StorageTek Automated Cartridge System Library Software

High Availability クラスタインストール、構成および操作

Release 8.4

E69741-01

2015 年 12 月

ORACLE

StorageTek Automated Cartridge System Library Software
High Availability クラスタインストール、構成および操作

E69741-01

目次

図の一覧

はじめに

このガイドには、Solaris SPARC ベースのシステムと x86 ベースのシステムの両方に Oracle の StorageTek Automated Cartridge System Library Software High Availability (ACSLS HA) 8.4 Cluster ソフトウェアをインストールして構成するためのガイドラインと手順が記載されています。

ACSLS HA 8.4 は、ZFS ファイルシステムを備えた Solaris 11.2 で ACSLS 8.4 をサポートするように特に設計されています。このバージョンでは、ユーザー定義のファイルシステムへの ACSLS ソフトウェアのインストールがサポートされます。

対象読者

このドキュメントは、Solaris 11 オペレーティングシステムおよび ZFS を良く理解し、Solaris Cluster 4.*x* の実践的な知識を持つ熟練した UNIX システム管理者を対象としています。

このドキュメントには、使用されるほとんどのテクノロジに関する適度な背景情報が記載されており、予想される標準のインストール手順のガイダンスを提供します。ただし、このドキュメントだけでは、UNIX システムの熟知と専門知識の暗黙の要件の代わりにはなりません。

ドキュメントのアクセシビリティについて

オラクルのアクセシビリティについての詳細情報は、Oracle Accessibility Program の Web サイト (`http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`) を参照してください。

Oracle Support へのアクセス
サポートをご契約のお客様には、My Oracle Support を通して電子支援サービスを提供しています。詳細情報は (`http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info`) か、聴覚に障害のあるお客様は (`http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs`) を参照してください。

表記規則

このドキュメントでは、次のテキスト表記規則を使用しています。

表記規則	意味
太字	太字は、アクションに関連付けられたグラフィカルユーザーインタフェースの要素、またはテキストや用語集で定義される用語を示します。

表記規則	意味
斜体	斜体は、マニュアルタイトル、強調、または特定の値を指定するプレースホルダ変数を示します。
モノスペース	モノスペースは、段落内のコマンド、URL、例のコード、画面に表示されるテキスト、またはユーザーが入力するテキストを示します。

1

第1章 はじめに

ACSLS HA は、コンポーネントやサブシステムに障害が発生した場合に、中断なしのテープライブラリの制御サービスを保証するために、デュアル冗長性、自動回復、および自動フェイルオーバー回復を提供するハードウェア構成およびソフトウェア構成です。このドキュメントでは、ACSLS ソフトウェアに高可用性を提供するために必要な構成、設定、およびテストの手順について説明します。

手順を開始する前に、完全なインストールプロセスを確認することをお勧めします。クラスタ化されたアプリケーションのインストールプロセスには、細部まで厳しく注意する必要のある複数の手順が伴います。通常、この手順は UNIX システム統合の専門家によって実行されます。

ACSLS HA システムに関連付けられたハードウェアおよびソフトウェアコンポーネントが複数あるため、完全なインストール手順を実行するには数日間かかることもあります。既存の本番ライブラリ環境の場合、ACSLS HA インストールの処理中でもライブラリが運用されるように、お客様は単純なスタンドアロン ACSLS サーバーをインストールすることをお勧めします。

構成は 2 ノードのクラスタです。重大なシステム障害を検出できるモニタリングソフトウェアを備えた 2 つの完全なサブシステム (1 つはアクティブ、1 つはスタンバイ) が含まれます。サブシステムに回復不能な障害が発生した場合に、制御をプライマリからスタンバイシステムに切り替えることができます。この構成では、冗長電源装置、および一般的なスイッチオーバーの必要なしに即座にサブシステムの通信障害から回復できる冗長ネットワークと I/O の相互接続が提供されます。

ACSLS HA では、Solaris Cluster のモニタリングおよびフェイルオーバー機能および Solaris オペレーティングシステムのマルチパス機能を活用することで、最小の停止時間で回復性の高いライブラリの制御操作が実現されます。Solaris では、中断なしのネットワーク接続を保証する IP マルチパス、およびシステムデータへの中断なしのアクセスを保証する RAID 1 を使用したマルチパスディスク I/O が提供されています。Solaris Cluster は、オペレーティングシステム、内部ハードウェア、および外部 I/O リソースを含むシステムリソースの健全性を監視し、必要に応じてシステムのスイッチオーバーを管理できます。ACSLS HA エージェン

トは、ACSLS アプリケーション、そのデータベース、そのファイルシステム、および StorageTek ライブラリリソースへの接続をモニターして、必要に応じて Solaris Cluster のフェイルオーバーサービスを起動します。

このような冗長構成では、クラスタフレームワークの内外で常に認識されている単一の論理ホスト ID が ACSLS Library Control Server に割り当てられています。この ID は、必要に応じてクラスタノード間で自動的に転送されますが、転送時の停止時間は最小限です。

プロジェクトに着手する前に、ここで説明するように、ACSLS HA をインストールおよび構成する完全なプロセスを確認します。必要に応じて、Oracle の Advanced Customer Services でインストール全体の忠告、支援、および処理に対応できます。

ACSLS ドキュメントについては、次にある OTN (Oracle Technical Network) を参照してください。

http://docs.oracle.com/

システム要件

ACSLS HA サーバーは、1 つの RAID ディスクアレイを共有する 2 台の Solaris サーバーノードで構成されます。

クライアントのオプション

ACSLS HA は、Automated Cartridge System アプリケーションプログラミングインタフェース (ACSAPI) ネットワークインタフェースを使用するすべての ACSLS クライアントをサポートします。単一のネットワーク IP アドレスは 2 つのサーバーノード間で共有されるため、ACSAPI クライアントが共通の仮想ホスト ID を使用して ACSLS を処理できます。

SMCE (SCSI メディアチェンジャーエミュレーション) を使用してファイバチャネルクライアントに提供される論理ライブラリは、この製品でサポートされません。

サーバーのオプション

ACSLS HA 8.4 は、Solaris Cluster 4.2 の最小ハードウェア要件を満たすシステム上で実行するようにしてください。特定のサーバープラットフォーム要件については、『*Oracle Solaris Cluster のシステム要件*』というタイトルのドキュメントを参照してください。

http://www.oracle.com/technetwork/jp/server-storage/solaris-cluster/overview/sysreq-cluster-166689-ja.pdf

単一障害点を回避するには、各 ACSLS HA サーバーノードに次のものを構成する必要があります。

- デュアル (冗長) 電源装置
- 6 つの 10/100/1000 Base-T Ethernet ポート
- 2 つのファイバチャネルポート (FC 接続のストレージを使用している場合)
- 2 つの SAS ポート (SAS 接続のストレージを使用している場合)

システムが論理ライブラリ用に使用するように構成されている場合は、SCSI クライアントアプリケーションをサポートしている構成に、1 つ以上の専用ファイバチャネルポートを追加する必要があります。

ファイバ接続のライブラリ (SL500 や SL150 など) 用に構成するには、1 つのファイバチャネルポートを追加する必要があります。

ストレージアレイのオプション

サポートされているディスクアレイサブシステムについては、『*Oracle Solaris Cluster Storage Partner Program*』というタイトルのドキュメントを参照してください。

http://www.oracle.com/technetwork/server-storage/solaris-cluster/partnerprogram-cluster-168135.pdf

ネットワーク要件

合計で 7 つの IP アドレスを予約するようにしてください。

1 - 論理ホスト (クラスタの仮想 IP (VIP))

2 - ノード 1 の IP アドレス

3 - ノード 2 の IP アドレス

4 - ライブラリインタフェース 1 のソース IP アドレス (ノード 1)

5 - ライブラリインタフェース 2 のソース IP アドレス (ノード 1)

6 - ライブラリインタフェース 1 のソース IP アドレス (ノード 2)

7 - ライブラリインタフェース 2 のソース IP アドレス (ノード 2)

ライブラリインタフェース 1 とライブラリインタフェース 2 は異なるサブネット上に存在することが理想的です (図2.1「各サーバーノード上の 2 つの Ethernet ポートに接続されている単一の HBCr ライブラリインタフェースカード」を参照)。

ソフトウェア要件

ACSLS HA 8.4 では、次のソフトウェアコンポーネントが必要です。

- Oracle Solaris 11.2 (SPARC または X86)
- Oracle Solaris Cluster 4.2

ACSLS HA 用のインストール前チェックリスト

新しい ACSLS HA システムをインストールする前、または ACSLS HA システムを新しいリリースにアップグレードする前に、ACSLS HA をインストールするお客様環境に関する情報を確認し、記録します。

次のインストール前チェックリストを記入すると、リスクが排除されます。また、このチェックリストによりインストールがスムーズに処理され、お客様の環境に関する詳細を待っている間も遅延されません。

Oracle サポート担当者

- このお客様をサポートする現地の Oracle 担当者はだれですか。
- Solaris システム管理の経験者はいますか。
- ACSLS の経験者はいますか。
- ACSLS HA の経験者はいますか。

お客様サポート担当者

- ACSLS HA サーバーやお客様のネットワークなどのアクティビティーをサポートするお客様のシステム管理担当者はだれですか。
- Solaris システム管理の経験者はいますか。
- ACSLS の経験者はいますか。
- ACSLS HA の経験者はいますか。
- ネットワーク管理者はだれですか。

ACSLS HA システム用のハードウェア

- Oracle Sun サーバーのモデルは何ですか。
- Solaris リリースおよび更新のレベルは何ですか。

- メモリー (最小 10G バイト)。
- ディスクをミラー化するには、各サーバーにデュアルブートドライブが必要です。
- 共有ディスクのモデルは何ですか。これは Solaris Cluster によってサポートされていますか。
- 各 ACSLS サーバーを共有ディスクアレイに接続するために使用される SAS またはファイバ HBA。
- 各 ACSLS サーバーに 6 つの Ethernet ポートが必要です。
- ACSLS がファイバ接続のライブラリ (SL500 または SL150) を管理している場合や、ACSLS にファイバターゲットモードのポートを使用している論理ライブラリが表示されている場合は、各 ACSLS サーバー上にファイバ HBA が必要です。
- お客様環境に Solaris サーバーおよびディスクアレイを接続するために必要な電源コードは何ですか。たとえば、HA のインストールが遅延しないようにするには、プラグがお客様サイトの電源コンセントと一致する必要があります。
- HA のインストールを開始する前に、HA サーバーおよび共有ディスクアレイが正しく構成されていることを確認します。
  - Ethernet 接続用の 6 ポートの要件を満たすために、各 HA サーバーに追加のネットワークインタフェースコントローラ (NIC) カードが搭載されている必要があります。
  - 外部の共有ディスクアレイのインタフェースタイプに注意し、互換性のある HBA が各サーバー上に構成されていることを確認します。
  - ACSLS がファイバ接続のライブラリ (SL500 や SL150 など) と通信する場合は、ファイバ HBA が必要です。
  - ACSLS にファイバターゲットモードを使用している論理ライブラリが提供されている場合は、Qlogic Fibre HBA が必要です。

ネットワーク情報

次のネットワーク情報を確認します。

2 台の HA ACSLS サーバーに割り当てられている IP アドレスとホスト名

- インストールするには、合計で 9 つの IP アドレスが必要です。
  - ACSLS サーバーノード 1 のローカルアドレス
  - ACSLS サーバーノード 2 のローカルアドレス
  - 論理ホストアドレス (2 台の HA ノード間で共有される仮想 IP アドレス)
  - ノード 1 からのライブラリ接続 a

  - ノード 1 からのライブラリ接続 b (デュアル TCP/IP またはマルチ TCP/IP による冗長化の場合)
  - ノード 2 からのライブラリ接続 a
  - ノード 2 からのライブラリ接続 b (デュアル TCP/IP またはマルチ TCP/IP による冗長化の場合)
  - ノード 1 上の ILOM
  - ノード 2 上の ILOM

- ACSAPI クライアントは ACSLS と通信しますか。
  - フェイルオーバーイベントのあと、アクティブノードに仮想 IP アドレスが指定されます。仮想 IP アドレスは、どのノードがアクティブであるかにかかわらず ACSLS クライアントからアクセスできます。
  - ACSAPI クライアント上で ACSLS HA の仮想 IP アドレスまたはホスト名を設定する方法を知っていますか。この設定は、クライアント上で実行されている ISV アプリケーションによって異なります。

- TCP/IP 接続のライブラリ (SL8500、SL3000、および 9310) と通信するために ACSLS で使用される IP アドレス。
- 単一障害点を回避するには、各冗長ライブラリ接続を、それ独自の別個のサブネットを介して配線することをお勧めします。ネットワークトラフィックのボリュームが大きいことによる問題を防ぐために、サブネットはライブラリ通信用に予約し、一般のネットワーク通信からの干渉を受けないようにしてください。
- 各 ACSLS HA ノード上のサービスプロセッサ (ILOM や ALOM など) にアクセスするために必要な IP アドレスとパスワード。

HLI ライブラリとの通信

ACSLS と TCP/IP 接続のライブラリ間の通信は、ブロードキャストトラフィックから保護されたサブネット上で行われていますか。

ファイアウォール

- ACSAPI クライアントと ACSLS HA システム間にファイアウォールがありますか。
- ACSLS HA システムとその管理対象ライブラリ間にファイアウォールがありますか。

ファイアウォールが存在する場合は、*ACSLS 8.4 の管理者ガイド*でファイアウォールセキュリティーのオプションに関する付録を参照して、ファイアウォールを介して通信するように ACSLS および ACSAPI クライアントを構成する方法の詳細を確認してください。

ファイバを使用した SCSI メディアチェンジャー

- この ACSLS システムには、ファイバチャネルターゲットモードのポートを使用している論理ライブラリがクライアントに表示されていますか。その場合は、QLogic Fibre HBA が必要です。

インストールメディア

- Oracle eDelivery の Web サイトからソフトウェア (Oracle Solaris Cluster、ACSLS、ACSLS HA、その他の必要なパッケージやパッチなど) をダウンロードできるように、HA サーバーからインターネットへの直接的または間接的なアクセスを構築しますか。

  間接的なアクセスの場合は、ファイルをインターネットから HA サーバーに転送できることを確認します。
- ソフトウェアをインターネットからローカルマシンに直接ダウンロードする場合は、ブラウザの構成に必要な完全なプロキシ情報をインストール時に入手できることを確認します。

ACSLS と通信するクライアント (バックアップまたは ILM) アプリケーション

- ACSAPI クライアント (バックアップや Information Lifecycle Management アプリケーションなど) は ACSLS と通信しますか。
  - ACSAPI クライアントが ACSLS と通信する場合、クライアントアプリケーション (NetBackup や Oracle SAM など) は何ですか。
  - 使用するクライアントのバージョンは何ですか。
  - クライアントが Windows 上で実行され、LibAttach を使用している場合、実行されている LibAttach のバージョンは何ですか。
- クライアントアプリケーションは、ファイバチャネルターゲットモードのポートを使用して ACSLS 論理ライブラリと通信しますか。
  - クライアントアプリケーション (NetBackup や Oracle SAM など) は何ですか。
  - 使用するクライアントのバージョンは何ですか。

ACSLS のユーザー ID とグループ

ACSLS では、*acsls* グループにユーザー ID (*acsss*、*acsdb*、および *acssa*) が必要です。

これらのユーザー ID および *acsls* グループは、ローカルで ACSLS HA サーバー上に設定できますか。それとも、サイトのセントラルユーザーおよびパスワード管理システムに統合する必要がありますか。

高レベルのインストール手順

ACSLS HA を完全にインストールするには、次の手順を実行する必要があります。

1. 一般的な外部ファイバチャネルまたは SAS2 ディスクアレイに接続されている 2 台の Solaris プラットフォームサーバーをインストールします。各サーバー上に Solaris 11.2 をインストールします。

   Oracle Technology Network ライブラリから入手可能な『*Oracle Solaris 11* システムのインストール』ドキュメントを参照してください。

