IBM PowerHA SystemMirror for AIX Standard Edition

バージョン 7.2.2

PowerHA SystemMirror コマンド

IBM PowerHA SystemMirror for AIX Standard Edition

バージョン **7.2.2**

PowerHA SystemMirror コマンド

IBM

お願い

本書および本書で紹介する製品をご使用になる前に、 109 ページの『特記事項』に記載されている情報をお読みください。

本書は、IBM PowerHA SystemMirror 7.2.2 Standard Edition for AIX および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原典： IBM PowerHA SystemMirror for AIX
Standard Edition
Version 7.2.2
PowerHA SystemMirror commands

目次

本書について

コマンドを使用して、PowerHA® SystemMirror® クラスターを管理および構成できます。各コマンドには構文および例があります。

強調表示

本書では、次の強調表示規則を使用しています。

太字	コマンド、サブルーチン、キーワード、ファイル、構造体、ディレクトリー、およびシステムによって名前が事前に定義されているその他の項目を表します。 さらに太字の強調表示は、ユーザーが選択するボタン、ラベル、およびアイコンなどのグラフィカル・オブジェクトも示します。
イタリック	ユーザーが入力する実際の名前または値のパラメーターを示します。
モノスペース	具体的なデータ値の例、表示される可能性があるテキストの例、プログラマーとして作成する可能性があるものに似たプログラム・コードの一部の例、システムからのメッセージ、またはユーザーが入力しなければならないテキストを示します。

AIX でのケース・センシティブ

AIX® オペレーティング・システムでは、すべてケース・センシティブとなっています。これは、英大文字と小文字を区別するという意味です。 例えば、**ls** コマンドを使用するとファイルをリストできます。 `LS` と入力すると、システムはそのコマンドが「`is not found`」と応答します。 同様に、**FILEA**、**FiLea**、および **filea** は、同じディレクトリーにある場合でも、3 つの異なるファイル名です。 予期しない処理が実行されないように、常に正しい大/小文字を使用するようにしてください。

ISO 9000

当製品の開発および製造には、ISO 9000 登録品質システムが使用されました。

関連情報

- PowerHA SystemMirror バージョン 7.2.2 for AIX PDF 資料は、『PowerHA SystemMirror 7.2.2 の PDF』のトピックで入手可能です。
- PowerHA SystemMirror バージョン 7.2.2 for AIX リリース・ノートは、『PowerHA SystemMirror 7.2.2 リリース・ノート』のトピックで入手可能です。

PowerHA SystemMirror コマンド

以下のコマンドは一般的に、クラスター環境に関する情報を入手するために、または特定の機能を実行するために使用します。以下のコマンドにはそれぞれ、構文および例があります。

コマンドの機能と制約事項の詳細については、マニュアル・ページを参照してください。PowerHA SystemMirror for AIX コマンドのマニュアル・ページは、`/usr/share/man/info/EN_US/a_doc_lib/cmds/powerha_cmds` ディレクトリーにインストールされています。

コマンドのマニュアル・ページ情報を表示するには、次のコマンドを使用してください。

`man` *command-name*

command-name は、PowerHA SystemMirror コマンドまたはスクリプトの実際の名前です。例えば、 `man clpasswd` と入力して PowerHA SystemMirror **clpasswd** コマンドに関する情報を取得します。

PowerHA SystemMirror コマンドの新機能

PowerHA SystemMirror コマンドのトピック集の中で、新規または大幅に変更された情報を以下に記載しています。

新規情報または変更情報の参照方法

この PDF ファイルでは、左マージンに新規情報と変更情報を識別するリビジョン・バー (|) が表示される場合があります。

2017 年 12 月

このトピック集に対して行われた更新の要約を以下に示します。

- ログ分析機能をサポートするために、 14 ページの『clanalyze コマンド』トピックが追加されました。
- 5 ページの『cl_ezupdate コマンド』 トピックで、ロールバック機能に関する情報が追加されました。
- 以下のトピックが更新されました。
  - 88 ページの『clRGinfo コマンド』
  - 49 ページの『clmgr コマンド』

2017 年 6 月

このトピック集に対して行われた更新の要約を以下に示します。

- 5 ページの『cl_ezupdate コマンド』 トピックが追加されました。

2016 年 12 月

このトピック集に対する更新の要約を以下に示します。

- CAA サービスのみ開始する `only` オプションを `[ START_CAA={no|yes|only} ]` 構文に追加しました。詳しくは、トピック: 49 ページの『clmgr コマンド』を参照してください。
- 欠落していたフラグを以下のトピックに追加しました。
  - 13 ページの『cl_rc.cluster コマンド』

– 106 ページの『rc.cluster コマンド』

cl_clstop コマンド

目的

システム・リソース・コントローラー (SRC) 機能を使用してクラスター・デーモンを停止します。

構文

```
cl_clstop [-cspoc "[-f] [-n NodeList | -g ResourceGroup]"] -f
cl_clstop [-cspoc "[-f] [-n NodeList | -g ResourceGroup]"] -g [-s] [-y] [-N | -R | -B]
cl_clstop [-cspoc "[-f] [-n NodeList | -g ResourceGroup]"] -gr [-s] [-y] [-N | -R | -B]
```

説明

cl_clstop コマンドは、クラスター・ノード全体にわたってクラスター・サービスをシャットダウンします。デフォルトでは、**cl_clstop** コマンドはすべてのクラスター・ノードにわたってクラスター・サービスを停止します。ただし、クラスター・サービスを停止するノードのリストを指定できます。 **cl_clstop** は、System Resource Controller (SRC) を使用してクラスター・デーモンを停止します。クラスター・デーモンを停止する際には、正常終了オプションまたは強制終了オプションのいずれかを使用します。このコマンドは、オプションで、/etc/inittab ファイル内のエントリーによってリブート時の自動開始を解除します。テークオーバーを伴う正常終了オプションを使用してクラスター・デーモンをシャットダウンする場合は、ノード・リストを指定する必要があります。デフォルトでは、**cl_clstop** コマンドは、クラスター内のすべてのノード、またはノード・リスト内のすべてのノードがネットワーク経由でアクセス可能であり、オンラインであることを必要とします。そうでない場合、**cl_clstop** コマンドは失敗します。

フラグ

-cspoc
以下の引数を C-SPOC オプションに使用できます。

-f C-SPOC コマンドにデフォルト検査をスキップするよう強制します。このフラグが設定され、クラスター・ノードがアクセス不可の場合、**cl_clstop** コマンドは警告を出し、他のノードでの実行を続けます。

-n NodeList
ノード・リストで指定されたノード上のクラスター・サービスをシャットダウンします。

-g ResourceGroup
cl_clstop コマンドを実行するリソース・グループに参加しているノードのリストを生成します。

-f シャットダウンを強制します。クラスター・デーモンは、ローカル・プロシージャーを実行することなく終了します。

-g テークオーバーなしで正常終了シャットダウンを行います。

-gr
正常終了シャットダウンを行います。リソースはこのノードによって解放され、他のノードによってテークオーバーされます。デーモンは正常に終了し、ノードはそのリソースを解放します。これらのリソースはテークオーバーされます。テークオーバーを伴う正常終了シャットダウンの場合は、ノード・リストを指定する必要があります。

-s サイレント・シャットダウンを実行します。このフラグは、**wall** コマンドによるシャットダウン・メッセージのブロードキャストを行いません。デフォルト設定では、ブロードキャストが行われます。

-y クラスター・ノードをシャットダウンする前に、オペレーターに確認を求めません。このフラグがデフォルトです。

-B 即時に、および以降のシステム再始動時に停止します。

-N 即時にシャットダウンします。

-R その後のシステム再始動時に停止し、/etc/inittab ファイル内のエントリーを除去します。

例

1. クラスター・プロセスが停止してリソースが解放される前に、警告メッセージをユーザーに送信することなく、node1 でテークオーバー付き正常終了オプションを使用してクラスター・ノードをシャットダウンする (リソースは解放します) には、次のように入力します。

   ```
   cl_clstop -cspoc "-n node1" -ysNgr
   ```

2. クラスター・プロセスが停止する前に、警告メッセージをユーザーにブロードキャストし、すべてのクラスター・ノード上のクラスターを強制的に即時シャットダウンする (リソースは解放しません) には、次のように入力します。

   ```
   cl_clstop -yNf
   ```

3. クラスター・プロセスが停止する前に、ユーザーに警告メッセージをブロードキャストし、全クラスター・ノードでの正常終了オプションを使用してクラスター・ノードをシャットダウンするには、次のように入力します。

   ```
   cl_clstop -yg
   ```

   注**:** -g フラグと -n フラグをいずれも指定しない場合は、すべてのクラスター・ノードに対してデフォルト・アクションが実行されます。

関連資料:

49 ページの『clmgr コマンド』

---

cl_convert コマンド

目的

PowerHA SystemMirror ソフトウェアの最新バージョンへのアップグレードには、以前のリリースから現行リリースへの構成データベースの変換が含まれます。 PowerHA SystemMirror をインストールすると、**cl_convert** が自動的に実行されます。ただし、インストールに失敗した場合は、コマンド・ラインから **cl_convert** を実行する必要があります。 **cl_convert** を実行するには、ルート・ユーザーのアクセス権が必要です。

構文

```
[-F] -v < release> [-s< simulationfile>][-i]
```

説明

このコマンドは、以前のバージョンの ODM データを新しいバージョンの ODM 構造にコピーします。新しいバージョンでフィールドが削除されている場合、そのデータは **/tmp/cl_convert_PowerHA SystemMirror_OLD** に保存されます。その際このコマンドにより、データは新しいバージョンの正しい形式になります。

新しいバージョンがインストールされると、インストール・スクリプトは **/etc/objrepos** ディレクトリーに格納されている PowerHA SystemMirrorxxx クラスに接尾部 OLD を追加し、新しいバージョンの新しい

PowerHA SystemMirrorxxx クラスを作成します。インストール・スクリプトは、PowerHA SystemMirrorxxxOLD 内のデータを PowerHA SystemMirrorxxx 内の対応する新しいクラスに変換する **cl_convert** コマンドを発行します。

コマンド・ラインから **cl_convert** コマンドを実行できますが、このコマンドは、PowerHA SystemMirrorxxx および PowerHA SystemMirrorxxxOLD ODM がすでに存在していると想定します。

-F オプションを指定して **cl_convert** コマンドを実行することもできます。このオプションを指定しない場合、**cl_convert** コマンドは、新しい ODM クラス PowerHA SystemMirror クラスターに構成済みデータがあるかどうかを検査します。データが存在する場合、コマンドは変換を実行せずに終了します。 **-F** オプションを指定した場合、コマンドは現行データを検査せずに処理を続行します。

cl_convert は、最終データを PowerHA SystemMirrorxxx ODM に書き込む前に、PowerHA SystemMirrorxxx および PowerHA SystemMirrorxxxOLD ODM を処理のために一時ファイル (**/tmp/tmpodmdir**) にコピーすることに注意してください。 **cl_convert** で何らかのエラーが検出されると、PowerHA SystemMirrorxxx ODM は上書きされません。エラーがない場合には、PowerHA SystemMirrorxxx ODM は上書きされ、インストール・スクリプトは PowerHA SystemMirrorxxxOLD ODM を除去します。

このコマンドは、次の conversion ディレクトリー内で実行する必要があります。

```
/usr/es/sbin/cluster/conversion
```

また、cl_convert は、ODMDIR に正しい値が設定されていることを前提としています。 cl_convert の結果は、**/tmp/clconvert.log** で確認できます。

フラグ

-F 強制フラグ。既存のエントリー数とは無関係に、cl_convert は既存の ODM オブジェクト・クラスを上書きします。このフラグを省略すると、cl_convert は PowerHA SystemMirror クラスター (以前の構成から常に存在) 内のデータの有無を検査し、データが検出された場合は終了します。

-v リリース・バージョン・フラグ。古いバージョンのリリース番号を示します。

重要**:** 変換元のバージョンを把握していない場合は、 cl_convert コマンドを使用しないでください。

-s <simulation_file>
シミュレーション・フラグ。結果の ODM データを新しい PowerHA SystemMirrorxxx ODM に再度書き込む代わりに、指定されたファイルにテキスト形式で書き込みます。

-i コピー無視フラグ。PowerHA SystemMirrorxxxOLD データを新しい PowerHA SystemMirrorxxx ODM にコピーするのではなく、新しい PowerHA SystemMirrorxxx ODM で直接操作することを指定します。これは主に、clconvert_snapshot で使用されます。

注**:** AIX 環境変数 ODMDIR に、変換したいディレクトリーを設定しておく必要があります。

例

クラスターがすでに前のリリースで構成されている場合は、PowerHA SystemMirror の新規バージョンのインストール中に、インストール・スクリプトが cl_convert を次のように呼び出します。

```
cl_convert -F -v <version of prior release>
```

cl_ezupdate コマンド

目的

通常、現在実行中のワークロードを中断せずに、クラスター全体で PowerHA SystemMirror および AIX のソフトウェア更新を管理します。

構文

```
cl_ezupdate [-v] -h

cl_ezupdate [-v] -Q {cluster|node|nim|lpp} [-N <node1,node2,...>]

cl_ezupdate [-v] {-Q {lpp|all} |-A|-R}
[-U -N <node1:"hdisk1 hdisk2 hdisk3",node2:hdisk2,...>]

cl_ezupdate [-v] {-Q {lpp|all} |-A|-R}
[-U -N <node1:"hdisk1 hdisk2 hdisk3",node2:hdisk2,...>]

cl_ezupdate [-v] {-Q {lpp|all} |-A|-R}
[-U <Multiple –N instances, each giving a "node:hdisk" pair> ...>]
 –s <repository> [-F]
```

説明

cl_ezupdate コマンドを使用すると、現行のクラスター構成および入手可能なソフトウェア更新 (AIX および PowerHA SystemMirror のサービス・パック、暫定修正、およびテクノロジー・レベルなど) に関する情報を照会することができます。また、**cl_ezupdate** コマンドを使用して、更新のインストールのプレビューや、更新の適用またはリジェクトも行えます。

cl_ezupdate ツールを使用するには、各ノードが、ユーザーがインストールしようとする更新にアクセスできる必要があります。更新は、ネットワーク・インストール管理 (NIM) サーバー上または共有ファイル・システム内に置くことができます。

cl_ezupdate ツールにより、各ノード上の使用可能な更新の自動比較が可能です。NIM を使用する場合には、すべてのノードを、同じ `lpp_source` リソースおよびコンテンツにアクセスするよう構成する必要があります。リポジトリーがローカル・ファイル・システム・ディレクトリーである場合、そのローカル・ノードが参照ノードです。ファイルシステムが存在しないか、またはいずれのノードでも空の場合、**cl_ezupdate** ツールにより、ローカル・ファイルのログ・ファイルが自動的にコピーされます。

cl_ezupdate ツールを実行しており、インストールまたはアンインストールのプロセス中にノードでエラーが発生した場合、**cl_ezupdate** ツールのロールバック機能を使用して、ノードを前の状態に戻すことができます。ロールバック機能を使用する際に、そのエラーが発生したノードのみをロールバックするのか、更新されたノードすべてをロールバックするのかを選択できます。

サービス・イメージのインストールまたは削除中にエラーが発生すると、ロールバック・プロセスでは、**alt_disk_copy** コマンドを使用することによって各ノード上で rootvg ボリューム・グループのコピーが作成され、その rootvg ボリューム・グループのコピーがリブートされます。ロールバック・プロセスの場合、rootvg ボリューム・グループのコピーを格納できる各ノードに、**hdisk** が 1 つ存在している必要があります。

フラグ

-A 入手可能な更新を、**–S** フラグで指定されたロケーションで適用します。

-C PowerHA SystemMirror オペレーティング・システムまたは AIX オペレーティング・システムのインストール済みの最新バージョンへのソフトウェア更新をコミットします。

-F サービス・パックのインストールを強制的に行います。暫定修正によりファイルセットがロックされており、更新のインストールが一時停止されている場合、このフラグは、ロックを解除して、サービス・パックをインストールします。

注: このフラグは、**–A** フラグと一緒に使用する必要があります。

-H **cl_ezupdate** コマンドのヘルプ情報を表示します。

-Q ネットワーク・インストール管理 (NIM) セットアップ、クラスター・ソフトウェア、または入手可能な更新の状況を照会します。値のオプションは、cluster、node、nim、または lpp です。

-N 更新のインストール先にするノードの名前を指定します。複数のノード名を指定する場合は、各ノード名をコンマで区切る必要があります。デフォルトでは、更新は、1 つのクラスター内のすべてのノード上にインストールされます。ロールバック機能を有効にするために **-U** フラグまたは **-u** フラグが指定された場合、**-N** フラグは <node name>:hdisk ペアを指定します。ノードに rootvg ボリューム・グループの hdisks が複数個ある場合、ノードを各 hdisks にマップするために、複数の **–N** 引数が必要です。以下に例を示します。

```
-N node1:hdisk1 –N node1:hdisk2 –N node1:hdisk3 –N node2:hdisk1
```

-P クラスター・インストールを preview モードで実行します。preview モードを使用する場合、すべてのインストール前提条件が検査されますが、更新がシステムにインストールされることはありません。

-R **-S** フラグによって指定されたロケーションにインストールおよび保管される、コミットされていないサービス・パックをリジェクトします。

-S インストールされる更新イメージのロケーションを指定します。ファイルシステム名を指定する場合、パスの先頭は、スラッシュ・キー (/) でなければなりません。スラッシュ・キー (/) を指定しない場合、NIM サーバーの lpp_source ロケーションが更新のインストールに使用されます。

-V 拡張ヘルプ情報を表示します。

-I 対話モードを指定します。この値を「**yes**」と指定すると、エラーが示されたときにロールバック機能が機能し続ける必要があります。interactive モードは、デフォルトで、アクティブです。値を「**no**」と指定すると、interactive モードはオフになり、ロールバック操作を開始する前にプロンプトは出されません。

-U 適用操作またはリジェクト操作中にエラーが発生した場合に、すべての変更済みノードのロールバックを使用可能にします。

-u 適用操作またはリジェクト操作中にエラーを検出したノードのみのロールバックを使用可能にします。

-X 各ノード上で **alt_disk_copy** コマンドを使用して **rootvg** ボリューム・グループを作成した後で終了します。**rootgv** ボリューム・グループの代替コピーを、以降の実行でロールバック操作用に使用するには、**-x** 引数を使用する必要があります。

-x ロールバック操作のために、各ノード上で **alt_disk_copy** コマンドを使用して **rootvg** ボリューム・グループのコピーを作成しないよう指定します。**rootvg** ボリューム・グループで障害が発生した場合、**-N** 引数に指定されたディスクをロールバック操作に使用できます。

-T **rootvg** ボリューム・グループのバックアップ操作のタイムアウト値を分数で指定します。指定されたタイムアウト値の前に **rootvg** ボリューム・グループがコピーされなかった場合、操作は終了します。このフラグのデフォルト値は infinite (無限) です。

