패킷에서 SSI에 의해 제공된 반환 포트 번호가 작동하지 않아서 네트워크 인터페이스 오류가 발생하는 경우에만 CSI에서 액세스됩니다. 이 경우에는 재시도 메커니즘에 따라 CSI가 SSI측 portmapper에 사용할 포트(SSI의 프로그램 번호에 따라 portmapper에 등록됨)를 질의합니다.

방화벽 뒤에서 ACSLS를 보호하기 위해서는 다음 설정이 필요합니다.

- ACSLS: 설정을 변경한 후에는 ACSLS를 다시 시작해야 합니다.
  - *CSI_TCP_RPCSERVICE* = TRUE
  - *CSI_UDP_RPCSERVICE* = FALSE

    (하지만 ACSLS와의 통신을 위해 UDP를 사용하는 클라이언트가 있을 경우 TRUE로 설정해야 함)
  - *CSI_USE_PORTMAPPER* = ALWAYS(*IF_DUAL_LAN_NOT_ENABLED* 가능)
  - *CSI_FIREWALL_SECURE* = TRUE
  - *CSI_INET_PORT* = <1024-65535, 50003 제외> 기본값 30031
- 클라이언트 SSI 설정 - 클라이언트가 방화벽 뒤에서 실행되도록 허용하는 환경 변수입니다.
  - *CSI_TCP_RPCSERVICE* = TRUE
  - *CSI_UDP_RPCSERVICE* = TRUE(FALSE 가능)
  - *SSI_INET_PORT* = 0

    이 변수는 ACSLS CSC Developer's Toolkit 2.3의 새로운 환경 변수이며, 이 시나리오에서는 필요하지 않습니다.
  - *CSI_HOSTPORT* = 0 또는 <1024-65535, 50003 제외> 기본값 30031

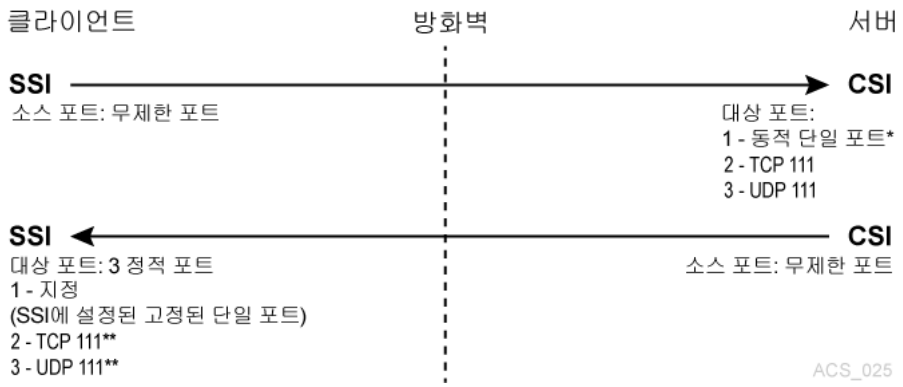
ACSLS 서버에서 portmapper를 사용할 경우에는 필요하지 않습니다. 0 이외의 값으로 정의된 경우에는 ACSLS 서버의 *CSI_INET_PORT*와 일치해야 합니다. 이 변수는 ACSLS CSC Developer's Toolkit 2.3(이상)의 새로운 환경 변수이며, 이 시나리오에서는 필요하지 않습니다.

CSI_INET_PORT(ACSLS 서버) 및 *CSI_HOSTPORT*(클라이언트)에 의해 지정된 포트를 사용해서 클라이언트가 ACSLS 서버에 요청을 전송할 수 있도록 방화벽을 구성합니다. 클라이언트가 ACSLS 서버에서 portmapper(포트 111)에 액세스하고 ACSLS가 클라이언트에서 portmapper(111)에 액세스할 수 있도록 허용합니다.

클라이언트측만의 방화벽 보안

이 예에서 방화벽 보안은 클라이언트측(SSI)에서만 구현됩니다. 이 시나리오를 지원하기 위해서는 CSC Toolkit 2.3(이상) 및 새로운 환경 변수가 필요합니다.

그림 M.2. 클라이언트측만의 방화벽 보안



이 예에서 "동적"은 시작 시 1024-65535 범위로 CSI에서 포트가 선택되며, 사용자가 포트를 지정하지 않으며, CSI를 새로 실행할 때마다 즉, CSI 실행 프로세스의 한 인스턴스가 다음 인스턴스로 바뀌어도 동일한 포트로 지정되지 않음을 의미합니다.

SSI측의 portmapper 111 포트는 CSI에서 드문 경우에만 질의됩니다. 이 포트는 응답 패킷을 다시 SSI로 전송할 때 해당 요청 패킷에서 SSI에 의해 제공된 반환 포트 번호가 작동하지 않아서 네트워크 인터페이스 오류가 발생하는 경우에만 CSI에서 액세스됩니다. 이 경우에는 재시도 메커니즘에 따라 CSI가 SSI측 portmapper에 사용할 포트(SSI의 프로그램 번호에 따라 portmapper에 등록됨)를 질의합니다.

방화벽 뒤에서 클라이언트 시스템을 보호하기 위해서는 다음 설정이 필요합니다.

- ACSLS: 설정을 변경한 후에는 ACSLS를 다시 시작해야 합니다.
  - *CSI_TCP_RPCSERVICE* = TRUE
  - *CSI_UDP_RPCSERVICE* = FALSE

    ACSLS와의 통신을 위해 UDP를 사용하는 클라이언트가 있을 경우, TRUE여야 합니다.

  - *CSI_USE_PORTMAPPER* = ALWAYS(*IF_DUAL_LAN_NOT_ENABLED* 가능)
  - *CSI_FIREWALL_SECURE* = FALSE
  - *CSI_INET_PORT* = 0
- 클라이언트 SSI 설정 - 클라이언트가 방화벽 뒤에서 실행되도록 허용하는 환경 변수입니다.
  - *CSI_TCP_RPCSERVICE* = TRUE
  - *CSI_UDP_RPCSERVICE* = FALSE(TRUE 가능)
  - *SSI_INET_PORT* = <1024 –65535, 50001 및 50004 제외>

    이 변수는 ACSLS CSC Developer's Toolkit 2.3(이상)의 새로운 환경 변수이며, 이 시나리오에서는 필요하지 않습니다.
  - *CSI_HOSTPORT* = 0 또는 <1024-65535, 50003 제외> 기본값 30031

    ACSLS 서버에서 portmapper를 사용할 경우에는 필요하지 않습니다. 0 이외의 값으로 정의된 경우에는 ACSLS 서버의 *CSI_INET_PORT*와 일치해야 합니다. 이 변수는 ACSLS CSC Developer's Toolkit 2.3(이상)의 새로운 환경 변수이며, 이 시나리오에서는 필요하지 않습니다.

다음을 허용하도록 방화벽을 구성해야 합니다.

- *CSI_INET_PORT*(ACSLS 서버) 및 *CSI_HOSTPORT*(클라이언트)에 의해 지정된 포트를 사용해서 클라이언트가 ACSLS 서버에 요청을 전송할 수 있어야 합니다.
- 클라이언트가 ACSLS 서버에서 portmapper(포트 111)에 액세스할 수 있어야 합니다.
- ACSLS 서버가 클라이언트에서 *SSI_INET_PORT*로 지정된 포트를 사용해서 클라이언트에 응답을 전송할 수 있어야 합니다.
- ACSLS 서버가 포트 111을 사용해서 클라이언트에서 portmapper를 질의할 수 있어야 합니다.

Portmapper를 사용하여 ACSLS 서버 및 클라이언트측 모두에서 방화벽 보안

이 예에서는 클라이언트(SSI) 및 ACSLS 서버(CSI) 모두 방화벽 보안 API를 구현합니다. 클라이언트 및 서버는 portmapper를 사용해서 포트를 식별합니다. 이 시나리오를 지원하기 위해서는 CSC Toolkit 2.3(이상) 및 새로운 환경 변수가 필요합니다.

그림 M.3. Portmapper를 사용하여 ACSLS 서버 및 클라이언트측 모두에서 방화벽 보안

SSI측의 portmapper 111 포트는 CSI에서 드문 경우에만 질의됩니다. 이 포트는 응답 패킷을 다시 SSI로 전송할 때 해당 요청 패킷에서 SSI에 의해 제공된 반환 포트 번호가 작동하지 않아서 네트워크 인터페이스 오류가 발생하는 경우에만 CSI에서 액세스됩니다. 이 경우에는 재시도 메커니즘에 따라 CSI가 SSI측 portmapper에 사용할 포트(SSI의 프로그램 번호에 따라 portmapper에 등록됨)를 질의합니다.

ACSLS 서버 및 클라이언트를 모두 보호하는 방화벽의 경우, 다음 설정이 필요합니다.

- ACSLS: 설정을 변경한 후에는 ACSLS를 다시 시작해야 합니다.
  - *CSI_TCP_RPCSERVICE* = TRUE
  - *CSI_UDP_RPCSERVICE* = FALSE

    하지만 ACSLS와의 통신을 위해 UDP를 사용하는 클라이언트가 있을 경우 TRUE로 설정해야 합니다.
  - *CSI_USE_PORTMAPPER* = ALWAYS(*IF_DUAL_LAN_NOT_ENABLED* 가능)
  - *CSI_FIREWALL_SECURE* = TRUE
  - *CSI_INET_PORT* = <1024-65535, 50003 제외> 기본값 30031
- 클라이언트 SSI 설정 - 클라이언트가 방화벽 뒤에서 실행되도록 허용하는 환경 변수입니다.
  - *CSI_TCP_RPCSERVICE* = TRUE
  - *CSI_UDP_RPCSERVICE* = FALSE
  - *SSI_INET_PORT* = <1024 –65535, 50001 및 50004 제외>
  - *CSI_HOSTPORT* = <1024-65535, 50003 제외> 기본값 30031

    이 값은 ACSLS 서버의 *CSI_INET_PORT*와 일치해야 합니다.

다음을 허용하도록 방화벽을 구성해야 합니다.

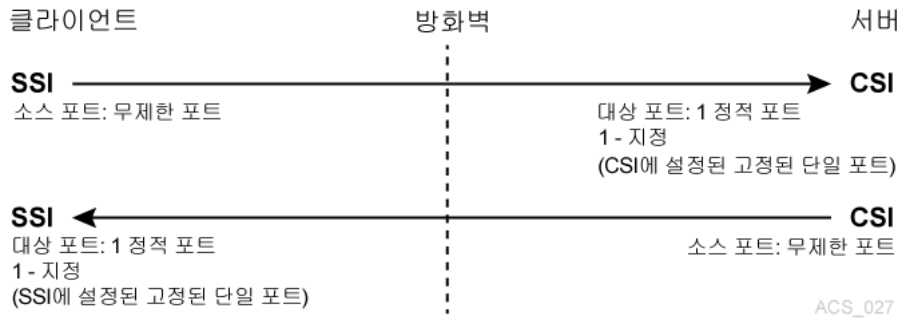
- *CSI_INET_PORT*(ACSLS 서버) 및 *CSI_HOSTPORT*(클라이언트)에 의해 지정된 포트를 사용해서 클라이언트가 ACSLS 서버에 요청을 전송할 수 있어야 합니다.
- 클라이언트가 ACSLS 서버에서 portmapper(포트 111)에 액세스할 수 있어야 합니다.

- ACSLS 서버가 클라이언트에서 *SSI_INET_PORT*로 지정된 포트를 사용해서 클라이언트에 응답을 전송할 수 있어야 합니다.
- ACSLS 서버가 포트 111을 사용해서 클라이언트에서 portmapper를 질의할 수 있어야 합니다.

Portmapper를 사용하지 않고 ACSLS 서버 및 클라이언트측 모두에서 방화벽 보안

이 예에서는 클라이언트(SSI) 및 ACSLS 서버(CSI) 모두 방화벽 보안 기능을 구현합니다. 클라이언트 및 서버는 "No Portmapper" 기능이 사용으로 설정됩니다. 이 시나리오를 지원하기 위해서는 CSC Toolkit 2.3(이상) 및 새로운 환경 변수가 필요합니다.

그림 M.4. Portmapper를 사용하지 않고 ACSLS 서버 및 클라이언트측 모두에서 방화벽 보안



최대한의 보안 구성을 위해서는 다음 설정이 필요합니다.

- ACSLS: 설정을 변경한 후에는 ACSLS를 다시 시작해야 합니다.
  - *CSI_TCP_RPCSERVICE* = TRUE
  - *CSI_UDP_RPCSERVICE* = FALSE
  - *CSI_USE_PORTMAPPER* = NEVER
  - *CSI_FIREWALL_SECURE* = TRUE
  - *CSI_INET_PORT* = <1024-65535, 50003 제외> 기본값 30031
- 클라이언트 SSI 설정 - 클라이언트가 방화벽 뒤에서 실행되도록 허용하는 환경 변수입니다.
  - *CSI_TCP_RPCSERVICE* = TRUE
  - *CSI_UDP_RPCSERVICE* = FALSE
  - *SSI_INET_PORT* = <1024 –65535, 50001 및 50004 제외>
  - *CSI_HOSTPORT* = <1024-65535, 50003 제외> 기본값 30031

    이 값은 ACSLS 서버의 *CSI_INET_PORT*와 일치해야 합니다.

다음을 허용하도록 방화벽을 구성해야 합니다.

- *CSI_INET_PORT*(ACSLS 서버) 및 *CSI_HOSTPORT*(클라이언트)에 의해 지정된 포트를 사용해서 클라이언트가 ACSLS 서버에 요청을 전송할 수 있어야 합니다.

- ACSLS 서버가 클라이언트에서 *SSI_INET_PORT*로 지정된 포트를 사용해서 클라이언트에 응답을 전송할 수 있어야 합니다.

ACSLS 서버에서 방화벽 보안 켜기

방화벽 보안 옵션을 켜기 위해서는 *acsss_config* 유틸리티를 사용해서 몇 가지 변수를 설정해야 합니다.

1. *acsss*로 로그인하십시오.
2. ACSLS 서버를 중지합니다.