   `http://www.oracle.com/technetwork/documentation/solaris-11-192991.html`
2. 基本的な Solaris システムを構成します。
   - ユーザーアクセス権限
   - マルチパスネットワークアクセスとディスク I/O

   21 ページの2章「*ACSLS HA 用の Solaris システムの構成* 」を参照してください
3. ZFS ファイルシステムを構成します。
   - ルートストレージプール
   - ACSLS ストレージプール

   3章「*ZFS を使用したファイルシステムの構成* 」を参照してください
4. ソフトウェアパッケージをダウンロードします。39 ページの4章「*ソフトウェアパッケージのダウンロード* 」を参照してください
   - ACSLS 8.4.0
   - Solaris Cluster 4.2
   - ACSLS HA 8.4.0

   39 ページの4章「*ソフトウェアパッケージのダウンロード* 」を参照してください
5. ACSLS 8.4.0 とパッチ更新 (ある場合) をインストールします。45 ページの5章「*ACSLS 8.4 のインストール* 」を参照してください
6. Solaris Cluster 4.2 とパッチ更新 (ある場合) をインストールします。6章「*Solaris Cluster 4.2 のインストール* 」を参照してください
7. ACSLS HA 8.4.0 をインストールします。7章「*ACSLS HA 8.4 のインストールと起動* 」を参照してください

8. ACSLS HA 用に Cluster 操作を調整します。67 ページの8章「*ACSLS HA の微調整*」を参照してください

2

第2章 ACSLS HA 用の Solaris システムの構成

基本的な Solaris 11.2 システムがインストールされると、次の 3 つの状況で、Solaris Cluster および ACSLS HA に固有の構成要件が必要となります。

1. */etc/hosts* を構成する。
2. *root* ユーザーのアクセス権限を構成する。
3. マルチパスインターネットアクセス用のネットワークインタフェースを構成する。
4. マルチパスディスク I/O を構成する。

/etc/hosts の構成

各ノードの */etc/hosts* ファイルには、ローカルホスト、2 つのクラスタノード名とそれらの IP アドレス、および論理ホストのエントリが含まれています。

```
127.0.0.1           localhost   loghost
129.99.99.101       thisNode.domain.com   thisNode
129.99.99.102       sisterNode
129.99.99.100       logicalHost.domain.com    logicalHost
```

root のアクセス構成

Solaris Cluster のモニタリングおよびフェイルオーバー操作は、どちらか一方のノードからクラスタ全体を制御する必要がある *root* ユーザーによって処理されます。*root* ユーザーを信頼できるユーザーとして確立して、セキュアシェル (*ssh*) 認証を使用したノード間のプライベートネットワークアクセスを提供します。

1. 外部システムから各ノードへの直接 root アクセスを有効にします。*/etc/user_attr* ファイルを編集して、*root* のロールが指定された行をコメントアウト (または削除) します。

   ```
   # root::::type=role
   ```

2. システムコンソール以外のポイントからのシステムへのログインを *root* に許可します。

   /etc/default/login ファイルを編集して、コンソールのみのアクセスが指定された行をコメントアウトします。

```
# CONSOLE=/dev/console
```

3. セキュアシェルを使用したログインアクセスを *root* に許可します。

   /etc/ssh/sshd_config ファイルを編集して、*PermitRootLogin* に「**yes**」を指定します。

   ```
   PermitRootLogin=yes
   ```

   隣接するノード上で手順 1 から 3 を繰り返します。

4. 信頼できるユーザーとして *root* を確立します。これにより、パスワードを必要とせずに認証が確立される姉妹ノードから各ノードへのログインプロトコルが root に設定されます。

   a. RSA 公開鍵と非公開鍵のペアを作成します。パスワードなしでのノードからノードへのログインを許可するには、パスフレーズを入力しないでください。

   ```
   # cd /root/.ssh
   # ssh-keygen -t rsa
   Enter file in which to save the key (//.ssh/id_rsa): ./id_rsa
   Enter passphrase (empty for no passphrase):
   Enter same passphrase again:
   Your identification has been saved in ./id_rsa.
   Your public key has been saved in ./id_rsa.pub.
   The key fingerprint is:
   1a:1b:1c:1d:1e:1f:2a:2b:2c:2d:2e:2f:ea:3b:3c:3d root@node1
   ```

   これにより、*/root/.ssh* ディレクトリに *id_rsa* と *id_rsa.pub* の 2 つのファイルが作成されます。

   b. *id_rsa.pub* を姉妹ノード上の */root/.ssh* ディレクトリにコピーします。

   ```
   # cat id_rsa.pub | ssh root@node2 /
     'cat >> /root/.ssh/authorized_keys'
   Password:
   ```

   c. 認証鍵が適切な場所に配置された状態で、リモートからパスワードなしでコマンドを発行できることをテストします。

   ```
   # hostname
   ```

```
node1
# ssh root@node2 hostname
node2
```

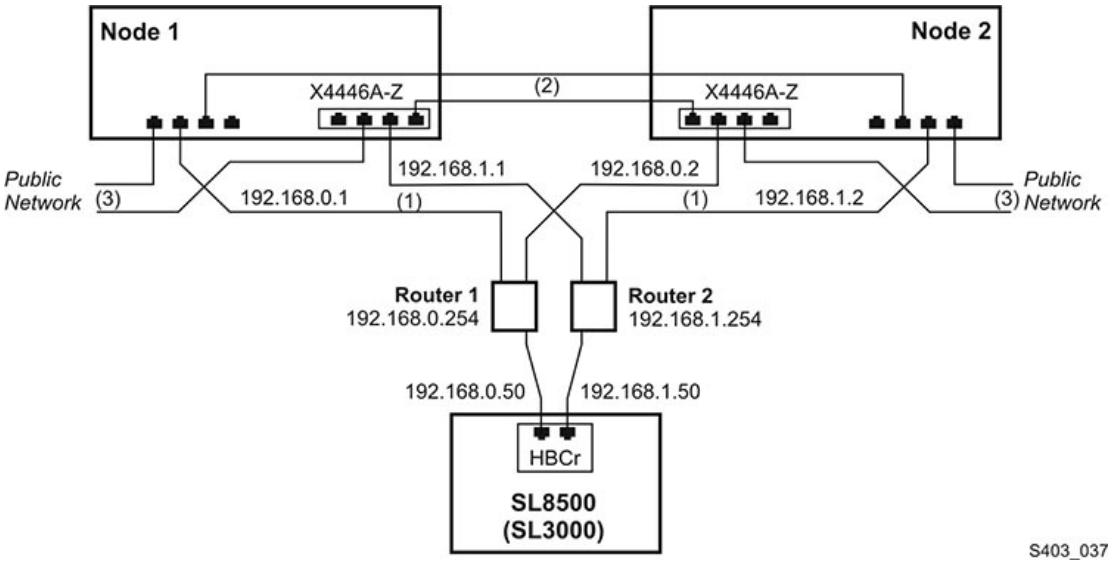
マルチパスネットワークの構成

冗長性は、高可用性コンピューティング全体のスキームです。冗長性はサーバーだけでなく、各サーバー上の各通信インタフェースにも適用されます。パブリックインタフェースの場合は、Internet Protocol Multi Pathing (IPMP) を Solaris 上での使用になります。Internet Protocol Multi Pathing は、一般的なシステムフェイルオーバーを必要とすることなく、失敗したネットワーク通信のための即時 NIC 回復を提供します。ライブラリインタフェースの場合は、2 つの独立したルート間で、2 つのネットワークインタフェースを使用したデュアル TCP/IP 接続を使用することを意味します。1 つのルートのいずれかの要素に障害が発生した場合、ACSLS は代替インタフェース経由で通信を続行します。

次の場合は、ACSLS HA に冗長ネットワーク接続が必要となります。

- パブリックなクライアント通信
- ライブラリ間の通信
- プライベートなノードクラスタ内通信

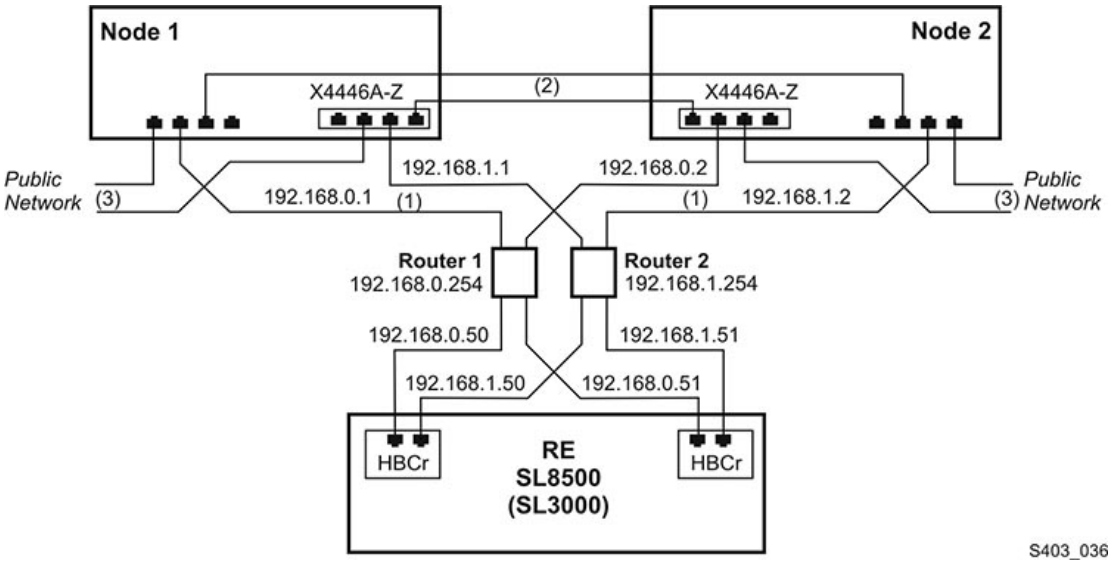
図2.1 各サーバーノード上の 2 つの Ethernet ポートに接続されている単一の HBCr ライブラリインタフェースカード



このセクションの図には、サーバーごとに別々の 2 つのコントローラを使用してアクセスできる 8 つの Ethernet ポートが示されています。6 つのポートを使用して、3 つの冗長接続を提供しています。この構成では、2 つのポートが使用されていません。複雑に見えますが、各サーバーから次の 3 つのデュアルパス Ethernet 接続があるだけです。

- サーバーとライブラリ間の通信
- プライベートネットワーク経由のサーバー間のハートビート交換
- パブリックネットワーク経由のサーバーとクライアント間の通信

図2.2 冗長電子装置を備えたライブラリ上のデュアル HBC 構成



冗長電子装置を備えたライブラリには、各サーバーノードから各 HBCr ライブラリコントローラへの独立したパスが 2 つあります。1 つの HBCr インタフェース上で両方のポートへの通信に障害が発生した場合は、ACSLS HA によって代替 HBCr カードに自動的に切り替えられます。この動作はすべて、代替サーバーノードへのフェイルオーバーを必要とせずに実現されます。

パブリックインタフェースと IPMP

Solaris IPMP では、NIC、ケーブル、スイッチなどのネットワークハードウェアを障害から保護するための冗長ネットワークインタフェースを構築するメカニズムが提供されています。Solaris ホスト上に IPMP を構成するときに、2 つ以上の物理ネットワークインタフェースを単一の IPMP グループに組み合わせます。

ネットワークインタフェース名と物理デバイスとのマッピングを表示するには、*`dladm show-phys`* コマンドを使用します。

例:

```
# dladm show-phys
LINK       MEDIA         STATE      SPEED  DUPLEX    DEVICE
net2       Ethernet      up         100    full      ixgbe1
net3       Ethernet      up         10000  full      ixgbe3
net0       Ethernet      up         10000  full      ixgbe2
net1       Ethernet      up         1000   full      ixgbe0
```

構成されているネットワークインタフェースの状態を表示するには、*ipadm* を使用します。

例:

```
# ipadm
NAME          CLASS/TYPE STATE  UNDER  ADDR
lo0           loopback   ok     --     --
   lo0/v4     static     ok     --     127.0.0.1/8
   lo0/v6     static     ok     --     ::1/128
net1          ip         ok     --     --
   net1/v4    static     ok     --     129.99.99.99/24
   net1/v6    addrconf   ok     --     fe99::999:999:ff23:ee02/10
net4          ip         ok     --     --
   net4/v4    static     ok     --     129.999.99.99/24
```

2 つのインタフェースが構成され、ACSLS HA の共通グループ ID に割り当てられている必要があります。これらのインタフェースの中には、サーバーのプライマリインタフェースが含まれている可能性があります。この場合、すでに *ip* アドレスが (グループではなく) インタフェースに割り当てられているため、このインタフェースの構成を解除してから、*ipmp* グループで再構成する必要があります。

この操作中はネットワーク通信が中断されるため、サーバーコンソールから次の手順を実行する必要があります。

既存のプライマリインタフェースの構成を解除するには:

ipadm delete-addr <primary interface>

例:

```
# ipadm delete-addr net0/v4
```

ipadm delete-ip <primary interface>

例:

```
# ipadm delete-ip net0
```

プライマリインタフェースを再構成するには:

ipadm create-ip <primary interface>

例:

```
# ipadm create-ip net0
```

2 つ目のプライマリインタフェースを作成するには:

```
ipadm create-ip <primary_interface>
```

例:

```
# ipadm create-ip net5
```

ipmp グループを作成するには:

ipadm create-ipmp <group_name>

例:

```
# ipadm create-ipmp ipmp0
```

注記:

グループ名は英数字である必要があります。

グループにホストの *ip-address* を割り当てるには:

ipadm create-addr -T static -a <ip-address> <group_name>

例:

```
# ipadm create-addr -T static -a 129.99.99.9  ipmp0
```

グループにプライマリインタフェースを追加するには:

ipadm add-ipmp -i <primary_interface> <group_name>

例:

```
# ipadm add-ipmp -i net0 ipmp0
```

グループに 2 つ目のインタフェースを追加するには:

ipadm add-ipmp -i <second_primary_interface> <group_name>

例:

```
# ipadm add-ipmp -i net5 ipmp0
```

ipmp を使用して *ipmp* 構成を確認します。

```
# ipadm
NAME              CLASS/TYPE STATE        UNDER      ADDR
ipmp0             ipmp       ok           --         --
   ipmp0/v4       static     ok           --         123.45.67.89/8
lo0               loopback   ok           --         --
   lo0/v4         static     ok           --         127.0.0.1/8
   lo0/v6         static     ok           --         ::1/128
net0              ip         ok           ipmp0      --
net5              ip         ok           ipmp0      --
```

ipmp0 グループに 2 つのネットワークインタフェースが構成されたことを確認します。ipmp0 グループにバージョン 4 の IP アドレスが割り当てられたことを確認します。

これらの変更を確定し、新しい構成でネットワーク通信を確立するために、システムのリブートが必要な場合があります。

姉妹ノード上でネットワークの構成を繰り返します。

start_acslsha.sh を使用してクラスタを起動すると、*ipmp* グループとクラスタのパブリック IP アドレスとのマッピングが確立されます。9章「*ACSLS クラスタの操作* 」を参照してください。

ライブラリインタフェース

残りの 2 つのネットワークインタフェースは、ライブラリの構成時に必要となります。この例では、net1 と net6 が使用されます。クラスタ化されたサーバーとライブラリ間のルーターが単一障害点とならないように、これらの 2 つの接続は別々のサブネット経由でルーティングさ

れています (図2.1「各サーバーノード上の 2 つの Ethernet ポートに接続されている単一の HBCr ライブラリインタフェースカード」と図2.2「冗長電子装置を備えたライブラリ上のデュアル HBC 構成」を参照)。

1. 各ノード上で 2 つのネットワークインタフェースを作成します。

```
# ipadm create-ip net1
# ipadm create-ip net6
```

2. インタフェースごとに *ip-address* を割り当てます。

```
# ipadm create-addr -T static -a <ip-address> net1/v4
# ipadm create-addr -T static -a <ip-address> net6/v4
```