出力ファイル

cl_ezupdate コマンドからの出力は、**/var/hacmp/EZUpdate/EZUpdate.log** ファイルにキャプチャーされます。

例

1. NIM サーバーに関する情報を表示するには、以下のコマンドを入力します。

   ```
   cl_ezupdate -Q nim
   ```

2. 入手可能な更新の内容を検査および表示するには、以下のコマンドを入力します。

   ```
   cl_ezupdate -Q lpp -S /tmp/lppsource/inst.images
   ```

3. 更新を apply モードでインストールするには、以下のコマンドを入力します。

   ```
   cl_ezupdate -A -S HA_v720_SP1
   ```

4. NIM サーバー上に置かれており、影響を受けるファイルセットが暫定修正でロックされている PowerHA SystemMirror 更新または AIX 更新のインストールを強制的に行うには、以下のコマンドを入力します。

   ```
   Cl_easyupdate –A –F –S HA_v720_SP1
   ```

5. 変更されるすべてのノードが前の rootvg 状態にロールバックされるように、ロールバック機能が有効な状態の apply モードで、NIM サーバー上にある 3 ノード・クラスターのすべてのノード上に更新をインストールするには、以下のコマンドを入力します。

   ```
   cl_ezupdate -A -U Multiple –N arguments are given,node2:hdisk5,node3:hdisk2 -S HA_v720_SP1
   ```

6. 変更されるすべてのノードが前の rootvg 状態にロールバックされるように、ロールバック機能が有効な状態の apply モードで、NIM サーバー上にある 3 ノード・クラスターのすべてのノード上に更新をインストールし、インストール・プロセス中にエラーが発生した場合にユーザーにプロンプトが出されることなくエラー・ノードをロールバックするには、以下のコマンドを入力します。

   ```
   cl_ezupdate -A –X No –U  -N node1:hdisk3,node2:hdisk5,node3:hdisk2 -S HA_v720_SP1
   ```

---

cl_lsfs コマンド

目的

共用ファイルシステムの特性を表示します。

注: 特定のフラグに関連する引数は、フラグの直後に指定しなければなりません。

構文

```
cl_lsfs [-cspoc"[-f] [-g ResourceGroup | -n Nodelist ]" [-q]  [-c | -l] FileSystem ]...
```

フラグ

-cspoc
次のいずれかの C-SPOC オプションを指定するために使用する引数。

-f - このオプションは、**cl_lsfs** コマンドと一緒に使用されると、効果がありません。

-g *ResourceGroup* - このコマンドが実行されるリソース・グループに参加するノードのリストを生成します。

-n *nodelist* - このリスト内のノードでコマンドを実行します。ノードが複数の場合は、リストするノードをコンマで区切ってください。

-c 異なる検索パターンを指定して、基礎となる AIX **lsfs** コマンドがデータを返したかどうかを判別します。

-l 出力をリスト形式にするように指定します。

-q 論理ボリューム・マネージャー (LVM) に論理ボリューム・サイズ (512 バイト・ブロック単位) を照

会し、JFS スーパーブロックにファイルシステム・サイズ、フラグメント・サイズ、圧縮アルゴリズム (ある場合)、および i ノードあたりのバイト数 (nbpi) を照会します。 **lsfs** コマンドにより報告されたファイルシステム特性に加えて、この情報が表示されます。

例

1. クラスター内のすべての共用ファイルシステムに関する特性を表示するには、次のように入力します。

   ```
   cl_lsfs
   ```

2. *resource_grp1* 内の参加ノード間で共用されるファイルシステムに関する特性を表示します。

   ```
   cl_lsfs -cspoc "-g resource_grp1"
   ```

---

cl_lsgroup コマンド

目的

PowerHA SystemMirror クラスターに存在するグループの属性を表示します。

注**:** 特定のフラグに関連する引数は、フラグの直後に指定しなければなりません。

構文

```
cl_lsgroup [-cspoc "[-f] -g ResourceGroup | -n Nodelist"] [-c|-f] [-a | -a List ] {ALL | Group [ ,Group] ... }
```

フラグ

-cspoc
次の C-SPOC オプションを指定するために使用する引数。

-f - このオプションは、**cl_lsgroup** コマンドと一緒に使用されると、効果がありません。

-g *ResourceGroup* - このコマンドが実行されるリソース・グループに参加するノードのリストを生成します。

-n *nodelist* - このリスト内のノードでコマンドを実行します。ノードが複数の場合は、リストするノードをコンマで区切ってください。

-a List
表示する属性を指定します。 *List* パラメーターは **chgroup** コマンドに定義された属性を含むことができます。複数の属性はブランク・スペースで区切る必要があります。-a フラグのみを使用して空のリストを指定すると、グループ名のみがリストされます。

-c 各グループの属性を、次のようにコロンで区切ったレコード・フォーマットで表示します。

```
# name: attribute1: attribute2:...

Group: value1:value2:     ...
```

-f グループ属性をスタンザ・フォーマットで表示します。各スタンザはグループ名で識別されます。次のように、1 つの「属性 = 値」の対が 1 行に表示されます。

```
group:
attribute1=value
attribute2=value
attribute3=value
```

ALL | group [group]...
表示するすべてのリソース・グループ、または 1 つ以上の特定のグループ

例

1. すべてのクラスター・ノードから finance グループの属性を表示するには、次のように入力します。

   ```
   cl_lsgroup finance
   ```

2. すべてのクラスター・ノードから finance グループの ID、メンバー (users)、および管理者 (adms) をスタンザ形式で表示するには、次のように入力します。

   ```
   cl_lsgroup -f -a id users adms finance
   ```

3. すべてのクラスター・ノードから、すべてのグループの属性をコロン区切り形式で表示するには、次のように入力します。

   ```
   cl_lsgroup -c ALL
   ```

---

cl_lslv コマンド

目的

共用論理ボリュームの属性を表示します。

注**:** 特定のフラグに関連する引数は、フラグの直後に指定しなければなりません。

構文

```
cl_lslv [-cspoc "[-f] [-g ResourceGroup | -n Nodelist "] ]  [-l | -m] LogicalVolume
```

フラグ

-cspoc
次のいずれかの C-SPOC オプションを指定するために使用する引数。

-f - このオプションは、**cl_lsfs** コマンドと一緒に使用されると、効果がありません。

-g *ResourceGroup* - このコマンドが実行されるリソース・グループに参加するノードのリストを生成します。

-n *Nodelist* - このリスト内のノードでコマンドを実行します。ノードが複数の場合は、リストするノードをコンマで区切ってください。

-l *Logical Volume*
共用論理ボリューム内の各物理ボリュームに関する情報をリストします。表示されるフィールドの詳細については、 **lslv** コマンドを参照してください。

-m *Logical Volume*
各論理区画に関する情報をリストします。表示されるフィールドの詳細については、**lslv** コマンドを参照してください。フラグを指定しなかった場合は、共用論理ボリュームとその基礎となる共用ボリューム・グループに関する情報が表示されます。表示されるフィールドの詳細については、**lslv** コマンドを参照してください。

例

1. 共用論理ボリューム *lv03* に関する情報を表示するには、次のように入力します。

   ```
   cl_lslv -cspoc -g resource_grp1 lv03
   ```

   論理ボリューム *lv03*、その論理区画および物理区画、および、論理ボリュームが属すボリューム・グループについての情報が表示されます。

2. ID を使用して特定の論理ボリュームに関する情報を表示するには、次のように入力します。

```
cl_lslv -g resource_grp1 00000256a81634bc.2
```

この論理ボリュームについて、使用可能なすべての特性と状況が表示されます。

---

cl_lsuser コマンド

目的

PowerHA SystemMirror クラスター上に存在するユーザーのユーザー・アカウント属性を表示します。

注: 特定のフラグに関連する引数は、フラグの直後に指定しなければなりません。

構文

```
cl_lsuser [-cspoc "[-f] [-g ResourceGroup | -n Nodelist]"]  [-c | -f] [-a List ] {ALL | Name [ ,Name ]... }
```

フラグ

-cspoc
次の C-SPOC オプションを指定するために使用する引数。

-f - このオプションは、**cl_lsuser** コマンドと一緒に使用されると、効果がありません。

-g *ResourceGroup* - このコマンドが実行されるリソース・グループに参加するノードのリストを生成します。

-n *Nodelist* - このリスト内のノードでコマンドを実行します。ノードが複数の場合は、リストするノードをコンマで区切ってください。

-a *Lists*
表示する属性を指定します。リスト変数には **chuser** コマンドに定義された属性を含めることができます。複数の属性はブランク・スペースで区切る必要があります。空のリストを指定した場合は、ユーザー名だけが表示されます。

-c 各ユーザーの属性を、次のようにコロンで区切ったレコード・フォーマットで表示するように指定します。

```
# name:  attribute1:  attribute2:  ...
User:      value1:      value2:      ...
```

-f ユーザー名により識別された各スタンザを、スタンザ・フォーマットで出力するように指定します。次のように、1 つの「属性 = 値」の対が 1 行に表示されます。

```
user:
attribute1=value
attribute2=value
attribute3=value
```

ALL | Name [name]...
すべてのユーザー、または指定した 1 人以上のユーザーに関する情報を表示します。

例

1. すべてのクラスター・ノードから、*smith* アカウントに関するユーザー ID およびグループ関連情報をスタンザ形式で表示するには、次のように入力します。

   ```
   cl_lsuser -fa id pgrp groups admgroups smith
   ```

2. すべてのクラスター・ノードから、ユーザー *smith* のすべての属性をデフォルト形式で表示するには、次のように入力します。

   ```
   cl_lsuser smith
   ```

3. クラスター上にいるすべてのユーザーのすべての属性を表示するには、次のように入力します。

```
cl_lsuser ALL
```

---

cl_lsvg コマンド

目的

共用ボリューム・グループに関する情報を表示します。

注: 特定のフラグに関連する引数は、フラグの直後に指定しなければなりません。

構文

```
cl_lsvg [-cspoc "[-f] [-g ResourceGroup | n- Nodelist ]" [-o] |[-l | -M | -p] Volume Group...INFO HERE
```

フラグ

-cspoc
次のいずれかを指定するために使用する引数。

-f - このオプションは、**cl_lsvg** コマンドと一緒に使用されると、効果がありません。

-g *ResourceGroup* - 参加するノードがボリューム・グループを共用する、リソース・グループの名前を指定します。それらのノードに対してコマンドが実行されます。

-n *Nodelist* - このリスト内のノードでコマンドを実行します。ノードが複数の場合は、リストするノードをコンマで区切ってください。

-p **VolumeGroup** パラメーターにより指定されたグループ内の物理ボリュームごとに、次の情報をリストします。

- Physical volume: グループ内の物理ボリューム

- PVstate : 物理ボリュームの状況

- **Total PPs** : 物理ボリューム上の物理区画の合計数

- Free PPs : 物理ボリューム上の空き物理区画の合計数

- **Distribution** : 物理ボリュームの各セクション (外部端、外部中間部、センター、内部中間部、内部端) 内に割り当てられた物理区画の数。

-l *VolumeGroup* パラメーターによって指定されるグループ内の各物理ボリュームに関する次の情報をリストします。

- LV : ボリューム・グループ内の論理ボリューム

- Type : 論理ボリューム・タイプ

- LPs : 論理ボリューム内の論理区画の数

- PPs: 論理ボリュームによって使用される物理区画の数

- PVs : 論理ボリュームによって使用される物理ボリュームの数

-M 物理ボリューム上のそれぞれの論理ボリュームごとに、次のフィールドをリストします。

- PVname: *PPnum* **[** *LVname* **:** *LPnum* **[:** *Copynum* **] [** *PPstate* **]]**

- **PVname** : システムにより指定された物理ボリュームの名前

- **PPnum** : 物理区画番号。1 から 1016 までの物理区画番号が使えます。

-o アクティブ・ボリューム・グループ (オンに変更されているグループ) のみをリストします。アクティブなボリューム・グループとは、使用できるボリューム・グループのことです。フラグが何も指定されていない場合に表示される情報については、 **lsvg** コマンドを参照してください。

例

1. クラスター内にあるすべての共用ボリューム・グループの名前を表示するには、次のように入力します。

   ```
   cl_lsvg
   nodeA: testvg
   nodeB: testvg
   ```

2. クラスター内にあるすべてのアクティブな共用ボリューム・グループの名前を表示するには、次のように入力します。

   ```
   cl_lsvg -o
   nodeA: testvg
   ```

3. 共用論理ボリューム *vg02* に関する情報を表示するには、次のように入力します。

   ```
   cl_lsvg -cspoc testvg
   ```

---

cl_nodecmd コマンド

目的

指定した一連のノード上で、1 つのコマンドを同時に実行します。

構文

```
cl_nodecmd [-q] [-cspoc "[-f] [-n nodelist | -g resourcegroup ]" ] command args
```

フラグ

-q 抑止モードを指定します。すべての標準出力が抑止されます。

-cspoc
次のいずれかの C-SPOC オプションを指定するために使用する引数。

-f - 強制的に **cl_nodecmd** に PowerHA SystemMirror バージョンの互換性検査とノードのアクセス可能性検証をスキップさせます。

-g *resource group* - このコマンドが実行されるリソース・グループに参加するノードのリストを生成します。

-n *nodelist* - このリスト内のノードでコマンドを実行します。ノードが複数の場合は、リストするノードをコンマで区切ってください。

command
ノード・リスト内のすべてのノードでコマンドを実行するよう指定します。

args
cl_nodecmd コマンドに渡す引数を指定します。

例

1. すべてのクラスター・ノードで **lspv** コマンドを実行します。

   ```
   cl_nodecmd lspv
   ```

2. ノード *beaver* および *dam* で **lsvg rootvg** コマンドを実行し、標準出力を抑止します。

   ```
   cl_nodecmd -cspoc "-n beaver,dam" lsvg rootvg
   ```

cl_rc.cluster コマンド

目的

オペレーティング・システム環境をセットアップし、すべてのクラスター・ノードでクラスター・デーモンを始動します。

構文

```
cl_rc.cluster [-cspoc "[-f] [-g ResourceGroup | -nNodeList ]"] [-boot]
[b] [-i | I] [-N | -R | -B] [-M | -A] [-x] [-r] [-v] [-C interactive|yes]
```

注: 特定のフラグに関連する引数は、フラグの直後に指定しなければなりません。

フラグ

-cspoc
次の C-SPOC オプションを指定するために使用する引数。

-f - 強制的に **cl_rc.cluster** に PowerHA SystemMirror バージョンの互換性検査とノードのアクセス可能性検証をスキップさせます。

-g *ResourceGroup* - 参加するノードがボリューム・グループを共用する、リソース・グループの名前を指定します。それらのノードに対してコマンドが実行されます。

-n *Nodelist* - 基礎となる AIX コマンドをノード・リスト内のノード全体で実行します。

-boot
IPAT が使用可能な場合、サービス・ネットワーク・インターフェースを、そのサービス・アダプターのブート・アドレスに置かれるように構成します。

-i クラスター情報 (**clinfoES**) デーモンをデフォルト・オプションで開始します。

-I トラップを有効にして、クラスター情報 (**clinfoES**) デーモンを始動します。

-b 始動をブロードキャストします。

-N 即座にデーモン (**inittab** の変更なし) を始動します。

-R PowerHA SystemMirror デーモンを、システム再始動時にのみ始動します (PowerHA SystemMirror 始動コマンドが **inittab** ファイルに追加されます)。

-B 即座にデーモンを始動し、PowerHA SystemMirror エントリーを **inittab** ファイルに追加します。

-C 問題が発生した場合の修正アクションに使用するモードを指定します。問題を自動的に修正するには、**yes** を指定してください。修正アクションのそれぞれが実行される前にプロンプトが出されるようにするには、**interactive** を指定してください。

-M 手動リソース獲得モードでクラスター・サービスを始動します。このオプションは、リソース・グループを手動でオンラインにする場合に使用します。

-A 自動リソース獲得モードでクラスター・サービスを始動します。 このオプションは、クラスターの起動時にリソース・グループを自動的にオンラインにする場合に使用します。これはデフォルト・オプションです。

-f 強制起動。クラスター・デーモンは、ローカル・プロシージャーの実行を初期化します。

-r 強制終了したあとで、クラスター・リソースを再獲得します。クラスターが強制終了されているときにいずれかのクラスター・リソース (IP ラベル、ディスク、アプリケーション) の状態を変更した場合は、このオプションを使用します。

-v 起動時 (auto ver sync) に検査エラーを無視します。

-x NFS クロスマウントを活動化します。

例

1. **clinfo** がすべてのクラスター・ノード上で実行している状態でクラスターを始動するには、次のコマンドを実行します。

   ```
   cl_rc.cluster -boot -i
   ```

2. トラップを有効にして、**clinfo** がすべてのクラスター・ノード上で実行している状態でクラスターを始動するには、次のコマンドを実行します。

   ```
   cl_rc.cluster -boot -I
   ```

---

clanalyze コマンド

目的

PowerHA SystemMirror ログ・ファイルにエラーがないか分析して、分析レポートを提供します。

構文

```
clanalyze -a -s <start_time> -e <end_time>
[-n <ALL|node1,node2,...>]

clanalyze -a -s <start_time> -e <end_time>
 -p <Error String> [-n <ALL|node1,node2,...>]
clanalyze -a -p <Error String>
[-n <ALL|node1,node2,...>]
clanalyze -a -o <all|recent>
 [-n <ALL|node1,node2,...>]
clanalyze -a -o <all|recent>
 -d <PATH of snap file>

clanalyze -a -p <Error String>
 -d <PATH of snap file>

clanalyze -a -s <start_time> -e <end_time>
 -p <Error String> -d <PATH of snap>

clanalyze -a -s <start_time> -e <end_time>
 -d <PATH of snap file>

clanalyze -a -u [-n <ALL|node1,node2,...>]

clanalyze -s <start_time> -e <end_time>
 -f <Path of log file> [-n <ALL|node1,node2,...>]

clanalyze -s <start_time> -e <end_time>
 -x <Path of log file> -d <Path of snap file>

clanalyze -c <Path to copy snap>

clanalyze -v [-n <ALL|node1,node2,...>]
```

説明

clanalyze コマンドは、以下のタスクを実行します。

- ログ・ファイルを分析し、エラー・ストリングまたはタイム・スタンプに基づいてエラー・レポートを提供します。
- AIX エラー・ログからコア・ダンプ・ファイルを分析します。

- **snap** ユーティリティーおよび **clsnap** ユーティリティーを使用して収集されたログ・ファイルを分析します。
- 指定されたエラー・ストリングに基づいてユーザー指定の **snap** ファイルを分析して、レポートを生成します。

フラグ

-a すべてのログ分析操作が、**-a** フラグによって実行されます。このフラグに引数は不要です。

-c <path to copy snap file>
ログ・ファイルを、ユーザーが指定したディレクトリーにコピーします。

-d 分析または抽出が行なわれる **snap** ファイルを指定します。

注**:** このフラグは、**–a** フラグと一緒に使用する必要があります。

-e ログ分析または抽出操作の終了時刻を表示します。形式は `YYYY-MM-DDTHH:MM:SS` です。

注**:** 次の例にある大文字 `T` は、フィールドの日付部分と時刻部分を区切っています。例えば、`2017-04-28T11:45:00` のようになります。

-f ライブ・クラスターから抽出されたログ・ファイルを表します。

-n 分析、抽出、または検証の一部であることが必要なクラスターのノード名を指定します。これは、名前または「**all**」のコンマ区切りリストである場合があります。「**all**」を指定した場合、クラスターのすべてのアクティブなノードが分析対象になります。