   주:

   새 방화벽 보안 변수를 적용하려면 ACSLS 서버를 중지해야 합니다.

   acsss disable
3. 구성 스크립트를 실행하기 위해 다음 명령을 입력합니다.

   acsss_config

   ACSLS 기능 구성 화면이 나타납니다.
4. 옵션 1 - *Set CSI tuning variables*를 선택합니다.

   다음을 제외하고 모든 변수에 대해 기본값을 수락합니다.
   a. 다음 프롬프트에서 값을 *TRUE*로 설정합니다.

      Changes to alter use of the TCP protocol will not take effect until the product is restarted. CSI support for RPC using the TCP protocol is enabled [TRUE].

      변수: *CSI_TCP_RPCSERVICE*

      TCP를 켜면 네트워크 통신을 위해 ACSLS의 클라이언트가 TCP 프로토콜을 사용할 수 있도록 보장합니다. ACSLS의 방화벽 보안 기능에서는 TCP만 지원되므로, 클라이언트가 이 프로토콜을 사용해서 네트워크 통신을 수행해야 합니다.
   b. 다음 프롬프트에서 값을 *FALSE*로 설정합니다.

      Changes to alter the use of the UDP protocol will not take effect until the product is restarted. CSI support for RPC using the UDP protocol is enabled [TRUE].

      변수: *CSI_UDP_RPCSERVICE*

주의:

이 UDP 프로토콜을 사용 중인 ACSLS 클라이언트가 없는지 확인하십시오. 방화벽 보안 ACSLS는 TCP에서만 실행됩니다.

UDP를 끄면 클라이언트가 이 프로토콜을 사용해서 서버에 액세스하지 않도록 보장합니다. 이렇게 하면 방화벽에서 ACSLS 플랫폼에 대한 모든 일반 UDP 액세스를 금지하여, 특별히 환경에 필요한 액세스만 허용할 수 있습니다.

해당 클라이언트가 방화벽 보안 기능을 구현하지 않는 한 portmapper 액세스를 위해 UDP 및 TCP 포트 111에 대한 클라이언트 액세스를 허용하고, 특별히 ACSLS portmapper에 대한 질의는 해제합니다.

c. 다음 프롬프트에서 값을 *NEVER*로 설정합니다.

Changes to alter use of the port mapper will not take effect until the product is restarted. Enable port mapper: (ALWAYS / NEVER /IF_DUAL_LAN_NOT_ENABLED) [IF_DUAL_LAN_NOT_ENABLED].

변수: *CSI_USE_PORTMAPPER*

NEVER로 설정하면 ACSLS 클라이언트가 해당 클라이언트 플랫폼에서 portmapper에 대한 외부 액세스를 금지할 수 있습니다.

중요: 이 설정으로 ACSLS 플랫폼에서 portmapper에 대한 외부 액세스를 해제할 수 **없습니다**. 이를 위해서는 ACSLS 클라이언트가 클라이언트 소프트웨어 구성 요소에서 방화벽 보안 변경을 채택하고, 클라이언트 소프트웨어 구성 요소에서 이 기능을 켜야 합니다.

이 기능은 ACSLS 서버가 클라이언트 플랫폼에서 portmapper 질의를 수행하지 않도록 보장합니다. 따라서 **클라이언트를 보호 중인** 모든 방화벽이 portmapper에 대한 액세스를 금지할 수 있습니다.

d. 다음 프롬프트에서 값을 *TRUE*로 설정합니다.

Enable CSI to be used behind a firewall (user-defined inbound port) (TRUE/FALSE) [FALSE]:

변수: *CSI_FIREWALL_SECURE*

*TRUE*를 선택하면 ACSLS가 인바운드 클라이언트 통신(TCP 연결)을 수락하기 위해 사용할 단일 포트를 지정할 수 있습니다. 이 변수는 단순히 이 기능을 사용으로 설정합니다. 특정 포트는 다음 변수에서 지정됩니다.

e. 다음 프롬프트에서는 ACSLS 서버에서 사용 가능한 고정 포트로 값을 설정합니다.

Port number used by the CSI to receive incoming ACSLS requests.

변수: *CSI_INET_PORT*

이 포트는 수신 네트워크 연결을 수락하기 위해 ACSLS CSI 구성 요소에서 사용되는 포트입니다. 1024-65535 범위(포트 50003 제외)로 포트를 지정합니다.

중요: 이 포트에서 수신 연결을 허용하도록 방화벽을 구성합니다. 이렇게 하면 ACSLS와 통신을 시작하려는 외부 클라이언트가 해당 포트만 사용할 수 있습니다. ACSLS portmapper에 대한 질의를 제거하는 기능이 클라이언트에 구현되지 않은 한 이 포트를 제외한 다른 모든 수신 포트 및 UDP/TCP 포트 111에 대한 연결을 금지할 수 있습니다. 이 경우 포트 111이 방화벽에서 금지될 수 있습니다. 이 포트에 대한 권장 기본값은 30031입니다. 이 포트는 대부분의 시스템에서 다른 프로세스에 사용될 가능성이 낮습니다(불가능하지는 않음). 포트 충돌이 있는지 확인하기 위한 단계는 부록 J. 문제 해결을 참조하십시오.

5. *E*를 선택해서 *acsss_config*를 종료합니다.

   변경사항이 저장됩니다.
6. 다음 명령을 입력하여 ACSLS를 다시 시작합니다.

   ```
   acsss enable
   ```

ACSLS 서버에서 방화벽 보안 끄기

방화벽 보안 기능을 켜기 위해 위에서 사용된 일부 변수는 해당 기능을 끄는 것과도 관련이 있습니다. 방화벽 보안 동작을 끄기 위해서는 아래 단계만 수행하면 되지만, 특정 사이트의 경우 다른 변수를 수정해야 할 수도 있습니다.

1. *acsss*로 로그인하십시오.
2. ACSLS 서버를 중지합니다.

   주:

   새 방화벽 보안 변수를 적용하려면 ACSLS 서버를 중지합니다.

   ```
   acsss disable
   ```
3. 구성 스크립트를 실행하기 위해 다음 명령을 입력합니다.

   ```
   acsss_config
   ```
4. 옵션 1 - *Set CSI tuning variables*를 선택합니다.

   방화벽 보안 기능을 구성할 때 설정된 다음 값을 변경합니다. 다음 변수를 변경합니다.
   a. 다음 프롬프트에서 값을 *ALWAYS*로 설정합니다.

      ```
      Changes to alter use of the port mapper will not take effect
      until the product is restarted. Enable port mapper: (ALWAYS /
      NEVER /IF_DUAL_LAN_NOT_ENABLED) [IF_DUAL_LAN_NOT_ENABLED].
      ```

      변수: *CSI_USE_PORTMAPPER*
   b. 다음 프롬프트에서 값을 *FALSE*로 설정합니다.

      ```
      Enable CSI to be used behind a firewall (user-defined inbound
      port) (TRUE/FALSE) [FALSE]:
      ```

변수: *CSI_FIREWALL_SECURE*

5. *E*를 선택해서 *acsss_config*를 종료합니다.

   변경사항이 저장됩니다.
6. 다음 명령을 입력하여 ACSLS를 다시 시작합니다.

   ```
   acsss enable
   ```

방화벽 보안 구성

다음 절에서는 ACSLS가 상주하는 네트워크 방화벽 구성과 관련된 지식이 필요합니다. 모든 방화벽은 "타사" 소프트웨어이며, 네트워크 환경 보호를 위한 올바른 설정과 관련한 세부정보는 업체마다 다양합니다. 다음 절의 내용은 방화벽 보안 정책을 위한 권장 사항이 아니며, ACSLS 제품과 관련해서 방화벽이 수행해야 하는 작업과 수행할 수 있는 작업에 대한 유용한 지침을 제공하기 위한 것입니다. 다른 보안 세부정보는 시스템 관리자에게 문의하십시오.

ACSLS 플랫폼을 위해 방화벽을 설정할 때의 세부정보는 다음과 같습니다.

- UDP 수신 및 송신 연결을 금지하기 위한 일반 규칙을 배치합니다.
- TCP 수신 연결을 금지하기 위한 일반 규칙을 배치합니다(TCP 송신 연결은 열린 상태로 유지되어야 함).
- ACSLS에서 사용하도록 지정한 포트에서 수신 TCP 연결을 허용하는 특정 규칙을 배치합니다. **중요:** 이 포트는 *acsss_config*로 구성한 것과 **일치해야 합니다.** 그렇지 않으면 ACSLS 서버에서 클라이언트 통신이 수신되지 않습니다.

모든 클라이언트에 방화벽 보안 기능이 구현되어 있는 경우 ACSLS 플랫폼의 portmapper에 대해 질의를 수행하지 않으면 됩니다. 클라이언트가 여전히 ACSLS 플랫폼에서 portmapper를 사용할 경우에는 다음을 추가해야 합니다.

- 잘 알려진 portmapper TCP 및 UDP 포트 111에서 수신 및 송신 연결을 허용하는 특정 규칙을 배치합니다.

예:

다음은 위의 모든 규칙을 배치하기 위해 iptables 기반 방화벽에 대해 배치된 규칙의 예입니다.

주:

이러한 규칙은 특정 방화벽에 대해 구성된 다른 규칙에 추가로 배치됩니다.

```
echo " - FWD: Allow all connections OUT and only existing/related IN"
$IPTABLES -A FORWARD -i $EXTIF -o $INTIF -m state --state /
ESTABLISHED,RELATED -j ACCEPT
# These rules allow client access to the portmapper
$IPTABLES -A FORWARD -p tcp -i $EXTIF --dport 111 -j ACCEPT
$IPTABLES -A FORWARD -p udp -i $EXTIF --dport 111    -j ACCEPT
```

```
# These rules allow client access to the ACSLS CSI for network communication
# Note: This assumes that the CSI firewall-secure port was specified as 30031
$IPTABLES -A FORWARD -p tcp -i $EXTIF --dport 30031 -j ACCEPT
# Catch all rule, all other forwarding is denied and logged.
$IPTABLES -A FORWARD -j drop-and-log-it
```

방화벽 보안 통신 문제 해결

ACSLS 플랫폼 및 클라이언트, 그리고 중간 방화벽까지 포함된 네트워크 통신 인터페이스의 문제 해결에는 여러 단계가 포함될 수 있습니다. ACSLS와 클라이언트 사이의 경로에 방화벽을 도입함으로써 네트워크 통신 오류의 원인이 더 증가할 가능성이 있습니다. 또한 다른 구성 요소의 설정에 따라 구성해야 하는 추가 구성 요소도 있습니다. 이러한 설정이 일치하지 않으면 네트워크 성능에 영향을 줄 수 있습니다. 다음은 ACSLS에서 모든 구성 작업을 수행했지만 해당 클라이언트, 방화벽 및 네트워크 통신이 작동하지 않는 경우에 확인하고 시도해야 할 작업 목록입니다.

1. ACSLS 플랫폼 확인:
   - ACSLS가 작동되어 실행 중입니까? 실행 중이 아니면 *acsss_event.log*에서 가능한 이유 또는 가능한 원인에 대한 단서를 확인합니다.
   - CSI가 ACSLS로 성공적으로 작동됩니까? 그렇지 않으면, *acsss_event.log*에서 원인에 대한 단서를 제공하는 정보 메시지를 찾을 수 있습니다. 일부 *configuration parameters* 또는 *port conflict*에 대한 잘못된 값이 가능한 원인일 수 있습니다.
   - *acsss_event.log*에 CSI 실패 원인이 되는 포트 충돌이 있습니까? 그렇다면, "*netstat*" 또는 비슷한 시스템 유틸리티를 사용해서 시스템에서 사용 중인 포트를 확인하고 사용 가능한 포트를 사용하도록 ACSLS를 구성해야 합니다. 포트가 일치하도록 방화벽을 다시 구성해야 합니다.
   - CSI가 필요한 포트에 등록되어 있습니까? '*rpcinfo -p*'를 사용해서 포트맵 테이블을 확인합니다. CSI는 프로그램 번호 300031 아래에 등록됩니다. 해당 프로그램 번호 아래에 등록된 포트가 필요한 포트인지 확인합니다. 기본값은 프로그램 번호보다 0이 하나 더 적은 30031입니다.
2. ACSLS 및 CSI가 작동되어 실행 중이고 올바르게 등록된 경우에는 이제 방화벽에서 ACSLS 플랫폼에 대한 액세스를 확인해야 합니다.
   - 기본 RPC를 통해 ACSLS에 연결할 수 있습니까? "*rpcinfo -t* <hostname> <*program-number*> <*version-number*>" 명령을 사용해서 간단한 RPC 요청을 CSI에 전송합니다. (시스템에서 "*man rpcinfo*"를 사용해서 *rpcinfo* 명령 및 해당 사용에 대해 자세한 정보를 확인합니다.) ACSLS가 포함된 방화벽 내(예: ACSLS 플랫폼 자체 내부)의 시스템 및 방화벽 외부의 시스템에서 이 작업을 수행합니다. 내부에서는 작동하지만 외부에서 작동하지 않을 경우 방화벽으로 인해 ACSLS 요청이 차단됩니다. 방화벽 구성 및 ACSLS 포트를 다시 확인합니다. 방화벽을 통해 portmapper에 액세스할 수 있는지 확인합니다(portmapper에 대한 액세스가 금지된 경우 방화벽 외부에서 이 테스트를 사용할 수 없음).
   - ACSLS 및 방화벽에 대해 구성된 포트가 일치합니까? 이러한 매개변수를 다시 확인합니다. 이 경우 네트워크 통신 오류가 원인일 수 있습니다. 구성된 값은 별도로 하고 위에 언급된 '*rpcinfo -p*' 명령을 수행해서 CSI가 실제로 필요한 포트 번호로 등록되었는지 확인합니다. 그렇지 않으면 *acsss_event.log*에서 원인에 대한 정보를 확인합니다.

- ACSLS가 요청을 수신하지만 응답을 다시 전송할 수 없습니까? *acsss_event.log*를 확인해서 CSI에서 네트워크 패킷 삭제가 여러 번 보고되거나 CSI가 네트워크 클라이언트와 통신할 수 없는 것으로 확인된 경우, 클라이언트 요청이 수신되지만 응답이 전송되지 않는 것입니다. 이러한 상태는 방화벽에 의해 전송이 차단되고 있음을 나타냅니다.