通常、アドレスオブジェクトには、インタフェースとプロトコルバージョンを使用した名前が付けられます (例: *net1/v4*)。

3. *ipmp* の構成を確認します。

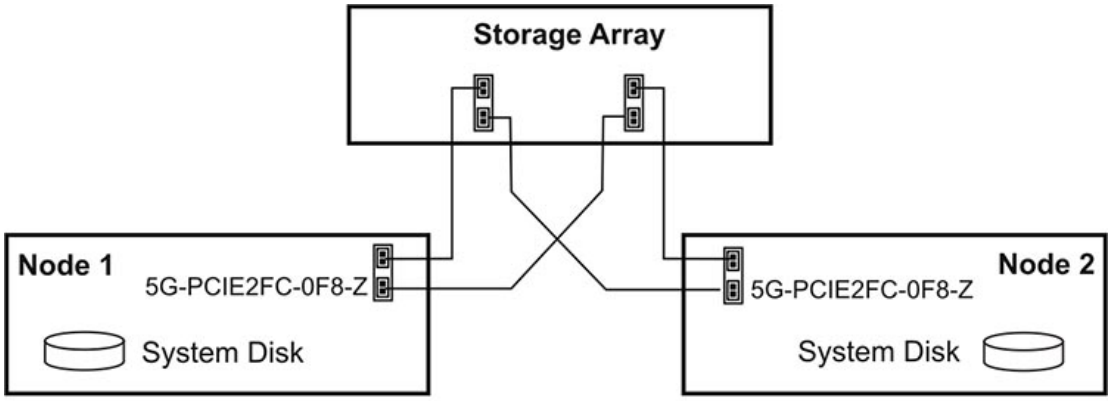
```
# ipadm
NAME              CLASS/TYPE STATE        UNDER      ADDR
ipmp0             ipmp       ok           --         --
   ipmp0/v4       static     ok           --         123.45.67.89/8
lo0               loopback   ok           --         --
   lo0/v4         static     ok           --         127.0.0.1/8
   lo0/v6         static     ok           --         ::1/128
net0              ip         ok           ipmp0      --
net1              ip         ok           --         --
   net1/v4        static     ok           --         192.168.0.1/8
net5              ip         ok           ipmp0      --
net6              ip         ok           --         --
   net6/v4        ip         ok           --         192.168.1.1/8
```

4. これらの設定を有効にするために、各ノードをリブートします。

マルチパスディスクの構成

外部の共有ディスクアレイは、2 台のホスト (それぞれ、サーバーとディスクアレイ間で冗長ファイバ接続または SAS 接続を使用) に接続します。

図2.3 外部の共有ストレージアレイに対するサーバー当たり 2 つのファイバ接続



S403_038

アレイは、接続されているホストに 2 つの仮想ドライブが表示されるように設定されています。

ディスクデバイスへのパスが複数あることが検出されると、Solaris 11.2 によって自動的にマルチパス (MPXIO) が設定されます。Solaris システムに、アレイ内の各仮想ディスクへの冗長接続が適切に構成されていることを確認します。

```
# mpathadm list lu

/dev/rdsk/c0t600A0B800049EE1A0000840552D3E2F9d0s
       Total Path Count: 2
       Operational Path Count: 2
/dev/rdsk/c0t600A0B800049EDD600000DAD52D3DA90d0s2
       Total Path Count: 2
       Operational Path Count: 2
```

mpathadm の出力には、2 つのディスクデバイス (動作可能なパス数は 2 つずつ) が表示されます。

SPARC システムで ACSLS HA を実行している場合、MPXIO が構成されるまで *mpathadm* には何も表示されません。何も表示されないか、適切でない構成が表示される場合は、この時点で Solaris 11.2 システム上で *stmsboot -e* を使用して、MPXIO を構成する手順に進みます。Oracle Technology Network ライブラリで『*Oracle Solaris の管理: SAN 構成およびマルチパス化*』ドキュメント (http://docs.oracle.com/cd/E26924_01/html/E26295) を参照してください。

Solaris I/O マルチパス対応デバイスは、それらが Solaris I/O マルチパスの制御下にあることを示す新しい名前を受け取ることに留意してください。デバイスには、元の名前とは異なる名前が付けられます。

```
Original device name:    c1t0d0
Name After stmds boot:   c0t600A0B800049EDD600000C9952CAA03Ed0
```

3

第3章 ZFS を使用したファイルシステムの構成

Solaris 11.2 は、ZFS ファイルシステムに基づいています。ディスクの I/O、ディスクのパーティション分割、ディスクのミラー化 (または RAID) は、完全に ZFS で処理されます。したがって、ディスクをパーティション分割する必要はありません (一般に、UFS ファイルシステムを使用した場合も同様です)。システムディスク全体が単一のパーティションとして表示されます。

ストレージアレイにはすでに RAID が構成されているため、ZFS を使用して追加の RAID レベルを構成することは、ACSLS ファイルシステムに不可欠ではありません。単純な JBOD ディスクを使用している場合は ZFS RAID が不可欠ですが、正規のディスクアレイを採用していれば、追加の RAID はオプションです。次の例では、どちらかのアプローチについて説明します。

ミラー化されたルートファイルシステムの作成

1. Solaris プラットフォームは、2 つの物理ディスクドライブで構成されます。ZFS のパフォーマンスを最適化するには、システムディスクとそのミラードライブをパーティション分割します。

   オペレーティングシステムをインストールする前の新しいシステムでは、パーティション 0 にディスク領域全体の (全部ではないが) 大部分が含まれるように、各システムディスクドライブをパーティション分割できます。ZFS はディスク全体にアクセスできる場合に、より高速かつ確実に動作します。2 つ目のディスク上の ZFS について定義されたパーティションのサイズが、プライマリディスク上で定義されたサイズと同じであることを確認します。

   Solaris 11.2 がすでにインストールされているシステムでは、プライマリシステムディスク上で *format* または *fdisk* を使用して、*root* パーティションのサイズを表示します。パーティションサイズが等しい 2 つ目のシステムディスクをフォーマットします。フォーマットが完了したら、ディスクにラベルを付けます。
2. システムが起動したら、*zpool status* コマンドを使用して、*rpool* を確認します。

   ```
   # zpool status
   ```

```
  pool: rpool
  state: ONLINE
  scan: none requested
  config:
     NAME                       STATE     READ WRITE CKSUM
     rpool                      ONLINE       0     0     0
      c0t5000C5000EA48903d0s0  ONLINE       0     0     0
```

3. 2 つ目のシステムディスクを特定し、そのデバイス ID を確認します。

```
# echo | format
AVAILABLE DISK SELECTIONS:
  0. c0t5000C5000EA48893d0 <SUN146G cyl 14087 alt 2 hd 24 sec 848>
    /scsi_vhci/disk@g5000c5000ea48893
  1. c0t5000C5000EA48903d0 <SUN146G cyl 14087 alt 2 hd 24 sec 848>
    /scsi_vhci/disk@g5000c5000ea48903
```

手順 2 で表示されたデバイスと同じサイズに近いサイズを持つ代替デバイスを選択します。この例では、2 つ目のディスク ID は c0t5000C5000EA48893d0s です

4. *rpool* に 2 つ目のディスクを追加します。

```
# zpool attach -f rpool /
       c0t5000C5000EA48903d0 /
       c0t5000C5000EA48893d0
```

ミラー化されたドライブの再同期化が開始され、ブートドライブの内容が 2 つ目のドライブにコピーされます。この動作には数分間かかるため、リブートによって中断されないようにしてください。

次のコマンドを使用すると、進行状況をモニターできます。

```
zpool status -v
```

注記 1: 再同期化が完了するまで、ステータスには、ディスクが縮退モードになっていることが表示されます。情報がプライマリディスクからミラーディスクにコピーされている間は、ディスクが縮退状態のままになります。

注記 2: ディスクに EFI ディスクのラベルが付けられているために *zpool attach* に失敗する場合は、『*Oracle Solaris の管理: デバイスとファイルシステム*』ドキュメント (http://docs.oracle.com/cd/E23824_01/pdf/821-1459.pdf) で説明されている処理に従ってください。EFI ディスクを SMI に変換するプロセスは次のとおりです。

```
# format -e
  (select the drive to serve as the rpool mirror).
  format> partition
  partition> print
  partition> label
    (specify label type "0")
    Ready to label? y
  partition> modify
    (select "1" All free Hog)
    Do you wish to continue ... yes
    Free Hog Partition[6]? (specify partition "0")
    (Specify a size of "0" to the remaining partitions)
    Okay to make this current partition table? yes
    Enter table name: "c1t1d0"
    Ready to label disk? y
 partition> quit
 format> quit
```

5. ミラー化された *rpool* の構成を確認します。

```
# zpool status
  pool: rpool
 state: ONLINE
  scan: resilvered 6.89G in 0h3m with 0 errors
config:
        NAME                        STATE     READ WRITE CKSUM
        rpool                       ONLINE       0     0     0
          mirror-0                  ONLINE       0     0     0
            c0t5000C5000EA48903d0   ONLINE       0     0     0
            c0t5000C5000EA48893d0   ONLINE       0     0     0
```

隣接するノード上で、この操作を繰り返します。

ACSLS アプリケーション用のミラー化されたファイルシステムの作成

ACSLS ファイルシステムは、外部の共有ストレージアレイ上の *zpool* に存在します。次の例では、2 つのディスクのみを使用した単純なミラー化アレイ (RAID 1) が採用されています。これらは実際のドライブである可能性もありますが、接続されているストレージアレイとは別個のドライブとして表示されている仮想デバイスである可能性が高いです。

ストレージアレイにはすでに RAID が構成されているため、ZFS を使用して追加の RAID レベルを構成することは、ACSLS ファイルシステムに不可欠ではありません。単純な JBOD ディスクを使用している場合は ZFS RAID が不可欠ですが、正規のディスクアレイを採用していれば、追加の RAID はオプションです。次の例では、どちらかのアプローチについて説明します。

1. 共有ストレージアレイを準備します。

   標準の構成では、ディスクアレイから 1 つの仮想ドライブを使用します。それ以外の場合、ZFS RAID ミラー化構成では、サイズが等しい 2 つの仮想ドライブが使用されます。ディスクアレイに備わっている管理ツールまたは Solaris のフォーマットユーティリティーを使用すると、サイズが等しくなるように 2 つの仮想ドライブをパーティション分割できます。
2. ACSLS をインストールするための目的のベースディレクトリを決定します。

   ACSLS 8.4 は、どのファイルシステムにもインストールできます。選択された基本ファイルシステムは、*rpool* システムにすでに存在するものであってはいけません。すでに存在する場合、既存のファイルシステムは、新しい *zpool* の下にそれを作成する前に破棄されます。

   デフォルトの */export/home* ベースディレクトリを ACSLS 用に使用する場合、Solaris 11.2 のデフォルトの *root* プールから */export* ファイルシステムを破棄する必要があります。

   /export/home が *rpool* に接続されているかどうかを確認するには、次のコマンドを実行します。

   ```
   # zfs list
   ```

rpool から */export/home* を切り離すには、まず保持するべきファイルやディレクトリを保存します。*/export/home* に現在アクティブになっているユーザーのホームディレクトリが存在しないことを確認します。次に、*zfs destroy* を使用して、*/export* の下にあるすべてのものを削除します。

```
# zfs destroy -r rpool/export
```

隣接するノード上で、この手順を繰り返して *rpool/export* を切り離します。

3. *format* を使用して、接続されているディスクアレイ上のドライブのデバイス名を特定します。

```
# echo | format
AVAILABLE DISK SELECTIONS:
   0. c0t5000C5000EA48893d0 <FUJITSU-MAY2073RCSUN72G-0501-68.37GB>
   /pci@0,0/pci1022,7450@2/pci1000,3060@3/sd@0,0
   /dev/chassis/SYS/HD0/disk
1. c0t5000C5000EA48893d0 <FUJITSU-MAY2073RCSUN72G-0501-68.37GB>
   /pci@0,0/pci1022,7450@2/pci1000,3060@3/sd@1,0
   /dev/chassis/SYS/HD1/disk
3. c0t600A0B800049EDD600000C9952CAA03Ed0 <SUN-LCSM100_F-50.00GB>
   /scsi_vhci/disk@g600a0b800049edd600000c9952caa03e
4. c0t600A0B800049EE1A0000832652CAA899d0 <SUN-LCSM100_F-50.00GB>
   /scsi_vhci/disk@g600a0b800049ee1a0000832652caa899
```

この例では、2 つのシステムディスクと、*c0t600A...* で始まるデバイス名を持つディスクアレイから表示されている 2 つの仮想ディスクがあります。

4. *acslspool* を作成します。

正規のディスクアレイを使用した標準の構成では、次のように *acslspool* を作成します。

```
# zpool create -m /export/home acslspool/
  /dev/dsk/c0t600A0B800049EDD600000C9952CAA03Ed0
```

手順 1 で提案されたように ZFS RAID が追加された場合、次のようにミラー化構成を作成します。

```
# zpool create -m /export/home acslspool mirror /
   /dev/dsk/c0t600A0B800049EDD600000C9952CAA03Ed0 /
   /dev/dsk/c0t600A0B800049EE1A0000832652CAA899d0
```

5. 新しい *acslspool* を検証します。

```
# zpool status acslspool
  pool: acslspool
 state: ONLINE
  scan: none requested
config:
  NAME                                        STATE   READ WRITE CKSUM
  acslspool                                   ONLINE   0     0     0
    mirror-0                                  ONLINE   0     0     0
      c0t600A0B800049EDD600000C9952CAA03Ed0   ONLINE   0     0     0
      c0t600A0B800049EE1A0000832652CAA899d0   ONLINE   0     0     0
```

注記:

RAID ディスクアレイを使用している場合、ミラー化された ZFS 構成はオプションです。

6. 新しいプールにテストファイルを作成し、確認します。

```
# cd /export/home
# date > test
# ls
test
# cat test
Tue Jan  7 11:48:05 MST 2015
```

7. プールをエクスポートします。

```
# cd /
# zpool export acslspool
```

8. (新しい現在のノードとして参照される) 隣接するノードにログインします。
9. 新しい現在のノードから、*/export/home* (または ACSLS 用のファイルシステム) が *root* プールのどこにもマウントされていないことを確認します。

```
# zfs list
```

すでにファイルシステムが *rpool* に存在する場合は、この現在のノード上で (前述の) 手順 2 を繰り返します。

10. 新しい現在のノードから *acslspool* をインポートし、このノード上に *acslspool* が存在することを確認します。

```
# zpool import acslspool
# zpool status
  pool: acslspool
 state: ONLINE
  scan: none requested
config:
NAME                                        STATE   READ WRITE CKSUM
acslspool                                   ONLINE   0     0     0
  mirror-0                                  ONLINE   0     0     0
    c0t600A0B800049EDD600000C9952CAA03Ed0   ONLINE   0     0     0
    c0t600A0B800049EE1A0000832652CAA899d0   ONLINE   0     0     0
```

zpool import に失敗した場合は、*zpool import -f* を使用して操作を試すことができます。

注記:

RAID ディスクアレイを使用している場合、ミラー化された ZFS 構成はオプションです。

---

11. 新しい現在のノード上にテストファイルが存在することを確認します。

```
# cd /export/home
# ls
test
# cat test
Tue Jan  7 11:48:05 MST 2015
```