-o 分析を適用します。このオプションは、「**all**」または「**recent**」として指定できます。「**all**」オプションを指定した場合、サポート対象のすべてのエラーについて、エラー分析が実行されます。「**recent**」を指定した場合は、サポート対象の最後のエラーが表示されます。

-p <error string>
指定のエラー・ストリングに基づいて分析を実行します。例えば、ストリングとして `Diskfailure` が指定された場合、分析は、ディスク障害エラーについて実行されます。

-s ログ分析または抽出の開始時刻を指定します。形式は `YYYY-MM-DDTHH:MM:SS` です。例えば、`2017-04-28T11:45:00` のようになります。

-u **errpt** ログを分析し、コア・ダンプに関連するログ情報をフィルタリングします。

-v **syslogd** や **errdaemon** などのデーモンの状況および構成を検査します。**v** フラグに、現在の状況および構成が示されます。

-x **snap** ファイルから抽出する必要のあるファイル名を指定します。

出力ファイル

clanalyze コマンドからの出力は、**/var/hacmp/log/loganalyzer/loganalyzer.log** ファイルに保管されます。

例

1. **clanalyze** コマンドは、すべての主要ログ・ファイルを分析し、開始時刻から終了時刻までに発生したイベントまたはエラーのレポートを表示します。この分析は、指定されたノードのログ・ファイルに対して実行されます。属性 `node` のデフォルト値は **all** です。指定の開始時刻から終了時刻までに発生したすべてのエラーまたはイベントのログ・ファイルを分析するには、以下のコマンドを入力します。

```
clanalyze -a -s "2017-04-28T13:45:00" -e "2017-04-28T13:45:00"
 [-n ALL|node1|node2]
```

2. **clanalyze** コマンドは、すべての主要ログ・ファイルについての指定されたエラーまたはイベントに対して分析を実行します。**clanalyze** コマンドは、広範囲に及ぶ検索を実行し、時間制限が適用不能であるためにエラーまたはイベントを分析します。特定のエラーまたはイベントについてログ・ファイルを分析するには、以下のコマンドを入力します。

   ```
   clanalyze -a -p "Disk failure" [-n ALL|node1|node2]
   ```

3. **clanalyze** コマンドは、該当するすべてのログ・ファイルに対して、ある期間内に発生した特定のエラーについて検索および分析を実行します。検索および分析は、特定のノードやクラスター内のすべてのノードのすべてのログ・ファイルに対して実行されます。ある期間内に発生したすべてのエラーまたはイベントのログ・ファイルを分析するには、以下のコマンドを入力します。

   ```
   clanalyze -a -s "2017-04-28T13:45:00" -e "2017-04-28T13:45:00"
   -p "Disk failure" [-n ALL|node1|node2]
   ```

4. 該当するすべてのエラー、または最新のエラーのみのログ・ファイルを分析するには、以下のコマンドを入力します。

   ```
   clanalyze -a -o "all/recent" [-n ALL|node1|node2]
   ```

5. コア・ダンプ固有のデータを **errpt** ログ・ファイルから抽出してそれを表示するには、以下のコマンドを入力します。

   ```
   clanalyze -a -u [-n ALL|node1|node2]
   ```

6. **snap** ファイルまたは **tar** ファイルの指定の開始時刻から終了時刻までに発生したすべてのエラーまたはイベントのログ・ファイルを分析するには、以下のコマンドを入力します。

   ```
   clanalyze -a -s "2017-04-28T13:45:00" -e "2017-04-28T13:45:00" -d <PATH of snap>
   ```

7. **clanalyze** コマンドは、すべての主要ログ・ファイルでのエラーまたはイベントに対して検索および分析を実行します。特別なエラーまたはイベントについてログ・ファイルを分析するには、以下のコマンドを入力します。

   ```
   clanalyze -a -p "Error String" -d <PATH of snap>
   ```

8. **snap** ファイルまたは **tar** ファイルについて指定の期間内に発生したすべてのエラーまたはイベントのログ・ファイルを分析するには、以下のコマンドを入力します。

   ```
   clanalyze -a -s "2017-04-28T13:45:00" -e "2017-04-28T13:45:00"
   -p "Error String" -d <PATH of snap>
   ```

9. あるクラスターのすべてのノードからログ・ファイルをコピーし、それらをリモート・ロケーションに保管するには、以下のコマンドを入力します。

   ```
   clanalyze -c /tmp/CLANALYZE
   ```

10. 特定のノード上のいくつかのデーモン (**syslogd** または **errdemon** など) の状況を調べるには、以下のコマンドを入力します。

    ```
    clanalyze -v [-n ALL|node1|node2]
    ```

11. ファイル **tar**、**.pax**、**.gz**、または **.Z** から特定のログ・ファイルを抽出するには、以下のコマンドを入力します。

    ```
    clanalyze -s <start_time> -e <end_time> -x <file_name> -d <PATH of snap>
    ```

    clanalyze コマンドは、**.tar** ファイルを入力として受け取り、特定の時刻範囲の間、ユーザー指定ファイルを抽出します。

12. タイム・スタンプのために、ライブ・ノードから特定のログ・ファイルを抽出するには、以下のコマンドを入力します。

    ```
    clanalyze -s <start_time> -e <end_time> -f <file_name> [-n ALL|node1|node2]
    ```

注:

- ログ・ファイルでデータが表示されない場合、このツールでは完全なデータが生成されない可能性があります。
- 開始時刻および終了時刻の形式は、YYYY-MM-DDTHH:MM:SS です。例: `2017-04-28T13:45:00`
- ログ・ファイル名は絶対パスでなければなりません。
- PowerHA SystemMirror ログ・ファイルはすべて、デフォルト・ディレクトリー内になくてはならず、他のディレクトリーに宛先指定されていてはなりません。
- **clanalyze** コマンドは、稼働環境分析とスナップ・ユーティリティーの両方について、PowerHA SystemMirror 7.2.2 以降でのみ機能します。
- ログ分析には、以下のエラー・ストリングがサポートされます。
  - Diskfailure
  - Interfacefailure
  - Networkfailure
  - Globalnetworkfailure
  - Nodefailure
  - Sitefailure

clconvert_snapshot コマンド

目的

このコマンドは、以前のバージョンの snapshot_file の ODM データを新しいバージョンの ODM 構造の形式にコピーします。

構文

`clconvert_snapshot -v` *`release`* `-s <` *`snapshotfile`* `>`

説明

clconvert_snapshot を実行することにより、クラスター・スナップショットを、前のバージョンの PowerHA SystemMirror から最新バージョンの PowerHA SystemMirror にアップグレードすることができます。このコマンドはデフォルトで、最新バージョンのソフトウェアに変換することを想定します。

新しいバージョンでフィールドが削除されている場合、そのデータは */tmp/cl_convert_PowerHA SystemMirror_OLD* に保存されます。その際このコマンドにより、データは新しいバージョンの正しい形式になります。

スナップショット・ファイルがアップグレードされると、以前のバージョンと同じ名前が割り当てられるので、以前のバージョンに復帰できなくなります。スナップショットの古いバージョンのコピーは、元の名前に *.old* 拡張子を付けて保管されます。

スナップショットを作成したのと同じノード上の **/usr/es/sbin/cluster/conversion** ディレクトリーで、`clconvert_snapshot` コマンドを実行する必要があります。

スナップショット・ファイルがアップグレードされ、クラスター内のすべてのノードに現在のレベルがインストールされると、アップグレードされたスナップショットを適用して、クラスターを始動できるようになります。

スクリプト clconvert_snapshot は、古いバージョンの ODM を作成し、ユーザーが指定したスナップショット・ファイルの値をその ODM に取り込みます。続いて、 *cl_convert* がそれらの ODM を現行バージョンに変換するために使用するものと同じコマンドを呼び出します。アップグレードされた ODM の新しいスナップショットが作成され、ユーザーが指定したスナップショット・ファイルにコピーされます。

clconvert_snapshot はインストール時に自動的に実行されないため、必ずコマンド・ラインから実行する必要があります。

表 *1. clconvert_snapshot* フラグ

フラグ	説明
-v	リリース・バージョン・フラグ。変換元のリリース番号を指定します。 **重要:** 変換元のバージョンを把握していない場合は、clconvert_snapshot コマンドを使用しないでください。
-s	スナップショット・ファイル・フラグ。変換するスナップショット・ファイルを指定します。スナップショット・ファイルのパスを指定しない場合、コマンドは **$SNAPSHOTPATH** 変数に指定されたパスを使用します。デフォルトは、**/usr/es/sbin/cluster/snapshots** です。

例

以下のコマンドを実行して、PowerHA SystemMirror 5.3 スナップショットを「mysnapshot」という名前の現行の PowerHA SystemMirror スナップショットに変換します。

```
clconvert_snapshot -v 5.3 -s mysnapshot
```

この「mysnapshot」ファイルはその後、**$SNAPSHOTPATH** 環境変数によって指定されたディレクトリーに配置されます。 **$SNAPSHOTPATH** 変数が指定 されていない 場合、このファイルは **/usr/es/sbin/cluster/snapshots** に配置されます。

---

clcheck_server コマンド

目的

PowerHA SystemMirror クラスター内のデーモンの状況を返します。

構文

```
clcheck_server daemon
```

説明

clcheck_server コマンドは、指定したデーモンの状況を返します。このコマンドは、デーモンの状況を確実に判別する必要があるシェル・スクリプト内で使用することを目的としています。このコマンドは、System Resource Controller (SRC) で提供される **lssrc** コマンドによって行われる以上の追加検査を行います。

clcheck_server コマンドを使用する前に、検査対象となるデーモンの目的を理解しておく必要があります。

フラグ

daemon
検査したいデーモンの名前を指定します。

例

clinfo デーモンの状況を確認するには、次のように入力します。

```
    if ! clcheck_server clinfoES
    then
 echo "clinfo is active"
    else
 echo "clinfo is inactive"
    fi
```

clfindres コマンド

目的

特定のリソース・グループまたはグループをクラスター構成内で検索します。

構文

```
clfindres [-s] [resgroup1] [resgroup2]...
```

説明

clfindres を実行すると **clRGinfo** が呼び出され、 **clfindres** のコマンド出力が **clRGinfo** コマンドの出力と同じになります。 したがって、 **clRGinfo** コマンドは、リソース・グループの状況と場所を検索するために使用してください。 **clfindres** コマンドの **-s** フラグは、省略した (ロケーションのみの) 出力を要求します。詳しくは、**clRGinfo** コマンドを参照してください。

clgetactivenodes コマンド

目的

すべてのクラスター・ノードの名前を取り出します。

構文

```
clgetactivenodes [-n nodename ] [-o odmdir ] [-ttimeout ]  [-v verbose ]
```

表 2. *clgetactivenodes* フラグ

フラグ	説明
-n nodename	指定したノードがアクティブであるかどうかを判別できます。
-o odmdir	ODM オブジェクト・リポジトリーのディレクトリーとして、デフォルトの **/etc/objrepos** ではなく、 *odmdir* を指定します。
-t timeout	アクティブ・ノードに関する情報を受信する最大時間間隔を指定します。
-v verbose	アクティブ・ノードに関する情報を詳細出力として表示することを指定します。

例

以下のコマンドを実行して、ノード *java* がアクティブであることを検証します。

```
clgetactivenodes -n java
```

clgetaddr コマンド

目的

指定したノード名の ping 可能なアドレスを返します。

構文

```
clgetaddr [-o odmdir ] nodename
```

-o 代替 ODM ディレクトリーを指定します。

例

ノード *seaweed* の PINGable アドレスを取得するには、次のように入力します。

```
clgetaddr seaweed
```

次のアドレスが戻されます。2361059035

cli_assign_pvids コマンド

目的

引数として渡される各ディスクに PVID を割り当て、その他すべてのクラスター・ノードをこれらの PVID で更新します。

構文

```
cli_assign_pvids  PhysicalVolume ...
```

説明

論理ボリューム・マネージャー (LVM) は、リスト内の各物理ボリュームに PVID を割り当て (まだ存在しない場合)、これらの PVID をすべてのクラスター・ノード上で認識されるようにします。

例

PVID をディスクのリストに割り当て、これらの PVID をクラスター全体で認識されるようにするには、次のように入力します。

```
cli_assign_pvids hdisk101 hdisk102 hdisk103
```

cli_chfs コマンド

目的

クラスター内のすべてのノード上のファイルシステムの属性を変更します。

構文

```
cli_chfs [ -m NewMountPoint ] [ -u MountGroup ] [ -p { ro | rw  } ]
         [ -t { yes | no } ] [ -a Attribute=Value ] [ -d Attribute ]
   FileSystem
```

説明

C-SPOC を使用してパラメーター指定で **chfs** コマンドを実行し、すべてのクラスター・ノード上のファイルシステム定義を更新します。

フラグ

-d Attribute
指定した属性を指定したファイルシステムの /etc/filesystems ファイルから削除します。

-m NewMountPoint
指定のファイルシステムに新しいマウント・ポイントを指定します。以下の値が有効です。

-p ファイルシステムの許可を設定します。以下の値が有効です。

ro 読み取り専用許可を指定します。

rw 読み取り/書き込み許可を指定します。

-t 指定したファイルシステムのアカウンティング属性を設定します。以下の値が有効です。

yes
ファイルシステム・アカウンティングはアカウンティング・サブシステムによって処理されます。

no ファイルシステム・アカウンティングはアカウンティング・サブシステムによって処理されません。これがデフォルト値です。

-u MountGroup
マウント・グループを指定します。マウント・グループは、マウントを個々に行うのではなく、1 つのグループとして行うことができるように、関連マウントをグループ化するために使用します。例えば、あるテストを実行する際に、いくつかのスクラッチ・ファイルシステムを一緒にマウントする必要がある場合は、それらのファイルシステムをテスト・マウント・グループに入れることができます。このマウント・グループは、**mount -t** のような 1 つのコマンドでマウントできます。

-a Attribute=Value
仮想ファイルシステム・タイプに応じて、Attribute=Value の対を指定します。複数の Attribute=Value の対を指定する場合は、複数の -a Attribute=Value パラメーターを指定します。

例

/test_fs という名前の共用ファイルシステムのサイズを変更するには、次のように入力します。

```
cli_chfs -a size=32768 /test_fs
```

関連情報:

chfs コマンド

---

cli_chlv コマンド

目的

クラスター内のすべてのノード上の論理ボリュームの属性を変更します。

構文

```
cli_chlv  [-a Position] [-b BadBlocks] [-d Schedule] [-e Range]
          [-L label] [-p Permission] [-r Relocate] [-s Strict]
          [-t Type] [-u Upperbound] [-v Verify] [-w MirrorWriteConsistency]
          [-x Maximum] [-U userid] [-G groupid] [-P modes] LogicalVolume
```

説明

C-SPOC を使用して、指定のパラメーターで **chlv** コマンドを実行し、すべてのクラスター・ノード上の論理ボリューム定義を更新します。

フラグ

-a Position
物理ボリュームの割り当てポリシー (物理ボリューム上の論理区画の位置) を設定します。以下の *Position* 変数が有効です。

- **m** 各物理ボリュームの外側の中央部分に論理区画を割り当てます。この変数がデフォルト設定です。
- **c** 各物理ボリュームの中央部分に論理区画を割り当てます。
- **e** 各物理ボリュームの外側の端の部分に論理区画を割り当てます。
- **ie** 各物理ボリュームの内側の端の部分に論理区画を割り当てます。
- **im** 各物理ボリュームの内部中央部分に論理区画を割り当てます。

-b BadBlocks
不良ブロック再配置ポリシーを設定します。 以下の *BadBlocks* 変数が有効です。

- **y** 不良ブロックの再配置を実行します。
- **n** 不良ブロックの再配置の実行を禁止します。

-d Schedule
2 つ以上の論理区画が書き込まれるときに、スケジューリング・ポリシーを設定します。ストライピングされた論理ボリュームをミラーリングするには、並列処理または順次処理を使用する必要があります。以下の *Schedule* 変数が有効です。

- **p** 並列スケジューリング・ポリシーを設定します。
- **ps** 並列書き込み/順次読み取りポリシー。ミラーはすべて並列に書き込まれますが、最初のミラーが使用可能であれば必ず最初のミラーから読み取られます。
- **pr** 並列書き込みおよび並列読み取りがすべてのミラーに対して行われます。このポリシーは、すべてのミラーにまたがってさらに論理ボリュームへの読み取りを広げようとする試みを除いては、並列ポリシーと類似しています。
- **s** 順次スケジューリング・ポリシーを設定します。この変数は、並列または順次の厳密性 (極度の厳密性) のポリシーを指定する場合に使用します。

-e Range
物理ボリュームの割り当てポリシーを設定します。割り当てポリシーとは、割り当てが最適になるボリュームを使用して拡張する物理ボリュームの数です。 *Range* 変数の値は、**-u** フラグで設定される *Upperbound* 変数によって制限されます。以下の *Range* 変数が有効です。

- **x** 物理ボリュームの最大数に論理区画を割り当てます。
- **m** 物理ボリューム間の最小数に論理区画を割り当てます。

-G Groupid
論理ボリューム・スペシャル・ファイルのグループ ID を指定します。

-L Label
論理ボリューム・ラベルを設定します。この変数の最大サイズは 127 文字です。

-n NewLogicalVolume
NewLogicalVolume 変数で指定された論理ボリュームの名前を変更します。論理ボリューム名はシステム全体で固有でなければならず、15 文字まで指定できます。

-p Permission
アクセス権を読み取り/書き込み、または読み取り専用に設定します。以下の *Permission* 変数が有効です。

w アクセス権を読み取り/書き込みに設定します。

r アクセス権を読み取り専用に設定します。読み取り専用論理ボリュームへの JFS ファイルシステムのマウントは、サポートされません。

-P Modes
論理ボリューム・スペシャル・ファイルのアクセス権 (ファイル・モード) を指定します。

-r Relocate
再編成中に論理ボリュームの再配置を許可するか禁止するかを指定します。以下の *Relocate* 変数が有効です。

y 再編成時の論理ボリュームの再配置を許可します。論理ボリュームがストライピングされている場合、**chlv** コマンドを使用して再配置フラグを y に変更することはできません。

n 再編成時の論理ボリュームの再配置を禁止します。

-s Strict
厳密な割り当てポリシーを決定します。 同じ物理ボリュームを共用するように、または共用しないように、論理区画のコピーを割り当てることができます。以下の *Strict* 変数が有効です。

y 厳密な割り当てポリシーを設定します。したがって、論理区画のコピーは、同一物理ボリュームを共用することはできません。

n 厳密な割り当てポリシーを設定しません。したがって、論理区画のコピーは、同一物理ボリュームを共用できます。

s 非常に厳密な割り当てポリシーを設定します。したがって、1 つのミラーに割り当てられた区画は、別のミラーからの区画と物理ボリュームを共用することができません。非超厳密論理ボリュームを超厳密論理ボリュームに変更する場合は、**-u** フラグを使用する必要があります。

-t Type
論理ボリューム・タイプを設定します。最大サイズは、31 文字です。論理ボリュームがストライピングされている場合、*Type* 変数を boot に変更することはできません。

-U Userid
論理ボリューム・スペシャル・ファイルのユーザー ID を指定します。

-u Upperbound
新たに割り当てるために物理ボリュームの最大数を設定します。*Upperbound* 変数の値は、1 から物理ボリュームの総数の範囲内です。極度の厳密さを使用する場合、上限は、ミラー・コピーごとに許される物理ボリュームの最大数を示します。ストライピングされた論理ボリュームを使用するときは、上限は *Stripe_width* 変数の倍数である必要があります。