3. 문제가 아직 해결되지 않은 경우 다음을 확인합니다.

   위 절차에서는 조사해야 할 여러 사항들을 보여줍니다. 그래도 문제가 해결되지 않았으면, 이제 통신이 중단되는 위치를 찾기 위해 몇 가지 세부적인 조사를 수행해야 합니다. 가장 좋은 방법은 Solaris의 'snoop'와 같은 네트워크 패킷 스니퍼 기능을 사용하는 것입니다. Solaris 기반 시스템에서 "man snoop"를 사용해서 *snoop* 명령 및 해당 사용에 대한 자세한 정보를 확인합니다.

   다른 네트워크 연결 시스템에서는 비슷한 패킷 추적 기능을 사용할 수 있습니다.

   - 이를 사용하기 위해서는 패킷이 연결되는 장소 및 패킷이 손실되는 장소를 보여주는 위치에서 패킷 스니핑을 수행해야 합니다. 이러한 위치는 방화벽 내부 및 외부에서 모두 비롯될 수 있습니다.
   - 또한 패킷 데이터를 조사하는 것도 도움이 될 수 있습니다. 어느 쪽에서든 portmapper 사용이 허용될 경우, 일부 PORTMAP 패킷이 표시될 수 있습니다.
   - 또한 ACSLS와 해당 클라이언트 사이에 전달되는 RPC 패킷을 찾을 수 있습니다.
   - 마지막으로 전송 레벨의 TCP 연결을 확인해도 각 연결측에서 사용되는 특정 포트를 확인할 수 있습니다. 이러한 정보는 특히 통신이 중지되는 위치를 찾기 위한 핵심 정보인 경우가 많습니다.

이러한 작업 수행에 대한 자세한 내용은 이 설명서의 범위를 벗어나며, 이 영역과 관련해서는 시스템 관리자의 지원을 받을 수 있습니다.

질문과 대답

- ACSLS에 방화벽 보안 솔루션이 필요한 이유는 무엇입니까?

  방화벽 보안 솔루션은 방화벽 뒤에서 ACSLS를 효과적으로 실행할 수 있게 해주며, 해당 방화벽에서 포트를 제한하여 보안을 크게 향상시킬 수 있습니다.
- 방화벽 보안 기능이 지원되는 ACSLS 릴리스는 무엇입니까?

  이 기능은 ACSLS 7.0.0 이상에서만 지원됩니다.
- 이 방화벽 보안 기능을 사용할 때 열어 두어야 하는 최대 포트 수는 몇 개입니까?

  수신 네트워크 연결을 허용하기 위해 설정할 수 있는 최대 포트 수는 3개입니다. 한 개는 ACSLS 네트워크 인터페이스를 위한 포트이고, 두 개는 portmapper(UDP 및 TCP 111)를 위한 포트입니다. 송신 포트는 일반적인 산업 보안 방식을 따르며 제한이 없습니다.
- 열어 둘 수 있는 최소 포트 수는 몇 개입니까?

  최소 개수는 1개입니다. 이 설정은 클라이언트(ISV 소프트웨어)가 해당 클라이언트에 방화벽 보안 기능을 구현하고, ACSLS 플랫폼에 상주하는 portmapper에 질의를 수행하지

않는 경우에 가능합니다. 이런 경우에 수신 연결을 위해 열어 두어야 하는 유일한 포트는 ACSLS 네트워크 인터페이스에서 사용되는 사용자 지정 TCP 포트 한 개입니다.

- 이 기능에서 포트 범위를 사용하지 않는 이유는 무엇입니까?

  포트 범위를 사용하는 데 따른 구조적 장점이 없으며, 몇 가지 보안상의 단점이 있습니다. 비방화벽 보안 ACSLS에서는 특정 플랫폼에서 사용 가능한 전체 동적 포트 범위로 구성되는 포트 범위가 사용됩니다. 이러한 구성은 사이트 보안을 잠재적으로 손상시킬 가능성이 있는 것으로 인식됩니다. ACSLS 성능에 부정적 영향을 주지 않으면서, 가능한 한 이러한 방식을 제한하는 것은 그러한 손상을 없애기 위한 것입니다. ACSLS 네트워크 인터페이스에는 특정 시점에 수신 포트가 하나만 사용되기 때문에, 포트가 ACSLS 플랫폼 전용으로 사용된다고 가정할 때 포트 1개 이상으로 범위를 확장할 이유가 없습니다.
- 선택한 포트가 내 시스템에서 해당 포트를 사용하는 다른 것과 충돌할 경우에는 어떻게 됩니까?

  포트를 사용자 지정 가능하게 설정하는 이유 중 하나입니다. 사용 가능한 특정 포트는 고객 사이트마다 달라질 수 있습니다. 0-1023 사이의 잘 알려진 예약된 포트는 사용하도록 허용되지 않습니다. 기본 포트 30031은 등록된 포트 범위 내에 있으면서, 동적 포트를 사용하는 다른 응용 프로그램에서 이를 사용할 가능성이 낮은 편입니다(불가능하지는 않음). 등록된 포트 범위에 있지만 이를 사용하도록 등록된 응용 프로그램이 없으므로 합리적으로 선택된 기본값입니다.
- 이 기능을 사용해서 방화벽으로 ACSLS 서버를 보호할 수 있습니까?

  예, 그렇습니다. 이 기능을 사용할 경우, ACSLS 서버를 방화벽의 신뢰할 수 있는 측면에 배치하고, 클라이언트가 반대쪽(신뢰할 수 없는) 측면 또는 동일 측면에서 서버에 액세스하도록 할 수 있습니다.
- 이 기능을 사용해서 방화벽으로 ACSLS 클라이언트(ISV 구성 요소)를 보호할 수 있습니까?

  잠재적으로 가능은 하지만, 그 자체로는 가능하지 않습니다. 이러한 시나리오를 실현하기 위해서는 클라이언트 소프트웨어 구성 요소(ACSLS의 클라이언트)에 StorageTek ACSLS 클라이언트 시스템 구성 요소 개발자 툴킷을 사용해서 제공되는 방화벽 보안 기능이 채택되어 있어야 합니다. 클라이언트 소프트웨어 제공자에게 연락해서 현재 업데이트 상태를 확인하십시오.
- 방화벽으로 내 클라이언트를 보호할 수 있으려면 어떻게 해야 합니까?

  클라이언트 소프트웨어 공급자에게 문의해야 합니다. 해당 CSC(클라이언트 소프트웨어 구성 요소)에서 방화벽 보안 변경이 채택되었는지 여부를 확인하십시오.
- portmapper는 어떻습니까? portmapper에 대한 액세스를 완전히 금지할 수 있습니까?

  클라이언트에 방화벽 보안 변경이 채택된 경우, ACSLS 플랫폼의 portmapper에 대한 클라이언트 질의를 차단할 수 있습니다. 이 경우, ACSLS 플랫폼을 보호하는 방화벽에서 portmapper에 대한 액세스를 금지할 수 있습니다. 다른 모든 경우에는 클라이언트가 ACSLS 서버측 portmapper에 의존해서 ACSLS 네트워크 인터페이스와의 연결을 지원하며, 클라이언트가 portmapper를 사용할 수 있어야 합니다.
- 내 ACSLS 서버 방화벽이 ACSLS 플랫폼 portmapper에 대한 액세스를 차단하기 위해 클라이언트에서 일부 변경사항을 구현해야 하는 이유는 무엇입니까?

ACSLS 플랫폼 질의를 수행하는 주체가 바로 클라이언트이기 때문입니다. 클라이언트가 계속해서 이러한 질의를 수행할 경우, ACSLS 플랫폼은 그러한 질의가 성공할 수 있도록 portmapper의 서비스를 계속 제공해야 합니다.

- portmapper는 좋은 방법이 아닌 것 같습니다. 이를 완전히 제거하지 않은 이유는 무엇입니까?

  portmapper는 레거시 클라이언트에 중요한 서비스를 제공합니다. 이를 완전히 제거하면 이러한 클라이언트가 필요한 인터페이스를 사용할 수 없게 됩니다. 간단히 말해서 코딩과 테스트를 다시 하고 새로운 비portmapper 인터페이스로 다시 인증하지 않는 한 레거시 클라이언트가 작동하지 않을 것입니다. 오라클은 이러한 방화벽 보안 솔루션에 대해 ACSLS에서 클라이언트로, 그리고 클라이언트에서 ACSLS로의 portmapper에 대한 질의 제거 기능을 제공했지만, 클라이언트 소프트웨어가 이를 따르도록 강제할 수는 없습니다. 따라서, 사이트의 클라이언트에 방화벽 보안 기능이 채택되고 더 이상 portmapper 서비스가 사용되지 않을 때까지는 최소한 선택적인 서비스로서 portmapper를 사용 가능한 상태로 유지해야 합니다.
- 내 클라이언트 중 일부에는 방화벽 보안 기능이 채택되었고, 일부는 그렇지 않습니다. 이를 활용하려면 어떻게 해야 합니까?

  이러한 기능이 채택된 클라이언트는 고유 방화벽을 통해 보호할 수 있습니다. 또한 portmapper의 잘 알려진 포트에 대한 액세스도 방화벽에서 제한할 수 있으며, 필요한 클라이언트에 대해서만 portmapper에 액세스를 허용하도록 구성할 수 있습니다. 세부 사항과 기능은 해당 사이트에서 사용 중인 특정 방화벽에 따라 달라집니다.
- RPC는 좋은 방법이 아닌 것 같습니다. 이를 완전히 제거하지 않은 이유는 무엇입니까?

  ACSLS 네트워크 인터페이스는 ACSLS가 처음 릴리스되었을 때부터 지금까지 RPC를 기반으로 하고 있습니다. RPC는 네트워크 통신 계층에서 다양한 이점을 제공하며 효과적이고, 안정적이며, 신뢰할 수 있는 메커니즘인 것으로 입증되었습니다. 하지만 일반적인 동적 포트 할당과 portmapper 사용으로 인해 RPC를 사용하는 플랫폼은 보안이 조금 더 까다로울 수 있습니다. 이러한 방화벽 보안 솔루션에서는 고객이 제한된 방식으로 방화벽을 구성할 수 있도록 지원하여 사용 중인 방화벽의 보안상 이점을 제공하는 방식으로, 이러한 영역이 갖고 있는 문제가 해결되었습니다.

  또한 ACSLS 네트워크 인터페이스에서 RPC를 완전히 제거한다면 모든 현재(레거시) ACSLS 클라이언트를 사용할 수 없게 되어, 코딩과 테스트를 다시 하고 해당 CSC(클라이언트 소프트웨어 구성 요소)를 다시 인증하지 않는 한 이러한 클라이언트가 ACSLS와 전혀 통신할 수 없을 것입니다.
- 방화벽 보안 기능은 내 ACSLS 클라이언트와 ACSLS 서버 사이의 네트워크 통신 성능 및 타이밍에 어떤 영향을 줍니까?

  새로운 방화벽 보안 기능으로 인한 성능 영향은 없습니다. 방화벽 사용은 성능에 영향을 줄 수 있지만, 이러한 영향은 각 고객의 방화벽 구현에 따른 운영상의 특성을 기반으로 합니다. 성능에 대한 영향이 무시할 수 있는 정도인 방화벽을 사용할 경우, ACSLS 및 해당 클라이언트는 방화벽 보안 기능을 설치하기 전과 같은 성능을 계속 제공할 것입니다. 또한 ACSLS 네트워크 인터페이스의 내결함성을 구성해서 방화벽으로 인한 지연 시간을 정상적으로 처리할 수도 있습니다.

- 방화벽 보안 기능은 나머지 ACSLS 작업에 어떤 영향을 줍니까?

  방화벽 보안 솔루션 설치로 인해 ACSLS 작업의 다른 부분에 미치는 영향은 없습니다.
- 방화벽 보안 기능은 내 클라이언트에서 ACSAPI를 통해 사용되는 ACSLS 기능에 어떤 영향을 줍니까?

  ACSAPI를 통해 제공되는(및 ACSLS 클라이언트가 현재 ACSLS와 통신하기 위해 사용하는) 전체 기능 세트는 방화벽 보안 기능 하에서도 이 기능을 설치하기 전과 동일하게 작동합니다. 특히 이 방화벽 보안 기능은 액세스 제어를 지원하며, ACSLS 제품에 추가된 모든 새로운 기능들을 지원합니다. ACSAPI의 전체 기능은 이 기능을 통해 계속해서 지원될 예정입니다.
- 방화벽 보안 기능이 ACSLS HA(고가용성) 솔루션에서 작동합니까?

  방화벽 보안 기능은 HA 작업에 부정적인 영향을 주지 않습니다. 하지만 HA 솔루션은 방화벽(즉, 방화벽의 양쪽에 HA 서버가 있는 구성)에서 실행되도록 설계되지 않았습니다. HA 솔루션은 portmapper에 대한 원격 액세스가 필요하므로, 방화벽의 양쪽 측면에서 각각 반대편의 서버를 실행하려는 시도가 수행된 경우, 방화벽이 그러한 액세스를 금지할 수 없습니다. 방화벽 실행과 관련해서 HA 설정에 부정적 영향을 줄 수 있는 다른 세부 설정들이 존재하며, 이렇게 되지 않도록 하는 것이 중요합니다.

  HA 서버가 방화벽의 동일한 보안 측면에 설정되어 있고, 그러한 HA 서버 세트를 방화벽 보안 기능으로 설정할 수 있는 경우, 방화벽 반대쪽에 있는 클라이언트는 비방화벽 보안 HA 솔루션과 동일한 성능 및 동작으로 방화벽을 통해 해당 서버와 상호 작용할 수 있습니다.
- 이 방화벽 보안 기능이 다른 StorageTek 소프트웨어 제품에서도 작동합니까?

  다른 StorageTek 제품 및 파트너 제품(즉, ACSLS와 통신하는 클라이언트 소프트웨어 구성 요소)과의 상호 운용성은 완전히 보존되어 있습니다. 이러한 제품은 수정 없이도 계속 작동할 수 있으며, 보안 방화벽 뒤 또는 해당 제품과 동일한 환경(현재 환경)에서 ACSLS 서버를 실행하면서 ACSLS 서버와 통신할 수 있습니다.
- 다른 StorageTek 소프트웨어 제품에도 동일한 방화벽 보안 기능이 포함됩니까?