4

第4章 ソフトウェアパッケージのダウンロード

この章では、ACSLS HA に必要な主なソフトウェアパッケージとパッチ、および製品ドキュメントのダウンロード手順について説明します。

ソフトウェアパッケージのダウンロード

ACSLS HA には 3 つの主なソフトウェアパッケージが必要です。

- ACSLS 8.4
- Oracle Cluster 4.2
- ACSLS HA 8.4

ソフトウェアパッケージを各サーバーノードにダウンロードする必要があります。パッケージを */opt* ディレクトリに入れることをお勧めします。

パッケージは Oracle Software Delivery Cloud から入手できます。

https://edelivery.oracle.com/

ACSLS 8.4 のダウンロード

1. システムで Web ブラウザを起動し、次の URL の Oracle Software Delivery Cloud Web サイトにナビゲートします。

   https://edelivery.oracle.com
2. Oracle サポート担当者から提供されたユーザー名およびパスワードでサインインします。
3. 輸出規制を読み、これに同意します。
4. 「製品のフィルタ基準」の「プログラム」ボックスにチェックマークを付けます。
5. 「製品」に対して「**acsls**」と入力し、「StorageTek Automated Cartridge System Library Software (ACSLS)」を選択します。
6. 「**プラットフォームの選択**」をクリックし、使用している Solaris プラットフォーム (Solaris または X86) を選択します。「**選択**」をクリックします。

7. 「選択された製品」画面で、「**続行**」をクリックします。
8. 「使用可能なリリース」の下で、使用している Solaris プラットフォームの ACSLS 8.4.0.0.0 リリースについてのボックスをクリックして、「**続行**」をクリックします。
9. 「オラクル標準の条件および規制」画面で、ライセンスの条件を確認して受諾します。「**続行**」をクリックします。
10. 「*V77685-xx*」パッケージを選択し、好みの場所に zip ファイルを保存します。

手順については、5章「*ACSLS 8.4 のインストール*」を参照してください。

PostgreSQL 8.4 のダウンロード (オプション)

ACSLS 8.4 をダウンロードするとき、PostgreSQL 8.3 用のパッケージが組み込まれます。これらは適切に動作し、ACSLS 8.4 がインストールされるときに自動的にインストールされます。ただし、PostgreSQL 8.4 をインストールする場合、*postgres 8.4 bz2* ファイルを `http://www.postgresql.org/` から各ノードの */opt* ディレクトリにダウンロードします。ACSLS がインストールされている場合、これは *.bz* ファイルを */opt* から検出し、PostgreSQL 8.4 を自動的にインストールします。詳細な説明については、ACSLS 8.4 パッケージに含まれている README.txt ファイルの「INSTALLING POSTGRESQL」のセクションを参照してください。

Oracle Cluster 4.2 のダウンロード

Oracle Cluster 4.2 をダウンロードするときに必要な 2 つの手順は次のとおりです。

- Solaris Cluster の基本イメージをダウンロードする。
- Solaris Cluster のパッチ更新が必要かどうかを判定する。

手順については、6章「*Solaris Cluster 4.2 のインストール*」を参照してください。

Solaris Cluster の基本イメージをダウンロードする

1. システムで Web ブラウザを起動し、次の URL の Oracle Software Delivery Cloud Web サイトにナビゲートします。

   `https://edelivery.oracle.com`
2. Oracle サポート担当者から提供されたユーザー名およびパスワードでサインインします。
3. 輸出規制を読み、これに同意します。
4. 「製品のフィルタ基準」の「プログラム」ボックスにチェックマークを付けます。
5. 「**Oracle Solaris Cluster**」と入力して、「**Enterprise Edition**」を選択します。
6. 「**プラットフォームの選択**」(SPARC または x86) をクリックして、「**選択**」をクリックします。

7. 「選択された製品」画面で、「**続行**」をクリックします。
8. 「使用可能なリリース」で、「**代替リリース**」を選択し、使用する Solaris プラットフォームの「**4.2.0.0.0 Enterprise Edition**」を選択します。「**続行**」をクリックします。
9. 著作権のライセンスを読んでこれに同意し、「**続行**」をクリックします。
10. 「*V46190-xx*」パッケージを選択し、好みの場所への zip ファイルを保存します。

Solaris のパッチ更新が必要かどうかを判定する

現在インストールされている Solaris バージョンを確認します。

```
# pkg info entire | grep Version
```

- Solaris バージョンが 11.2.12 以下の場合は、Oracle Solaris Cluster のパッチ更新は必要ありません。
- Solaris バージョンが 11.2.13 以上である場合、Cluster のパッチ更新が必要になります。
  1. Oracle Support Web サイトに移動します。

     http://support.oracle.com
  2. 「**サインイン**」をクリックして、Oracle サポート担当者から提供されたユーザー名とパスワードを入力します。
  3. 「**パッチと更新版**」をクリックします。
  4. 「パッチ検索」で、「**製品またはファミリ (拡張)**」をクリックします。
  5. 「製品」ダイアログボックスで、「**Solaris Cluster**」と入力します。
  6. 「リリース」ボックスで、下向き矢印をクリックして「**Solaris Cluster 4.2.0**」をチェックします。「**検索**」をクリックします。
  7. プラットフォーム (SPARC または X86) に一致する ORACLE SOLARIS CLUSTER 4.2.5.*x.x* REPO ISO イメージを探してそのパッチを選択します。「**ダウンロード**」をクリックします。
  8. 「パッチ検索」で情報を確認して「**ダウンロード**」をクリックします。
  9. パッチパッケージを選択し、好みの場所に zip ファイルを保存します。

ACSLS HA 8.4 のダウンロード

1. システムで Web ブラウザを起動し、次の URL の Oracle Software Delivery Cloud Web サイトにナビゲートします。

   https://edelivery.oracle.com

2. Oracle サポート担当者から提供されたユーザー名およびパスワードでサインインします。
3. 輸出規制を読み、これに同意します。
4. 「製品のフィルタ基準」で「**プログラム**」をチェックします。
5. 「**acsls**」と入力し、「StorageTek Automated Cartridge System Library Software (ACSLS) High-Availability Agent (HA)」を選択します。
6. プラットフォーム (SPARC または x86) を選択して、「**選択**」をクリックします。
7. 「選択された製品」画面で、「**続行**」をクリックします。
8. プラットフォームに対して ACSLS HA を検証し、「**続行**」をクリックします。
9. 著作権のライセンスを読んでこれに同意し、「**続行**」をクリックします。
10. zip ファイルを選択して「**ダウンロード**」をクリックします。
11. 著作権のライセンスを読んでこれに同意し、「**続行**」をクリックします。
12. 「*V75269-xx*」パッケージを選択し、好みの場所に zip ファイルを保存します。

手順については、7章「*ACSLS HA 8.4 のインストールと起動* 」を参照してください。

パッチのダウンロード

ACSLS 8.4、Solaris Cluster 4.2、および ACSLS HA 8.4 についてのパッチが入手できる場合があります。Oracle Support サイトでパッチの更新を確認します。

`https://support.oracle.com`

1. Oracle ID とパスワードでサインインします。
2. 「**パッチと更新版**」タブを選択します。
3. 「**検索**」タブで、「**製品またはファミリ (拡張)**」をクリックします。
4. 前述のソフトウェアパッケージについての手順に従います。

製品ドキュメントのダウンロード

関連する製品ドキュメントをダウンロードするには、次の手順を実行します。

`http://docs.oracle.com`

ACSLS の場合:

1. 「Storage」を見つけて「Storage Software」を選択します。
2. 「StorageTek ACSLS Manager documentation」、「Automated Cartridge System Library Software 8.4」の順に選択します。

3. 「View Library」を選択します。

Solaris Cluster の場合:

1. 「Operating Systems」を見つけて「Operating Systems」を選択します。
2. 「Oracle Solaris Cluster」から「Oracle Solaris Cluster 4.2」を選択します。
3. Cluster 4.2 についての必要な言語およびドキュメントを選択します。

5

第5章 ACSLS 8.4 のインストール

ACSLS 8.4 のインストールの詳細は、『*StorageTek Automated Cartridge System Library Software 8.4 インストールガイド*』に記載されています。ACSLS 8.4 をインストールするには、この大まかな手順に従ってください。

最初のノードへのインストール

1. ACSLS 8.4 を各サーバーの */opt* ディレクトリにダウンロードします。
2. ダウンロードした zip ファイルを unzip します。
3. *acslspool* が現在のノードにマウントされていることを確認します

```
# zfs list
NAME                      USED  AVAIL  REFER  MOUNTPOINT
acslspool                1.60G  47.4G  1.60G  /export/home
rpool                    6.97G  60.0G  4.58M  /rpool
rpool/ROOT               4.39G  60.0G    31K  legacy
rpool/ROOT/solaris       4.39G  60.0G  3.17G  /
rpool/ROOT/solaris/var   1.22G  60.0G  1.21G  /var
rpool/VARSHARE           95.5K  60.0G  95.5K  /var/share
rpool/dump               1.55G  60.0G  1.50G  -
rpool/swap               1.03G  60.0G  1.00G  -
node2:# clrg resume acsls-rg
```

4. ACSLS インストールディレクトリに移動して、パッケージのインストールスクリプトを実行します。

```
# cd /opt/ACSLS_8.4.0
# ./pkg_install.sh
```

5. このノードの */etc/passwd* を調べます。ユーザー *acsss*、*acssa*、および *acsdb* に割り当てられたユーザー ID 番号とグループ ID 番号を書き留めます。

```
# tail -3 /etc/passwd
# grep acsls /etc/group
```

隣接するノードにインストールするときは、2 番目のノードで割り当てられたユーザー ID 番号が、ここに一覧表示されている対応する ID 番号と一致することを確認します。

6. ACSLS 環境を入手して、パッケージのインストールスクリプトを実行します。

```
# . /var/tmp/acsls/.acsls_env
# cd $ACS_HOME/install
# ./install.sh
```

> **注記:**
>
> インストールスクリプトがデータベースのバックアップディレクトリを求めるプロンプトを表示したら、共有ディスクアレイにマウントされているディレクトリを指定してください。ACSLS インストールディレクトリ (*$installDir*) の下にあるパスを使用します。たとえば、*/export/home* に ACSLS をインストールする場合、データベースのバックアップファイルには */export/home/backup* を使用します。

7. 接続ライブラリが接続されていることを確認します。

```
# su - acsss
$ testlmutcp <library ip address>
```

8. ライブラリ構成ルーチンを実行します。

```
$ acsss_config
```

9. ACSLS 8.4 に対するパッチ更新がないか確認します。パッチが存在する場合は、指示に従ってダウンロードおよびインストールします。
10. 現在のノードから *acslspool* をエクスポートします。

```
$ exit
# cd /
# zpool export acslspool
```

ユーザーまたは操作が現在 ACSLS ファイルシステムでアクティブな場合、この操作は失敗します。

隣接するノードへのインストール

STKacsls パッケージ内のほとんどのファイルは共有ディスクアレイ (これらのファイルがすでにインストールされている場所) に解凍されますが、ACSLS ユーザーは各ノードに追加され、多数のシステムファイル (SMF の起動スクリプトと停止スクリプト、cron ジョブ) を各ノードにインストールする必要があるため、隣接するノードに引き続き ACSLS 8.4 をインストールする必要があります。

1. 隣接するノードにログインし、acslspool をインポートします。

   ```
   # zpool import acslspool
   ```

   この操作が失敗する場合、*zpool import -f acslspool* を試行してください。
2. ACSLS パッケージのインストールディレクトリに移動して、パッケージをインストールします。

   ```
   # cd /opt/ACSLS_8.4.0
   # ./pkg_install.sh
   ```

   一部の ACSLS ファイルは (共有ドライブ上の) インストールディレクトリにすでに存在することがインストールルーチンによって示されます。それらを上書きすることを選択できます。
3. ユーザー *acsss*、*acssa*、および *acsdb* のためにこのノードで割り当てられたユーザー ID とグループ ID を調べます。

   ```
   # tail -3 /etc/passwd
   # grep acsls /etc/group
   ```

   これらの番号が最初のノードで割り当てられた番号と一致することを確認します。デフォルトでは、GID は 100 ですが、GID 100 がすでに使用中の場合はこの番号は変わります。

   UID および GID の番号が 2 つのノード間で一致することが重要です。これらの番号が 2 番目のノードで異なる場合、2 番目のノードの UID および GID の番号が最初のノードの番号と一致するように、*/etc/passwd* ファイルを編集します。

次に、*$ACS_HOME* 内のファイルのグループ ID が *acsls* のグループ所有権を持っていることを確認します。ない場合は、*STKacsls* パッケージをアンインストールして再インストールする必要があります。

```
# pkgrm STKacsls
# pkgadd STKacsls
```

4. ACSLS 環境を継承して、インストールシェルスクリプトを実行します。

```
# . /var/tmp/acsls/.acsls_env
# cd $ACS_HOME/install
# ./install.sh
```

注記 1: インストールスクリプトがデータベースのバックアップディレクトリを求めるプロンプトを表示したら、共有ディスクアレイにマウントされているディレクトリを指定してください。ACSLS インストールディレクトリ (*$installDir*) の下にあるパスを使用します。たとえば、*/export/home* に ACSLS をインストールする場合、データベースのバックアップファイルには */export/home/backup* を使用します。

注記 2: 2 番目のノードで *install.sh* を実行するときは、ACSLS GUI を再インストールする必要はありません。GUI の *Acsls GUI* ドメインがすでに存在することがインストールスクリプトによって示されたときは、GUI を再インストールするためのプロンプトに対しては単に「**no**」と応答して、GUI を削除するためのプロンプトには必ず「**no**」と応答してください。

5. ACSLS 8.4 へのパッチが最初のノードに追加された場合は、このノード上でパッチのインストールを繰り返します。
6. 接続ライブラリが接続されていることを確認します。

```
# su - acsss
$ testlmutcp <library ip address>
```

7. ライブラリ構成ルーチンを実行します。

```
$ acsss_config
```

HA サーバーとライブラリとの間の冗長性を有効にするには、図2.1「各サーバーノード上の 2 つの Ethernet ポートに接続されている単一の HBCr ライブラリインタフェースカード」または図2.2「冗長電子装置を備えたライブラリ上のデュアル HBC 構成」を参照してください。*`acsss_config`* (オプション 8) の実行時には、必ず、各 ACS への 2 つの接続を定義して、各接続のライブラリ IP アドレスを入力してください。

6

第6章 Solaris Cluster 4.2 のインストール

Solaris Cluster のインストールについては、Oracle Technology Network のサイトから入手可能な『*Oracle Solaris Cluster ソフトウェアのインストールガイド*』で詳細に説明されています(このドキュメントの39 ページの4章「*ソフトウェアパッケージのダウンロード*」を参照)。

ACSLSHA 8.4 は、Oracle Solaris Cluster 4.2 を使用した Solaris 11.2 でサポートされています。

Cluster パッケージのインストール

Cluster ソフトウェアをインストールするには、この手順に従います。

1. */opt/OSC* ディレクトリを作成します。

```
# mkdir /opt/OSC
```

2. 「Oracle Cluster 4.2 のダウンロード」で決定した操作システムバージョンによっては、Solaris Cluster パッケージの 1 つまたは 2 つの ISO イメージがダウンロードされていることがあります。ダウンロードされた各 Cluster パッケージを */opt/OSC* ディレクトリに移動します。
3. パッケージを unzip します。unzip されたパッケージごとに ISO イメージを特定します。
4. 各 ISO イメージから擬似デバイスを作成します。

```
#  /usr/sbin/lofiadm -a /opt/OSC/V46190-01.iso

#  /usr/sbin/lofiadm -a /opt/OSC/osc-4_2_5_1_0-repo-incr.iso
```

   これらのそれぞれについて作成した */dev/lofi* インスタンス番号を追跡します。
5. */opt/OSC* ディレクトリで、擬似デバイスごとにマウントポイントを作成します。

```
# mkdir mnt
```

```
# mkdir mnt1
```

6. 擬似デバイスをこれらのマウントポイントにマウントします。

```
# mount –F hsfs –o ro /dev/lofi/1 /opt/OSC/mnt

# mount –F hsfs –o ro /dev/lofi/2 /opt/OSC/mnt1
```

7. 次のいずれかを選択します。
   - Solaris バージョンが 11.2.13 以上の場合は、手順 8 に進みます。
   - Solaris バージョンが 11.2.12 以下であって、基本 Cluster イメージのみをダウンロードしている場合、その基本イメージのリポジトリを公開できます。

     ```
     #  pkg set publisher –G  ’*’  –g  file:/opt/OSC/mnt/repo  ha-cluster
     ```