-v Verify
論理ボリュームに対する書き込み検査状態を設定します。これにより、論理ボリュームに対するすべての書き込みは、書き込み後の読み取りによる検査を受けるか、受けないかのどちらかになります。以下の *Verify* 変数が有効です。

y 論理ボリュームに対する書き込みはすべて、書き込み後の読み取りにより検査されます。

n 論理ボリュームに対する書き込みはすべて、書き込み後の読み取りにより検査されません。

-w MirrorWriteConsistency
以下の *MirrorWriteConsistency* 変数が有効です。

y アクティブ・ミラー書き込み整合性の保持をオンにします。この変数は、通常の入出力処理中に、論理ボリュームのミラー・コピーでのデータ整合性を検査します。

p パッシブ・ミラー書き込み整合性の保持をオンにします。この変数は、システム割り込み後のボリューム・グループの同期化中に、ミラー・コピーでのデータ整合性を検査します。この機能はビッグ・ボリューム・グループでのみ使用できます。

n ミラー書き込み整合を行いません。

-x Maximum
論理ボリュームに割り当てることのできる論理区画の最大数を設定します。 論理ボリュームあたりの論理区画の最大数は 32,512 です。

例

lv01 という名前の論理ボリュームの物理ボリューム割り当てを変更するには、次のように入力します。

```
cli_chlv -e m lv01
```

関連情報:

chlv コマンド

---

cli_chvg コマンド

目的

クラスター内のすべてのノード上のボリューム・グループの属性を変更します。

構文

```
cli_chvg  [ -s Sync { y | n }] [ -L LTGSize ] [ -Q { n | y } ] [ -u ]
          [ -t [factor ] ] [ -M { y | n | s } ] [ -B ] [ -C  ] VolumeGroup
```

説明

C-SPOC を使用して、指定のパラメーターで **chvg** コマンドを実行し、更新されたボリューム・グループ定義をすべてのクラスター・ノード上で使用可能にします。

フラグ

-B ボリューム・グループを大容量ボリューム・グループ形式に変更します。 このフラグは、最大 128 の物理ボリュームおよび 512 の論理ボリュームに対応できます。クラスター・ノードに不整合の物理区画がある場合は、このフラグを使用できません。このフラグを使用するには、VGDA 拡張用の各物理ボリュームで、十分な空き区画が使用可能であることが必要です。

VGDA はディスクのエッジにあり、拡張には連続スペースを必要とするため、ディスクのエッジに空き区画が必要です。これらの区画は、アプリケーション・データ用に割り当てられている場合、同じディスク上の他の空き区画に移行されます。残りの物理区画は、VGDA 使用のために区画が減少したことを反映するために、番号が振り直されます。この処理により、ボリューム・グループ内のすべての物理ボリュームで論理区画から物理区画へのマッピングが変更されます。

将来必要になる可能性のある回復操作に備えて論理ボリュームのマッピングを保存した場合は、変換操作の完了後にマップを再度生成できます。マップ・オプションを指定してボリューム・グループのバックアップを取り、それらのマップを使用した復元を予定している場合、区画番号が (減少によって) 存在しない可能性があるため、復元操作は失敗するおそれがあります。マップ・オプションを使用する場合は、変換処理の開始前および終了直後に論理ボリュームのバックアップを取ることをお勧めします。

VGDA スペースは大幅に増加しているため、どの VGDA 更新操作 (論理ボリュームの作成、論理ボリュームの変更、物理ボリュームの追加など) も、実行にかなり時間がかかる可能性があります。

-C ボリューム・グループを拡張コンカレント機能付きボリューム・グループに変更します。ボリューム・グループを非コンカレント・モード (オンに変更済み) から拡張コンカレント・モードに変更します。この処理では、拡張コンカレント・モードを活動化する前に、ボリューム・グループを他のすべてのノードで再インポートすることが必要です。ボリューム・グループをコンカレント・モード (オンに変更済み) から拡張コンカレント・モードに変更します。このフラグは PowerHA SystemMirror クラスターでのみ使用でき、そのクラスターは拡張コンカレント・ボリューム・グループを活動化する前に構成しておく必要があります。

-L LTGSize
ボリューム・グループがオンに変更済みである場合に、論理トラック・グループ・サイズをディスクの共通最大転送サイズに設定します。 *LTGSize* 変数の値は、0、128、256、512、または 1024 でなければなりません。*LTGSize* 変数は、ボリューム・グループ内のすべてのディスクの最大転送サイズ以下でなければなりません。 *LTGSize* 変数のデフォルト値は 128 です。*LTGSize* に 0 を指定した場合、**varyonvg** コマンドは論理トラック・グループ・サイズをディスクの共通最大転送サイズに設定します。

-M ボリューム・グループのミラー・プール構造を変更します。以下の値が有効です。

y ボリューム・グループ内に作成された各論理ボリューム・コピーをミラー・プールに割り当てる必要があります。

n ミラー・プールのユーザーに制限は設けられません。このオプションがデフォルト値です。

s ボリューム・グループに超厳密なミラー・プールが強制されます。

注**:** ローカル物理ボリュームおよびリモート物理ボリュームは同一のミラー・プールに所属できません。1 つのボリューム・グループは最大 3 つのミラー・プールで構成できます。各ミラー・プールには、ボリューム・グループ内の各論理ボリュームのコピーが少なくとも 1 つ含まれていなければなりません。

-Q 物理ボリュームのクォーラム (定足数) を失った後でボリューム・グループを自動的にオフに変更するかどうかを決定します。デフォルト値は yes です。変更は、ボリューム・グループを次回に活動化したときに有効になります。以下の値が有効です。

y ボリューム・グループは、物理ボリュームのクォーラムを失うと自動的にオフに変更されます。

n ボリューム・グループは、物理ボリュームをすべて失うまでアクティブです。

-s Sync
VolumeGroup 変数で指定されたボリューム・グループの同期特性を設定します。このフラグは、ミラーリングされていない論理ボリュームには影響しません。このフラグは、コンカレント機能付きボリューム・グループに対してはサポートされません。

自動同期は、論理ボリューム・マネージャー (LVM) デバイス・ドライバーが LVM_SA_STALEPP を AIX オペレーティング・システム・エラー・ログに記録した後にのみ試行される回復メカニズムです。他のパス (例えば、**mklvcopy** コマンド) によって不整合になった区画は、自動的には再同期されません。以下の値が有効です。

y 不整合区画を自動的に同期化しようとします。

n 不整合区画の自動同期を禁止します。この値がボリューム・グループのデフォルト設定です。

-t factor
factor で指定される、物理ボリュームあたりの物理区画数の制限を変更します。 32 物理ボリューム・グループの場合、factor は 1 から 16 の範囲内でなければなりません。 128 物理ボリューム・グループの場合、factor は 1 から 64 の範囲内でなければなりません。

factor を指定しない場合、factor には、ボリューム・グループ内の最大ディスクの物理区画数が、factor 値に 1016 を乗算した値より小さくなるような最小値が設定されます。

factor を指定した場合、ボリューム・グループ用の物理ボリュームあたりの物理区画の最大数は、factor 値に 1016 を乗算した値に変更されます。

factor の値を決定する際には、以下の情報を確認してください。

- このフラグは、スケーラブル・タイプのボリューム・グループでは無視されます。
- ボリューム・グループがコンカレント・モードでオンに変更された場合、このフラグは使用できません。
- ボリューム・グループに不整合の物理区画がある場合、factor 値は変更できません。
- このボリューム・グループで許可されている物理ボリュームの最大数は、MAXPVS を factor 値で除算した値 (MAXPVS/factor) に削減されます。
- 既存のボリューム・グループをスケーラブル・ボリューム・グループ形式に変更すると、関連するすべての論理ボリュームのデバイス・サブタイプ (`IOCINFO ioctl()` 呼び出しによって報告される) が、以前のサブタイプにかかわらず *DS_LVZ* に変更されます。この変更によって、報告されたサブタイプ以外の論理ボリュームの動作が変更されることはありません。

-u ボリューム・グループをアンロックします。このフラグは、別の LVM 操作の異常終了によってボリューム・グループがロック状態のままになっている場合 (コマンド・コア・ダンプまたはシステム・クラッシュなど) に使用できます。このフラグを使用するには、ボリューム・グループが別の LVM コマンドで使用されていないことを事前に確認する必要があります。

例

vg01 という名前のボリューム・グループのクォーラムをオフにするには、次のように入力します。

```
cli_chvg -Q n vg01
```

関連情報:

chvg コマンド

cli_crfs コマンド

目的

新しいファイルシステムを作成し、クラスター内のすべてのノードで使用可能にします。

構文

```
cli_crfs  -v VfsType { -g VolumeGroup | -d Device } [ -l LogPartitions ]
    -m MountPoint [ -u MountGroup ] [ -A { yes | no } ]
    [ -p {ro | rw } ] [ -a Attribute=Value ... ] [ -t { yes | no } ]
```

説明

C-SPOC を使用して、指定のパラメーターで **crfs** コマンドを実行し、更新されたファイルシステム定義をすべてのクラスター・ノード上で使用可能にします。

フラグ

-a Attribute=Value
仮想ファイルのシステム依存属性および値の組を指定します。属性/値の組を複数指定するには、複数の -a Attribute=Value パラメーターを指定します。

-d Device
ファイルシステムを作成するデバイスまたは論理ボリュームのデバイス名を指定します。このフラグは、既存の論理ボリューム上にファイルシステムを作成するために使用します。

-g VolumeGroup
ファイルシステムを作成する既存のボリューム・グループを指定します。ボリューム・グループは、1 つ以上の物理ボリュームの集合です。

-l LogPartitions
ログ論理ボリュームのサイズを論理区画数で指定します。このフラグは、ログ・デバイスを持たない JFS ファイルシステムおよび JFS2 ファイルシステムにのみ適用されます。

-m MountPoint
ファイルシステムを使用できるようにするディレクトリー、つまりマウント・ポイントを指定します。相対パス名を指定した場合、/etc/filesystems ファイルに挿入される前に絶対パス名に変換されます。

-p ファイルシステムの許可を設定します。

ro 読み取り専用許可

rw 読み取り/書き込み許可

-t ファイルシステムがアカウンティング・サブシステムによって処理されるかどうかを指定します。以下の値が有効です。

yes
ファイルシステムでアカウンティングが使用可能です。

no ファイルシステムでアカウンティングが使用可能ではありません。この値がデフォルト設定です。

-u MountGroup
マウント・グループを指定します。

-v VfsType
仮想ファイルシステムのタイプを指定します。 *agblksize* 属性はファイルシステムの作成時に設定さ

れ、ファイルシステムの作成後に変更することはできません。 *size* 属性は最小ファイルシステム・サイズを定義します。ファイルシステムの作成後に *size* 属性を使用してファイルシステム・サイズを小さくすることはできません。

例

lv01 という名前の既存の論理ボリューム上に JFS ファイルシステムを作成するには、次のように入力します。

```
cli_crfs -v jfs -d lv01 -m /tstvg -a 'size=32768'
```

関連情報:

crfs コマンド

---

cli_crlvfs コマンド

目的

新しい論理ボリュームおよびファイルシステムを作成し、それらをクラスター内のすべてのノードで使用可能にします。

構文

```
cli_crlvfs -v VfsType -g VolumeGroup  [ -l LogPartitions ] -m MountPoint
        [ -u MountGroup ] [ -A { yes | no } ] [ -p {ro | rw } ]
  [ -a Attribute=Value ... ] [ -t { yes | no } ]
```

説明

C-SPOC を使用して、指定のパラメーターで **crfs** コマンドを実行し、更新されたファイルシステム定義をすべてのクラスター・ノード上で使用可能にします。

フラグ

-a Attribute=Value
仮想ファイルのシステム依存属性および値の組を指定します。属性/値の組を複数指定するには、複数の -a Attribute=Value パラメーターを指定します。

-g VolumeGroup
ファイルシステムを作成する既存のボリューム・グループを指定します。ボリューム・グループは、1 つ以上の物理ボリュームの集合です。

-l LogPartitions
ログ論理ボリュームのサイズを論理区画数で指定します。このフラグは、ログ・デバイスを持たない JFS ファイルシステムおよび JFS2 ファイルシステムにのみ適用されます。

-m MountPoint
ファイルシステムを使用できるようにするディレクトリー、つまりマウント・ポイントを指定します。相対パス名を指定した場合、/etc/filesystems ファイルに挿入される前に絶対パス名に変換されます。

-p ファイルシステムの許可を設定します。

ro 読み取り専用許可

rw 読み取り/書き込み許可

-t ファイルシステムがアカウンティング・サブシステムによって処理されるかどうかを指定します。以下の値が有効です。

yes
ファイルシステムでアカウンティングが使用可能です。

no ファイルシステムでアカウンティングが使用可能ではありません。この値がデフォルト設定です。

-u MountGroup
マウント・グループを指定します。

-v VfsType
仮想ファイルシステムのタイプを指定します。 *agblksize* 属性はファイルシステムの作成時に設定され、ファイルシステムの作成後に変更することはできません。 *size* 属性は最小ファイルシステム・サイズを定義します。ファイルシステムの作成後に *size* 属性を使用してファイルシステム・サイズを小さくすることはできません。

例

vg01 という名前のボリューム・グループ上に JFS ファイルシステムを作成するには、次のように入力します。

```
cli_crlvfs -v jfs -g vg01 -m /tstvg -a 'size=32768'
```

---

cli_extendlv コマンド

目的

ボリューム・グループ内から未割り当ての物理区画を追加することにより、クラスター内のすべてのノードで論理ボリュームのサイズを拡大します。

構文

```
cli_extendlv [ -a Position ] [ -e Range ] [ -u Upperbound ] [ -s Strict ]
             LogicalVolume Partitions [ PhysicalVolume ... ]
```

説明

C-SPOC を使用して、指定のパラメーターで **extendlv** コマンドを実行し、更新された論理ボリューム定義をすべてのクラスター・ノード上で使用可能にします。

フラグ

-a Position
物理ボリュームの割り当てポリシー (物理ボリューム上の論理区画の位置) を設定します。以下の *Position* 変数が有効です。

m 各物理ボリュームの外側の中央部分に論理区画を割り当てます。この変数がデフォルト設定です。

c 各物理ボリュームの中央部分に論理区画を割り当てます。

e 各物理ボリュームの外側の端の部分に論理区画を割り当てます。

ie 各物理ボリュームの内側の端の部分に論理区画を割り当てます。

im 各物理ボリュームの内部中央部分に論理区画を割り当てます。

-e Range
物理ボリュームの割り当てポリシーを設定します。割り当てポリシーとは、割り当てが最適になるボリ

ユームを使用して拡張する物理ボリュームの数です。 *Range* 変数の値は、**-u** フラグで設定される *Upperbound* 変数によって制限されます。以下の *Range* 変数が有効です。

x 物理ボリュームの最大数に論理区画を割り当てます。

m 物理ボリューム間の最小数に論理区画を割り当てます。

-s Strict

厳密な割り当てポリシーを決定します。 同じ物理ボリュームを共用するように、または共用しないように、論理区画のコピーを割り当てることができます。以下の *Strict* 変数が有効です。

y 厳密な割り当てポリシーを設定します。したがって、論理区画のコピーは、同一物理ボリュームを共用することはできません。

n 厳密な割り当てポリシーを設定しません。したがって、論理区画のコピーは、同一物理ボリュームを共用できます。

s 非常に厳密な割り当てポリシーを設定します。したがって、1 つのミラーに割り当てられた区画は、別のミラーからの区画と物理ボリュームを共用することができません。非超厳密論理ボリュームを超厳密論理ボリュームに変更する場合は、**-u** フラグを使用する必要があります。

-u Upperbound

新たに割り当てるために物理ボリュームの最大数を設定します。*Upperbound* 変数の値は、1 から物理ボリュームの総数の範囲内です。極度の厳密さを使用する場合、上限は、ミラー・コピーごとに許される物理ボリュームの最大数を示します。ストライピングされた論理ボリュームを使用するときは、上限は *Stripe_width* 変数の倍数である必要があります。

例

lv01 という名前の論理ボリュームのサイズを論理区画 3 つ分だけ増やすには、次のように入力します。

```
cli_extendlv lv01 3
```

関連情報:

extendlv コマンド

---

cli_extendvg コマンド

目的

物理ボリュームをクラスター内のすべてのノード上のボリューム・グループに追加します。

構文

```
cli_extendvg VolumeGroup PhysicalVolume ...
```

説明

C-SPOC を使用して、指定のパラメーターで **extendvg** コマンドを実行し、更新されたボリューム・グループ定義をすべてのクラスター・ノード上で使用可能にします。

このコマンドを実行する前に、組み込む物理ボリューム (ディスク) がすべてのクラスター・ノードで使用可能であり、PVID が割り当て済みであることを確認する必要があります。

例

hdisk101 および *hdisk111* という名前のディスクを *vg01* という名前のボリューム・グループに追加するには、次のように入力します。

```
cli_extendvg vg01 hdisk101 hdisk111
```

関連情報:

extendvg コマンド

---

cli_importvg コマンド

目的

新しいボリューム・グループ定義を、クラスター内のすべてのノード上の物理ボリュームのセットからインポートします。

構文

```
cli_importvg [ -y VolumeGroup ] [ -V MajorNumber ]  PhysicalVolume
```

説明

C-SPOC を使用して、指定のパラメーターで **importvg** コマンドを実行します。このコマンドにより、各クラスター・ノードの論理ボリューム・マネージャー (LVM) は、ボリューム・グループ内のディスク上の LVM 情報を読み取り、ローカル・ボリューム・グループ定義を更新します。

フラグ

-V MajorNumber
インポートされたボリューム・グループのメジャー番号を指定します。

-y VolumeGroup
新しいボリューム・グループに使う名前を指定します。このフラグを使用しない場合は、システムによって新しい名前が自動的に生成されます。ボリューム・グループ名には、以下の文字のみを組み込むことができます。

- A から Z
- a から z
- 0 から 9
- _ (下線文字)
- - (マイナス文字)
- . (ピリオド文字)

例

hdisk07 という名前の物理ボリュームからの *bkvg* という名前のボリューム・グループをすべてのクラスター・ノード上で使用可能にするには、次のように入力します。

```
cli_importvg -y bkvg hdisk07
```

関連情報:

importvg コマンド

cli_mirrorvg コマンド

目的

構文

```
cli_mirrorvg  [-S | -s] [-Q] [-c Copies] [-m] VolumeGroup [PhysicalVolume...]
```

説明

C-SPOC を使用して、指定のパラメーターで **mirrorvg** コマンドを実行し、更新されたボリューム・グループ定義をすべてのクラスター・ノード上で使用可能にします。

フラグ

-c Copies
mirrorvg コマンドの実行後に各論理ボリュームに必要な最小コピー数を指定します。 **mklvcopy** コマンドを個別に使用することにより、一部の論理ボリュームでは、**mirrorvg** コマンドの実行後に指定された最小数よりも多くのコピーを持つことができる場合があります。指定できる最小値は 2、最大値は 3 です。値 1 は無視されます。