  다른 StorageTek 제품은 단순히 방화벽 보안 ACSLS와 동일한 환경에 사용한다고 해서 방화벽 보안의 이점을 얻을 수 없습니다. 각 제품은 방화벽 보안 ACSLS에서 작동할 수 있지만(이전 질문 참조), 이러한 각 제품을 고유 방화벽 뒤에 배치하는 것은 특정 제품 자체에서 해결해야 하는 문제입니다. 일부 StorageTek 제품은 그러한 제품이 실행되는 플랫폼 보호를 위해 사용되는 방화벽에서 일부 제한을 허용하는 내장 정책이 이미 포함되어 있습니다. 또한 ACSLS에 대해 클라이언트로 작동하는 모든 제품에는 ACSLS CSC Developer’s Toolkit 2.3(이상)의 일부로 제공되는, ACSLS에 대한 방화벽 보안 변경 채택 옵션이 포함되어 있습니다.

부록 N

부록 N. CSCI

이 장에서는 CSCI(클라이언트 서버 통신 인터페이스) 구성 요소에 대해 설명합니다.

이 장에서는 다음 항목에 대해 다룹니다.

- CSCI 개요
- 오류 메시지
- 환경 변수

CSCI 설치 시에는 운영체제 설치 및 ACSLS 설치와 구성을 따라야 합니다.

> **주:**
>
> ACSLS가 올바르게 작동하기 위해서는 START_CSCI_PROCESS 구성 변수를 TRUE로 설정해야 합니다. 이 작업은 acsss_config의 옵션 1에서 수행합니다. 6장. *ACSLS* 동작을 제어하는 변수 설정을 참조하십시오.

CSCI 개요

CSCI(클라이언트 서버 통신 인터페이스)는 클라이언트 시스템과 서버 시스템 간에 요청 및 응답을 전달하는 독립형 구성 요소입니다. CSCI는 MVS 또는 RMLS/CSC 클라이언트가 ACSLS와 통신할 수 있게 해주는 선택적인 구성 요소입니다. CSCI를 사용하지 않을 경우에는 기본적으로 CSI를 사용해서 ACSLS와 통신합니다.

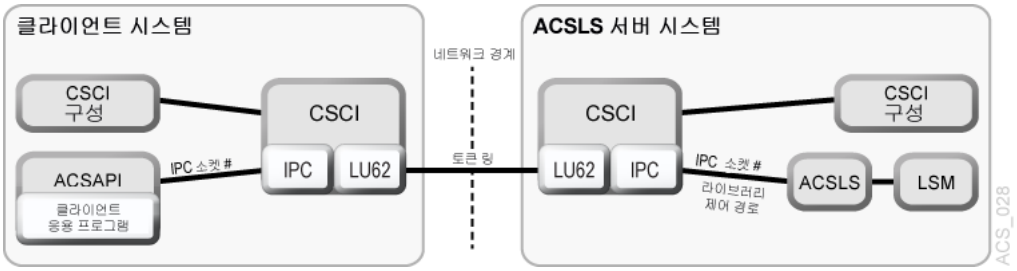
CSCI 연결

CSCI는 ACSLS 서버를 MVS 클라이언트 또는 RMLS/CSC 클라이언트와 연결하는 IPC 및 TCP/IP 연결을 제공합니다.

아키텍처 및 부속 시스템

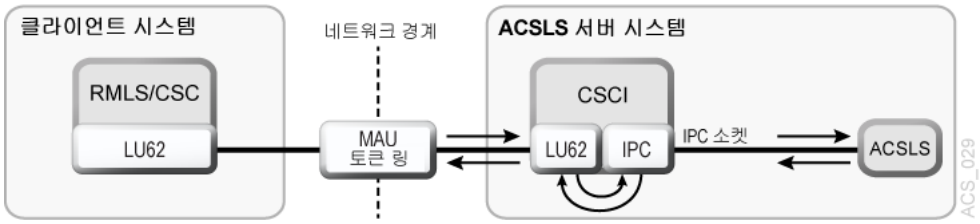
다음 그림은 전반적인 CSCI 아키텍처 및 부속 시스템을 보여줍니다.

그림 N.1. CSCI 아키텍처 및 부속 시스템



다음 그림은 CSCI 시스템이 네트워크 인터페이스와 연결하는 방법을 보여줍니다. 서버측에서 데이터는 TCP/IP 인터페이스를 통해 네트워크 허브로부터 CSCI 인터페이스로 이동합니다. 여기에서 데이터는 IPC 인터페이스를 통해 양방향으로 ACSLS 소프트웨어로 이동하지만, 반이중 모드에서는 한 번에 한 방향으로만 이동합니다.

그림 N.2. 토큰 링 인터페이스를 사용하는 CSCI 시스템



오류 메시지

CSCI 설치 및 구성 오류 메시지는 다음 조건과 관련이 있습니다.

- 부적절한 소유권
- 부적절한 액세스 권한 설정
- 정의되지 않은 환경 변수
- 다양한 기타 조건
- 부적절한 CSCI 및 TCP/IP 구성

일반 오류 메시지

Communication failure, verb = cmroutine, rc = xx, errno = yyy IPC failure on Socket CSCIzzzzz

설명: 이름이 지정된 연결에서 읽기를 실패했습니다.

변수:

- xx는 IBM CPI 반환 코드입니다.
- yyy는 */usr/include/sys/errno.h*의 UNIX errno 값입니다.
- *CSCIzzzzz*는 사용 중이던 소켓 번호입니다.
- *cmroutine*은 실패한 IBM CPI 루틴 이름입니다.

Read failed for connection: connection_name

설명: 이름이 지정된 연결에서 읽기를 실패했습니다.

변수: connection_name은 읽기를 실패한 연결의 이름입니다.

Write failed for connection: connection_name

설명: 이름이 지정된 연결에서 쓰기를 실패했습니다.

변수: connection_name은 쓰기를 실패한 연결의 이름입니다.

환경 변수

CSCI는 ACSLS 실행 파일이 상주하는 /$ACS_HOME/bin 디렉토리의 TPN(트랜잭션 프로그램 이름)인 *csciTcpServer.sh*에 의해 호출됩니다. 이 셸 스크립트는 CSCI 환경 변수를 정의합니다.

다음 표에서는 서버 부속 시스템에 대한 CSCI 환경 변수 및 ACSLS/CSCI 소프트웨어의 배포에 제공된 것과 비슷한 셸 스크립트에 대해 설명합니다. 이 셸 스크립트는 서버에서 TCP/IP로 시작됩니다.

표 N.1. CSCI 서버 부속 시스템 환경 변수

이름	설명
START_CSCI_PROCESS	ACSLS 시작 시 CSCI 자동 시작(TRUE 또는 FALSE)
CSCI_TRACE	이 변수는 CSCI 추적이 ON 또는 OFF인지 여부를 나타냅니다.
CSCI_SLEEP_TIME	이 변수는 연결 테이블(폴링 시간 초과)을 통한 각 라운드 로빈 루프 끝에서 PER 일시 정지 시간을 나타냅니다. 기본값은 100밀리초입니다.
CSCI_INITIAL_CONNECTION	이 변수는 첫번째로 열 CSCI의 초기 연결 이름을 나타냅니다. 이 이름은 CSCI의 초기 시작 중 i/o가 필요한 연결 서비스 이름입니다. 클라이언트는 일반적으로 IPC를 먼저 열고, 서버는 TCP/IP를 먼저 엽니다.
CSCI_SERVER_NAME	이 변수는 CSCI 서버의 이름을 나타냅니다. 단일 CSCI 서버에 정의된 모든 CSCI의 클라이언트에 대해 동일하게 설정해야 합니다. 서버 이름은 해당 CSCI 클라이언트 CSCI_SERVER_NAME과 일치해야 합니다. 서버와 클라이언트 CSCI_SERVER_NAME을 대조하여 시스템 간에 엔드 투 엔드 논리적 링크를 얻을 수 있습니다. 마지막으로 이 이름은 나중에 다른 환경 변수에서 ”connectionname”에 대해 사용됩니다. CSCI_SERVER_NAME에 지정된 값은 이후 변수의 connectionname 부분에 복제되어야 합니다.
CSCI_connectionname_NET_TYPE	이 변수는 CIF 부속 시스템에서 이 CSCI가 구성된 네트워크 유형을 나타냅니다. NETTYPE은 이 CSCI에서 사용되는 전송 계층을 정의합니다. 유효한 지정은 LU62 또는 IPC입니다.
CSCI_connectionname_INPUT_SOCKET	이 변수는 이 CSCI에 대한 입력으로 사용할 입력 소켓 번호를 나타냅니다. 이 번호는 서버 또는 클라이언트 응용 프로그램 출력 소켓 번호와 일치해야 합니다.
CSCI_connectionname_CON_TYPE	이 변수는 이 CSCI 유형을 나타냅니다. 이 *CON_TYPE*은 CSCI를 SERVER 또는 CLIENT로 정의합니다.
CSCI_connectionname_TRANSLATE_TYPE	이 변수는 실행 중 적용할 이 CSCI XDR 유형을 나타냅니다. 이 변수는 XDR 또는 NONE으로 지정할 수 있습니다.
CSCI_connectionname_DESTINATION_SOCKET	이 변수는 CSCI 서버에서 사용되는 대상 또는 출력 소켓 번호를 나타냅니다. 이 소켓 번호는 CSCI 서버 응용 프로그램에 대해 일치하는 입력 소켓 번호가 됩니다.

부록 O

부록 O. 매체 관리

이 장에서는 서로 다른 밀도로 기록된 동일 유형의 매체를 관리하는 방법 및 이전 밀도로 쓰여진 카트리지와 새 밀도로 쓰여진 카트리지를 구분하는 방법에 대해 설명합니다.

개요

이 개요에서는 다음과 같은 항목에 대해 다룹니다.

- 제한 사항 및 발생 가능한 문제
- 서로 다른 밀도에서 동일 매체에 기록되는 테이프 드라이브 예
- 권장 해결 방법

제한 사항

새로운 테이프 드라이브가 소개될 때는 기존 테이프 매체를 사용하면서도 더 높은 밀도로 기록을 수행하는 경우가 많습니다. 이러한 새 드라이브는 이전 밀도로 기록된 테이프를 읽을 수 있지만, 이전 밀도로 쓰기를 수행할 수는 없습니다. 이전 테이프 드라이브는 높은 밀도로 읽기 또는 쓰기를 수행할 수 없습니다.

이러한 제한 사항으로 인해 다음과 같은 문제가 발생할 수 있습니다.

- 새 밀도로 쓰여진 테이프가 이전 드라이브에 마운트된 경우, 이전 드라이브가 테이프를 읽을 수 없습니다.
- 스토리지 관리 응용 프로그램이 나중에 파일을 추가하여 부분적으로 사용된 테이프를 채우려고 시도할 때, 새 테이프 드라이브가 읽을 수 있지만 쓸 수 없는 다른 밀도를 사용해서 테이프가 쓰여진 경우 작업이 실패합니다.

라이브러리에 이전 테이프 드라이브와 새로운 테이프 드라이브가 혼용된 경우에는 매체 유형이 동일한 테이프 카트리지를 관리해야 합니다.

예

다음 예에서는 동일한 매체에 기록되지만, 밀도가 서로 다른 테이프 드라이브를 보여줍니다.

- T10000A 및 T10000B 테이프 드라이브에서 사용되는 T10000 매체

  T10000B 테이프 드라이브는 T10000A와 동일한 매체를 사용하지만 T10000A 밀도의 두 배로 데이터를 기록합니다. T10000B는 T10000A 매체를 읽을 수 있으며, T10000B 밀도 데이터를 기록하기 위해 이를 재생 이용(테이프 처음부터 쓰기)할 수 있지만, 이전에 기록된 T10000A에 데이터를 첨부할 수 없습니다.

T10000A 드라이브는 T10000A 밀도 데이터를 쓰기 위해 T10000B 카트리지를 재생 이용할 수 있지만, T10000B 카트리지에서 읽거나 데이터를 첨부할 수 없습니다.

- T9840A, T9840B, T9840C 및 T9840D 테이프 드라이브에서 사용되는 9840 매체

  T9840A, T9840B, T9840C 및/또는 T9840D 드라이브가 혼합된 경우에는 9840 매체를 관리해야 합니다. 이유:

  - T9840A 및 T9840B

    T9840A 및 T9840B 테이프 드라이브는 동일 매체를 사용하며 동일 밀도로 기록합니다.

  - T9840C

    T9840C는 T9840A 및 T9840B와 동일한 매체를 사용하지만 두 배의 밀도로 기록합니다.

  - T9840D

    T9840D는 T9840C 드라이브 밀도보다 거의 두 배로 기록합니다.

    T9840A, T9840B, T9840C 및/또는 T9840D 드라이브가 혼합된 경우에는 9840 매체를 관리해야 합니다.

- T9940A 및 T9940B 테이프 드라이브에서 사용되는 9940 매체
  - T9940A

    T9940A 드라이브는 단일 밀도 데이터 쓰기를 위해 T9940B 카트리지를 재생 이용할 수 있지만, T9940B 카트리지에서 읽거나 데이터를 첨부할 수 없습니다.

  - T9940B

    T9940B 테이프 드라이브는 T9940A와 동일한 매체를 사용하지만 T9940A 밀도의 두 배로 데이터를 기록합니다. T9940B는 T9940A 매체를 읽을 수 있으며, 두 배 밀도로 데이터를 쓰기 위해 이를 재생 이용할 수 있습니다. 하지만 이전에 기록된 T9940A 카트리지에 데이터를 첨부할 수 없습니다.

해결 방법

ACSLS에는 두 개 이상의 드라이브 유형이 쓰기를 수행할 수 있지만, 기록 밀도가 호환되지 않는 경우에 공통 매체를 관리하는 데 사용할 수 있는 도구가 포함되어 있습니다. 클라이언트 응용 프로그램은 이러한 도구를 사용해서 데이터 읽기/첨부 비호환성을 관리해야 합니다.

다음과 같은 방법을 사용해서 ACS 내에서 밀도가 서로 다른 공통 매체에서 기록을 수행하는 드라이브를 관리합니다.