     このセクションの手順 13 に進んでパッケージをインストールします。
8. 次の手順では、OSC 基本パッケージを読み取り/書き込みファイルシステムにコピーし、パッチ更新を基本パッケージにマージします。

   OSC パッケージをマージするための読み取り/書き込みファイルシステムを作成します。

```
# cd /opt/OSC

# mkdir merged_iso
```

9. 基本 OSC イメージリポジトリを作成されたディレクトリにコピーします。

```
# cp –r  mnt/repo  merged_iso
```

10. マージされたディレクトリに 2 つのイメージを一緒に同期させます。

```
# rsync –aP mnt1/repo  merged_iso
```

11. リポジトリ用に検索インデックスを再構築します。

```
# pkgrepo rebuild –s  merged_iso/repo
```

12. マージされたパッケージの *ha-cluster* リポジトリを公開します。

```
# pkg set-publisher –g file:/opt/OSC/merged_iso/repo ha-cluster
```

13. Oracle Solaris Cluster パッケージをインストールします。

```
# pkg install --accept ha-cluster-full
```

隣接するノード上で手順 1 から 13 を繰り返します。

scinstall ルーチン

Solaris Cluster のインストールルーチンは、2 台のノード間で一連のチェックを実行して、両方のサーバーからシステム操作をモニターでき、起動およびフェイルオーバーアクションを制御できることを確認します。

準備の手順:

1. *scinstall* を実行する前に、インストールされたクラスタユーティリティーへのパスを含む *root* 用の環境を構築しておくと便利です。*/root/.profile* ファイルを編集します。*/usr/cluster/bin* が含まれるようにパス文を変更します。

```
export PATH=/usr/cluster/bin:/usr/bin:/usr/sbin
```

この変更は、必ず各ノード上で行なってください。新しいパスを継承させるには、ログアウトして再度ログインするか、または単に *su -* を実行します。

2. *rpc/bind* の *config/local_only* プロパティーが「*false*」に設定されていることを確認します。

```
# svccfg -s network/rpc/bind listprop config/local_only
```

このプロパティーで「*true*」が返される場合は、「*false*」に設定する必要があります。

```
# svccfg -s network/rpc/bind setprop config/local_only=false
```

もう一度、確認します。

```
# svccfg -s network/rpc/bind listprop config/local_only
```

3. Cluster ソフトウェアのための重要なハードウェア設定要件は、2 つのノード間でのクラスタ操作のための中断なしの通信を確保するために予約されている 2 つのプライベートネットワーク接続が存在することです。

   図2.1「各サーバーノード上の 2 つの Ethernet ポートに接続されている単一の HBCr ライブラリインタフェースカード」では、これらの物理的な接続が (2) というラベルで示されています。単一障害点で Cluster の内部通信が中断される可能性がないように、各接続は別々のネットワークアダプタから始まっています。*scinstall* ルーチンは、2 つの接続をそれぞれチェックして、ほかのネットワークトラフィックが流れていないことを確認します。最後に、*scinstall* は、2 つのライン間の通信が機能していることを確認します。物理的な接続が確認されたら、このルーチンは各インタフェースを 172.16 で始まる内部のプライベートアドレスに割り当てます。

   scinstall を実行する前に、このプライベート接続用に設定された各サーバー上で、2 つのネットワークポートに割り当てられているネットワークデバイス ID を確認するようにしてください。*dladm show-phys* を実行して、インタフェースの割り当てを表示します。

   ```
   # dladm show-phys
   ```

4. どちらか一方のノードからクラスタを表示するには、論理ホスト名と IP アドレスを確立する必要があります。この論理ホストは、アクティブなホストが node1 と node2 のどちらから実行されているのかに関係なく、確実にネットワーク通信に応答します。

   論理ホスト名と論理 IP アドレスが含まれるように、両方のノード上で */etc/hosts* ファイルを更新します。ACSLS HA (「ACSLS HA の構成」) を起動すると、このホストがアクティブになります。

5. Cluster を正常にインストールするには、Solaris Common Agent Container を有効にします。エージェントコンテナが有効になっていることを確認します。

   ```
   # cacaoadm status
   ```

   システム起動時に、ステータスの応答でエージェントコンテナが「DISABLED」であることが示された場合は、次のように有効にします。

   ```
   # cacaoadm enable
   ```

scinstall の実行

2 台のノードのどちらか一方で、*scinstall* コマンドを実行してから、この手順に従います。

1. メインメニューから「**Create a new cluster**」を選択します。
2. サブメニューから「**Create a new cluster**」を選択します。
3. 初期のデフォルト値を受け入れます。
4. 「**Typical**」インストールを選択します。
5. クラスタに名前 (*acsls_cluster* など) を割り当てます。
6. クラスタノードのプロンプトで、隣接するノードのホスト名を入力します。ノードリストが正しい場合は、それを受け入れます。
7. この目的のために特定された 2 台のプライベートノードの相互接続を定義します。TCP リンクを物理的な接続に割り当てることをインストールルーチンに許可します。
8. プロンプトに従って、クラスタを作成します。定足数デバイスとして機能する特定のデバイスを特定した場合を除いて、定足数デバイスを選択することを *scinstall* ルーチンに許可します。
9. クラスタチェックが*両方のノードで失敗した*ことがユーティリティーでレポートされても心配しないでください。マイナーな警告でも失敗はレポートされます。各ノードのレポートを確認し、重大なエラーや違反が返されていないか探してください。このルーチンでは、操作中に発生したエラーや警告に関する詳細がレポートされているログファイルへのパスが表示されます。ログファイルを確認し、特定された重大な問題または中程度の重大な問題を修正します。

   scinstall ルーチンは 1 つのノードから実行され、Solaris Cluster を両方のノード上にインストールします。ルーチンが 1 つのノードを構成し、そのノードをリブートし、次に 2 番目のノードを構成し、その 2 番目のノードをリブートすることを確認します。

クラスタ構成の確認

1. クラスタ内に両方のノードが含まれていることを確認します。

```
# clnode list -v
Node                Type
----                ----
node1               cluster
node2               cluster
```

2. Solaris Cluster で使用可能なデバイスのリストを表示します。

```
# cldevice list -v
DID Device   Full Device Path
d1           node1:/dev/rdsk/c0t600A0B800049EDD600000C9952CAA03Ed0
d1           node2:/dev/rdsk/c0t600A0B800049EDD600000C9952CAA03Ed0
d2           node1:/dev/rdsk/c0t600A0B800049EE1A0000832652CAA899d0
d2           node2:/dev/rdsk/c0t600A0B800049EE1A0000832652CAA899d0
d3           node1:/dev/rdsk/c1t0d0
d4           node1:/dev/rdsk/c1t1d0
d5           node2:/dev/rdsk/c1t0d0
d6           node2:/dev/rdsk/c1t1d0
```

この例では、共有ディスクデバイスは d1 および d2 です。一方、d3 および d4 は node1 のブートデバイス、d5 および d6 は node2 のブートデバイスです。d1 および d2 には、どちらか一方のノードからアクセスできます。

3. 定足数は、3 つ以上のデバイスで構成されます。これはアクティブなノードになるノードを特定するために、起動イベント時に使用されます。

完全な定足数が構成されたことを確認します。

```
# clquorum list -v
Quorum              Type
------              ----
d1                  shared_disk
node1               node
node2               node
```

オプションで、2 番目の shared_disk を定足数デバイスのリストに追加できます。

```
# clquorum add d2
# clquorum list -v
Quorum              Type
------              ----
d1                  shared_disk
d2                  shared_disk
```

```
node1                node
node2                node
```

共有ディスクデバイスがリストに表示されない場合は、それらのデバイス ID を確認してからそれを定足数に追加します。

a. 共有ディスクごとのデバイス ID を特定します。

```
# cldevice list -v
```

b. *clsetup* を実行して、定足数デバイスを追加します。

```
# clsetup

Select '1' for quorum.
Select '1' to dd a quorum device.
Select 'yes' to continue.
Select 'Directly attached shared disk'
Select 'yes' to continue.
Enter the device id (d<n>) for the first shared drive.
Answer 'yes' to add another quorum device.
Enter the device id for the second shared drive.
```

c. *clquorum show* を実行して、定足数メンバーシップを確認します。

```
# clquorum show
```

4. クラスタ構成全体を確認します。

```
# cluster check -v | egrep -v "not applicable|passed"
```

違反しているインスタンスがリスト内にないか探します。

5. 登録されているリソースタイプのリストを確認します。

```
# clrt list
SUNW.LogicalHostname:4
SUNW.SharedAddress:2
```

```
SUNW.gds:6
```

SUNW.gds がリストに表示されていない場合は、登録します。

```
# clrt register SUNW.gds
```

clrt list を実行して、確認します。

7

第7章 ACSLS HA 8.4 のインストールと起動

SUNWscacsls パッケージには、Oracle Solaris Cluster と通信する ACSLS エージェントソフトウェアが含まれています。これには、ACSLS と Solaris Cluster との間の正しい操作を確保する特殊な構成ファイルとパッチが含まれています。

基本インストール手順

1. ダウンロードした *SUNWscacsls.zip* ファイルを /opt 内で unzip します。

```
# cd /opt
# unzip SUNWscacsls.zip
```

2. *SUNWscacsls* パッケージをインストールします。

```
# pkgadd -d .
```

3. 隣接するノードで手順 1 と 2 を繰り返します。
4. 2 つのノードのいずれかで *acslspool* がマウントされたままになっていることを確認します。

```
# zpool status acslspool
```

acslspool がマウントされていない場合、ほかのノードを確認します。

acslspool がいずれのノードでもマウントされていない場合、次のようにして現在のノードにインポートします。

```
# zpool import -f acslspool
```

zpool status で確認します。

5. *acslspool* を所有するノードの */opt/ACSLSHA/util* ディレクトリに移動し、*copyUtils.sh* スクリプトを実行します。この操作は、両方のノードで重要なファイルを更新するか適切な場所にコピーします。隣接するノードでこの操作を繰り返す必要はありません。

```
# cd /opt/ACSLSHA/util
# ./copyUtils.sh
```

6. *acslspool* がアクティブになっているノードで、*acsss* ユーザーとして ACSLS アプリケーションを起動し (*acsss enable*)、これが動作していることを確認します。発生した問題をすべて解決します。主な問題は、ノードで STKacsls パッケージを削除して再インストールすることで解決することがあります。

   STKacsls パッケージを再インストールする必要がある場合、パッケージのインストール後に */opt/ACSLSHA/util/copyUtils.sh* スクリプトを実行します。

7. *acsls* を停止します。

```
# su - acsss
$ acsss shutdown
$ exit
#
```

8. アクティブノードから *acslspool* をエクスポートします。

```
# zpool export acslspool
```

   注記:

   ユーザー *acsss* がログインしている場合、ユーザーシェルが *acslspool* 内のいずれかの場所でアクティブな場合、または *acsss* サービスが有効なままになっている場合、この操作は失敗します。

9. 隣接するノードから *acslspool* をインポートします。

```
# zpool import acslspool
```

10. このノードで ACSLS アプリケーションを起動して、ライブラリの操作が成功したことを確認します。発生した問題をすべて解決します。主な問題は、ノードで STKacsls パッケージを削除して再インストールすることで解決することがあります。

STKacsls パッケージを再インストールする必要がある場合、パッケージのインストール後に */opt/ACSLSHA/util/copyUtils.sh* スクリプトを実行します。

ACSLS HA の構成

この手順では、Solaris Cluster によって管理および制御される 3 つの ACSLS リソースを作成します。

- *acsls-rs* は ACSLS アプリケーションそのものです。
- *acsls-storage* は、ACSLS が存在する ZFS ファイルシステムです。
- *<logical host>* は仮想 IP です (両方のノードに共通するネットワーク ID)。「/etc/hosts の構成」を参照してください。

これらのリソースハンドルが作成されたら、これらは *acsls-rg* の名前の下にある共通リソースグループに割り当てられます。

これらのリソースを構成するには、最初に *acslspool* がマウントされていることを確認し (*zpool list*)、次に */opt/ACSLSHA/util* ディレクトリに移動して *acsAgt configure* を実行します。

```
# cd /opt/ACSLSHA/util
# ./acsAgt configure
```

このユーティリティーでは論理ホスト名の入力を求めるプロンプトが表示されます。論理ホストが */etc/hosts* ファイルで定義されていること、および対応する IP アドレスが、2章「*ACSLS HA 用の Solaris システムの構成*」の章で定義されている *ipmp* グループにマップされていることを確認してください。さらに、*acsAgt configure* を実行する前に、*zpool list* を使用して、*acslspool* が現在のサーバーノードにマウントされていることを確認します。

この構成手順は完了するまで 1 分以上かかる場合があります。リソースハンドルが作成されると、ACSLS アプリケーションを起動しようとする操作が行われます。

ACSLS クラスタ操作のモニタリング

ACSLS クラスタの操作を表示するとき、複数の視点が存在します。Solaris Cluster が ACSLS アプリケーションを 1 分に 1 回プローブするとき、プローブが実行されるたびにプローブの結果が表示されると便利です。プローブがノードのスイッチオーバーイベントをトリガーするステータスを返すとき、1 つのノードでシャットダウンアクティビティーが表示され、

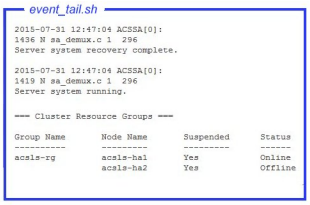
隣接するノードで起動アクティビティーが表示されると便利です。一般的に、ACSLS アプリケーションの動作の健全性を時系列に沿って示すビューがあれば便利です。

操作上のもっとも重要な視点は、ACSLS の観点です。*acsss_event.log* の末尾には、刻一刻と変わるシステム全体の健全性についての最適な指標を表示できます。

/opt/ACSLSHA/util ディレクトリのツール *event_tail.sh* は、どちらか一方のノードから *acsss_event.log* への直接アクセスを提供します。このツールから提供されるビューは、制御が 1 つのノードから別のノードに切り替わってもアクティブなままになります。通常の ACSLS アクティビティーに加えて、このツールは ACSLS Cluster リソースグループ (*acsls-rg*) のステータス変更を動的に追跡し、1 つのノードがオフラインになり別のノードがオンラインになるときのリアルタイムビューが可能です。このツールは次のようにシェルから発行します。

```
# /opt/ACSLSHA/util/event_tail.sh
```

図7.1 event_tail.sh の例



単一ノードの観点から開始アクティビティーおよび停止アクティビティーを表示するには、次のようにそのノードから *start_stop_log* を表示します。

```
# tail -f /opt/ACSLSHA/log/start_stop_log
```

アクティブなノード上の定期的なプローブの結果を表示するには:

```
# tail -f /opt/ACSLSHA/log/probe_log
```

Solaris Cluster および ACSLS Cluster エージェントは、重要なイベントの詳細を Solaris システムログ (*var/adm/messages*) に送信します。指定されたノード上のシステムログを表示するために、*/opt/ACSLSHA/log* ディレクトリ内にリンクが提供されます。

```
# tail -f /opt/ACSLSHA/log/messages
```

ha_console.sh ユーティリティー

クラスタ構成には多くの視点があり、時間の経過に応じてクラスタ制御が 1 つのノードから別のノードに移行するため、システムの操作アクティビティーを単一の観点から刻一刻と追

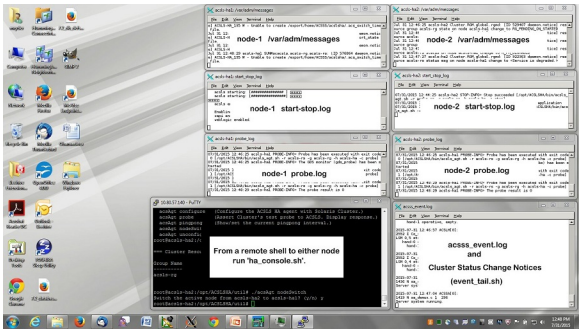
跡することが困難な場合があります。*ha_console.sh* ユーティリティーを使用すれば、包括的なビューをさらに簡単に提供できます。