-m exact map
論理ボリュームのミラーリングを、オリジナル・コピーでの正確な物理区画の順序で行います。正確なマップ・コピーが配置されている物理ボリュームを指定する必要があります。正確なマッピングを行うためのスペースが不足している場合、このコマンドは失敗します。新しいドライブを追加するか、ボリューム・グループ全体の正確な論理ボリューム・マッピングを満たす異なるドライブのセットを選択する必要があります。指定するディスクは、(ディスク全体が使用されているかどうかにかかわらず)ミラーリングするドライブのサイズ以上であることが必要です。論理ボリュームのいずれかが既にミラーリングされている場合、コマンドは失敗します。

-Q Quorum Keep
ボリューム・グループの内容がミラーリングされている場合、デフォルトでは、ボリューム・グループのクォーラムは使用不可になります。ミラーリング完了後もボリューム・グループ・クォーラム要件を保持したい場合に、このフラグを使用できます。後でクォーラムを変更する方法については、**chvg** コマンドを参照してください。

-S Background Sync
mirrorvg コマンドを即時に返し、ボリューム・グループの **syncvg** コマンドをバックグラウンドで開始します。このフラグを使用した場合、ミラーリングがその同期化を完了する時期は明確ではありません。ただし、ミラーの一部分が同期化されると、その部分は論理ボリューム・マネージャー (LVM) によって即時にミラーリングに使用されます。

-s Disable Sync
どのタイプのミラー同期も実行することなく **mirrorvg** コマンドを即時に返します。このフラグを使用した場合、論理ボリューム用にミラーが存在していても、**syncvg** コマンドで同期化されるまではオペレーティング・システムで使用されません。

例

vg01 という名前の共用ボリューム・グループ内のすべての論理ボリュームに 2 つのコピーを指定するには、次のように入力します。

```
cli_mirrorvg -c 2 vg01
```

関連情報:

cli_mklv コマンド

目的

クラスター内のすべてのノード上に新しい論理ボリュームを作成します。

構文

```
cli_mklv [ -a Position ] [ -b BadBlocks ] [ -c Copies ]  [ -d Schedule ]
         [ -e Range ] [ -i ] [ -L Label ] [ -o y / n ] [ -r Relocate ]
         [ -s Strict ] [ -t Type ] [ -u UpperBound ] [ -v Verify ]
         [ -w MirrorWriteConsistency ] [ -x Maximum ] [ -y NewLogicalVolume |
         -Y Prefix ] [ -S StripSize ] [ -U Userid ] [ -G Groupid ] [ -P Modes ]
   VolumeGroup NumberOfLPs [ PhysicalVolume ... ]
```

説明

C-SPOC を使用してパラメーター指定で **mklv** コマンドを実行し、新しい論理ボリューム定義をすべてのクラスター・ノード上で使用可能にします。

フラグ

-a Position
物理ボリュームの割り当てポリシー (物理ボリューム上の論理区画の位置) を設定します。以下の *Position* 変数が有効です。

- **m** 各物理ボリュームの外側の中央部分に論理区画を割り当てます。この変数がデフォルト設定です。
- **c** 各物理ボリュームの中央部分に論理区画を割り当てます。
- **e** 各物理ボリュームの外側の端の部分に論理区画を割り当てます。
- **ie** 各物理ボリュームの内側の端の部分に論理区画を割り当てます。
- **im** 各物理ボリュームの内部中央部分に論理区画を割り当てます。

-b BadBlocks
不良ブロック再配置ポリシーを設定します。 以下の *BadBlocks* 変数が有効です。

- **y** 不良ブロックの再配置を実行します。
- **n** 不良ブロックの再配置の実行を禁止します。

-c Copies
mirrorvg コマンドの実行後に各論理ボリュームに必要な最小コピー数を指定します。 **mklvcopy** コマンドを個別に使用することにより、一部の論理ボリュームでは、**mirrorvg** コマンドの実行後に指定された最小数よりも多くのコピーを持つことができる場合があります。指定できる最小値は 2、最大値は 3 です。値 1 は無視されます。

-d Schedule
2 つ以上の論理区画が書き込まれるときに、スケジューリング・ポリシーを設定します。ストライピングされた論理ボリュームをミラーリングするには、並列処理または順次処理を使用する必要があります。以下の *Schedule* 変数が有効です。

- **p** 並列スケジューリング・ポリシーを設定します。
- **ps** 並列書き込み/順次読み取りポリシー。ミラーはすべて並列に書き込まれますが、最初のミラーが使用可能であれば必ず最初のミラーから読み取られます。

pr 並列書き込みおよび並列読み取りがすべてのミラーに対して行われます。このポリシーは、すべてのミラーにまたがってさらに論理ボリュームへの読み取りを広げようとする試みを除いては、並列ポリシーと類似しています。

s 順次スケジューリング・ポリシーを設定します。この変数は、並列または順次の厳密性 (極度の厳密性) のポリシーを指定する場合に使用します。

-e Range
物理ボリュームの割り当てポリシーを設定します。割り当てポリシーとは、割り当てが最適になるボリュームを使用して拡張する物理ボリュームの数です。 *Range* 変数の値は、**-u** フラグで設定される *Upperbound* 変数によって制限されます。以下の *Range* 変数が有効です。

x 物理ボリュームの最大数に論理区画を割り当てます。

m 物理ボリューム間の最小数に論理区画を割り当てます。

-G Groupid
論理ボリューム・スペシャル・ファイルのグループ ID を指定します。

-L Label
論理ボリューム・ラベルを設定します。この変数の最大サイズは 127 文字です。

-n NewLogicalVolume
NewLogicalVolume 変数で指定された論理ボリュームの名前を変更します。論理ボリューム名はシステム全体で固有でなければならず、15 文字まで指定できます。

-p Permission
アクセス権を読み取り/書き込み、または読み取り専用に設定します。以下の *Permission* 変数が有効です。

w アクセス権を読み取り/書き込みに設定します。

r アクセス権を読み取り専用に設定します。読み取り専用論理ボリュームへの JFS ファイルシステムのマウントは、サポートされません。

-P Modes
論理ボリューム・スペシャル・ファイルのアクセス権 (ファイル・モード) を指定します。

-r Relocate
再編成中に論理ボリュームの再配置を許可するか禁止するかを指定します。以下の *Relocate* 変数が有効です。

y 再編成時の論理ボリュームの再配置を許可します。論理ボリュームがストライピングされている場合、**chlv** コマンドを使用して再配置フラグを y に変更することはできません。

n 再編成時の論理ボリュームの再配置を禁止します。

-s Strict
厳密な割り当てポリシーを決定します。 同じ物理ボリュームを共用するように、または共用しないように、論理区画のコピーを割り当てることができます。以下の *Strict* 変数が有効です。

y 厳密な割り当てポリシーを設定します。したがって、論理区画のコピーは、同一物理ボリュームを共用することはできません。

n 厳密な割り当てポリシーを設定しません。したがって、論理区画のコピーは、同一物理ボリュームを共用できます。

s 非常に厳密な割り当てポリシーを設定します。したがって、1 つのミラーに割り当てられた区画は、別のミラーからの区画と物理ボリュームを共用することができません。非超厳密論理ボリュームを超厳密論理ボリュームに変更する場合は、**-u** フラグを使用する必要があります。

-S StripSize
ストリップごとのバイト数を指定します (ストリップ・サイズにアレイ内のディスク数を乗算するとストライプ・サイズになります)。有効な値には、 4K、8K、16K、32K、64K、128K、256K、512K、1M、2M、4M、8M、16M、32M、64M、および 128M が含まれます。ストライピングされた論理ボリュームをこのフラグを使用して作成する場合、**-d**、**-e**、および **-s** の各フラグは使用できません。

-t Type
論理ボリューム・タイプを設定します。標準タイプは以下のとおりです。

- jfs (ジャーナル・ファイル・システム)
- jfslog (ジャーナル・ファイル・システム・ログ)
- jfs2 (拡張ジャーナル・ファイルシステム)
- jfs2log (拡張ジャーナル・ファイルシステム・ログ)
- paging (ページング・スペース)

このフラグを使用して、その他の論理ボリュームのタイプを定義することができます。boot タイプのストライプ化論理ボリュームの作成はできません。デフォルト値は jfs です。タイプ jfslog または jfs2log で論理ボリュームを作成する場合は、その論理ボリュームを使用できるように、C-SPOC が自動的に **logform** コマンドを実行します。

-U Userid
論理ボリューム・スペシャル・ファイルのユーザー ID を指定します。

-u Upperbound
新たに割り当てるために物理ボリュームの最大数を設定します。*Upperbound* 変数の値は、1 から物理ボリュームの総数の範囲内です。極度の厳密さを使用する場合、上限は、ミラー・コピーごとに許される物理ボリュームの最大数を示します。ストライピングされた論理ボリュームを使用するときは、上限は *Stripe_width* 変数の倍数である必要があります。

-v Verify
論理ボリュームに対する書き込み検査状態を設定します。これにより、論理ボリュームに対するすべての書き込みは、書き込み後の読み取りによる検査を受けるか、受けないかのどちらかになります。以下の *Verify* 変数が有効です。

y 論理ボリュームに対する書き込みはすべて、書き込み後の読み取りにより検査されます。

n 論理ボリュームに対する書き込みはすべて、書き込み後の読み取りにより検査されません。

-w MirrorWriteConsistency
以下の *MirrorWriteConsistency* 変数が有効です。

y アクティブ・ミラー書き込み整合性の保持をオンにします。この変数は、通常の入出力処理中に、論理ボリュームのミラー・コピーでのデータ整合性を検査します。

p パッシブ・ミラー書き込み整合性の保持をオンにします。この変数は、システム割り込み後のボリューム・グループの同期化中に、ミラー・コピーでのデータ整合性を検査します。この機能はビッグ・ボリューム・グループでのみ使用できます。

n ミラー書き込み整合を行いません。

-x Maximum
論理ボリュームに割り当てることのできる論理区画の最大数を設定します。 論理ボリュームあたりの論理区画の最大数は 32,512 です。

-y NewLogicalVolume
システムが生成した名前の代わりに使用する論理ボリューム名を指定します。論理ボリュームの名前は

システム全体で固有の名前でなければならず、可能な文字数の範囲は 1 から 15 です。新しい名前は、ボリューム・グループが定義されているすべてのノードで固有にする必要があります。他のデバイスのデバイス構成データベースで事前定義デバイス・データベース (PdDv) クラスにすでに定義されている接頭部を、名前の先頭に使用することはできません。

-Y Prefix
新しい論理ボリュームのシステム生成名の接頭部の代わりに使用する接頭部の値を指定します。接頭部の値は 13 文字以内でなければなりません。他のデバイスのデバイス構成データベースで事前定義デバイス・データベース (PdDv) クラスにすでに定義されている接頭部を、名前の先頭に使用することはできません。また、別のデバイスですでに使用されている名前は使用できません。

例

1 つの論理区画と合計 2 つのデータのコピーを持つ論理ボリュームを *vg02* という名前のボリューム・グループに作成するには、次のように入力します。

```
cli_mklv -c 2 vg01 1
```

関連情報:

mklv コマンド

---

cli_mklvcopy コマンド

目的

クラスター内のすべてのノード上の論理ボリューム内の各論理区画で、コピー数を増やします。

構文

```
cli_mklvcopy  [ -a Position ] [ -e Range ] [ -k ] [ -s Strict ]
  [ -u UpperBound ] LogicalVolume Copies [ PhysicalVolume... ]
```

説明

C-SPOC を使用してパラメーター指定で **mklvcopy** コマンドを実行し、更新された論理ボリューム定義をすべてのクラスター・ノード上で使用可能にします。

フラグ

-a Position
物理ボリュームの割り当てポリシー (物理ボリューム上の論理区画の位置) を設定します。以下の *Position* 変数が有効です。

- **m** 各物理ボリュームの外側の中央部分に論理区画を割り当てます。この変数がデフォルト設定です。
- **c** 各物理ボリュームの中央部分に論理区画を割り当てます。
- **e** 各物理ボリュームの外側の端の部分に論理区画を割り当てます。
- **ie** 各物理ボリュームの内側の端の部分に論理区画を割り当てます。
- **im** 各物理ボリュームの内部中央部分に論理区画を割り当てます。

-e Range
物理ボリュームの割り当てポリシーを設定します。割り当てポリシーとは、割り当てが最適になるポリ

ユームを使用して拡張する物理ボリュームの数です。 *Range* 変数の値は、**-u** フラグで設定される *Upperbound* 変数によって制限されます。以下の *Range* 変数が有効です。

x 物理ボリュームの最大数に論理区画を割り当てます。

m 物理ボリューム間の最小数に論理区画を割り当てます。

-k 新しい区画でデータと同期をとります。

-s Strict
厳密な割り当てポリシーを決定します。 同じ物理ボリュームを共用するように、または共用しないように、論理区画のコピーを割り当てることができます。以下の *Strict* 変数が有効です。

y 厳密な割り当てポリシーを設定します。したがって、論理区画のコピーは、同一物理ボリュームを共用することはできません。

n 厳密な割り当てポリシーを設定しません。したがって、論理区画のコピーは、同一物理ボリュームを共用できます。

s 非常に厳密な割り当てポリシーを設定します。したがって、1 つのミラーに割り当てられた区画は、別のミラーからの区画と物理ボリュームを共用することができません。非超厳密論理ボリュームを超厳密論理ボリュームに変更する場合は、**-u** フラグを使用する必要があります。

-u Upperbound
新たに割り当てるために物理ボリュームの最大数を設定します。*Upperbound* 変数の値は、1 から物理ボリュームの総数の範囲内です。極度の厳密さを使用する場合、上限は、ミラー・コピーごとに許される物理ボリュームの最大数を示します。ストライピングされた論理ボリュームを使用するときは、上限は *Stripe_width* 変数の倍数である必要があります。

例

各論理区画に合計 3 つのコピーが存在するように、物理区画を *lv01* という名前の論理ボリュームの論理区画に追加するには、次のように入力します。

```
cli_mklvcopy lv01 3
```

関連情報:

mklvcopy コマンド

---

cli_mkvg コマンド

目的

クラスター内のすべてのノードにボリューム・グループを作成します。

構文

```
cli_mkvg   [ -B ] [ -t factor ] [ -C ] [ -G ] [ -x ] [ -s Size ]
     [ -V MajorNumber ] [ -v LogicalVolumes ] [ -y VolumeGroup ]
     PhysicalVolume ...
```

説明

C-SPOC を使用してパラメーター指定で **mkvg** コマンドを実行し、新しい論理ボリューム定義をすべてのクラスター・ノード上で使用可能にできます。

フラグ

-B ビッグ・タイプのボリューム・グループを作成します。このタイプのボリューム・グループには、最大

128 の物理ボリュームおよび 512 の論理ボリュームを収容できます。 vgda スペースは大幅に増加しているため、どの vgda 更新操作 (論理ボリュームの作成、論理ボリュームの変更、および物理ボリュームの追加) も、実行にかなり時間がかかる可能性があります。

-C 拡張コンカレント機能付きボリューム・グループを作成します。このフラグは構成済みの PowerHA SystemMirror クラスターでのみ使用できます。このフラグを使用して、拡張コンカレント機能付きボリューム・グループを作成できます。

拡張コンカレント機能付きボリューム・グループは、グループ・サービスを使用します。グループ・サービスは、PowerHA SystemMirror で使用でき、このモードでボリューム・グループを活動化する前に構成されていなければなりません。

64 ビット・カーネルでは、拡張コンカレント機能付きボリューム・グループのみがサポートされます。コンカレント機能付きボリューム・グループは 64 ビット・カーネルではサポートされません。

-p partitions
ボリューム・グループ内の区画の総数を指定します。 *partitions* 変数は 1024 区画の単位で表されます。このフラグには以下の値が有効です。

- 32
- 64
- 128
- 256
- 512
- 768
- 1024
- 2048

デフォルト値は 32 k (32768 区画) です。 **chvg** コマンドを使用して、区画数を最大 2048 k (2097152 区画) まで増やすことができます。このフラグは **-s** フラグと併用する場合にのみ有効です。

-s size
各物理区画にメガバイト数 (MB) を設定します。 size 変数は、1 (1 MB) から 131072 (128 GB) までのメガバイト単位で表されます。 size 変数は 2 の累乗 (1、2、4、8 など) でなければなりません。 32 物理ボリューム・グループおよび 128 物理ボリューム・グループのデフォルト値は、物理ボリュームあたり 1016 物理区画の制限内に収まる最小値です。スケーラブル・ボリューム・グループのデフォルト値は、物理ボリュームあたり 2040 物理区画を収容する最小値です。

-t factor
factor で指定される、物理ボリュームあたりの物理区画数の制限を変更します。 32 物理ボリューム・グループの場合、factor は 1 から 16 の範囲内でなければなりません。 128 物理ボリューム・グループの場合、factor は 1 から 64 の範囲内でなければなりません。このボリューム・グループの物理ボリュームあたりの物理区画の最大数は、factor x 1016 に変更されます。デフォルトは、factor x 1016 の物理区画の制限内に収まる最小の値です。ボリューム・グループに収容できる物理ボリュームの最大数は maxpvs/factor です。 **-s** フラグを使用する場合、このフラグは無視されます。

-V majornumber
作成するボリューム・グループのメジャー番号を指定します。

-v 作成できる論理ボリューム数を指定します。このフラグには以下の値が有効です。

- 256
- 512

- 1024
- 2048
- 4096

デフォルト値は 256 です。 **chvg** コマンドを使用して、論理ボリューム数を最大 4096 まで増やすことができます。このフラグは **-s** フラグと併用する場合にのみ有効です。最後の論理ボリュームは、メタデータ用に予約済みです。

-y volumegroup
ボリューム・グループ名を、自動的に生成するのではなく指定します。ボリューム・グループ名はシステム全体で固有でなければなりません。この名前には 1 文字から 15 文字を使用できます。他のデバイスのデバイス構成データベースで事前定義デバイス・データベース (PdDv) クラスにすでに定義されている接頭部を、名前の先頭に使用することはできません。作成されたボリューム・グループ名は、標準出力に送信されます。ボリューム・グループ名には、以下の文字のみを組み込むことができます。

- a から z
- 0 から 9
- _ (下線文字)
- - (マイナス文字)
- . (ピリオド文字)

例

hdisk3、*hdisk5*、および *hdisk6* という名前のディスクを含み、物理区画サイズを 1 メガバイトに設定したボリューム・グループを作成するには、次のように入力します。

```
cli_mkvg -s 1 hdisk3 hdisk5 hdisk6
```

関連情報:

mkvg コマンド

---

cli_on_cluster コマンド

目的

クラスター内のすべてのノードでコマンドを実行します。

構文

```
cli_on_cluster  [ -S | -P ]  'command string'
```

説明

コマンドを root として、すべてのクラスター・ノードで順次または並列に実行します。コマンドからの出力 (stdout および stderr) はコマンド・ラインに表示されます。出力の各行の先頭にはノード名があり、続いてコロンが示されています。

フラグ

-S クラスター内の各ノードで、一度に 1 つのコマンドを実行します。コマンドが終了すると、次のコマンドが実行されます。

-P クラスター内のすべてのノードで、コマンドを同時に並行して実行します。

例

クラスター内のすべてのノードをリブートするには、次のように入力します。

```
cli_on_cluster -S 'shutdown -Fr'
```

cli_on_node コマンド

目的

任意のコマンドをクラスター内の特定ノードで実行します。

構文

```
cli_on_node  [ -V <volume group> | -R <resource group | -N <node> ] 'command string'
```