- ACS에 있는 모든 이전 드라이브를 동시에 새 드라이브로 교체합니다.

  이 방법은 가장 간단하고 안전한 전략입니다. 이 전략을 사용하면 서로 다른 밀도를 사용하는 드라이브 조합을 관리하는 데 따른 문제를 방지할 수 있습니다. 이렇게 할 수 없는 경

우에는 두번째 글머리 기호에 설명된 것처럼 이전 드라이브를 새 드라이브로 점진적으로 교체할 수 있습니다.

주:

이전 드라이브를 새 드라이브로 교체한 후 이전 기록 밀도로 쓰여진 테이프에는 파일을 첨부하지 마십시오. Veritas NetBackup에서 이 작업은 테이프 일시 중지를 통해 수행됩니다.

- 이전 드라이브를 새 드라이브로 점진적으로 교체합니다.

  이를 위해서는 서로 다른 밀도로 기록된 공통 매체를 관리해야 합니다. 이를 위해서는 다음과 같은 방법을 선택할 수 있습니다.
  - 카트리지가 SL8500 및 SL3000 라이브러리에서 마운트 해제될 때 반환된 기록 형식 정보를 사용해서 매체를 관리합니다. 이 방법에 대해서는 다음 절에서 설명합니다.
  - 각 형식에 대해 별도의 ACSLS 풀을 만듭니다.
  - 백업 응용 프로그램(예: Veritas NetBackup, Legato NetWorker, IBM Tivoli 또는 CA BrightStor)의 기능을 사용해서 매체 풀을 관리합니다.

ACSLS에서 보고된 기록 밀도를 사용하여 공통 매체 관리

카트리지가 최신 라이브러리의 최신 테이프 드라이브에서 마운트 해제되면 해당 카트리지의 기록 형식이 ACSLS에 보고됩니다. ACSLS는 이러한 기록 형식을 해당 데이터베이스에 저장합니다. *`display volume`* 명령을 사용하면 이 정보를 표시할 수 있습니다.

기록 형식은 다음과 같이 보고됩니다.

- 라이브러리:
  - SL3000
  - SL8500(4.10 이상 펌웨어)
- 테이프 드라이브:
  - 모든 T10000 테이프 드라이브(1.38 이상 펌웨어)
  - T9840A, T9840C 및 T9840D(T9840B를 제외한 모든 T9840 테이프 드라이브)(1.4.2 이상 펌웨어)
  - T9940A 및 T9940B 테이프 드라이브(1.4.2 이상 펌웨어)

아래 예에서는 T10000A 및 T10000B 형식으로 기록된 T10000 매체를 관리하는 방법에 대해 설명합니다. 다음 표에 따라 사용자 상황에 맞는 절차를 적용하십시오.

이전 형식 드라이브	새 형식 드라이브
T10000A	T10000B
T9940A	T9940B
T9840A/T9840B	T9840C 또는 T9840D
T9840C	T9840D

주:

아래 설명에서는 명령 및 유틸리티에 대해 다음과 같은 구문 규약이 사용됩니다.

- 있는 그대로 입력되는 명령 및 유틸리티는 **굵게** 표시됩니다.
- 변수(사용자가 정확한 값을 입력해야 함)는 기울임꼴로 표시됩니다.

절차

T10000B 드라이브가 설치된 다음 기록 형식 정보를 사용해서 호환되는 테이프 드라이브에 카트리지를 마운트합니다. T10000A 밀도로 기록된 카트리지는 T10000A 드라이브 또는 T10000B 드라이브에서 읽을 수 있지만, T10000A 드라이브만 T10000A 형식으로 기록된 데이터를 첨부할 수 있습니다. T10000B 드라이브만 T10000B 형식으로 기록된 카트리지에서 읽기 또는 첨부를 수행할 수 있습니다.

스크래치 카트리지를 마운트할 때는 특별한 처리가 필요하지 않습니다. 테이프 시작 부분부터 카트리지 쓰기를 수행할 때는 이전 기록 형식이 문제가 되지 않습니다.

기존에 데이터가 기록된 카트리지에 대해 읽기 또는 쓰기를 수행할 드라이브를 선택하려면 이 절차를 따릅니다. 다음 명령은 카트리지 기록 형식을 읽고 쓸 수 있는 드라이브를 식별합니다.

비스크래치 카트리지 마운트

1. 카트리지의 기록 형식을 표시합니다.

   ```
   display volume vol_id -f recording_format_family recording
   _format_model
   ```

   이 볼륨의 기록 형식이 표시됩니다.
2. 다음 *query* 명령을 사용해서 호환되는 드라이브를 식별합니다.

   ```
   query mount vol_id
   ```

   카트리지와 호환되는 드라이브가 표시됩니다.
   - 카트리지를 마운트하기 전에 올바른 드라이브 유형을 선택합니다.

     이 *query* 명령은 T10000A 및 T10000B 드라이브를 모두 반환합니다(두 드라이브 유형 모두 T10000 매체와 호환되기 때문).
   - 드라이브 유형을 사용해서 T10000A를 T10000B와 구분합니다.

     목록에서 호환되는 첫번째 드라이브를 선택하면 전달 작업이 최소화되어 라이브러리 성능이 향상됩니다.
3. 다음 명령을 사용해서 기록 형식과 호환되는 드라이브에 카트리지를 마운트합니다.

   ```
   mount vol_id drive_id
   ```

새 기록 형식으로 마이그레이션

- 카트리지의 모든 데이터가 만료된 경우에는 테이프의 시작 부분부터 새로운 기록 형식으로 카트리지 쓰기를 다시 수행할 수 있습니다.

  만료된 카트리지에 대한 자세한 내용은 “LSM 채우기 ”를 참조하십시오.

- 모든 T10000A 드라이브가 변환되었거나 T10000B 드라이브로 교체된 다음에는 T10000A 형식으로 기록된 카트리지에 데이터가 첨부되지 않은 경우 모든 T10000 매체를 T10000B 드라이브가 사용할 수 있습니다.

ACSLS 풀에서 다른 밀도로 기록된 공통 매체 관리

아래 예에서는 T9940A 및 T9940B 밀도로 기록된 9940 매체를 관리하는 방법에 대해 설명합니다. 다음 절차는 서로 다른 테이프 드라이브가 서로 다른 밀도의 공통 매체에서 기록되는 모든 경우에 적용됩니다. 다음 표에 따라 사용자 상황에 맞는 절차를 적용하십시오.

이전 형식 드라이브	새 형식 드라이브
T10000A	T10000B
T9940A	T9940B
T9840A/T9840B	T9840C 또는 T9840D
T9840C	T9840D
SDLT 220	SDLT 320

절차

1. ACS에서 모든 9940 데이터(비스크래치) 카트리지를 식별하고 이를 T9940A 풀에 지정합니다. 이 작업은 T9940A 드라이브가 있는 ACS에서 T9940B 드라이브를 설치하기 전에 수행합니다.

   이제 9940 스크래치 카트리지를 T9940A 또는 T9940B 풀에 지정할 수 있습니다.

   a. 다음 명령을 사용해서 T9940A 및 T9940B 매체 풀을 정의합니다.

      ```
      define pool pool_id
      ```

   b. 모든 9940 매체를 보고합니다.

      이 작업은 아래에 표시된 것처럼 *display volume* 명령(옵션 1) 또는 사용자 정의 *volrpt*(옵션 2)를 사용해서 수행할 수 있습니다.

      결과 파일이 기록됩니다.

      - 옵션 1

        display volume 명령을 사용해서 결과를 파일에 기록합니다.

        ```
        display volume * -media STK2P > filename
        ```

        설명:

        모든 9940 테이프 카트리지(매체 STK2P)가 보고됩니다.

        filename은 결과를 기록할 파일 이름입니다. 카트리지가 있는 **ACS ID**도 나열됩니다.

        출력 내용을 읽고 필요에 따라 특정 ACS에서 카트리지를 선택합니다.

하나의 ACS에서 카트리지를 선택하려면 *-home* 피연산자를 사용해서 이 ACS의 카트리지만 선택합니다.

```
display volume * -home acs_id.*,*,*,* -media STK2P> filename
```

- 옵션 2

  ACS의 모든 볼륨에 대해 사용자 정의 *volrpt*를 사용합니다. 결과 파일이 기록됩니다.

  ```
  volrpt -d -f custom_volrpt_file -a acs_id > filename
  ```

  설명:

  *custom_volrpt_file*은 사용자 정의 *volrpt*에서 보고된 필드를 지정하는 파일 이름입니다. 다음 필드가 보고됩니다.

  ```
  VOLUME_ID      6   2

  MEDIA_TYPE     7   2

  VOLUME_TYPE    4   2
  ```

  *acs_id*는 관리 중인 ACS의 ID입니다.

  *filename*은 결과를 기록할 파일 이름입니다.

출력 내용을 읽고 매체 유형이 STK2P인 볼륨만 선택합니다.

c. 선택한 볼륨을 적합한 풀에 지정합니다.

다음 명령을 사용해서 모든 비스크래치(*VOLUME_TYPE* **=** *D*) 카트리지를 T9940A 풀에 지정합니다.

```
set scratch off pool_id vol_id
```

다음 명령을 사용해서 스크래치 카트리지(*VOLUME_TYPE* **=** *S*)를 T9940A 또는 T9940B 풀에 지정합니다.

```
set scratch pool_id vol_id
```

2. T9940B 드라이브가 설치된 다음 풀을 사용해서 호환되는 테이프 드라이브에 카트리지를 마운트합니다.

T9940A 기록 밀도로 기록된 카트리지는 T9940A 드라이브 또는 T9940B 드라이브에서 읽을 수 있지만, T9940A 드라이브만 9940A 매체에 데이터를 첨부할 수 있습니다. T9940B 드라이브만 9940B 기록 밀도로 기록된 카트리지에서 읽기 또는 첨부를 수행할 수 있습니다.

다음 명령을 사용하면 T9940A 또는 T9940B 드라이브에 적합한 풀에서 카트리지 식별, 선택 및 마운트를 수행할 수 있습니다. 올바른 드라이브에 카트리지를 마운트할 수 있는 풀을 사용하십시오.

주:

올바른 드라이브 유형으로 스크래치 카트리지를 마운트하려면 절차 A를 따릅니다. 이미 데이터가 기록된 비스크래치 카트리지를 마운트하려면 절차 B를 따릅니다.

절차 A - 스크래치 카트리지 마운트

a. 드라이브를 질의해서 드라이브 유형을 식별합니다.

```
query drive drive_id | all
```

b. 원하는 드라이브에 마운트하려는 카트리지에 대해 올바른 풀(매체 유형)을 식별합니다.
c. 스크래치 카트리지를 지정된 풀의 드라이브에 마운트합니다.

```
mount * drive_id pool_id
```

절차 B - 비스크래치 카트리지 마운트

a. 카트리지와 호환되는 드라이브 상태를 표시합니다.

```
display volume vol_id -f pool
```

이 볼륨의 풀이 표시됩니다.

b. 질의 명령을 사용해서 호환되는 드라이브를 식별합니다.

```
query mount vol_id
```

카트리지와 호환되는 드라이브가 표시됩니다.

카트리지를 마운트하기 전에 올바른 드라이브 유형을 선택합니다. 이 질의 명령은 T9940A 및 T9940B 드라이브를 모두 반환합니다(두 드라이브 유형 모두 9940 매체와 호환되기 때문).

드라이브 유형을 사용해서 T9940A를 T9940B와 구분합니다.

c. 선택한 드라이브에서 카트리지를 마운트합니다.

```
mount vol_id drive_id
```

3. 카트리지의 모든 데이터가 만료되었으면, 새 기록 밀도를 위해 스크래치 카트리지를 풀로 마이그레이션할 수 있습니다. T9940A 및 T9940B 드라이브는 해당 밀도로 카트리

지를 다시 초기화할 수 있으므로, 스크래치 카트리지를 다른 풀에 다시 지정할 수 있습니다.

```
set scratch pool_id vol_id
```

4. 모든 T9940A 드라이브가 변환되었거나 T9940B 드라이브로 교체된 다음에는 데이터가 T9940A 기록 밀도로 기록된 카트리지에 첨부되지 않은 경우 모든 9940 매체를 T9940B 드라이브가 사용할 수 있습니다.

부록 P

부록 P. XAPI 클라이언트 인터페이스

이 장에서는 ACSLS에 대한 XAPI 클라이언트 인터페이스를 소개합니다.

ACSLS 서버에 대한 XAPI 클라이언트 인터페이스

XAPI(XML API)는 StorageTek 클라이언트 및 서버가 TCP/IP를 통한 일반 프로토콜을 사용하여 통신할 수 있는 API입니다. ACSLS 8.4 이상 릴리스는 XAPI 지원과 함께 구성될 수 있습니다.

MVS 기반 서버(실제 테이프의 경우 HSC 및/또는 가상 테이프의 경우 VTCS)를 사용해야 했던 클라이언트는 이제 ACSLS를 사용하여 StorageTek 라이브러리를 관리할 수 있습니다.

XAPI 및 XCMD 구성, 관리 및 운영자 명령과 지원되는 XAPI 사용자 요청 및 ACSLS XAPI 컨트롤 변수에 대한 자세한 내용은 ELS 설명서 세트에 있는 *StorageTek Enterprise Library Software XAPI Client Interface to ACSLS Server* 설명서를 참조하십시오.

ACSLS XAPI 서비스

ACSLS XAPI 서비스 설치 및 설치 해제 절차는 *StorageTek ACSLS 8.4 Installation Guide*를 참조하십시오.

설치되면 ACSLS를 사용하여 XAPI 구성 요소가 시작되고 중지됩니다.

- XAPI 서비스와 함께 ACSLS를 시작하려면 Unix 명령 프롬프트에 다음을 입력합니다.

  ```
  acsss enable
  ```

- ACSLS 및 XAPI 서비스를 중지하려면 Unix 명령 프롬프트에 다음을 입력합니다.

  ```
  acsss disable
  ```

- ACSLS, XAPI 서비스, ACSLS 데이터베이스 및 모든 ACSLS 구성 요소를 중지하려면 Unix 명령 프롬프트에 다음을 입력합니다.

  ```
  acsss shutdown
  ```

XAPI 변수

ACSLS *acsss_config* 또는 *dv_config* 유틸리티를 사용하여 ACSLS XAPI 정적 변수를 표시하고 설정합니다.