ACSLS HA システムのいずれかのノードにリモートデスクトップからログインし、*ha_console.sh* を実行します。このユーティリティーは、ログイン ID (*who am i*) をチェックして*表示内容*の送信先を判断します。ローカルコンソールまたはデスクトップシステムから HA ノードに直接ログインし、表示を参照します。問題が発生した場合は、*/opt/ACSLSHA/log* ディレクトリの *gnome-terminal.log* 内でメッセージを探してください。

```
# /opt/ACSLSHA/util/ha_console.sh
```

このユーティリティーは、このセクションに記載されているすべてのログを両方のノードからモニターします。これはローカルのコンソール画面で 7 つの gnome-terminal ウィンドウを起動します。画面上に次のようにウィンドウを整理すると、便利なことがあります。

図7.2 gnome-terminal ウィンドウの整理



1 つの端末表示から、ACSLS クラスタコンプレックス全体の包括的なビューが表示されます。

リモートシステムは表示データをローカル画面に送信するため、ローカルシステム上で X-11 アクセスを開きます。UNIX システムでは、これを実行するコマンドは *xhost +* です。Windows システムでは、*xming* や *exceed* などの X-11 クライアントソフトウェアをインストールする必要があります。

ha_console.sh の使用に問題がある場合は、ローカルシステムから各ノードに対して複数のログインセッションを開いて、このセクションに記載されているさまざまなログを表示できます。

クラスタ操作の確認

1. *acslsha* が起動して Solaris Cluster に登録されたら、クラスタコマンドを使用して、ACSLS リソースグループとそれに関連するリソースのステータスを確認します。

```
# clrg status
   === Cluster Resource Groups ===
   Group Name       Node Name       Suspended      Status
   ----------       ---------       ---------      ------
   acsls-rg         node1           No             Online
                    node2           No             Offline

# clrs status
   === Cluster Resources ===
   Resource Name       Node Name      State        Status Message
   -------------       ---------      -----        --------------
   acsls-rs            node1          Online       Online
                       node2          Offline      Offline
   acsls-storage       node1          Online       Online
                       node2          Offline      Offline
   <logical host>      node1          Online       Online
                       node2          Offline      Offline
```

2. 初期テストを容易にするために、クラスタのフェイルオーバー準備を一時的に中断します。

```
# clrg suspend acsls-rg
# clrg status
```

3. アクティブノードからスタンバイへのクラスタの切り替え操作をテストします。

```
# cd /opt/ACSLSHA/util
# ./acsAgt nodeSwitch
```

スイッチオーバーアクティビティーは、前のセクションで説明されている手順を使用して複数の観点からモニターできます。

4. ACSLS サーバーの論理ホスト名を使用して、ACSLS クライアントシステムからネットワーク接続を確認します。

```
# ping acsls_logical_host
# ssh root@acsls_logical_host hostname
```

```
passwd:
```

この操作は、アクティブノードのホスト名を返します。

5. ACSLS 操作を確認します。

```
# su acsss
$ acsss status
```

6. 反対側のノードから手順 3、4、および 5 を繰り返します。
7. クラスタのフェイルオーバー準備を再開します。

```
# clrg resume acsls-rg
# clrg status
```

8. 次の一連のテストには、ノードのフェイルオーバー動作の検証が含まれます。

   複数のフェイルオーバーシナリオを連続して実行するために、デフォルトの応答間隔を 20 分から 5 分に下げます。(詳細は、8章「*ACSLS HA の微調整*」の章を参照してください。)テスト目的の場合、デフォルト設定を下げるのが簡単です。

   応答間隔を変更するには、*/opt/ACSLSHA/util* ディレクトリに移動して、*acsAgt pingpong* を実行します。

```
# ./acsAgt pingpong
Pingpong_interval
   current value:  1200 seconds.
   desired value: [1200] 300
Pingpong_interval : 300 seconds
```

9. アクティブノードをリブートして、2 つのシステムコンソールと、「ACSLS クラスタ操作のモニタリング」で提示した観点から操作をモニターします。スタンバイノードへの自動フェイルオーバー操作を確認します。
10. 手順 4 で提示されているように、クライアントシステムから論理ホストへのネットワークアクセスを確認します。
11. ACSLS 操作が新しいノードでアクティブになったら、このノードをリブートして、反対側のノードへのフェイルオーバーアクションを調べます。

ha_console.sh を使用して操作をモニターする場合、リブートノードに関連するウィンドウが表示されなくなることがあります。そのノードが再稼働したら、いずれかのノードで *ha_console.sh* コマンドをもう 1 回実行して、新しくリブートしたノードからウィンドウを復元します。

12. 手順 4 で提示されているネットワークの確認を繰り返します。

9章「*ACSLS クラスタの操作*」には、フェイルオーバーシナリオの完全なセットが提供されています。ACSLS HA システムを本番環境に配置する前に、これらのシナリオをいくつでもテストできます。システムを本番環境に戻す前に、推奨される応答間隔の設定に復元してフェイルオーバーが常時繰り返されないようにします。

8

第8章 ACSLS HA の微調整

この章では、ライブラリコンプレックス内で最適なフェイルオーバーポリシーを設定する方法、不要なフェイルバックイベントを回避するためにデフォルトの応答間隔を調整する方法、およびフェイルオーバーイベントの電子メール通知を登録する方法について説明します。

ライブラリ通信用のフェイルオーバーポリシーの定義

ACSLS HA エージェントは、ACSLS と接続済みライブラリの間の通信を常にモニターします。そのような通信は、ACSLS の連続操作にとって重要です。ただし、ライブラリとの通信に失敗したときに行うべきアクションは、ローカルの ACSLS HA 管理者が決定するポリシーによって異なります。

ポリシー表 *$ACS_HOME/acslsha/ha_acs_list.txt* を使用すると、ローカル管理者は、HA の回復が必要な ACS にとって望ましいフェイルオーバーアクションを定義できます。ライブラリとの通信の失敗時には、管理者の指示に応じて、代替ノードで正常な ACS 通信を確認した場合、ACSLS HA エージェントはそのノードにフェイルオーバーします。

複数の ACS 環境では、単一の ACS との通信が失敗したときには、ACSLS HA システムがフェイルオーバーすることが望ましい場合があります。ただし、あらゆるフェイルオーバーアクションは接続されたすべてのライブラリ上での本番環境に影響を及ぼすため、管理者は一般的なフェイルオーバーアクションをデータセンター内のさらに重大な ACS (1 つまたは複数) に限定することを選択する場合もあります。ライブラリとの通信が失われたときにクラスタのフェイルオーバーアクションが必要な ACS ごとに、ポリシーレコードが *ha_acs_list.txt* に作成されます。各レコードには 2 つのフィールドがあります。

```
ACS Number   Fail-over Action (true or false)
```

最初のフィールドは ACS ID で、2 番目のフィールドはブール値 *true* または *false* です。ポリシー設定の論理は次のとおりです。

- 2 番目のフィールドが *false* のとき、ACS への通信が失敗して復元できない場合でも、ACSLS HA エージェントは代替ノードへのクラスタのフェイルオーバーアクションを開始しません。

- 2 番目のフィールドが *true* のとき、プライマリノードからの通信を再確立するための試行がすべて失敗したあとで、ACSLS HA エージェントはクラスタのフェイルオーバーアクションを表明します。代替ノードでライブラリの接続が確認された場合にのみ、システムはフェイルオーバーします。

このファイルに一覧表示されていない ACS では、デフォルトのアクションは *false* です。

冗長電子装置 (RE) を備えたライブラリ

冗長電子装置 (RE) を備えたライブラリでは、ACSLS HA エージェントは、クラスタのフェイルオーバーアクションを使用する前に、通信を代替の RE パスに切り替えようとします。この RE の切り替えアクションは、単一の SL8500、SL3000、またはデュアル LMU を備えた古い 9310 にのみ適用されます。パーティション化されたライブラリでは、RE の自動切り替えは試行されません。

フェイルオーバーの *`Pingpong_interval`* の設定

Solaris Cluster の *`Pingpong_interval`* は、最初のクラスタのフェイルオーバーイベント後に完全な回復を復元できない場合に繰り返しのフェイルオーバーアクションを防止するタイムアウトプロパティーです。

これは、ACSLS リソースグループに関する、ユーザーによる変更が可能なプロパティーです。デフォルト値は 20 分に設定されます。この設定では、ACSLS HA エージェントによってフェイルオーバーアクションがリクエストされるとすぐに、最初のフェイルオーバーイベントが発生します。ただし、フェイルオーバーアクションをトリガーする可能性がある状態が、新しいクラスタノードでクリアされない場合、定義済みの応答間隔が期限切れになるまで、次のフェイルオーバーアクションは遅延されます。これによって、根本にある問題が解決されるまで、あるクラスタノードと別のクラスタノードの間での制御の不必要なスラッシングが防止されます。

このプロパティーのデフォルト設定を変更するには、ファイル *`$ACS_HOME/acslsha/pingpong_interval`* で定義されたデフォルトの数値を変更します。この数値は秒単位で表されます。

デフォルト設定である 1200 秒は、ほとんどの中規模から大規模のライブラリ構成で適切な設定です。このプロパティーに最適なタイムアウト値は、ライブラリ構成内に存在する LSM とテープドライブの実際の数によって異なります。ライブラリ構成が大きいほど、フェイルオーバーイベント後の回復に時間がかかるため、10 個を超える LSM または 40 台のドライブ、またはその両方を使用して構成されたシステムでは、この数値はより長い間隔に設定するべきです。

40 個の LSM 構成では設定 1800 (30 分) が推奨されるのに対して、1 - 4 個の LSM で構成された小さいライブラリでは設定 900 (15 分) が推奨されます。

ここで加えた変更は、ACSLS HA をコマンド *acsAgt configure* で再構成するまで有効になります。

```
# cd /opt/ACSLSHA/util
# ./acsAgt configure
```

このコマンドは、*acsls-rg* リソースグループがすでにアクティブであっても実行できます。これは通常の HA 操作に影響を与えずに新しいデフォルト設定を登録します。

pingpong_interval 設定は、*acsAgt pingpong* を使用してテストのために動的に変更できます。このコマンドで設定された値は *acsAgt configure* を使用してリソースグループを再起動するまで有効のままになります。

システムイベントの電子メール通知の登録

管理責務を持つユーザーは、システムのブートイベントや ACSLS HA クラスタのフェイルオーバーイベントなど、システムイベントの自動電子メール通知を登録してもかまいません。

そのようなイベントを登録するには、ユーザーは、次のディレクトリのそれぞれのファイルに電子メールアドレスを追加する必要があります。

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

ヘッダーの注釈の下にある単一の行に、目的の各受信者の電子メールアドレスを入力します。それ以降、システムがブートするか、HA クラスタがスタンバイノードにフェイルオーバーするたびに、登録済みの各ユーザーは電子メールで通知されます。

この機能では、sendmail サービスが ACSLS サーバーで有効になっていること、およびネットワークファイアウォールの制限でデータセンターからの電子メール通信が許可されていることを前提としています。

9

第9章 ACSLS クラスタの操作

Solaris Cluster は、1 つのサーバーノードから別のサーバーノードへと動作制御を移動させることで、重大な障害シナリオが発生した場合に自動的にシステムを回復するように設計されています。ただし、Solaris システムで発生するほとんどの障害では、回復するために完全なシステムのスイッチオーバーアクションは必要ありません。

- ネットワーク通信で発生した障害は、Solaris IPMP によって通知なしに迅速に処理されます。
- システムディスクの障害は、Solaris ZFS によって通知なしに自動的に処理されます。
- 接続されているストレージアレイ内の単一のディスクドライブで発生した障害は、ストレージアレイのファームウェアによって自動的に回復されます。ストレージアレイにディスク障害から回復する機能が不足している場合は、Solaris ZFS の制御下で、ミラー化構成での代替ドライブへの中断なしのディスク I/O が実現されます。
- 共有アレイへの HBA ポートに障害が発生した場合は、Solaris によって自動的に代替ポートに切り替えられます。同様に、共有アレイ上のコントローラモジュールに障害が発生した場合や、相互接続ケーブルが切断された場合も、Solaris によって即座にディスクリソースに接続している代替パスに戻されます。
- ライブラリ通信パスの障害は、ACSLS のデュアル TCP/IP ロジックによって自動的に回復されます。障害が発生したライブラリコントローラカードからの操作は、ライブラリの冗長電子装置 (RE) に関連付けられた ACSLS HA ロジックによって自動的に回復されます。
- ACSLS で動作している複数のプロセスのいずれかにエラーが発生した場合は、ACSLS デーモンによって即座にエラーが発生したプロセスがリブートされます。
- ACSLS デーモン自体にエラーが発生した場合や、残りの ACSLS サービスのいずれかの動作が停止した場合は、Solaris Service Management Facility (SMF) によって即座にエラーが発生したサービスが再起動されます。

前述のシナリオはすべて、Solaris Cluster が関与せずに迅速かつ自動的に処理されます。ただし、その他の重大な障害によってアクティブなサーバーノード上の ACSLS 操作が影響を受ける場合は、制御を代替ノードに切り替えるように、ACSLS HA から Solaris Cluster に指示されます。

ACSLS HA は起動後に、システムを 1 分に 1 回調査して、次のイベントのいずれかが発生していないかどうかを監視します。

- 接続されているライブラリへの通信が失われた。
- ACSLS 論理ホストへのネットワーク接続が失われた。
- クライアント呼び出し用の RPC リスナーポートへの接続が失われた。
- ACSLS ファイルシステムへのアクセスが失われた。
- ACSLS SMF サービスの保守状態を回復できない。

前述のイベントのいずれかが発生すると、Cluster のフェイルオーバーがトリガーされます。また、アクティブなサーバーノード上のシステムに致命的な状況が発生した場合にも、Solaris Cluster はフェイルオーバーを実行します。

ACSLS のクラスタ制御の開始

クラスタのフェイルオーバー制御をアクティブにするには:

```
# cd /opt/ACSLSHA/util
# ./acsAgt configure
```

ユーティリティーでは論理ホスト名の入力を求めるプロンプトが表示されます。論理ホストが */etc/hosts* ファイルで定義されていること、および対応する IP アドレスが、2章「*ACSLS HA 用の Solaris システムの構成*」の章で定義されている *ipmp* グループにマップされていることを確認してください。*acsAgt configure* を実行する前に、*zpool list* を使用して、*acslspool* が現在のサーバーノードにマウントされていることを確認します。

このアクションによって、ACSLS のクラスタ制御が開始されます。Solaris Cluster はシステムをモニターして、ACSLS の詳細と Solaris システム全般の健全性を確認するために 1 分に 1 回プローブします。致命的な状況であると判断されると、代替ノード上でアクションが開始されます。

ACSLS リソースグループのクラスタステータスをチェックするには:

```
# clrg status
```

次のことが表示されます。

- 各ノードのステータスを表示します。
- アクティブなノードを特定します。

- フェイルオーバーアクションが一時停止されているかどうかを示します。

acsls-storage へのフェイルオーバーポリシーの設定

アクティブノードと共有 RAID ディスクデバイスとの通信が失われた場合は常に、そのノードをリブートするポリシーを *acsls-storage* リソースに設定することをお勧めします。このアクションにより、アクティブノードはディスクに接続できなくなると制御を放棄するため、Solaris Cluster が制御を代替ノードに渡せるようになります。*Failover_mode* を SOFT から HARD に設定することで、共有ストレージデバイスへの通信が失われたときは常にアクティブノードが確実にリブートされるようになります。

既存の *Failover_mode* を表示するには、次のコマンドを実行します。

```
#  clrs show -v acsls-storage | grep Failover
```

Failover_mode を次のように HARD に設定します。

```
# clrs set -p Failover_mode=HARD  acsls-storage
```