説明

明示的に指定したノードで、あるいは指定したボリューム・グループまたはリソース・グループを所有するクラスター・ノードで、root としてコマンドを実行します。コマンドからの出力 (stdout および stderr) はすべてコマンド・ラインに表示されます。

フラグ

-V volume group
指定したボリューム・グループがオンに変更済み状態にあるノードで、コマンドを実行します。ボリューム・グループが複数のノードでコンカレント・モードでオンに変更済み状態にある場合、コマンドはすべてのノードで実行されます。

-R resource group
指定したリソース・グループを現在所有しているノードでコマンドを実行します。

-N node
指定したノードでコマンドを実行します。このフラグは PowerHA SystemMirror ノード名を識別します。

例

awesome という名前のノードで **ps -efk** コマンドを実行するには、次のように入力します。

```
cli_on_node -N awesome 'ps -efk'
```

cli_reducevg コマンド

目的

物理ボリュームをボリューム・グループから除去し、更新された変更をすべてのクラスター・ノード上で使用可能にします。物理ボリュームがボリューム・グループからすべて除去されると、そのボリューム・グループはすべてのクラスター・ノード上で削除されます。

構文

```
cli_reducevg  VolumeGroup PhysicalVolume ...
```

説明

C-SPOC を使用してパラメーター指定で **reducevg** コマンドを実行し、更新されたボリューム・グループ定義をすべてのクラスター・ノード上で使用可能にします。

例

hdisk10 という名前の物理ディスクを *vg01* という名前のボリューム・グループから除去するには、次のように入力します。

```
cli_reducevg vg01 hdisk101
```

関連情報:

reducevg コマンド

---

cli_replacepv コマンド

目的

ボリューム・グループ内の物理ボリュームを別の物理ボリュームで置換し、変更をすべてのクラスター・ノードで使用可能にします。

構文

```
cli_replacepv  SourcePhysicalVolume  DestinationPhysicalVolume
```

説明

C-SPOC を使用してパラメーター指定で **replacepv** コマンドを実行し、更新されたボリューム・グループ定義をすべてのクラスター・ノード上で使用可能にします。

例

hdisk10 という名前のディスクを、*hdisk10* ディスクを所有するボリューム・グループ内の *hdisk20* という名前のディスクで置換するには、次のように入力します。

```
cli_replacepv hdisk10 hdisk20
```

関連情報:

replacepv コマンド

---

cli_rmfs コマンド

目的

ファイルシステムをクラスター内のすべてのノードから除去します。

構文

```
cli_rmfs [ -r ] FileSystem
```

説明

C-SPOC を使用してパラメーター指定で **rmfs** コマンドを実行し、ファイルシステム定義をすべてのクラスター・ノードから除去します。

フラグ

-r ファイルシステムのマウント・ポイントを除去します。

例

/test_fs という名前の共用ファイルシステムを除去するには、次のように入力します。

```
cli_rmfs -r /test_fs
```

関連情報:

rmfs コマンド

---

cli_rmlv コマンド

目的

論理ボリュームをクラスター内のすべてのノードから除去します。

構文

```
cli_rmlv  LogicalVolume ...
```

説明

C-SPOC を使用してパラメーター指定で **rmlv** コマンドを実行し、更新された論理ボリューム定義をすべてのクラスター・ノード上で使用可能にします。

例

lv01 という名前の共用論理ボリュームを変更するには、次のように入力します。

```
cli_rmlv lv01
```

関連情報:

rmlv コマンド

---

cli_rmlvcopy コマンド

目的

コピーをクラスター内のすべてのノード上の論理ボリュームから除去します。

構文

```
cli_rmlvcopy LogicalVolume Copies [ PhysicalVolume... ]
```

説明

C-SPOC を使用してパラメーター指定で **rmlvcopy** コマンドをすべてのクラスター・ノード上で実行します。

例

単一コピーのみが残るように、*lv01* という名前の論理ボリュームに属する各論理区画のコピー数を削減するには、次のように入力します。

```
cli_rmlvcopy lv01 1
```

関連情報:

rmlvcopy コマンド

cli_syncvg コマンド

目的

パラメーター指定で **syncvg** コマンドを実行し、更新されたボリューム・グループ定義をすべてのクラスター・ノード上で使用可能にします。

構文

```
cli_syncvg  [-f] [-H] [-P NumParallelLps] {-l|-v} Name
```

説明

C-SPOC を使用して **syncvg** コマンドを実行します。これにより、各クラスター・ノードの論理ボリューム・マネージャー (LVM) は、ボリューム・グループ内のディスク上の LVM 情報を読み取ります。このコマンドはまた、ローカル・ボリューム・グループ定義を更新します。

フラグ

-f 有効な物理コピーを (不整合であっても) 選択し、論理区画の他のすべてのコピーに伝搬することを指定します。

-H コンカレント・ボリューム・グループがアクティブである他のクラスター・ノード上の選択したボリューム・グループに対する書き込み操作を、同期操作が完了するまで遅らせます。このフラグを使用する場合、クラスター内のすべてのノードで **cli_syncvg** コマンドの -P フラグがサポートされている必要はありません。ボリューム・グループがコンカレント・モードでオンに変更されていない場合、このフラグは無視されます。

-l *Name* 変数が論理ボリューム・デバイス名を表すことを指定します。

-P NumParallelLps
並行して同期化する論理区画の数を指定します。 *NumParallelLps* 変数の有効範囲は 1 から 32 です。*NumParallelLps* 変数は、システム、ボリューム・グループ内のディスク、システム・リソース、およびボリューム・グループ・モードに固有でなければなりません。

-v *Name* 変数がボリューム・グループ・デバイス名であることを指定します。

例

v01 という名前のボリューム・グループ上のコピーを同期化するには、次のように入力します。

```
cli_syncvg -v vg01
```

関連情報:

syncvg コマンド

cli_unmirrorvg コマンド

目的

クラスター内のすべてのノード上のボリューム・グループをミラーリング解除します。

構文

```
cli_unmirrorvg  [ -c Copies ] VolumeGroup [ PhysicalVolume ... ]
```

説明

C-SPOC を使用してパラメーター指定で **unmirrorvg** コマンドを実行し、更新されたボリューム・グループ定義をすべてのクラスター・ノード上で使用可能にします。

フラグ

-c Copies
unmirrorvg コマンドの実行後に各論理ボリュームに必要な最小コピー数を指定します。すべての論理ボリュームに同じコピー数を持たせたくない場合は、**rmlvcopy** コマンドを使用して手動でミラーを削減します。このフラグを使用しない場合は、デフォルト値 1 が使用されます。

例

vg01 という名前の共用ボリューム・グループの単一コピーを指定するには、次のように入力します。

```
cli_unmirrorvg -c 1 vg01
```

関連情報:

unmirrorvg コマンド

---

cli_updatevg コマンド

目的

ボリューム・グループの実際の現行状態に合わせて、すべてのクラスター・ノード上のボリューム・グループの定義を更新します。

構文

```
cli_updatevg VolumeGroup
```

説明

C-SPOC を使用して **updatevg** コマンドを実行します。これにより、各クラスター・ノードの論理ボリューム・マネージャー (LVM) は、ボリューム・グループ内のディスク上の LVM 情報を読み取り、ローカル・ボリューム・グループ定義を更新します。

例

すべてのクラスター・ノード上の *vg11* という名前のボリューム・グループのボリューム・グループ定義を更新するには、次のように入力します。

```
cli_updatevg vg11
```

---

cllscf コマンド

目的

クラスター・トポロジー情報をリストします。

構文

```
cllscf
```

説明

cllscf コマンドは、クラスター構成、ネットワーク構成、およびアダプター構成の各 ODM オブジェクト・クラスで定義されているクラスター・トポロジー情報をリストします。 **cllscf** コマンドはクラスター構成情報を要約します。

例

デフォルトの、またはアクティブなクラスター構成で定義されているクラスター情報を表示するには、次のように入力します。

```
cllscf
```

このコマンドによって、以下の例のような出力情報が表示されます。

```
# /usr/es/sbin/cluster/utilities/cllscf
Cluster Name:    hadev11_cluster
Cluster Type:    Standard
Heartbeat Type:  Unicast
Repository Disk: hdisk10 (00c0f592e54367f2)

There were 2 networks defined: net_ether_01, net_ether_02
There are 2 nodes in this cluster

NODE hadev11:
        This node has 0 service IP label(s):

NODE hadev12:
        This node has 0 service IP label(s):

Breakdown of network connections:

Connections to network net_ether_01
        Node hadev11 is connected to network net_ether_01 by these interfaces:
                hadev11

        Node hadev12 is connected to network net_ether_01 by these interfaces:
                hadev12

Connections to network net_ether_02
        Node hadev12 is connected to network net_ether_02 by these interfaces:
                hadev12_en1_boot
                hadev12_en2_boot
```

関連資料:

49 ページの『clmgr コマンド』

cllsdisk コマンド

目的

指定したリソース・チェーン内のアクセス可能なディスクの PVID をリストします。

構文

```
cllsdisk {-g Resource Group }
```

例

次のコマンドを実行して、リソース・グループ *grp3* のすべての参加ノードからアクセスできるディスクの PVID をリストします。

```
cllsdisk -g grp3
```

cllsfs コマンド

目的

リソース・グループのすべての参加ノードからアクセスできる共用ファイルシステムをリストします。

構文

```
cllsfs {-g resource group } [-n]
```

表 *3. cllsfs* フラグ

フラグ	説明
-g resource group	ファイルシステムをリストする対象のリソース・グループの名前を指定します。
-n	リソース・グループ内のファイルシステムを共用しているノードをリストします。

注**: cllsfs** コマンドをコマンド・ラインから実行しないでください。『共用 LVM コンポーネントの管理』の説明に従って、SMIT インターフェースを使用してファイルシステム情報を検索します。

cllsgrp コマンド

目的

クラスター用に構成されているすべてのリソース・グループをリストします。

構文

```
cllsgrp
```

説明

クラスター内のすべてのリソース・グループの名前を表示します。

例

クラスターのリソース・グループ情報を表示するには、次のように入力します。

```
cllsgrp
```

このコマンドは以下の出力を表示します。

```
grp1
grp2
grp3
grp4
```

cllsparam コマンド

目的

実行時パラメーターをリストします。

構文

```
cllsparam {-n nodename } [-c] [-s] [-d odmdir ]
```

フラグ

-n nodename
情報をリストするノードを指定します。

-c コロン出力形式を指定します。

-s **-c** フラグとともに使用され、英語ではなくネイティブ言語を指定します。

-d odmdir
代替 ODM ディレクトリーを指定します。

例

以下の例を実行して、ノード abalone のランタイム・パラメーターを表示します。

```
cllsparam -n abalone
```

---

cllsres コマンド

目的

PowerHA SystemMirror for AIX 構成データベースのリソース・データを名前および引数別にソートします。

構文

```
cllsres [-g group ] [-e] [-c] [-s] [-d odmdir ] [-qquery ]
```

フラグ

-g group
リストするリソース・グループの名前を指定します。

-c コロン出力形式を指定します。

-e ユーザー定義のリソース・リストを「resourcetype=resourcenames」形式で拡張します。

-s **-c** フラグとともに使用され、英語ではなくネイティブ言語を指定します。

-d odmdir
代替 ODM ディレクトリーを指定します。

-q query
ODM を取り出すための検索基準を指定します。検索条件の詳細については、 **odmget** のマニュアル・ページを参照してください。

例

1. すべてのリソース・グループのソース・データをリストするには、以下のコマンドを実行します。
   ```
   cllsres
   ```
2. grp1 リソース・グループのリソース・データをリストするには、以下のコマンドを実行します。
   ```
   cllsres -g grp1
   ```
3. grp1 リソース・グループのファイルシステム・リソース・データをリストするには、以下のコマンドを実行します。

```
cllsres -g grp1 -q"name = FILESYSTEM"
```

cllsserv コマンド

目的

アプリケーション・コントローラーを名前別にリストします。

構文

```
cllsserv [-c] [-h] [-n name ] [-d odmdir ]
```

フラグ

-c コロン出力形式を指定します。

-h 見出しを出力することを指定します。

-n name
情報を検査する対象のアプリケーション・コントローラーを指定します。

-d odmdir
代替 ODM ディレクトリーを指定します。

例

1. すべてのアプリケーション・コントローラーをリストするには、以下のコマンドを実行します。

   ```
   cllsserv
   ```

2. test1 アプリケーション・コントローラーの情報をコロンで区切られたフォーマットでリストするには、以下のコマンドを実行します。

   ```
   cllsres -c -n test1
   ```

cllsvg コマンド

目的

クラスター内でノードが共用するボリューム・グループをリストします。 構成されたリソース・グループのすべての参加ノードがアクセスできるボリューム・グループは、共用されていると見なされます。 リストされるボリューム・グループは、いずれかのリソース・グループでリソースとして構成されている場合も、 されていない 場合もあります。 **-s** と **-c** がいずれも選択されていない場合は、共用ボリューム・グループとコンカレント・ボリューム・グループの両方がリストされます。

構文

```
cllsvg {-g resource group } [-n] [-v] [-s | -c]]
```

フラグ

-g resource group
リスト対象のボリューム・グループを含み、そのボリューム・グループを共用しているノードが参加しているリソース・グループの名前を指定します。

-n nodes
各リソース・グループに参加しているすべてのノードを指定します。

-v オンに変更され、他のコマンド・ライン基準に一致するボリューム・グループのみをリストします。

-s 他の基準も満たす共用ボリューム・グループのみをリストします。

-c 他の基準も満たすコンカレント・ボリューム・グループのみをリストします。

例

grp1 リソース・グループに含まれるすべての共用ボリューム・グループをリストするには、以下のコマンドを実行します。

```
cllsvg -g grp1
```

---

clmgr コマンド

目的

clmgr コマンドは、端末またはスクリプトを使用して PowerHA SystemMirror のクラスター操作を実行するための、整合性のある、信頼できるインターフェースを提供します。

構文

clmgr コマンドの完全な構文は以下のとおりです。

```
clmgr {[-c|-d <DELIMITER>] [-S] | [-x]}
[-v][-f][-D] [-T <#####>]
[-l {error|standard|low|med|high|max}][-a {<ATTR#1>,<ATTR#2>,...}] <ACTION> <CLASS> [<NAME>]
[-h | <ATTR#1>=<VALUE#1> <ATTR#2>=<VALUE#2> <ATTR#n>=<VALUE#n>]
```

```
clmgr {[-c|-d <DELIMITER>] [-S] | [-x]}
[-v][-f] [-D] [-T <#####>]
[-l {error|standard|low|med|high|max}][-a {<ATTR#1>,<ATTR#2>,...}]
[-M]- "
<ACTION> <CLASS> [<NAME>] <ATTR#1>=<VALUE#1> <ATTR#n>=<VALUE#n>]
.
.
."
         ACTION={add|modify|delete|query|online|offline|...}
          CLASS={cluster|site|node|network|resource_group|...}

    clmgr {-h|-?} [-v]
    clmgr [-v] help
```

```
# clmgr add nova -h
clmgr add nova <NovaLink> \
[ TIMEOUT={<#>} ] \
[ RETRY_COUNT={<#>} ] \
[ RETRY_DELAY={<#>} ] \
[ USER_NAME={<#>} ] \
[ CHECK_Nova={<yes>|<no>} ]
```

```
# clmgr modify nova -h
clmgr modify nova <NovaLink> \
[ TIMEOUT={<#>} ] \
[ RETRY_COUNT={<#>} ] \
[ RETRY_DELAY={<#>} ] \
[ USER_NAME={<#>} ] \
[ PASSWORD={<#>} ] \
[ CHECK_Nova={<yes>|<no>} ]
```

```
# clmgr delete nova -h
clmgr delete nova {<NOVA>[,<NOVA#2>,...] | ALL}
```

```
# clmgr manage cluster nova -h
clmgr manage cluster nova \
[ DEFAULT_NOVA_TIMEOUT=# ] \
[ DEFAULT_NOVA_RETRY_COUNT=# ] \
[ DEFAULT_NOVA_RETRY_DELAY=# ] \
[ CONNECTION_TYPE=<SSH> ]
```

以下は、**clmgr** コマンドを使用するための基本形式です。

```
clmgr <ACTION> <CLASS> [<NAME>] [<ATTRIBUTES...>]
```

clmgr コマンドについては、コマンド・ラインでヘルプが使用可能です。例えば、フラグやパラメーターを何も指定しないで **clmgr** コマンドを実行すると、選択可能な ACTION のリストが表示されます。 コマンド・ラインで CLASS を指定しないで `clmgr ACTION` と入力すると、結果は、指定した ACTION について選択可能な CLASS がすべてリストされます。NAME または ATTRIBUTES を指定しないで `clmgr ACTION CLASS` と入力した場合は少し異なります。これは、ACTION と CLASS の組み合わせによっては、追加のパラメーターが不要な場合があるからです。このシナリオでヘルプを表示するには、`clmgr ACTION CLASS` コマンドに `-h` フラグを付加することによって、明示的にヘルプを要求する必要があります。各 **clmgr** コマンドの個々の ATTRIBUTES に関するヘルプをコマンド・ラインから表示することはできません。

説明

clmgr コマンドは、高度な整合性があり、学習しやすく使用しやすいコマンドです。実行時の整合性に加えて、**clmgr** は、スクリプト記述が容易になる整合した戻りコードを提供します。クラスター情報の収集ができるだけ簡単になるように、データ照会にも複数の出力形式が用意されています。

すべての **clmgr** コマンド操作は clutils.log ファイルに記録されます。ここには、実行されたコマンドの名前、コマンドの開始時刻と終了時刻、およびコマンドを開始したユーザー名が含まれます。

注**:** リソース・グループに複数の依存関係がある場合、複数のリソース・グループを移動するために **clmgr** コマンドを使用することはできません。

フラグ

ACTION

実行する操作を記述します。

注**:** ACTION には大/小文字の区別がありません。すべての ACTION フラグについて、短いエイリアスが用意されています。 例えば、`rm` は `delete` のエイリアスです。 エイリアスは、コマンド・ラインでの利便性のために提供されるものであり、スクリプトで使用してはなりません。

以下の 4 つの ACTION フラグは、サポートされるほとんどすべての CLASS オブジェクトで使用可能です。

- add (エイリアス: a)
- query (エイリアス: q、ls、get)
- modify (エイリアス: mod、ch、set)
- delete (エイリアス: de、rm、er)

残りの ACTION は、通常、サポートされる CLASS オブジェクトの小さなサブセットでのみサポートされます。

- クラスター、ノード、リソース・グループの場合:
  - start (エイリアス: online、on)

- stop (エイリアス: offline、off)

- リソース・グループ、サービス IP、永続 IP の場合:
  - move (エイリアス: mv)
- クラスター、インターフェース、ログ、ノード、スナップショット、ネットワーク、アプリケーション・モニターの場合:
  - manage (エイリアス: mg)
- クラスターおよびファイル・コレクションの場合:
  - sync (エイリアス: sy)
- クラスター、メソッドの場合:
  - verify (エイリアス: ve)
- ログ、レポート、スナップショットの場合:
  - view (エイリアス: vi)
- リポジトリー:
  - replace (エイリアス: rep、switch、swap)