변경사항을 적용하려면 ACSLS를 다시 시작해야 합니다.

- *dv_config*
  - *dv_config -d* - 모든 ACSLS 변수를 표시합니다.
  - *dv_config -p <variable_name>* - XAPI 변수를 업데이트합니다.

  6장. *ACSLS* 동작을 제어하는 변수 설정을 참조하십시오.
- *acsss_config*

  옵션 9를 선택하여 XAPI 변수를 표시하고 변경합니다. 이 옵션은 XAPI 서버가 설치된 경우에만 표시됩니다.

```
ACSLS Feature Configuration

Enter the number followed by Return for your choice from the following menu to
 configure product behavior in that area.

Press ? followed by the Return key for help.

   1: Set CSI tuning variables
   2: Set event logging variables
   3: Set general product behavior variables
   4: Set access control variables
   5: Set automatic backup parameters
   6: Rebuild Access Control information
   7: Event Notification settings
   8: Define or Change Library Configuration
   9: Set XAPI server variables
   E: Exit

Menu choice:
```

XAPI 변수는 다음과 같습니다.

- *XAPI_PORT*

  프롬프트: *XAPI* 서버에 대한 사용자 정의 인바운드 포트 변경사항은 *XAPI* 서버를 다시 시작해야 적용됩니다. 수신 *XAPI* 요청을 받기 위해 *XAPI* 서버에서 사용되는 포트 번호입니다. *[50020]:*

  이 옵션은 클라이언트의 TCP 요청 수신을 위해 XAPI 서버에서 사용되는 포트를 지정합니다. XAPI 서버에서 사용되는 포트를 정의하려면 1024에서 65535 사이의 숫자를 입력하십시오. 포트 50003은 지정하지 **마십시오**.

  이 변수를 적용하려면 XAPI 서버를 다시 시작해야 합니다.
- *XAPI_WORK_PATH*

프롬프트: *XAPI* 작업 디렉토리 변경사항은 *XAPI* 서버를 다시 시작해야 적용됩니다. *XAPI* 로그 및 추적 정보를 *[$ACS_HOME/log/xapi*] 디렉토리에 배치합니다.

XAPI 서버 작업 파일이 배치되는 디렉토리를 선택하십시오. 설치되면 XAPI 서버가 정보를 *$ACS_HOME/log/xapi* 디렉토리에 기록합니다. 일반적인 사용에서 이 변수의 값은 변경되지 않습니다. *$ACS_HOME*을 포함하는 파일 시스템에 디스크 공간 문제가 있는 경우 대체 경로를 지정할 수 있습니다. 지정된 경로가 절대 경로여야 합니다(/ 또는 *$ACS_HOME*으로 시작되는 경로).

이 변수를 적용하려면 XAPI 서버를 다시 시작해야 합니다.

- *XAPI_LOG_SIZE*

  프롬프트: *XAPI* 로그 크기 변경사항은 *XAPI* 서버가 다시 시작될 때까지 적용되지 않습니다. 최대 *XAPI* 로그 크기*(MB)*입니다. *[20]*:

  이 옵션에서는 MB로 표현된 XAPI 로그에 대한 임계값 크기를 지정합니다(여기서는 "1048576바이트"로 정의됨). 음수가 아닌 숫자를 입력하십시오. 이 옵션의 기본값은 20입니다.

  이 변수를 적용하려면 XAPI 서버를 다시 시작해야 합니다.

- *XAPI_LOG_FILE_NUM*

  프롬프트: 보존할 *XAPI* 로그 아카이브 파일 수입니다. *[10]:*

  이 옵션에서는 보존할 아카이브된 XAPI 로그 파일 수를 지정합니다. 현재 vlog.file 크기가 임계값 크기를 초과하면 해당 로그 파일은 0-n 접미사가 붙은 이름으로 바뀝니다. 0은 가장 최근 파일이며, n은 가장 오래된 파일입니다. 아카이브된 파일은 *API_WORK_PATH* 디렉토리에 저장됩니다. 지정된 아카이브된 로그 수에 도달하는 경우 새 파일이 아카이브 디렉토리에 추가될 때마다 가장 오래된 파일이 이 디렉토리에서 제거됩니다. 1개에서 99개 사이의 아카이브된 파일을 보존할 수 있습니다. 보존할 아카이브된 로그 파일 수를 지정하려면 1에서 99 사이의 숫자를 입력하십시오.

  이 변수를 적용하려면 XAPI 서버를 다시 시작해야 합니다.

- *XAPI_TRACE_SIZE*

  프롬프트: *XAPI* 추적 크기 변경사항은 *xapi* 서버를 다시 시작해야 적용됩니다. 최대 *XAPI* 추적 크기*(MB)*입니다. *[50]:*

  이 옵션은 XAPI 추적에 대한 임계값 크기(MB)를 지정합니다(여기서는 "1048576바이트"로 정의됨). 음수가 아닌 숫자를 입력하십시오. 이 옵션의 기본값은 50입니다.

  이 변수를 적용하려면 XAPI 서버를 다시 시작해야 합니다.

- *XAPI_TRACE_FILE_NUM*

  프롬프트: 보존할 *XAPI* 추적 아카이브 파일 수입니다. *[10]:*

이 옵션에서는 보존할 아카이브된 XAPI 추적 파일 수를 지정합니다. 현재 vtrace.file 크기가 임계값 크기를 초과하면 해당 추적 파일은 0-n 접미사가 붙은 이름으로 바뀝니다. 0은 가장 최근 파일이며, n은 가장 오래된 파일입니다. 아카이브된 파일은 *XAPI_WORK_PATH* 디렉토리에 저장됩니다. 지정된 아카이브된 로그 수에 도달하는 경우 새 파일이 아카이브 디렉토리에 추가될 때마다 가장 오래된 파일이 이 디렉토리에서 제거됩니다. 1개에서 99개 사이의 아카이브된 파일을 보존할 수 있습니다. 보존할 아카이브된 로그 파일 수를 지정하려면 1에서 99 사이의 숫자를 입력하십시오.

이 변수를 적용하려면 XAPI 서버를 다시 시작해야 합니다.

- *XAPI_STARTUP_FILE*

  프롬프트: *XAPI* 시작 파일 이름 변경사항은 *XAPI* 서버를 다시 시작해야 적용됩니다. 컨트롤 매개변수가 있는 *XAPI* 시작 파일의 이름입니다. *[xapi_startup_file]:*

  이 옵션은 XPI 시작 파일의 이름을 지정합니다. 이 파일은 *XAPI_WORK_PATH* 디렉토리에 있으며 XAPI 시작 매개변수를 포함합니다.

  이 변수를 적용하려면 XAPI 서버를 다시 시작해야 합니다.

- *XAPI_TAPEPLEX_NAME*

  프롬프트: *XAPI Tapeplex* 이름 변경사항은 *XAPI* 서버를 다시 시작해야 적용됩니다. *XAPI Tapeplex*의 이름입니다. *[ ]*:

  이 옵션에서는 XAPI Tapeplex의 이름을 지정합니다. 이 변수를 적용하려면 XAPI 서버를 다시 시작해야 합니다.

  8자를 초과하지 않는 텍스트를 입력하십시오.

부록 Q

부록 Q. ACSLS의 접근성 기능

오라클은 장애가 있는 사용자가 액세스할 수 있는 고품질 정보 기술을 제공하기 위해 노력하고 있습니다. ACSLS의 특정 접근성 기능이 여기에 요약되어 있습니다.

사용자 인터페이스

이 절에서는 ACSLS 사용자 인터페이스에 대해 설명합니다.

ACSLS(텍스트 전용) CLI(명령줄 인터페이스)

ACSLS의 기본 CLI는 *cmd_proc*라고 부릅니다. ACSLS 작업자는 다음을 포함해서 *cmd_proc*로부터 모든 물리적 라이브러리 작업을 조작할 수 있습니다.

- 마운트, 마운트 해제, 넣기, 꺼내기 및 이동 작업과 같은 테이프 이동 작업.
- 테이프 드라이브 및 테이프 볼륨 위치와 상태, lsm 상태, lmu 상태 및 CAP 상태에 대한 질의 작업.
- 모든 물리적 라이브러리 리소스를 온라인 및 오프라인으로 전환하는 제어 작업 및 여러 라이브러리 인터페이스 간 전환 기능(switch lmu).
- 테이프 소유권, 풀 관리, 작업 제어를 위한 잠금 기능, 청소 카트리지 관리와 같은 테이프 관리 작업.

CLI에서 액세스할 수 없는 관리 기능은 다음과 같습니다.

- 논리적 라이브러리 관리: 논리적 라이브러리 만들기, 테이프 드라이브 지정 및 지정 해제, 논리적 라이브러리에 대한 테이프 볼륨 지정 및 지정 해제를 위한 기능.
- 논리적 라이브러리 작동: 구성된 드라이브에 대한 논리적 볼륨 마운트 및 마운트 해제 기능.

두번째 CLI인 *lib_cmd*는 ACSLS GUI와 함께 논리적 라이브러리를 관리하는 데 사용됩니다. *lib_cmd*를 사용하면 다음을 수행할 수 있습니다.

- 논리적 라이브러리를 만들거나 삭제합니다.
- 논리적 라이브러리의 속성을 편집합니다.
- 논리적 라이브러리에 볼륨 또는 드라이브를 지정 및 지정 해제합니다.
- 클라이언트 응용 프로그램과 논리적 라이브러리 사이의 매핑을 정의합니다.
- 논리적 라이브러리의 상태를 표시합니다.
- 물리적 또는 논리적 라이브러리에서 볼륨 또는 드라이브의 상태를 표시합니다.
- 논리적 라이브러리 또는 지정된 드라이브를 온라인 또는 오프라인으로 전환합니다.

기본 관리 인터페이스: Unix 셸 기능

ACSLS 설치 및 구성, ACSLS 데이터베이스 백업 및 복원, 보조 유틸리티 작동을 위한 기본 기능은 Solaris에서 제공되는 기본 UNIX 셸(sh, csh, ksh, bash)을 통해 액세스할 수 있습니다.

ACSLS GUI(그래픽 사용자 인터페이스)

ACSLS GUI는 세 가지 프레임으로 구분됩니다.

- 위쪽 프레임에는 5개의 제목이 있는 탐색 버튼과 사용자 ID, 서버 ID 및 대시보드 레벨 상태를 포함한 요약 정보가 표시된 ACSLS GUI 마스트헤드가 항상 표시됩니다.
- 왼쪽 프레임에는 확장 가능한 탐색 트리가 항상 표시됩니다.
- 오른쪽 프레임에 표시되는 내용은 사용자가 탐색 트리 또는 마스트헤드의 Preferences 버튼으로부터 선택한 컨텐츠를 명제화함에 따라 동적으로 변경됩니다.

웹 기반 GUI를 통해 사용자는 모든 논리적 라이브러리 작업 및 대부분의 물리적 라이브러리 작업을 수행할 수 있습니다. GUI 라이브러리 작업에는 다음이 포함됩니다.

- 논리적 라이브러리 만들기, 논리적 라이브러리에 대한 볼륨 지정 또는 지정 해제, 볼륨 소유권 또는 청소 속성 설정을 위한 관리 기능.
- 마운트, 마운트 해제, 넣기 및 꺼내기와 같은 물리적 및 논리적 테이프 이동 작업.
- 물리적 테이프 드라이브 및 테이프 볼륨 위치와 상태, lsm 상태, lmu 상태 및 CAP 상태에 대한 질의 작업.
- 논리적 라이브러리에 지정된 물리적 라이브러리 리소스에 대한 질의 작업.
- 물리적 및 논리적 라이브러리 리소스를 온라인 및 오프라인으로 전환(*vary*)하는 제어 작업.

GUI에서 액세스할 수 없는 관리 기능은 다음과 같습니다.

- 여러 라이브러리 인터페이스 간 전환 기능(switch lmu).
- 풀 관리 및 작업 제어를 위한 잠금 기능을 포함하는 특정 테이프 관리 작업.

이러한 작업은 *cmd_proc* 인터페이스를 통해 처리됩니다.

ACSLS GUI의 접근성 프로비전

- 컨텐츠 프레임에 표시되는 모든 페이지는 페이지 상단의 제목으로 식별됩니다.
- GUI의 모든 그래픽 요소에는 텍스트에 상응하는 도구 설명이 제공됩니다. 대체 텍스트를 표시하려면 마우스를 사용해서 해당 아이콘 위로 커서를 가져갑니다.
- 컨텐츠 프레임에 표시되는 모든 데이터 테이블은 열 및 행 헤더로 식별됩니다.
- 확장 가능한 탐색 트리는 기본적으로 완전히 확장되도록 구성할 수 있습니다.
- 마스트헤드에는 결함 상태 조건을 나타내는 색상별 대시보드 요소가 포함됩니다. 색맹 사용자는 각 상태 아이콘 옆에서 동일한 메시지를 의미하는 0보다 큰 숫자를 확인할 수 있으며, 에런트 상태를 공유하는 라이브러리 리소스 수를 확인할 수 있습니다.

ACSLS에서 접근성 모드를 구성하는 데 필요한 특수 단계

이 절에서는 ACSLS에서 접근성 모드를 구성하는 데 필요한 특수 단계에 대해 설명합니다.

GUI 트리 메뉴

ACSLS GUI는 확장 또는 축소된 트리 메뉴를 제공합니다. 트리 메뉴를 축소하면 복잡한 메뉴가 숨겨지고 처음 몇 개 항목만 표시됩니다. 이러한 메뉴 항목은 선택했을 때 다시 확장됩니다. 키보드 탐색 또는 메뉴에 대한 오디오 요약을 사용할 수 있는 시각 장애가 있는 사용자를 위해서는 보기에 숨겨진 GUI의 부분을 축소된 트리 메뉴에서 크게 표시할 수 있습니다. 확장된 트리 메뉴를 사용으로 설정하려면 다음 절차를 수행합니다.