クラスタ制御下での ACSLS の操作および保守

クラスタ制御がアクティブになると、通常の方法で ACSLS を操作できます。ACSLS の起動および停止には、標準の *acsss* 制御ユーティリティーを使用します。クラスタ制御下では、ユーザーはスタンドアロンの ACSLS サーバーでアプリケーションを起動および停止するときと同じ方法で、ACSLS サービスを起動および停止します。操作は、次のような標準の *acsss* コマンドを使用して管理されます。

```
acsss enable
acsss disable
acsss db
```

これらのコマンドを使用して手動で *acsss* サービスを起動または停止しない場合は、Solaris Cluster がフェイルオーバーアクションに介入します。同様に、Solaris SMF コマンド (*svcadm* など) を使用しない場合も Cluster が介入します。*acsss* サービスが異常終了した場合や中断された場合は常に、Cluster ではなく、主に SMF がこれらのサービスの再起動に対処します。

Solaris Cluster は、次の状況が発生した場合にのみ、隣接するノード上の制御の復元に介入します。

- ACSLS ファイルシステムとの通信が失われた。

- すべての冗長化されたパブリック Ethernet ポートとの通信が失われた。
- 指定したライブラリとの通信が失われ、回復できない。

クラスタ制御の一時停止

保守アクティビティーによってクラスタの不要なフェイルオーバーイベントがトリガーされるおそれがある場合は、*acsls* リソースグループのクラスタ制御を一時停止できます。

クラスタ制御を一時停止するには:

```
# clrg suspend acsls-rg
```

リソースグループが一時停止されている間は、このようなアクションがトリガーされるような状況が発生しても、Solaris Cluster は隣接するノードへの制御の切り替えを試みません。

この一時停止により、ライブラリの生成がフル稼働中の場合でも、より侵襲的なシステム修復が可能です。

一時停止モード中にアクティブなノードのリブートが発生した場合は、リブート後に *acslspool* がマウントされず、ACSLS 操作が停止されます。このような状況を解消するには、クラスタ制御を再開します。

クラスタ制御を再開するには:

```
# clrg resume acsls-rg
```

共有ディスクリソースが現在のノードにマウントされている場合は、正常な操作が再開されます。ただし、アクティブ化時に Solaris Cluster で、*zpool* がマウントされていないことが検出された場合は、隣接するノードに制御が即座に切り替えられます。隣接するノードにアクセスできない場合、制御は現在のノードに切り替わります。クラスタは *acslspool* をマウントしてこのノード上で ACSLS サービスを起動しようとします。

ACSLS HA クラスタの電源切断

次の手順では、ACSLS HA システムの電源を切断する必要がある場合の、安全な電源切断シーケンスについて説明します。

1. クラスタ内のアクティブなノードを特定します。

   ```
   # clrg status
   ```

オンラインのノードを検索します。

2. *root* としてアクティブなノードにログインし、ACSLS リソースグループの Solaris Cluster 制御を停止します。

```
# clrg suspend acsls-rg
```

3. ユーザー *acsss* に切り替え、*acsss* サービスを停止します。

```
# su - acsss
$ acsss shutdown
```

4. *acsss* としてログアウトし、ノードの電源を正常に切断します。

```
$ exit
# init 5
```

5. 代替ノードにログインし、*init 5* を使用して電源を切断します。
6. 物理的な電源スイッチを使用して、共有ディスクアレイの電源を切断します。

一時停止された ACSLS クラスタシステムの電源投入

制御されたシャットダウンの前にアクティブであったノード上で ACSLS 操作を復元するには:

1. ローカルで物理的な電源スイッチを使用するか、またはリモートで Sun Integrated Lights Out Manager を使用して、両方のノードの電源を投入します。
2. 共有ディスクアレイの電源を投入します。
3. *root* として、どちらか一方のノードにログインします。
4. *acsss* としてログインするか、*$ACS_HOME* ディレクトリを一覧表示してみれば、どちらか一方のノードに共有ディスクリソースがマウントされていないことがわかります。クラスタのモニタリングを再開するには、次のコマンドを実行します。

```
# clrg resume acsls-rg
```

このアクションによって、Solaris Cluster は、システムをシャットダウンしたときにアクティブであったノードに共有ディスクをマウントします。また、このアクションによって自動的に *acsss* サービスがリブートし、通常の操作が再開されます。

単一ノードクラスタの作成

ほかのノードの保守中に、あるノード上のスタンドアロンサーバー環境から ACSLS 操作を続行する必要がある場合があります。これは、ハードウェアの保守、オペレーティングシステムのアップグレード、または Solaris Cluster へのアップグレードの状況に当てはまります。

次の手順を使用して、スタンドアロンの ACSLS サーバーを作成します。

1. 非クラスタモードで目的のノードをリブートします。

   ```
   # reboot -- -x
   ```

   SPARC サーバーで Open Boot Prom (OBP) からブートして非クラスタモードになるには:

   ```
   ok: boot -x
   ```

   X86 サーバー上で、GRUB ブートメニューを編集する必要があります。

   a. システムの電源を投入します。
   b. GRUB ブートメニューが表示されたら、「**e**」(編集) を押します。
   c. サブメニューから矢印キーを使用して、「**kernel /platform/i86pc/multiboot**」を選択します。これを選択したら、「**e**」を押します。
   d. 編集モードで、マルチブートオプション *`kernel /platform/i86pc/multiboot -x`* に *`-x`* を追加して、「**return**」をクリックします。
   e. マルチブートオプション *`-x`* が選択されている状態で「**b**」を押して、そのオプションでブートします。
2. ブートサイクルが完了したら、root としてログインし、ACSLS Z プールをインポートします。

   ```
   # zpool import acslspool
   ```

   ディスクリソースが別のノードに関連付けられたままになっている場合は、必要に応じて、*`-f`* (強制) オプションを使用します。

   ```
   # zpool import -f acslspool
   ```

3. *`acsss`* サービスを起動します。

```
# su - acsss
$ acsss enable
```

10

第10章 ソフトウェアコンポーネントのインストール、アップグレード、および削除

この章では、ACSLS HA に関連したさまざまなコンポーネントをアップグレードまたは削除するときに従う手順について説明します。

ACSLS パッチのインストール

STKacsls パッケージに対応したパッチをインストールするには:

1. クラスタ制御を一時停止します。

```
# clrg suspend acsls-rg
```

2. */opt* ディレクトリにパッチをダウンロードし、パッケージを unzip します。
3. */opt/ACSLS_8.x.x* ディレクトリに移動し、パッチの *README.txt* ファイルの指示に従います。
4. ACSLS 操作を無効にし、隣接するノードに制御を切り替えます。

```
# su - acsss
$ acsss shutdown
$ exit
# cd /opt/ACSLSHA/util
# acsAgt nodeSwitch
```

5. 新しいノード上で ACSLS パッチをインストールします。
6. 更新された ACSLS 構造に ACSLS HA の変更を適用します。

```
# cd /opt/ACSLSHA/util
# ./copyUtils.sh
```

7. ACSLS サービスを有効にします。

```
# su - acsss
# acsss enable
```

8. *acsls-rg* リソースグループのクラスタ制御を再開します。

```
# clrg resume acsls-rg
```

ACSLS パッケージの削除

ACSLS をアップグレードする際に、ACSLS パッケージの削除が必要な場合があります。そのために、クラスタ制御を無効にし、両方のノード上で ACSLS サービスを停止してから、各ノード上でパッケージを削除する必要があります。次の手順を使用します。

1. クラスタ制御を一時停止します。

```
node1:# clrg suspend acsls-rg
```

2. アクティブなノード上で、ACSLS を停止します。

```
node1:# su - acsss
node1:$ acsss shutdown
node1:$ exit
node1:#
```

3. 共有ディスクアレイ上のファイルシステムをエクスポートします。

```
node1:# cd /
node1:# zpool export acslspool
```

ユーザー *acsss* でログインしている場合は、この操作に失敗します。

4. 代替ノードにログインし、共有ディスクアレイをインポートします。

```
node1:# ssh <alternate node>
node2:# zpool import acslspool
```

5. ACSLS を停止します。

```
node2:# su - acsss
```

```
node2:$ acsss shutdown
node2:$ exit
node2:#
```

6. *STKacsls* パッケージを削除します。

```
node2:# pkgrm STKacsls
```

7. 元のノードに戻り、*STKacsls* パッケージを削除します。

```
node2:# exit
node1:# pkgrm STKacsls
```

ACSLS アップグレードリリースのインストール

新しい ACSLS リリースをインストールする前に、両方のノード上で *STKacsls* パッケージを削除する必要があります。上のセクションで説明した手順を参照してください。新しいパッケージをインストールするには、この手順に従います。

1. */opt* ディレクトリに STKacsls パッケージをダウンロードし、パッケージを unzip します。代替ノード上で、この手順を繰り返します。
2. Solaris Cluster が一時停止されている状態で、共有ディスクアレイ (*acslspool*) が現在のノードにマウントされていることを確認します。

```
node1:# zpool list
```

acslspool がマウントされていない場合は、代替ノードにログインします。どちらのノードにもマウントされていない場合は、*acslspool* をインポートします。

3. */opt/ACSLS_8.x.x* ディレクトリに移動し、*README.txt* ファイルの指示に従います。
4. *acslspool* をエクスポートします。

```
node1:# zpool export acslspool
```

ユーザー *acsss* でログインしている場合は、この操作に失敗します。

5. 代替ノードにログインし、手順 1 から 3 を繰り返します。
6. */opt/ACSLSHA/util* ディレクトリに移動し、*copyUtils.sh* を実行します。

```
node2:# cd /opt/ACSLSHA/util
node2:# ./copyUtils.sh
```

7. ACSLS ライブラリ制御を起動します。

```
node2:# su - acsss
node2:$ acsss enable
node2:$ exit
node2:#
```

8. *acsls* リソースグループのクラスタ制御を再開します。

```
node2:# clrg resume acsls-rg
```

ACSLS HA の再インストールまたはアップグレードのインストール

SUNWscacsls パッケージを再インストールするには、ACSLS ライブラリの動作を停止して *acsls-rg* リソースグループをオフラインにすることが必要です。

1. ACSLS を停止します。

```
# su - acsss
$ acsss shutdown
$ exit
#
```

2. カスタマイズされたすべての変更を *ha_list.txt* および *pingpong_interval* に保存します。

```
# cd $ACS_HOME/acslsha
# cp ha_list.txt ha_list.save
# cp pingpong_interval pingpong_interval.save
```

3. *acsls-rg* リソースグループをオフラインにします。

```
# clrg offline acsls-rg
```

このアクションにより、*acslspool* が ZFS ファイルシステムからアンマウントされます。

4. *acsls-rg* リソースグループの構成を解除します。

```
# cd /opt/ACSLSHA/util
# ./acsAgt unconfigure
```

5. 更新された *SUNWscacsls* パッケージを両方のノードにダウンロードして unzip します。
6. acslspool を 2 つのノードのいずれかに再マウントします。

```
# zpool import -f acslspool
```

7. ACSLS を開始して、適切に機能することを確認します。

```
# su - acsss
$ acsss enable
$ exit
#
```

8. *copyUtils.sh* ユーティリティーを実行します。

```
# cd /opt/ACSLSHA/uti
# ./copyUtils.sh
```

9. 手順 2 で保存したカスタマイズされたファイルをすべて復元します。

```
# cd $ACS_HOME/acslsha
# cp ha_list.save ha_list.txt
# cp pingpong_interval.save pingpong_interval
```

10. *acsls-rg* リソースグループを構成します。

```
# cd /opt/ACSLSHA/util
# ./acsAgt configure
```

このアクションによって、ACSLS が停止および再起動されます。

数分以内に、*clrg status* コマンドによって、*acsls-rg* リソースグループがオンラインに戻り、ACSLS が動作可能であることが表示されます。

Solaris Cluster のアップグレード

Solaris Cluster を削除またはアップグレードする前に、ACSLS を停止して *acsls-rg* リソースグループの構成を解除します。

```
# su - acsss
$ acsss shutdown
$ exit
# cd /opt/ACSLSHA/util
# ./acsAgt unconfigure
```

具体的なアップグレード手順については、最新の Solaris Cluster ドキュメントを参照してください。Solaris Cluster をアップグレードする一般的なコマンドは、次のとおりです。

```
# scinstall -u
```

Solaris Cluster のインストールが更新されたら、*acsls-rg* リソースグループを構成します。

```
# cd /opt/ACSLSHA/util
# ./acsAgt configure
```

Solaris Cluster の削除

1. ACSLS および *acsls-rg* リソースグループを停止します。

   ```
   # su - acsss
   $ acsss shutdown
   $ exit
   # cd /opt/ACSLSHA/util
   # ./acsAgt unconfigure
   ```

2. 両方のノードを非クラスタモードでリブートします。

   ```
   # reboot -- -x
   ```

3. 両方のノードが起動したら、どちらか一方のノードからログインして、Solaris Cluster パッケージを削除します。

   ```
   # scinstall -r
   ```

11

第11章 クラスタのロギング、診断、およびテスト

この章では、ACSLS HA のインストールのテスト、問題の診断、およびシステムで発生する可能性がある問題のトラブルシューティングに使用できるさまざまなリソースについて説明します。

クラスタ操作全般のモニタリング

起動イベントまたはスイッチオーバーイベント中に発生するアクティビティーは、2 つのノード間で幅広く分散されます。したがって、テスト中に全体的な操作を監視するために選ばれる視点によって、イベントの発生時に展開イベントを表示するための能力が大きく左右される可能性があります。包括的なビューを設定するための手順については、「ha_console.sh ユーティリティー」に記載されています。

テスト中に HA 全体の動作を監視するために推奨されるダッシュボード構成には、ノードごとに 4 つのウィンドウから成る 8 つのシェルウィンドウが含まれます。

1. *root* 用のコマンドシェルは、必要に応じてさまざまなコマンドを発行するために各ノードに予約します。
2. システムの */var/adm/messages* ファイルの末尾を表示するためのウィンドウを各ノードに設定します。

   ```
   # tail -f /var/adm/messages
   ```

   Solaris Cluster はすべての情報メッセージをこのログファイルに出力します。
3. *acsls-rs* リソースの *start_stop log* の末尾を表示するための別のウィンドウを各ノードに設定します。

   ```
   # tail -f /var/cluster/logs/DS/acsls-rg/acsls-rs/start_stop_log.txt
   ```

   acsls_agt.sh 起動スクリプトによって公開されたすべてのメッセージはここに表示されます。

4. *acsls-rs* プローブログの末尾を表示するために、各ノードの 3 番目のウィンドウを設定する必要があります。

```
# tail -f /var/cluster/logs/DS/acsls-rg/acsls-rs/probe_log.txt
```

アプリケーションが起動すると、Solaris Cluster は ACSLS リソースを 1 分ごとにプローブします。数値コードが各プローブからクラスタに戻され、結果は *probe_log.txt* ファイルに出力されます。各プローブによって、標準的な 5 つの戻り値のいずれかがこのログに公開されて表示されます。

```
  0 -  The probe found that ACSLS is healthy and functioning normally.
  1 -  The probe may not have completed due to a functional error.
  2 -  The probe reports that ACSLS is in a transitional state.
  3 -  The ACSLS application has been intentionally placed offline.
201 -  A condition was detected that requires fail-over action.
```

コード *201* に応答する場合のみ、Solaris Cluster はフェイルオーバーアクションを開始します。そのようなアクションを実行させる条件は、9章「*ACSLS クラスタの操作* 」の章に一覧表示されています。Cluster プローブからのほかのすべての戻りコードは情報提供と見なされ、クラスタの応答アクションは実行されません。

テスト用のサンプルプローブは、コマンド行からいつでも実行できます。コマンド *acsAgt probe* を次のように使用します。

```
#/opt/ACSLSHA/util/acsAgt probe
```

前述のすべてのログには、Solaris Cluster から認識されているシステムビューが反映されます。*$ACS_HOME/log/* ディレクトリ内の 2 つの追加のログは、ACSLS アプリケーションレベルからのビューを提供します。*acsss_event.log* には、アプリケーションが起動された時点から ACSLS によって検出されたすべての重要なイベントが報告されます。また、SMF によって検出された ACSLS 起動のすべての問題は *acsls_start.log* に記録されます。