CLASS

ACTION が実行される対象のオブジェクトのタイプ。

注**:** CLASS には大/小文字の区別がありません。すべての CLASS オブジェクトについて、短いエイリアスが用意されています。例えば、`fc` は `file_collection` のエイリアスです。エイリアスは、コマンド・ラインでの利便性のために提供されるものであり、スクリプトで使用してはなりません。

以下は、サポートされる CLASS オブジェクトの完全なリストです。

- cluster (エイリアス: cl)
- repository (エイリアス: rp)
- site (エイリアス: st)
- node (エイリアス: no)
- interface (エイリアス: in、if)
- network (エイリアス: ne、nw)
- resource_group (エイリアス: rg)
- service_ip (エイリアス: si)
- persistent_ip (エイリアス: pi)
- application_controller (エイリアス: ac、app)
- application_monitor (エイリアス: am、mon)
- tape (エイリアス: tp)
- dependency (エイリアス: de)
- file_collection (エイリアス: fi、fc)
- snapshot (エイリアス: sn、ss)
- method (エイリアス: me)
- volume_group (エイリアス: vg)
- logical_volume (エイリアス: lv)
- file_system (エイリアス: fs)

- physical_volume (エイリアス: pv、disk)
- mirror_pool (エイリアス: mp)
- user (エイリアス: ur)
- group (エイリアス: gp)
- ldap_server (エイリアス: ls)
- ldap_client (エイリアス: lc)
- event
- hmc
- cod (エイリアス: cuod、dlpar)

NAME
ACTION が実行される対象の、タイプ CLASS の、指定されたオブジェクト。

ATTR=VALUE
ACTION と CLASS の組み合わせに固有な属性のペアと値のペアを持つオプションのフラグ。このペア・フラグは、構成の設定を指定したり、特定の操作を調整したりするために使用します。

ATTR=VALUE の指定は、query アクションで使用する場合は、属性ベースの検索およびフィルター処理を実行するために使用できます。この目的で使用する場合には、単純なワイルドカードを使用できます。例えば、"*" はゼロ個以上の任意の文字に一致し、"?" はゼロ個または 1 個の任意の文字に一致します。

注**:** ATTR の場合、必ず (すべての文字を) 完全に入力しなければならないというわけではありません。属性を、指定された操作について使用可能な一連の属性から一意的に識別するのに必要な数だけの先頭からの文字を入力するだけですみます。add クラスター操作の場合であれば、FC_SYNC_INTERVAL と入力する代わりに FC と入力しても同じ結果が得られます。

-a 指定された属性のみを表示し、query、add、および modify ACTION でのみ有効です。属性名には大/小文字の区別がなく、標準の UNIX ワイルドカード ("*" および "?") を使用することができます。

-c すべてのデータをコロン区切りのフォーマットで表示し、query、add、および modify ACTION でのみ有効です。

-d *query*、*add*、および *modify* ACTION フラグでのみ有効で、すべてのデータを、指定の区切り文字で区切られた形式で表示するよう要求します。

-D 必要なリソースがクラスター内にまだ定義されていない場合に、デフォルト値を使用してそのリソースを作成しようとする **clmgr** コマンドの依存関係メカニズムを使用不可に設定します。

-f 対話式プロンプトを無効にし、現在の操作を試みることを強制します (操作の強制が可能である場合)。

-h ヘルプ情報を表示します。

-l 保守容易性について以下のトレース・ロギング値をアクティブにします。

- Error : エラーが検出された場合はログ・ファイルの更新のみを行います。
- Standard: clmgr の各操作について基本情報をログに記録します。
- Low: 各関数の入り口および出口の基本トレースを実行します。
- Med: *low* トレースを実行するほか、関数の入り口パラメーター値と関数からの戻り値を追加記録します。
- High: *med* トレースを実行するほか、実行の各行のトレースも記録しますが、ルーチン関数とユーティリティー関数は省略します。

- Max: *high* トレースを実行するほか、ルーチン関数とユーティリティー関数を追加記録します。さらに、関数の開始メッセージと終了メッセージに時刻と日付のタイム・スタンプを追加記録します。

注**:** トレース・データはすべて clutils.log ファイルに書き込まれます。このフラグは、問題をトラブルシューティングするためには理想的なフラグです。

-M
各行に 1 つの操作を指定する形式で、複数の操作を指定し、**clmgr** の 1 回の呼び出しで実行できるようにします。すべての操作は共通のトランザクション ID を共用します。

-S データを列見出しを抑止して表示し、query ACTION および -c フラグでのみ有効です。

-T トランザクション ID は記録される出力すべてに適用されます。これは、1 つ以上のアクティビティーを、分析の目的でログから抽出できる出力の単一部分にグループ化するときに役立ちます。このフラグは、問題をトラブルシューティングするためには理想的なフラグです。

-v 最大の詳細さで情報を出力します。

注**:** query ACTION で特定のオブジェクト名を指定しないでこのフラグを使用すると、指定されたクラスのすべてのインスタンスが表示されます。例えば `clmgr -v query node` と入力すると、すべてのノードとその属性が表示されます。このフラグを add または modify ACTION で使用すると、操作が完了した後の結果の属性が表示されます (この操作が成功した場合のみ)。

-x すべてのデータを単純な XML フォーマットで表示し、query、add、および modify ACTION でのみ有効です。

構文

以下のセクションでは、指定可能なすべての **clmgr** 操作の構文について説明します。

- アプリケーション・コントローラー
- アプリケーション・モニター
- クラスター
- CoD
- 依存関係
- EFS
- イベント
- フォールバック・タイマー
- ファイル・コレクション
- ファイルシステム
- グループ
- HMC
- インターフェース
- LDAP サーバー
- LDAP クライアント
- ログ
- メソッド
- ミラー・グループ

- ミラー・ペア
- ミラー・プール
- ネットワーク
- ノード
- 永続 IP/ラベル
- 物理ボリューム
- レポート
- リポジトリー
- リソース・グループ
- サービス IP/ラベル
- サイト
- スナップショット
- ストレージ・エージェント
- ストレージ・システム
- テープ
- ユーザー
- ボリューム・グループ

クラスター

```
clmgr add cluster ¥
      [ <cluster_label> ] ¥
      [ NODES=<host>[,<host#2>,...] ] ¥
      [ TYPE={NSC|SC} ] ¥
      [ HEARTBEAT_TYPE={unicast|multicast} ] ¥
         [ CLUSTER_IP=<IP_Address> ] ¥
      [ REPOSITORIES=<disk>[,<backup_disk>,...] ] \
      [ FC_SYNC_INTERVAL=## ] ¥
      [ RG_SETTLING_TIME=## ] ¥
      [ MAX_EVENT_TIME=### ] ¥
      [ MAX_RG_PROCESSING_TIME=### ] ¥
      [ DAILY_VERIFICATION={Enabled|Disabled} ] ¥
      [ VERIFICATION_NODE={Default|<node>} ] ¥
      [ VERIFICATION_HOUR=<00..23> ] ¥
      [ VERIFICATION_DEBUGGING={Enabled|Disabled} ] ¥
      [ HEARTBEAT_FREQUENCY=<min...max> ] \
      [ GRACE_PERIOD=<min...max> ] \
      [ SITE_POLICY_FAILURE_ACTION={fallover|notify} ] ¥
      [ SITE_POLICY_NOTIFY_METHOD="<FULL_PATH_TO_FILE>" ] ¥
      [ SITE_HEARTBEAT_CYCLE=<min..max> ] \
      [ SITE_GRACE_PERIOD=<min...max> ] \
      [ TEMP_HOSTNAME={disallow|allow} ] ¥
      [ MONITOR_INTERFACES={enable|disable} ] ¥
      [ NETWORK_FAILURE_DETECTION_TIME=<0..590> ]


clmgr add cluster ¥
      [ <cluster_label> ] ¥
      [ NODES=<host>[,<host#2>,...] ] ¥
      [ TYPE="LC" ] ¥
      [ HEARTBEAT_TYPE={unicast|multicast} ] ¥
      [ FC_SYNC_INTERVAL=## ] ¥
      [ RG_SETTLING_TIME=## ] ¥
      [ MAX_EVENT_TIME=### ] ¥
      [ MAX_RG_PROCESSING_TIME=### ] ¥
      [ DAILY_VERIFICATION={Enabled|Disabled} ] ¥
```

```
[ VERIFICATION_NODE={Default|<node>} ] ¥
[ VERIFICATION_HOUR=<00..23> ] ¥
[ VERIFICATION_DEBUGGING={Enabled|Disabled} ] ¥
[ HEARTBEAT_FREQUENCY=<min...max> ] \
[ GRACE_PERIOD=<min...max> ] \
[ SITE_POLICY_FAILURE_ACTION={fallover|notify} ] ¥
[ SITE_POLICY_NOTIFY_METHOD="<FULL_PATH_TO_FILE>" ]
[ SITE_HEARTBEAT_CYCLE=<min...max> ] \
[ SITE_GRACE_PERIOD=<min...max> ] \
[ TEMP_HOSTNAME={disallow|allow} ] ¥
[ MONITOR_INTERFACES={enable|disable} ] ¥
[ NETWORK_FAILURE_DETECTION_TIME=<0..590> ]
```

表 *4.* 頭字語とその意味

頭字語	意味
NSC	Nonsite cluster (非サイト・クラスター: サイトは定義されません。)
SC	Stretched cluster (拡張クラスター: 限定距離のデータ複製に適した簡易インフラストラクチャー。サイトを定義する必要があります。)
LC	Linked cluster (リンク・クラスター: 長距離のデータ複製に適したフル機能のインフラストラクチャー。サイトを定義する必要があります。)

注**:** *CLUSTER_IP* は、クラスター・タイプが *NSC* または *SC* の場合にのみ使用できます。*LC* クラスターの場合、各サイトにマルチキャスト・アドレスを設定する必要があります。

注**:** *REPOSITORIES* オプションは、クラスター・タイプ *NSC* または *SC* でのみ使用できます。 *LC* クラスターの場合、*REPOSITORIES* オプションはサイトごとに識別されます。 *REPOSITORIES* オプションでは 7 個のディスクを使用できます。最初のディスクはアクティブ・リポジトリー・ディスクであり、以降のディスクはバックアップ・リポジトリー・ディスクです。

```
clmgr modify cluster ¥
             [ NAME=<new_cluster_label> ] ¥
             [ NODES=<host>[,<host#2>,...] ] ¥
             [ TYPE={NSC|SC} ] ¥
             [ HEARTBEAT_TYPE={unicast|multicast} ] ¥
             [ CLUSTER_IP=<IP_Address> ] ¥
             [ REPOSITORIES=<disk>[,<backup_disk>,...] ] \
             [ FC_SYNC_INTERVAL=## ] ¥
             [ RG_SETTLING_TIME=## ] ¥
             [ MAX_EVENT_TIME=### ] ¥
             [ MAX_RG_PROCESSING_TIME=### ] ¥
             [ DAILY_VERIFICATION={Enabled|Disabled} ] ¥
             [ VERIFICATION_NODE={Default|<node>} ] ¥
             [ VERIFICATION_HOUR=<00..23> ] ¥
             [ VERIFICATION_DEBUGGING={Enabled|Disabled} ] ¥
             [ HEARTBEAT_FREQUENCY=<min....max> ] \
             [ GRACE_PERIOD=<min..max> ] \
             [ SITE_POLICY_FAILURE_ACTION={fallover|notify} ] ¥
             [ SITE_POLICY_NOTIFY_METHOD="<FULL_PATH_TO_FILE>" ]
             [ SITE_HEARTBEAT_CYCLE=<min..max> ] \
             [ SITE_GRACE_PERIOD=<min..max> ] \
             [ TEMP_HOSTNAME={disallow|allow} ] ¥
             [ MONITOR_INTERFACES={enable|disable} ] ¥
             [ LPM_POLICY={manage|unmanage} ] ¥
             [ HEARTBEAT_FREQUENCY_DURING_LPM=### ] ¥
             [ NETWORK_FAILURE_DETECTION_TIME=<0,5...590> ] \
             [ CAA_AUTO_START_DR={Enabled|Disabled} ] \
```

```
                [ CAA_DEADMAN_MODE={assert|event} ] \
                [ CAA_REPOS_MODE={assert|event} ] \
                [ CAA_CONFIG_TIMEOUT=<0..2147483647> ]
```

注: *REPOSITORIES* オプションは、クラスター・タイプ *NSC* または *SC* でのみ使用できます。 *LC* クラスターの場合、*REPOSITORIES* オプションはサイトごとに識別されます。 *REPOSITORIES* オプションでは 6 個のバックアップ・リポジトリー・ディスクを使用できます。

```
clmgr modify cluster ¥
                [ NAME=<new_cluster_label> ] ¥
                [ NODES=<host>[,<host#2>,...] ] ¥
                [ TYPE="LC" ] ¥
                [ HEARTBEAT_TYPE={unicast|multicast} ] ¥
                [ FC_SYNC_INTERVAL=## ] ¥
                [ RG_SETTLING_TIME=## ] ¥
                [ MAX_EVENT_TIME=### ] ¥
                [ MAX_RG_PROCESSING_TIME=### ] ¥
                [ DAILY_VERIFICATION={Enabled|Disabled} ] ¥
                [ VERIFICATION_NODE={Default|<node>} ] ¥
                [ VERIFICATION_HOUR=<00..23> ] ¥
                [ VERIFICATION_DEBUGGING={Enabled|Disabled} ] ¥
                [ HEARTBEAT_FREQUENCY=<min...max> ] \
                [ GRACE_PERIOD=<min..max> ] \
                [ SITE_POLICY_FAILURE_ACTION={fallover|notify} ] ¥
                [ SITE_POLICY_NOTIFY_METHOD="<FULL_PATH_TO_FILE>" ]
                [ SITE_HEARTBEAT_CYCLE=<min..max> ] \
                [ SITE_GRACE_PERIOD=<min..max> ] \
                [ TEMP_HOSTNAME={disallow|allow} ] ¥
                [ MONITOR_INTERFACES={enable|disable} ] ¥
                [ LPM_POLICY={manage|unmanage} ] ¥
                [ HEARTBEAT_FREQUENCY_DURING_LPM=### ] ¥
                [ NETWORK_FAILURE_DETECTION_TIME=<0,5...590> ] \
                [ CAA_DEADMAN_MODE={assert|event} ] \
                [ CAA_REPOS_MODE={assert|event} ] \
                [ CAA_CONFIG_TIMEOUT=<0..2147483647> ]
```

```
clmgr modify cluster ¥
      [ SPLIT_POLICY={none|tiebreaker|manual|NFS} ] ¥
      [ TIEBREAKER=<disk> ] ¥
      [ MERGE_POLICY={none|majority|tiebreaker|manual|NFS} ] ¥
      [ NFS_QUORUM_SERVER=<server> ] ¥
      [ LOCAL_QUORUM_DIRECTORY=<local_mount>] ¥
      [ REMOTE_QUORUM_DIRECTORY=<remote_mount>] ¥
      [ QUARANTINE_POLICY=<disable|node_halt|fencing|halt_with_fencing>] ¥
      [ CRITICAL_RG=<rgname> ] ¥
      [ NOTIFY_METHOD=<method> ] ¥
      [ NOTIFY_INTERVAL=### ] ¥
      [ MAXIMUM_NOTIFICATIONS=### ] ¥
      [ DEFAULT_SURVIVING_SITE=<site> ] ¥
      [ APPLY_TO_PPRC_TAKEOVER={yes|no} ] ¥
      [ ACTION_PLAN={reboot|disable_rgs_autostart|disable_cluster_services_autostart} ]
```

注: サイトが完全に定義されて同期された後であって、サイトがすでに使用中の場合は、クラスター・タイプを変更できません。

```
clmgr query cluster [ ALL | {CORE,SECURITY,SPLIT-MERGE,HMC,ROHA} ]
clmgr delete cluster [ NODES={ALL|<node>[,<node#2>,...}] ]
```

注: 削除アクションでは、すべての使用可能ノードからのクラスターの完全削除がデフォルトです。

```
clmgr discover cluster
clmgr recover cluster
clmgr sync cluster ¥
      [ VERIFY={yes|no} ] ¥
```

```
      [ CHANGES_ONLY={no|yes} ] ¥
      [ DEFAULT_TESTS={yes|no} ] ¥
      [ METHODS=<method#1>[,<method#2>,...] ] ¥
      [ FIX={no|yes} ] ¥
      [ LOGGING={standard|verbose} ] ¥
      [ LOGFILE=<PATH_TO_LOG_FILE> ] ¥
      [ MAX_ERRORS=## ] ¥
      [ FORCE={no|yes} ]
```

注: オプションはすべて検証用パラメーターです。よって、VERIFY が yes に設定されている場合にのみ有効です。

```
clmgr manage cluster {reset|unlock}

clmgr manage cluster security ¥
      [ LEVEL={Disable|Low|Med|High} ] ¥
      [ ALGORITHM={DES|3DES|AES} ]¥
      [ MECHANISM={OpenSSL|SSH} ] ¥
         CERTIFICATE=<PATH_TO_FILE> ¥
         PRIVATE_KEY=<PATH_TO_FILE>
```

注: SSL または SSH の MECHANISM を指定した場合は、ユーザー定義の証明書ファイルと秘密鍵ファイルを指定する必要があります。

```
clmgr manage cluster security ¥
      [ LEVEL={Disable|Low|Med|High} ] ¥
      [ ALGORITHM={DES|3DES|AES} ]¥
      [ MECHANISM="SelfSigned" ] ¥
      [ CERTIFICATE=<PATH_TO_FILE> ] ¥
      [ PRIVATE_KEY=<PATH_TO_FILE> ]
```

注: 自己署名の MECHANISM を指定した場合、証明書ファイルおよび秘密鍵ファイルの指定はオプションです。両方ともに指定されなければ、自動的にデフォルトのペアが生成されます。
GRACE_PERIOD のデフォルトは 21600 秒 (6 時間) です。REFRESH のデフォルトは 86400 秒 (24 時間) です。

```
clmgr manage cluster hmc ¥
            [ DEFAULT_HMC_TIMEOUT=<MINUTES> ] ¥
            [ DEFAULT_HMC_RETRY_COUNT=<INTEGER> ] ¥
            [ DEFAULT_HMC_RETRY_DELAY=<SECONDS> ] ¥
            [ DEFAULT_HMCS_LIST=<HMCS> ]

clmgr manage cluster roha ¥
            [ ALWAYS_START_RG={YES|NO} ] ¥
            [ ADJUST_SPP_SIZE={YES|NO} ]¥
            [ FORCE_SYNC_RELEASE={YES|NO} ] ¥
            [ AGREE_TO_COD_COSTS={YES|NO} ] ] ¥
            [ ONOFF_DAYS=<DAYS>} ]
            [ RESOURCE_ALLOCATION_ORDER={enterprise_pool_first|free_pool_first} ]


clmgr verify cluster ¥
      [ CHANGES_ONLY={no|yes} ] ¥
      [ DEFAULT_TESTS={yes|no} ] ¥
      [ METHODS=<method#1>[,<method#2>,...] ] ¥
      [ FIX={no|yes} ] ¥
      [ LOGGING={standard|verbose} ] ¥
      [ LOGFILE=<PATH_TO_LOG_FILE> ] ¥
      [ MAX_ERRORS=## ]
      [ SYNC={no|yes} ] ¥
      [ FORCE={no|yes} ]
```