1. ACSLS GUI 마스트헤드의 가장 오른쪽 위 모서리에 있는 3개 버튼 중 첫번째인 Preferences 버튼을 찾습니다. Preferences 버튼을 누르면 Preferences 페이지가 GUI의 오른쪽 프레임에 표시됩니다.
2. Preferences 프레임의 Per-user Preferences 부분에서 'Default Tree Menu' 드롭다운 메뉴를 찾습니다.
3. 드롭다운 화살표를 누르고 "expanded"를 선택합니다.
4. 확장 옵션을 선택한 다음에는 Preferences 프레임의 오른쪽 위 모서리에 있는 2개 버튼을 찾아서 "set" 버튼을 누릅니다.

용어

absent cartridge(존재하지 않는 카트리지) 데이터베이스에 있지만 카트리지에 대한 모든 기록된 위치를 카탈로그로 지정할 때 찾지 못한 카트리지입니다. 보존 기간이 0이 아닌 값으로 설정된 경우에 볼륨 상태가 STATUS_VOLUME_ABSENT로 변경됩니다.

ACS Automated Cartridge System을 참조하십시오.

ACS Event Logger(ACSEL, ACS 이벤트 로거) 다른 ACSLS 구성 요소에서 메시지를 수신하고 이를 이벤트 로그에 기록하는 소프트웨어 구성 요소입니다.

ACS ID ACS의 고유 식별자입니다.

ACS Library Handler(ACSLH, ACS 라이브러리 처리기) LMU와 직접 통신하는 ACSLM 부분입니다.

ACS Library Manager(ACSLM, ACS 라이브러리 관리자) 라이브러리 요청 및 응답을 검증하고 경로 지정하는 소프트웨어 구성 요소입니다.

ACS Library Software(ACSLS, ACS 라이브러리 소프트웨어) 여러 라이브러리의 컨텐츠를 관리하고, ACS 테이프 드라이브에서 카트리지 마운트 및 마운트 해제를 위해 라이브러리 하드웨어를 제어합니다.

ACS library(ACS 라이브러리) 라이브러리는 하나 이상의 ACS, 연결된 테이프 드라이브 및 ACS에 있는 카트리지로 구성됩니다.

ACS System Administrator(ACSSA, ACS 시스템 관리자) 명령 프로세서와 나머지 시스템 사이의 인터페이스입니다.

ACSEL ACS Event Logger(ACS 이벤트 로거)를 참조하십시오.

ACSLH ACS Library Handler(ACS 라이브러리 처리기)를 참조하십시오.

ACSLM ACS Library Manager(ACS 라이브러리 관리자)를 참조하십시오.

ACSLS ACS Library Software(ACS 라이브러리 소프트웨어)를 참조하십시오.

ACSLS database(ACSLS 데이터베이스) 데이터 또는 청소 카트리지의 위치 및 상태에 대한 정보가 포함된 ACSLS 데이터베이스입니다. 이 정보에는 셀 위치, 스크래치 상태 등이 포함됩니다.

ACSLS database(ACSLS 데이터베이스) 라이브러리 구성 및 라이브러리에 있는 모든 데이터 또는 청소 카트리지의 위치 및 ID를 추적하기 위해 ACSLS에서 사용되는 데이터베이스입니다.

ACSLS platform(ACSLS 플랫폼)	ACSLS에 대해 적합한 환경을 제공하는 서버 하드웨어 및 소프트웨어입니다.
ACSSA	ACS System Administrator(ACS 시스템 관리자)를 참조하십시오.
ADI	Application Data Interchange(응용 프로그램 데이터 교환)의 약어입니다.
audit(감사)	라이브러리의 전체 또는 일부 컨텐츠에 대한 물리적 인벤토리입니다.
Automated Cartridge System(ACS, 자동화된 카트리지 시스템)	전달 포트를 통해 연결된 하나 이상의 라이브러리로 구성된 라이브러리 부속 시스템입니다.
automated library(자동화된 라이브러리)	library(라이브러리)를 참조하십시오.
backing library(원형 라이브러리)	논리적 라이브러리가 생성되는 물리적 라이브러리를 식별합니다.
beginning of tape(BOT, 테이프 시작 부분)	테이프에서 기록된 데이터가 시작되는 위치입니다.
BOT	Beginning of Tape(테이프 시작 부분)를 참조하십시오.
CAP	Cartridge Access Port(카트리지 액세스 포트)를 참조하십시오.
CAP ID	CAP 위치에 대한 고유 식별자입니다. CAP ID는 ACS ID, LSM 번호 및 CAP 번호로 구성됩니다.
Cartridge Access Port(CAP, 카트리지 액세스 포트)	LSM의 도어 패널에 내장된 양방향 포트로, 데이터 또는 청소 카트리지의 수동 넣기 또는 자동 꺼내기를 위해 제공됩니다.
cartridge drive(CD, 카트리지 드라이브)	2개 또는 4개 카트리지 드라이브 및 연관된 전원/공기 공급 장치가 포함된 장치입니다.
cartridge tape I/O driver(카트리지 테이프 I/O 드라이버)	카트리지 부속 시스템에 대해 명령(예: 읽기, 쓰기 및 되감기)을 실행하는 운영체제 소프트웨어입니다.
cartridge transport(카트리지 전송)	테이프에서 데이터를 쓰고 읽는 헤드를 통해 카트리지에서 테이프를 이동하는 전자 기계 장치입니다. 전송은 작동에 필요한 전기 및 공기를 공급하는 전원/공기 소스와 구별됩니다. cartridge drive(카트리지 드라이브)를 참조하십시오.
cartridge(카트리지)	데이터 레코딩 테이프 길이를 포함하는 플라스틱 하우징입니다. 테이프는 전송 장치에 로드될 때 자동으로 감깁니다. 자동 감기를 위한 플라스틱 리더 블록이 테이프에 부착되어 있습니다. 카트리지 스파인에는 볼륨 ID가 나열된 OCR/바코드 레이블이 포함될 수 있습니다.

CCI	client computing system(클라이언트 컴퓨팅 시스템)을 참조하십시오.
CD	cartridge drive(카트리지 드라이브)를 참조하십시오.
cell(셀)	LSM에서 카트리지가 보관되는 콘센트입니다.
channel(채널)	호스트 및 기본 스토리지를 입력 및 출력 제어 장치에 연결하는 장치입니다.
client applications(클라이언트 응용 프로그램)	테이프 카트리지 컨텐츠를 관리하는 소프트웨어 응용 프로그램입니다. 이러한 응용 프로그램은 ACSLS와 상호 작용을 통해 테이프 카트리지에 액세스합니다. 클라이언트 시스템에는 임의 개수의 클라이언트 응용 프로그램이 상주할 수 있습니다.
client computing system(클라이언트 컴퓨팅 시스템)	컴퓨터 및 실행 가능한 운영체제 이미지입니다.
client software(클라이언트 소프트웨어)	이 소프트웨어는 테이프 카트리지 컨텐츠를 관리하고, 카트리지에 대한 요청을 생성하고, 카트리지에서 데이터를 이동합니다. 클라이언트 소프트웨어는 ACSLS의 일부가 아닙니다.
Client System Component(클라이언트 시스템 구성 요소)	클라이언트 컴퓨팅 시스템의 운영체제와 ACSLS 사이의 인터페이스를 제공하는 소프트웨어입니다.
Client System Interface(CSI, 클라이언트 시스템 인터페이스)	ACS 라이브러리 관리자와 클라이언트 시스템 구성 요소 사이에 메시지를 변환하고 경로를 지정하는 소프트웨어 구성 요소입니다.
command access control(명령 액세스 제어)	명령에 대한 액세스를 제한합니다.
command area(명령 영역)	요청을 입력하고 응답을 수신하는 cmd_proc 인터페이스의 아래쪽 영역입니다.
command processor(cmd_proc, 명령 프로세서)	ACSSA의 화면 인터페이스입니다. cmd_proc를 사용하면 7장에 설명된 명령을 입력할 수 있습니다.
complex partitioning(컴플렉스 분할)	라이브러리 컴플렉스에서 전달 포트를 통해 연결된 여러 SL8500 라이브러리 간 분할입니다. 여기에는 또한 최대 16개 분할 영역에 대한 지원이 포함됩니다.
control path adapter(제어 경로 어댑터)	클라이언트 컴퓨팅 시스템의 제어 프로토콜을 StorageTek 라이브러리 제어 시스템의 제어 프로토콜로 변환하는 하드웨어 장치입니다.
control unit(CU, 제어 장치)	한 채널과 최대 15개의 카트리지 전송 사이에 논리적으로 준비되는 마이크로프로세서 기반 장치입니다. CU는 채널 명령을 전송 명령으로 변환하고 전송 상태를 채널에 전송합니다.

CSC	Client System Component(클라이언트 시스템 구성 요소)의 약어입니다.
CSE	Customer Services Engineer(고객 서비스 엔지니어)의 약어입니다.
CSI	Client System Interface(클라이언트 시스템 인터페이스)를 참조하십시오.
CSI variables(CSI 변수)	CSC와 CSI 사이의 통신을 미세 조정하기 위한 여러 옵션을 정의하는 데 사용됩니다. 이러한 변수는 acsss_config 프로그램에서 변경할 수 있습니다.
CU	control unit(제어 장치)을 참조하십시오.
cycle error messages(순환 오류 메시지)	라이브러리 또는 ACSLS 오류를 나타내는 메시지입니다.
data path adapter(데이터 경로 어댑터)	클라이언트 컴퓨팅 시스템의 데이터 프로토콜을 StorageTek 제어 장치의 데이터 프로토콜로 변환하는 하드웨어 장치입니다.
data path(데이터 경로)	테이프 카트리지에 대한 클라이언트 응용 프로그램의 읽기/쓰기 액세스를 사용으로 설정하는 네트워크 경로입니다.
database(데이터베이스)	상호 연관된 데이터 레코드의 모음입니다. ACSLS Database(ACSLS 데이터베이스)도 참조하십시오.
display area(표시 영역)	라이브러리 상태와 관련된 메시지를 수집하는 cmd_proc 인터페이스의 위쪽 영역입니다.
Dual TCP/IP(이중 TCP/IP)	호스트 소프트웨어(ACSLS 또는 HSC)와 라이브러리 간에 2개의 별도 호스트 연결을 제공합니다
dynamic configuration(동적 구성)	ACSLS가 온라인 및 실행 중인 상태로 유지되는 동안 ACSLS 라이브러리(및 구성 요소)에 대해 구성 변경사항을 구현할 수 있습니다.
ejected cartridge(배출된 카트리지)	라이브러리에서 배출된 카트리지입니다. 보존 기간이 0이 아닌 값으로 설정된 경우 카트리지 상태가 STATUS_VOLUME_EJECTED로 변경됩니다.
end of tape(EOT, 테이프 끝)	테이프에서 기록된 데이터가 끝나는 위치입니다.
EOT	end of tape(테이프 끝)를 참조하십시오.
EPO	Emergency Power Off(비상 전원 끄기)의 약어입니다.
EPROM	erasable programmable read only memory(소거 및 프로그래밍 가능 읽기 전용 메모리)를 참조하십시오.
erasable programmable read-only memory(EPROM, 소거	소거 및 재프로그래밍이 가능한 특별한 메모리 칩입니다.

및 프로그래밍 가능 읽기 전용 메모리)	
Event Log(이벤트 로그)	ACSEL에서 유지 관리되고 라이브러리 및 ACSLS 이벤트를 기술하는 메시지가 포함된 파일입니다.
Event Logger(이벤트 로거)	ACS Event Logger(ACS 이벤트 로거)를 참조하십시오.
external label identifiers(외부 레이블 식별자)	물리적 테이프 카트리지 식별을 위해 사용되는 카트리지의 외부 가장자리에 있는 6자리 영숫자 레이블입니다. A~Z의 대문자, 0~9의 숫자, $, # 및 공백으로 구성될 수 있습니다.
Fast Load(빠른 로드)	빠른 로드가 사용으로 설정된 경우 마운트 작업을 실행하는 FC 개시자는 ACSLS에서 작업이 검증 및 수락된 다음, 카트리지 이동이 시작되기 전에 성공한 응답을 수신합니다.
full installation(전체 설치)	새로운 고객 사이트 또는 새 라이브러리가 설치된 기존 사이트에 필요한 전체 소프트웨어 설치입니다.
HLI	Host/Library Interface(호스트/라이브러리 인터페이스)의 약어입니다. ACSLS가 라이브러리와 통신하는 단방향입니다.
HLI-attached(HLI 연결)	HLI를 통해 ACSLS에 연결된 라이브러리입니다. 이러한 라이브러리는 직렬 인터페이스(직렬 연결) 또는 TCP/IP 인터페이스(TCP/IP 연결)를 통해 연결할 수 있습니다.
home location(홈 위치)	제공된 카트리지와 연관된 셀입니다.
Host Software Component(HSC, 호스트 소프트웨어 구성 요소)	IBM 메인프레임에서 실행되는 소프트웨어로, 여러 라이브러리를 하나의 라이브러리 서버로 제어합니다.
I/O	Input/Output(입력/출력)의 약어입니다.
ID	identifier(식별자) 또는 identification(ID)의 약어입니다.
in-transit cartridges(이동 중인 카트리지)	소스와 대상 위치 사이에 있는 카트리지입니다. 카트리지는 전달 포트, 로봇 손 또는 플레이그라운드에 있을 경우 이동 중인 것으로 간주됩니다.
Initial Program Load(IPL, 초기 프로그램 로드)	시스템 재설정을 활성화하고, 웨이크업 진단을 시작하고(EPROM), 기능 코드를 로드하는 프로세스입니다.
inline diagnostics(인라인 진단)	부속 시스템 구성 요소에 있는 기능 마이크로코드를 사용해서 시간 공유 기반으로 작동하는 동안 부속 시스템의 구성 요소를 테스트하는 루틴입니다.
IPC	Interprocess Communication(프로세스간 통신)의 약어입니다.