Cluster のモニタリングユーティリティー

Solaris Cluster ユーティリティーは、*/usr/cluster/bin* ディレクトリにあります。

- ACSLS リソースグループの現在の状態を表示する場合: *clrg list -v*

- 2 つのクラスタノードの現在のステータスを表示する場合: *clrg status*
- リソースグループのステータスを表示する場合: *clrs status*
- ノード、定足数デバイス、およびクラスタリソースでの詳細ステータスを取得する場合: *cluster status*
- クラスタ構成内の詳細なコンポーネントリストの場合: *cluster show*
- リソースグループでの各 Ethernet ノードのステータスを表示する場合: *clnode status -m*
- 各ノード上のさまざまな *acsls-rg* リソースのステータスを表示する場合: *scstat -g*
- ハートビートネットワークリンクの健全性を表示する場合: *clintr status*
- IPMP ステータスを表示する場合: *scstat -i*
- ノードステータスを表示する場合: *scstat -n*
- 定足数構成とステータスを表示する場合: *scstat -q* または *clq status*
- タイムアウト値を含む詳細なクラスタリソースを表示する場合: *clresource show -v*

回復およびフェイルオーバーのテスト

このセクションでは、回復およびフェイルオーバーのテストの状態、モニタリング、およびテストについて説明します。

回復状態

システムのフェイルオーバーイベントを必要とせずに回復できる、致命的なシステム状態は多数存在します。たとえば、IPMP では、各グループ内の 1 つの Ethernet 接続が何らかの理由で失敗することがありますが、通信は代替のパスを通じて途切れずに再開されます。

共有ディスクアレイは、各サーバー上の 2 つの別個のポートを使用してサーバーに接続するべきです。1 つのパスが中断された場合、ディスク I/O 操作は、代替のパスを通じて中断することなく再開されます。

ACSLS は、Solaris Service Management Facility (SMF) によってモニターされるいくつかのソフトウェアサービスで構成されます。ユーザー acsss は、*acsss status* コマンドを使用してそれぞれの *acsss* サービスを一覧表示できます。これらのサービスには、PostgreSQL データベース、WebLogic Web アプリケーションサーバー、および ACSLS アプリケーションソフトウェアがあります。特定のサービスが Solaris システムで失敗した場合、システムのフェイルオーバーを必要とせずに、SMF はそのサービスを自動的にリブートします。

acsls サービス自体は、親の *acsss_daemon* によってモニターされる多数の子プロセスで構成されます。ACSLS サブプロセスを一覧表示するには、(ユーザー *acsss* として) コマンド

psacs を使用します。何らかの理由でいずれかの子プロセスが異常終了した場合、親は即時にその子をリブートして、通常の操作を回復します。

回復のモニタリング

システムリソース (ディスク I/O や Ethernet 接続など) の回復を確認するために最適な場所は、システムログ */var/adm/messages* です。

SMF は、モニターするソフトウェアサービスごとに特定のログを保持します。このログには、起動、再起動、およびシャットダウンイベントが表示されます。サービスログへのフルパスを取得するには、コマンド *svcs -l service-name* を実行します。ACSLS サービスは、*acsss* コマンド *$ acsss status* を使用して一覧表示できます。サブプロセスは、コマンド *$ acsss p-status* を使用して一覧表示できます。

ACSLS サブプロセスの回復を表示するには、*acsss_event.log* (*$ACS_HOME/ACSSS/log/acsss_event.log*) をモニターできます。このログには、ACSLS サブプロセスに関するすべての回復イベントが表示されます。

回復テスト

冗長なネットワーク接続は、Solaris IPMP ロジックによって自動的に再開されます。共有ディスクアレイへの中断されたデータ接続は、冗長データパスでは Solaris によって自動的に再開されます。Solaris Service Management Facility の制御下にあるサービスは、SMF によって自動的に再開されます。

実際のフェイルオーバーイベントに関するテストでは、*$ACS_HOME/acslsha/pingpong_interval* ファイルで定義されたプロパティー設定を確認してください。フェイルオーバーイベントを引き起こす可能性がある状態にもかかわらず、指定された *pingpong_interval* 内に前のフェイルオーバーイベントが発生した場合、Solaris Cluster はフェイルオーバーアクションを開始しません。

応答間隔を表示または動的に変更するには、*/opt/ACSLSHA/util* ディレクトリに移動して、*acsAgt pingpong* を実行します。

```
# ./acsAgt pingpong
Pingpong_interval
   current value:  1200 seconds.
   desired value: [1200] 300
Pingpong_interval : 300 seconds.
```

次のいずれかまたはすべての手法を使用して、HA インストールの回復力を評価します。

1. ACSLS が操作可能な間に、アクティブノード上の各 IPMP グループから 1 つの Ethernet 接続を切断します。*# scstat -i* を使用してステータスをモニターします。

   /var/adm/messages で反応を調べます。ACSLS の操作は、この手順によって中断されません。

2. Cluster *Failover_mode* が **HARD** に設定されていることを確認します。ACSLS が操作可能な間に、アクティブサーバーから共有ディスクリソースへの 1 つのファイバまたは SAS 接続を切断します。

   /var/adm/messages で反応を調べます。ACSLS の操作は、この手順によって中断されません。

   このテストをそれぞれの冗長 I/O 接続で繰り返します。

3. *acsss_daemon* を停止することによって ACSLS を突然終了します。*pkill acsss _daemon* を使用します。

   svcs -l acsls を実行して、サービスログを見つけます。

   acsss_daemon が停止したときのこのログの末尾を表示します。サービスが SMF によって自動的に再起動されることがわかります。*acsls shutdown* を使用して *acsls* を停止した場合、類似したアクションが確認されます。

4. SMF を使用して、*acsls* サービスを無効にします。

   これは *svcadm disable acsls* を使用して root として行うか、*acsss disable* を使用してユーザー *acsss* として行うことができます。

   SMF がこのシャットダウンイベントを管理するため、*acsls* サービスの再起動は試行されません。これは必要な動作です。*acsls* サービスは SMF の下で再起動する必要があります。*root* として、コマンド *svcadm enable acsls* を使用します。または、ユーザー *acsss* として、*acsss-enable* コマンドを使用します。

5. *acsdb* サービスを停止します。

   ユーザー *acsdb* として、*.acsls_env* ファイルを探します。

   ```
   $ su acsdb
   $ . /var/tmp/acsls/.acsls_env
   ```

   ここで、次のコマンドを使用して PostgreSQL データベースを突然無効にします。

```
pg_ctl stop /
     -D $installDir/acsdb/ACSDB1.0/data /
     -m immediate
```

このアクションによってデータベースが停止し、*acsls* プロセスも停止します。*svcs -l acsdb* を実行して、*acsdb* サービスログを見つけます。

データベースの停止時に、*acsdb* サービスログと *acsls* サービスログの両方の末尾を確認します。*acsdb* サービスが停止すると、*acsls* サービスも停止することがわかります。両方のサービスが、SMF によって自動的に再起動されます。

6. ACSLS が操作可能な間に、ユーザー *acsss* として *psacs* を実行して、*acsss_daemon* の下で実行されているサブプロセスのリストを取得します。

   これらのサブプロセスの 1 つを停止します。*acsss_event.log* を調べて、サブプロセスが再起動されて回復手順が起動されることを確認します。

フェイルオーバー状態

Solaris Cluster ソフトウェアは Solaris システムをモニターし、システムのフェイルオーバーイベントを必要とする致命的な状態を探します。これには、ユーザーが開始したフェイルオーバー (*acsAgt nodeSwitch* または *clrg switch -n*)、アクティブノードのシステムのリブート、システムのハング、致命的なメモリー障害、またはアクティブノードでの回復不能な I/O 通信があります。また、Solaris Cluster は、特定のアプリケーション用に設計された HA エージェントをモニターします。ACSLS HA エージェントは、次の状態でシステムのフェイルオーバーイベントをリクエストします。

- アクティブノードと論理ホストの間の TCP/IP 通信が失われている。
- *$ACS_HOME* ファイルシステムがマウントされていない。
- データベースバックアップファイルシステム (*$ACS_HOME/.../backup*) がマウントされていない。
- ファイル *$ACS_HOME/acslsha/ha_acs_list.txt* に指定されていて、必要な状態はオンラインであり、通常であれば *switch lmu* は不可能であるか失敗する、ACS に対応するライブラリへの通信が失われている。

フェイルオーバーのモニタリング

場合によっては、コマンド *# clrg status* を使用して、各ノードのフェイルオーバーステータスをモニターできることがあります。

フェイルオーバーアクティビティーは、*start_stop_log* の末尾を調べてモニターすることもできます。

```
# tail -f /var/cluster/logs/DS/acsls-rg/acsls-rs/start_stop_log.txt
```

診断フェイルオーバー操作の実行時に、両方のノードで */var/adm/messages* ファイルを表示する (*tail -f*) と役に立つことがあります。「ACSLS クラスタ操作のモニタリング」を参照してください。

フェイルオーバーのテスト

1. クラスタのフェイルオーバーイベントを開始する単純なコマンドは、*acsAgt nodeSwitch* です。

   ```
   # acsAgt nodeSwitch
   ```

   または、同等のクラスタコマンドを使用します。

   ```
   # clrg switch -n <node name> acsls_rg
   ```

   このアクションによって ACSLS アプリケーションが停止し、アクティブサーバーからスタンバイシステムに処理が切り替えられます。オプション *-M -e* は、新しいノードで SMF サービスを有効にするようクラスタサーバーに指示します。「ACSLS クラスタ操作のモニタリング」を参照してください。
2. アクティブノードでのシステムのリブートによって、代替ノードへの HA の切り替えが即時に開始されます。

   この操作は、ACSLS が新しいアクティブノードで実行されて終了します。スタンバイシステムはその新しい役割がアクティブノードであると想定するため、スタンバイノードで、*/var/adm/messages* ファイルの末尾を調べます。コマンド *# clrg status* を定期的に実行することもできます。
3. *init 5* を使用して、アクティブサーバーノードの電源を切って、システムのフェイルオーバーを確認します。
4. アクティブサーバーノードと共有ディスクストレージ配列との間の両方のデータ回線を抜いて、スタンバイノードへのシステムの切り替えを確認します。
5. 特定のライブラリがポリシーファイル *ha_acs_list.txt* に一覧表示されていると想定して、アクティブサーバーノードとそのライブラリとの間の両方の Ethernet 通信回線を切断します。

スタンバイノードへのシステムのフェイルオーバーを確認します。

追加のテスト

ミラー化されたブートドライブがホットプラグ可能である場合、ブートドライブの 1 つを無効にして、システムが完全に操作可能なままであることを確認できます。1 つのブートドライブを無効にして、システムをリブートし、代替のブートドライブからノードが起動することを確認します。2 つの各ノード上のブートドライブごとにこのアクションを繰り返します。

アクティブノードから 1 つの電源を外しても、システムは代替の電源で完全に操作可能なままになります。

12

第12章 トラブルシューティングのヒント

ACSLS HA 8.4 は、Solaris 11.2 下の 2 ノードシステムで動作する ACSLS アプリケーションを、Solaris Cluster 4.2 の制御下にある IPMP および ZFS と統合したものです。

ACSLS が実行中であることの確認

ACSLS サービスがアクティブノードで実行されていることを確認するには、ユーザー *acsss* として次のコマンドを使用します。

```
# su - acsss
$ acsss status
```

1 つ以上のサービスが無効になっている場合、*$ acsss enable* を使用して有効にします。

ステータスの表示によって、1 つ以上の ACSLS サービスが保守モードになっていることが明らかになった場合、コマンド *$ acsss 1-status* を実行します。

障害があるサービスのログファイルへのパスを探して、サービスが保守モードになっている理由を説明するヒントをそのログで確認します。

1 つ以上の ACSLS サービスが保守モードになっている場合、*acsss* コマンドを使用してそのサービスを無効にしてから有効にすることでクリアできます。

```
$ acsss shutdown
$ acsss enable
```

root として *# svcadm clear <service name>* を使用して、個々のサービスをクリアします。

根本的な障害を修正するまで、サービスはクリアされません。

問題の原因を明らかにする手段として、特定の動作ログも確認するべきです。これらの大部分は、*$ACS_HOME/log* ディレクトリにあります。

確認するべきプライマリログは *acsss_event.log* です。このログには、ACSLS の操作全体に関するほとんどのイベントが記録されます。

問題が ACSLS GUI または論理ライブラリの操作に関係している場合、関連するログは *$ACS_HOME/log/sslm* ディレクトリにあります。

ACSLS GUI および WebLogic の場合、*AcslsDomain.log*、*AdminServer.log*、および *gui_trace.logs* を探してください。

WebLogic に関するインストールの問題は、*weblogic.log* にあります。

論理ライブラリの問題の場合、論理ライブラリの構成後に、*slim_event.logs* と *smce_stderr.log* を調べます。

共有ディスクリソースへの接続の対処

1. *acsls-storage* リソースがアクティブなクラスタノードに対してオンラインになっていることを確認します。

   ```
   # clrs status acsls-storage
   ```

2. *acsls-storage* リソースがオンラインではない場合、アクティブノードでリソースが ZFS にマウントされているかどうかを確認します。

   ```
   # zpool status
   ```

   acslspool がアクティブノードでマウントされていない場合、スタンバイノードでマウントされているかどうかを確認します。

   ```
   # ssh standby hostname zpool status
   ```

   共有ディスクリソースがスタンバイノードでマウントされている場合、クラスタの制御をそのノードに切り替えます。

   ```
   # clrg switch -n standby hostname acsls-rg
   ```

3. *acslspool* がアクティブノードでマウントされておらず、*acsls-storage* リソースがオフラインになっている場合、アクティブノードで *acslspool* を表示できるかどうかを確認します。

   ```
   # zpool import (no argument)
   ```

注記:

acsls-storage がオフラインになっている場合にのみ、この操作は機能します。オフラインにするには、コマンド *clrs disable acsls-storage* を使用します。

アクティブノードから *acslspool* を表示できる場合、インポートを試行します。

```
# zpool import -f acslspool
```

import 操作が成功する場合、*acsls-storage* リソースを Solaris Cluster に対してオンラインにします。

```
# clrs enable acsls-storage
```

アクティブノードで *acslspool* を表示できない場合、共有ドライブへの物理接続をトラブルシューティングする必要があります。

論理ホストに ping できない場合

1. 論理ホスト名が Solaris Cluster に登録されていることを確認します。

   ```
   # clrslh list
   ```

2. アクティブノードを判別します。

   ```
   # clrg status | grep -i Online
   ```

3. アクティブノードに ping できることを確認します。

   ```
   # ping <node name>
   ```

4. *logical-host* 名前リソースがアクティブノードに対してオンラインになっていることを確認します。

   ```
   # clrslh status
   ```

   論理ホストがオンラインではない場合、有効にしてください。

```
# clrs enable <logical host>
```

5. パブリックグループに割り当てられた IP インタフェースの状態を確認します。

```
# ipadm
```

出力の表示で、パブリック *ipmp* グループの各メンバーの *ok* 状態を確認します。

6. パブリックグループ (*ipmp0*) 内のインタフェースごとに、その物理的な状態を確認します。

```
# dladm show-phys
```

7. 論理ホストが、パブリック *ipmp* グループ内にある 2 つのインタフェースのいずれか (手順 5 で確認) に plumb されていることを確認します。

```
# arp <logical-hostname>
# ifconfig net0
# ifconfig net4
```

この例では、net0 と net4 がパブリック *ipmp* グループに割り当てられたことを想定しています。

2 つのインタフェースのうちいずれかの MAC アドレスは、論理ホスト名に割り当てられた MAC アドレスと一致します。

ノード間の相互接続の確認

2 つのノード間の Cluster の通信が失われたためにクラスタの制御が失敗したと考えられる場合、次のようにして、Cluster のプライベート相互接続を確認します。

```
# cluster status -t interconnect
```

索引