IPL	Initial Program Load(초기 프로그램 로드)를 참조하십시오.
journal(저널)	마지막 체크포인트 이후 데이터베이스에서 변경된 항목의 순차적 로그입니다.
LAD	Lock Access Door(잠금 액세스 도어)의 약어입니다.
LAN	local area network(근거리 통신망)를 참조하십시오.
large CAP(LCAP, 큰 CAP)	각각 10개 셀의 이동식 매거진 4개에 스토리지 셀이 배열되어 있는 40 카트리지 CAP입니다. 매거진은 호스트 소프트웨어에 40개 셀이 포함된 단일 열로 표시됩니다.
LCAP	large CAP(큰 CAP)를 참조하십시오.
LCU	Library Control Unit(라이브러리 제어 장치)을 참조하십시오.
LED	Light Emitting Diode(발광 다이오드)를 참조하십시오.
LibAttach	Windows NT 클라이언트 시스템에서 상주하며, 클라이언트 응용 프로그램과 CSI 사이의 메시지를 변환 및 경로를 지정하는 소프트웨어 구성 요소입니다.
library configuration options(라이브러리 구성 옵션)	고객은 라이브러리에 있는 ACS 수 및 각 ACS와 서버 시스템 사이의 연결을 지정할 수 있습니다.
library control component(라이브러리 제어 구성 요소)	ACS에서 카트리지의 마운트 및 마운트 해제를 제어하는 소프트웨어입니다.
library control processor(라이브러리 제어 프로세서)	적절한 소프트웨어 추가를 통해 라이브러리 제어 소프트웨어 작동을 지원하는 올바르게 구성된 컴퓨터 하드웨어입니다.
library control software(라이브러리 제어 소프트웨어)	라이브러리 제어 구성 요소, 클라이언트 시스템 인터페이스 및 라이브러리 유틸리티를 포함하는 ACSLS의 소프트웨어 구성 요소입니다.
library control system(라이브러리 제어 시스템)	라이브러리 제어 소프트웨어(ACSLS)로 로드된 라이브러리 제어 플랫폼입니다.
Library Control Unit(라이브러리 제어 장치)	데이터 및 청소 카트리지 선택, 마운트, 마운트 해제 및 교체를 제어하는 LSM의 일부입니다.
library drive(라이브러리 드라이브)	클라이언트 시스템에 연결되어 있고 클라이언트 시스템에서 제어되는 LSM에 연결된 카트리지 전송입니다. 라이브러리 드라이브는 자동 테이프 카트리지 마운트 및 마운트 해제 작업 중 LCU와 상호 작용합니다. 라이브러리

	드라이브는 테이프 데이터 전송 작업 중 클라이언트 응용 프로그램과 상호 작용합니다. 라이브러리 드라이브는 ACSLM에서 개별적으로 처리할 수 있으며, 클라이언트 응용 프로그램에서 개별적으로 액세스할 수 있습니다. Cartridge Transport(카트리지 전송)를 참조하십시오.
library errors(라이브러리 오류)	라이브러리가 오프라인이거나, 하드웨어 오류가 발생했거나, 라이브러리를 사용할 수 없는 등의 이유로 발생하는 오류입니다.
Library Management Unit(LMU, 라이브러리 관리 단위)	LSM을 관리하고, 해당 리소스를 할당하고, ACSLS와 통신하는 ACS의 부분입니다.
Library Storage Module(LSM, 라이브러리 스토리지 모듈)	카트리지, 카트리지 드라이브, CAP 및 이들을 이동하는 데 필요한 로봇에 스토리지 영역을 제공하는 ACS 구조입니다.
library(라이브러리)	라이브러리는 하나 이상의 ACS, 연결된 테이프 드라이브, ACS 볼륨, ACS를 제어 및 관리하는 ACSLS 소프트웨어로 구성됩니다.
light emitting diode(LED, 발광 다이오드)	적은 양의 에너지를 사용하고 주로 on/off 상태를 표시하기 위해 사용되는 발광 장치입니다.
LMU	Library Management Unit(라이브러리 관리 단위)을 참조하십시오.
local area network(LAN, 근거리 통신망)	네트워크의 모든 구성 요소가 다른 구성 요소에 액세스할 수 있는 컴퓨터 네트워크입니다. 이 유형은 LMU와 연결된 LSM 사이의 인터페이스 유형입니다.
LSM	Library Storage Module(라이브러리 스토리지 모듈)을 참조하십시오.
LSM ID	LSM의 고유 식별자입니다. LSM ID는 ACS ID 및 LSM 번호로 구성됩니다.
media validation(매체 검증)	T10000C 및 T10000D 드라이브의 지정 풀을 사용해서 T10000 카트리지를 확인하는 기능입니다. 검증은 테스트된 각 테이프 카트리지에 대한 "성공" 또는 "실패" 결과를 제공합니다.
missing cartridge(누락된 카트리지)	데이터베이스에 있지만, 찾을 수 없는 카트리지입니다. 오프라인 LSM 또는 통신하지 않는 드라이브로 인해 카트리지에 대해 기록된 가능한 위치를 조사할 수 없으면, 카트리지가 ABSENT 대신 MISSING으로 표시됩니다. 카트리지 상태는 STATUS_VOLUME_MISSING으로 변경됩니다.
Multiple TCP/IP(다중 TCP/IP)	여러 라이브러리에 대한 TCP/IP 연결을 사용하여 호스트 소프트웨어(ACSLS 또는 HSC)와 SL8500 라이브러리 컴플렉스 간 중복 통신 경로를 제공합니다.
network adapter(네트워크 어댑터)	네트워크와 특정 연결 장비 사이의 전기 및 논리적 인터페이스를 제공하는 장비입니다.

Network Interface(NI, 네트워크 인터페이스)	네트워크 연결을 유지 관리하고 메시지 교환을 제어하는 서버 시스템과 클라이언트 시스템 사이의 인터페이스입니다. NI는 서버 시스템 및 각 클라이언트 시스템에 상주합니다.
NI	Network Interface(네트워크 인터페이스)를 참조하십시오.
OCR	Optical character recognition(광학 문자 인식)의 약어입니다.
ONC	Open network computing(개방형 네트워크 컴퓨팅)의 약어입니다.
Open Systems Interconnection(OSI, 개방형 시스템 상호 연결)	International Organization for Standardization(국제 표준화 기구)의 소프트웨어 아키텍처 모델입니다. OSI 모델은 데이터 처리 시스템의 상호 연결을 위한 표준을 제공합니다.
OSI	Open Systems Interconnection(개방형 시스템 상호 연결)을 참조하십시오.
OSLAN	Open Systems Local Area Network(개방형 시스템 근거리 통신망)의 약어입니다.
Partition(분할 영역)	ACSLS에서 개별 ACS로 관리되는 물리적 분할 영역에 지정된 라이브러리 셀, 카트리지, 드라이브 및 CAP의 분할 영역입니다.
Pass-Thru Port(PTP, 전달 포트)	하나의 LSM에서 다중 LSM ACS에 있는 다른 LSM으로 카트리지를 전달할 수 있게 해주는 메커니즘입니다.
PCAP	priority CAP(우선 순위 CAP)를 참조하십시오.
playground(플레이그라운드)	전원이 켜진 다음 LSM 초기화가 완료되기 전에 진단 카트리지 및 이동 중으로 확인된 카트리지를 저장하기 위해 사용되는 특수 셀(LSM 내부)의 예약된 영역입니다.
pool(풀)	하나 이상의 유사한 기능 또는 속성을 가진 테이프 카트리지 모음(예: 스크래치 테이프 풀)입니다.
POST	Power-on self-test(전원 공급 자가 테스트)의 약어입니다.
priority CAP(PCAP, 우선 순위 CAP)	카트리지 우선 순위 넣기 및 꺼내기에 사용되는 단일 카트리지 CAP입니다.
processing errors(처리 오류)	처리 또는 네트워크 통신 실패로 인해 발생하는 오류입니다.
PROM	Programmable read-only memory(프로그램 가능 읽기 전용 메모리)의 약어입니다.
PTP	Pass-Thru Port(전달 포트)를 참조하십시오.
RDBMS	Relational database management system(관계형 데이터베이스 관리 시스템)의 약어입니다.

redo log files(리두 로그 파일)	ACSLS 데이터베이스를 복원하는 데 사용되는 백업 파일입니다.
Redundant Electronics(중복 전자 부품)	선택적인 SL8500 RE(중복 전자 부품) 기능은 엔터프라이즈 라이브러리에서 페일오버 보호를 제공합니다. RE는 두 세트의 라이브러리 컨트롤러 카드를 사용합니다. 특정 시간에 한 세트는 활성 상태이고 다른 세트는 대기 상태입니다. 활성 라이브러리 컨트롤러는 ACSLS 또는 SLConsole의 명령에 응답하여 대기 라이브러리 컨트롤러로 페일오버할 수 있습니다. 라이브러리 카드에 장애가 발생할 경우 라이브러리에서 자동 페일오버를 시작할 수 있습니다.
relational database(관계형 데이터베이스)	데이터 항목 간 관계에 따라 구성 및 액세스되는 데이터베이스입니다. 관계는 테이블로 표현됩니다.
ROM	Read-only memory(읽기 전용 메모리)의 약어입니다.
RPC	Remote Procedure Call(원격 프로시저 호출)의 약어입니다.
SCAP	standard CAP(표준 CAP)를 참조하십시오.
scratch(스크래치)	비어 있거나 유용한 데이터를 포함하지 않음을 나타내는 테이프 카트리지의 속성입니다.
SCSI	Small computer serial interface(소형 컴퓨터 직렬 인터페이스)의 약어입니다.
Serial-attached(직렬 연결)	HLI-attached(HLI 연결)를 참조하십시오.
server system user(서버 시스템 사용자)	서버 시스템에서 ACSLS 명령, 유틸리티 또는 프로시저를 호출하는 사용자입니다. 서버 시스템 사용자는 일반적으로 사이트 및 유지 관리 관리자(예: 라이브러리 운영자, 테이프 라이브러리 관리자, 시스템 관리자, CSE 및 시스템 관리자)입니다.
server system(서버 시스템)	ACSLS에 대해 상주하는 라이브러리 부분이며, 현재는 라이브러리 제어 시스템이라고 부릅니다. 라이브러리 제어 시스템은 라이브러리와 클라이언트 시스템 사이의 인터페이스 역할을 수행합니다.
servo(서보)	피드백을 사용해서 프로세스를 제어하는 시스템입니다.
silo(사일로)	LSM에서 일반적으로 사용되는 용어입니다. Library Storage Module(라이브러리 스토리지 모듈)을 참조하십시오.
SIMM	Single inline memory module(단일 인라인 메모리 모듈)의 약어입니다.
SLOT	cell(셀)을 참조하십시오.
SQL	structured query language(구조적 질의어)를 참조하십시오.

SRN	service request number(서비스 요청 번호)를 참조하십시오.
SSI	Storage Server Interface(스토리지 서버 인터페이스)를 참조하십시오.
SSR	Software Support Representative(소프트웨어 지원 담당자)의 약어입니다.
Standard CAP(SCAP, 표준 CAP)	7개 고정 셀의 3개 행에 스토리지 셀이 배열되어 있는 21 카트리지 CAP입니다.
Storage Server Interface(SSI, 스토리지 서버 인터페이스)	클라이언트 시스템에서 상주하며, 클라이언트 응용 프로그램과 CSI 사이의 메시지를 변환 및 경로를 지정하는 소프트웨어 구성 요소입니다.
StorageTek Library Console	StreamLine 라이브러리에 대해 사용되는 운영자 패널 소프트웨어 응용 프로그램입니다.
structured query language(SQL, 구조적 질의어)	데이터베이스에서 데이터를 정의, 액세스 및 업데이트하는 데 사용되는 언어입니다.
system resource variable(시스템 리소스 변수)	ACSLS에서 사용되는 시스템 리소스의 양을 제어하는 데 사용됩니다.
system unit(시스템 장치)	라이브러리 제어 플랫폼입니다.
tape library management system (TLMS, 테이프 라이브러리 관리 시스템)	클라이언트 응용 프로그램의 유형입니다.
TCP	Transmission Control Protocol(전송 제어 프로토콜)의 약어입니다.
TLMS	tape library management system(테이프 라이브러리 관리 시스템)을 참조하십시오.
TOD	Time of day(일 중 시간)의 약어입니다.
UDP	User Datagram Protocol(사용자 데이터그램 프로토콜)의 약어입니다.
UNIX	처음에 Bell Laboratories(현재의 UNIX Systems Laboratories, Inc.)에서 개발되었고 다양한 컴퓨터 시스템에서 사용되는 운영체제입니다.
unsolicited messages(요청하지 않은 메시지)	오류를 나타내거나 특정 루틴 작업이 수행될 수 있는 경우를 알리는 메시지입니다.
UOC	Usable on codes(사용성 부호)의 약어입니다.

upgrade installation(업그레이드 설치)	기존 고객 사이트에서 새 버전의 ACSLS를 설치할 때 수행됩니다.
user selectable features and options variables(사용자 선택 가능 기능 및 옵션 변수)	다양한 사용자 선택 가능 기능 및 옵션을 정의하는 데 사용됩니다.
validation errors(검증 오류)	cmd_proc로 수행된 형식 및 구문 검증으로부터 발생한 오류입니다.
venter	Virtual enter(가상 넣기)의 약어입니다. 가상 레이블을 사용한 레이블이 지정되지 않은 카트리지 넣기를 의미합니다.
virtual label(가상 레이블)	물리적 레이블이 누락되었거나 판독할 수 없는 상태일 때 카트리지에 지정할 수 있는 논리적 볼륨 ID(volser)입니다.
volser	Volume Serial Number(볼륨 일련 번호)의 약어입니다.
volume access control(볼륨 액세스 제어)	일반적으로 클라이언트에 의한 볼륨에 대한 액세스를 제한합니다.
volume identifier(볼륨 식별자)	데이터베이스에 대해 데이터 카트리지를 고유하게 식별하는 6자의 문자열입니다.
volume serial number(volser, 볼륨 일련 번호)	외부 레이블 식별자와 동의어입니다.
volume(볼륨)	데이터 또는 청소 카트리지입니다.
WTM	write tape mark(테이프 표시 쓰기)의 약어입니다.
XDR	External data representation(외부 데이터 표현)의 약어입니다.
XML	Extensible Markup Language(확장 가능한 마크업 언어)의 약어입니다. 웹 기반의 구조화된 문서 및/또는 데이터에 대한 범용 형식입니다.

색인

기호

ㅅ

ㅇ