注: FORCE オプションは SYNC が yes に設定されている場合に使用できます。

```
clmgr offline cluster ¥
      [ WHEN={now|restart|both} ] ¥
      [ MANAGE={offline|move|unmanage} ] ¥
      [ BROADCAST={true|false} ] ¥
      [ TIMEOUT=<seconds_to_wait_for_completion> ]
      [ STOP_CAA={no|yes} ]
clmgr online cluster ¥
      [ WHEN={now|restart|both} ] ¥
      [ MANAGE={auto|manual} ] ¥
      [ BROADCAST={false|true} ] ¥
      [ CLINFO={false|true|consistent} ] ¥
      [ FORCE={false|true} ] ¥
      [ FIX={no|yes|interactively} ] ¥
      [ TIMEOUT=<seconds_to_wait_for_completion> ] ¥
      [ START_CAA={no|yes|only} ]
```

注: RG_SETTLING_TIME 属性は、始動ポリシーが「最初に使用可能なノードでオンライン」であるリソース・グループのみに影響します。cluster のエイリアスは cl です。

注: STOP_CAA オプションおよび START_CAA オプションは、Cluster Aware AIX (CAA) クラスター・サービスをオフラインまたはオンラインにします。これらのオプションは、具体的な既知のニーズがある場合、または IBM® サポートの指示があった場合に使用します。CAA クラスター・サービスは、非アクティブにしないでください。クラスター化環境で問題を検出する機能が無効になるからです。only オプションは、CAA サービスのみ開始します。

リポジトリー

```
clmgr add repository <disk>[,<backup_disk#2>,...] ¥
      [ SITE=<site_label> ]¥
      [ NODE=<reference_node> ]
```

注: アクティブ・リポジトリーがまだ定義されていない場合は、最初のディスクがアクティブ・リポジトリーとして使用されます。リスト内のその他のディスクは、すべてバックアップ・リポジトリー・ディスクとして定義されます。標準クラスターおよび拡張クラスターのクラスターごとに、最大 6 つのバックアップ・リポジトリー・ディスクを指定できます。リンク・クラスターのサイトごとに、最大 6 つのバックアップ・リポジトリー・ディスクを指定できます。

```
clmgr replace repository [ <new_repository> ]  ¥
      [ SITE=<site_label> ]¥
      [ NODE=<reference_node>]
```

注: ディスクが指定されていない場合は、バックアップ・リストの最初のディスクが使用されます。

```
clmgr query repository [ <disk>[,<disk#2>,...] ]
clmgr delete repository {<backup_disk>[,<disk#2>,...] | ALL}¥
      [ SITE=<site_label> ]¥
      [ NODE=<reference_node> ]
```

注: アクティブ・リポジトリー・ディスクを削除することはできません。バックアップ・リポジトリーのみを除去できます。

サイト

```
clmgr add site <sitename> ¥
      NODES=<node>[,<node#2>,...] ¥
      [ SITE_IP=<multicast_address> ] ¥
      [ RECOVERY_PRIORITY={MANUAL|1|2} ] ¥
      [ REPOSITORIES=<disk>[,<backup_disk>,...] ]
```

注: *REPOSITORIES* オプションは、クラスター・タイプ *LC* でのみ使用できます。 *REPOSITORIES* オプションでは 7 個のディスクを使用できます。最初のディスクはアクティブ・リポジトリー・ディスクであり、以降のディスクはバックアップ・リポジトリー・ディスクです。

```
clmgr modify site <sitename> ¥
      [ NAME=<new_site_label> ] ¥
      [ NODES=<node>[,<node#2>,...] ] ¥
      [ SITE_IP=<multicast_address> ] ¥
      [ RECOVERY_PRIORITY={MANUAL|1|2} ] ¥
      [ REPOSITORIES=<backup_disk>[,<backup_disk>,...] ] \
      [ HMCS=<hmc>[,<hmc#2>,...] ]
```

注: *SITE_IP* 属性は、クラスター・タイプが *LC* (リンク・クラスター)、クラスター・ハートビート・タイプが *multicast* である場合にのみ使用できます。

注: *REPOSITORIES* オプションは、クラスター・タイプ *LC* でのみ使用できます。 *REPOSITORIES* オプションでは 6 個のバックアップ・リポジトリー・ディスクを使用できます。

```
clmgr query site [ <sitename>[,<sitename#2>,...] ]
clmgr delete site {<sitename>[,<sitename#2>,...] | ALL}
clmgr offline site <sitename> ¥
      [ WHEN={now|restart|both} ] ¥
      [ MANAGE={offline|move|unmanage} ] ¥
      [ BROADCAST={true|false} ] ¥
      [ TIMEOUT=<seconds_to_wait_for_completion> ] ¥
      [ STOP_CAA={no|yes} ]
clmgr online site <sitename> ¥
      [ WHEN={now|restart|both} ] ¥
      [ MANAGE={auto|manual} ] ¥
      [ BROADCAST={false|true} ] ¥
      [ CLINFO={false|true|consistent} ] ¥
      [ FORCE={false|true} ] ¥
      [ FIX={no|yes|interactively} ] ¥
      [ TIMEOUT=<seconds_to_wait_for_completion> ] ¥
      [ START_CAA={no|yes|only} ]
clmgr manage site respond {continue|recover}
```

注: *site* のエイリアスは st です。

注: STOP_CAA オプションおよび START_CAA オプションは、Cluster Aware AIX (CAA) クラスター・サービスをオフラインまたはオンラインにします。これらのオプションは、具体的な既知のニーズがある場合、または IBM サポートの指示があった場合に使用します。CAA クラスター・サービスは、非アクティブにしないでください。クラスター化環境で問題を検出する機能が無効になるからです。only オプションは、CAA サービスのみ開始します。

ノード

```
clmgr add node <node> ¥
      [ COMMPATH=<ip_address_or_network-resolvable_name> ] ¥
      [ RUN_DISCOVERY={true|false} ] ¥
      [ PERSISTENT_IP=<IP> NETWORK=<network>
        {NETMASK=<255.255.255.0 | PREFIX=1..128} ] ¥
      [ START_ON_BOOT={false|true} ] ¥
      [ BROADCAST_ON_START={true|false} ] ¥
      [ CLINFO_ON_START={false|true|consistent} ] ¥
      [ VERIFY_ON_START={true|false} ] ¥
      [ SITE=<sitename> ]
clmgr modify node <node> ¥
      [ NAME=<new_node_label> ] ¥
      [ COMMPATH=<new_commpath> ] ¥
      [ PERSISTENT_IP=<IP> NETWORK=<network>
```

```
        {NETMASK=<255.255.255.0 | PREFIX=1..128} ] ¥
      [ START_ON_BOOT={false|true} ] ¥
      [ BROADCAST_ON_START={true|false} ] ¥
      [ CLINFO_ON_START={false|true|consistent} ] ¥
      [ VERIFY_ON_START={true|false} ] ¥
      [ HMCS=<hmc>[,<hmc#2>,...] ] \
      [ ENABLE_LIVE_UPDATE={true|false} ]
clmgr query node [ {<node>|LOCAL}[,<node#2>,...] ]
clmgr delete node {<node>[,<node#2>,...] | ALL}
clmgr manage node undo_changes
clmgr recover node <node>[,<node#2>,...]
clmgr online node <node>[,<node#2>,...] ¥
      [ WHEN={now|restart|both} ] ¥
      [ MANAGE={auto|manual} ] ¥
      [ BROADCAST={false|true} ] ¥
      [ CLINFO={false|true|consistent} ] ¥
      [ FORCE={false|true} ] ¥
      [ FIX={no|yes|interactively} ] ¥
      [ TIMEOUT=<seconds_to_wait_for_completion> ] ¥
      [ START_CAA={no|yes|only} ]
clmgr offline node <node>[,<node#2>,...] ¥
      [ WHEN={now|restart|both} ] ¥
      [ MANAGE={offline|move|unmanage} ] ¥
      [ BROADCAST={true|false} ] ¥
      [ TIMEOUT=<seconds_to_wait_for_completion> ] ¥
      [ STOP_CAA={no|yes} ]
```

注: TIMEOUT 属性のデフォルトは 120 秒です。*node* のエイリアスは no です。

注: STOP_CAA オプションおよび START_CAA オプションは、Cluster Aware AIX (CAA) クラスター・サービスをオフラインまたはオンラインにします。これらのオプションは、具体的な既知のニーズがある場合、または IBM サポートの指示があった場合に使用します。CAA クラスター・サービスは、非アクティブにしないでください。クラスター化環境で問題を検出する機能が無効になるからです。only オプションは、CAA サービスのみ開始します。

ネットワーク

```
clmgr add network <network> ¥
      [ TYPE={ether|XD_data|XD_ip} ] ¥
      [ {NETMASK=<255.255.255.0 | PREFIX=1..128} ] ¥
      [ IPALIASING={true|false} ] ¥
      [ PUBLIC={true|false} ]
```

注: デフォルトでは、IPv4 ネットワークはネットマスク *255.255.255.0* を使用して構成されます。IPv6 ネットワークを作成するには、有効なプレフィックスを指定してください。

```
clmgr modify network <network> ¥
      [ NAME=<new_network_label> ] ¥
      [ TYPE={ether|XD_data|XD_ip} ] ¥
      [ {NETMASK=<255.255.255.0> | PREFIX=1..128} ] ¥
      [ PUBLIC={true|false} ] ¥
      [ RESOURCE_DIST_PREF={AC|ACS|C|CS|CPL|ACPL|ACPLS|NOALI} ] ¥
        [ SOURCE_IP=<service_or_persistent_ip> ]
```

注: RESOURCE_DIST_PREF 属性に指定できる値は、次のとおりです。

AC アンチコロケーション

ACS
ソースを持つアンチコロケーション

C コロケーション

CS ソースを持つコロケーション

CPL
永続ラベルを持つコロケーション

ACPL
永続ラベルを持つアンチコロケーション

ACPLS
永続ラベルとソースを持つアンチコロケーション

NOALI
最初のエイリアスを使用不可にする

注: RESOURCE_DIST_PREF 属性が CS または ACS の値を使用する場合、SOURCE_IP 属性はサービス・ラベルであることが必要です。

```
clmgr query network [ <network>[,<network#2>,...] ]
clmgr delete network {<network>[,<network#2>,...] | ALL}
```

注: *network* のエイリアスは ne および nw です。

インターフェース

```
clmgr add interface <interface> ¥
      NETWORK=<network> ¥
      [ NODE=<node> ] ¥
      [ TYPE={ether|XD_data|XD_ip} ] ¥
      [ INTERFACE=<network_interface> ]
clmgr modify interface <interface> ¥
      NETWORK=<network>
clmgr query interface [ <interface>[,<if#2>,...] ]
clmgr delete interface {<interface>[,<if#2>,...] | ALL}
clmgr discover interfaces
```

注: interface は IP アドレスか IP ラベルのいずれかです。NODE 属性のデフォルトはローカル・ノード名です。 TYPE 属性のデフォルトは ether です。<network_interface> は en1、en2、en3 のようになります。*interface* のエイリアスは in および if です。

リソース・グループ

```
clmgr add resource_group <resource_group>[,<rg#2>,...] \
NODES=nodeA1,nodeA2,...                                 ¥
[ SECONDARYNODES=nodeB2[,nodeB1,...]                  ] ¥
[ SITE_POLICY={ignore|primary|either|both}            ] ¥
[ STARTUP={OHN|OFAN|OAAN|OUDP}                        ] ¥
[ FALLOVER={FNPN|FUDNP|BO}                            ] ¥
[ FALLBACK={NFB|FBHPN}                                ] ¥
  [ FALLBACK AT=<FALLBACK TIMER>                    ] ¥
[ NODE_PRIORITY_POLICY={default|mem|cpu|                ¥
                        disk|least|most}              ] ¥
[ NODE_PRIORITY_POLICY_SCRIPT=</path/to/script>       ] ¥
[ NODE_PRIORITY_POLICY_TIMEOUT=###                    ] ¥
[ SERVICE_LABEL=service_ip#1[,service_ip#2,...]       ] ¥
[ APPLICATIONS=appctlr#1[,appctlr#2,...]              ] ¥
[ SHARED_TAPE_RESOURCES=<TAPE>[,<TAPE#2>,...]         ] ¥
[ VOLUME_GROUP=<VG>[,<VG#2>,...]                      ] ¥
[ FORCED_VARYON={true|false}                          ] ¥
[ VG_AUTO_IMPORT={true|false}                         ] ¥
[ FILESYSTEM=/file_system#1[,/file_system#2,...]      ] ¥
[ DISK=<raw_disk>[,<raw_disk#2>,...]                  ] ¥
[ FS_BEFORE_IPADDR={true|false}                       ] ¥
[ WPAR_NAME="wpar_name"                               ] ¥
[ EXPORT_FILESYSTEM=/expfs#1[,/expfs#2,...]           ] ¥
[ EXPORT_FILESYSTEM_V4=/expfs#1[,/expfs#2,...]        ] ¥
```

```
      [ STABLE_STORAGE_PATH="/fs3"                       ] ¥
      [ NFS_NETWORK="nfs_network"                        ] ¥
      [ MOUNT_FILESYSTEM=/nfs_fs1;/expfs1,/nfs_fs2;,... ] ¥
      [ MIRROR_GROUP=<replicated_resource>               ] ¥
      [ FALLBACK_AT=<FALLBACK_TIMER>                     ]

STARTUP:
    OHN ----- Online Home Node (default value)
    OFAN ---- Online on First Available Node
    OAAN ---- Online on All Available Nodes (concurrent)
    OUDP ---- Online Using Node Distribution Policy

FALLOVER:
    FNPN ---- Fallover to Next Priority Node (default value)
    FUDNP --- Fallover Using Dynamic Node Priority
    BO ------ Bring Offline (On Error Node Only)

FALLBACK:
    NFB ----- Never Fallback
    FBHPN --- Fallback to Higher Priority Node (default value)

NODE_PRIORITY_POLICY:
    default - next node in the NODES list
    mem ----- node with most available memory
    disk ---- node with least disk activity
    cpu ----- node with most available CPU cycles
    least --- node where the dynamic node priority script
              returns the lowest value
    most ---- node where the dynamic node priority script
              returns the highest value
```

注: NODE_PRIORITY_POLICY ポリシーが確立されるのは、FALLOVER ポリシーが FUDNP に設定されている場合のみです。

```
SITE_POLICY:
     ignore -- 無視
     primary - 1 次サイトを優先
     either -- Online on Either Site (一方のサイトでオンライン)
     both ---- 両方のサイトでオンライン

clmgr modify resource_group <resource_group>              ¥
      [ NAME=<new_resource_group_label>               ] ¥
      [ NODES=nodeA1[,nodeA2,...]                       ] ¥
      [ SECONDARYNODES=nodeB2[,nodeB1,...]              ] ¥
      [ SITE_POLICY={ignore|primary|either|both}        ] ¥
      [ STARTUP={OHN|OFAN|OAAN|OUDP}                    ] ¥
      [ FALLOVER={FNPN|FUDNP|BO}                        ] ¥
      [ FALLBACK={NFB|FBHPN}                            ] ¥
      [ FALLBACK_AT=<FALLBACK_TIMER>                    ] ¥
      [ NODE_PRIORITY_POLICY={default|mem|cpu|            ¥
                              disk|least|most}          ] ¥
      [ NODE_PRIORITY_POLICY_SCRIPT=</path/to/script>   ] ¥
      [ NODE_PRIORITY_POLICY_TIMEOUT=###                ] ¥
      [ SERVICE_LABEL=service_ip#1[,service_ip#2,...]   ] ¥
      [ APPLICATIONS=appctlr#1[,appctlr#2,...]          ] ¥
      [ VOLUME_GROUP=volume_group#1[,volume_group#2,...]] ¥
      [ FORCED_VARYON={true|false}                      ] ¥
      [ VG_AUTO_IMPORT={true|false}                     ] ¥
      [ FILESYSTEM=/file_system#1[,/file_system#2,...]  ] ¥
      [ DISK=<raw_disk>[,<raw_disk#2>,...]              ] ¥
      [ FS_BEFORE_IPADDR={true|false}                   ] ¥
      [ WPAR_NAME="wpar_name"                           ] ¥
      [ EXPORT_FILESYSTEM=/expfs#1[,/expfs#2,...]       ] ¥
      [ EXPORT_FILESYSTEM_V4=/expfs#1[,/expfs#2,...]    ] ¥
      [ STABLE_STORAGE_PATH="/fs3"                      ] ¥
      [ NFS_NETWORK="nfs_network"                       ] ¥
```

```
      [ MOUNT_FILESYSTEM=/nfs_fs1;/expfs1,/nfs_fs2;,... ] ¥
      [ MIRROR_GROUP=<replicated_resource>              ] ¥
      [ FALLBACK_AT=<FALLBACK_TIMER>                    ]
```

注: appctlr という値は application_controller の省略語です。

```
clmgr query resource_group [ <resource_group>[,<rg#2>,...] ]
clmgr delete resource_group {<resource_group>[,<rg#2>,...] |
                             ALL}
clmgr online { resource_group <resource_group>[,<rg#2>,...] | ALL} ¥
      [ NODES={<node>[,<node#2>,...] | ALL}]
clmgr offline resource_group {<resource_group>[,<rg#2>,...] | ALL} ¥
      [ NODES={<node>[,<node#2>,...] | ALL} ]
```

注: NODES 属性の特別な ALL ターゲットは、コンカレント・リソース・グループにのみ適用可能です。

```
clmgr move resource_group <resource_group>[,<rg#2>,...] ¥
      {NODE|SITE}=<node_or_site_label> ¥
      [ SECONDARY={false|true} ] ¥
      [ STATE={online|offline} ] ¥
```

注: *SITE* 属性および *SECONDARY* 属性は、クラスターにサイトが構成されている場合にのみ適用可能です。*STATE* が明示的に指定されていない場合は、リソース・グループ *STATE* は変更されないままになります。*resource_group* のエイリアスは rg です。

フォールバック・タイマー

```
clmgr add fallback_timer <timer> ¥
      [ YEAR=<####> ] ¥
      [ MONTH=<{1..12 | Jan..Dec}> ] ¥
      [ DAY_OF_MONTH=<{1..31}> ] ¥
      [ DAY_OF_WEEK=<{0..6 | Sun..Sat}> ] ¥
        HOUR=<{0..23}> ¥
        MINUTE=<{0..59}>
clmgr modify fallback_timer <timer> ¥
      [ YEAR=<{####}> ] ¥
      [ MONTH=<{1..12 | Jan..Dec}> ] ¥
      [ DAY_OF_MONTH=<{1..31}> ] ¥
      [ DAY_OF_WEEK=<{0..6 | Sun..Sat}> ] ¥
      [ HOUR=<{0..23}> ] ¥
      [ MINUTE=<{0..59}> ] ¥
      [ REPEATS=<{0,1,2,3,4 |
        Never,Daily,Weekly,Monthly,Yearly}> ]
clmgr query fallback_timer [<timer>[,<timer#2>,...] ]
clmgr delete fallback_timer {<timer>[,<timer#2>,...] |¥
                            ALL}
```

注: *fallback_timer* のエイリアスは fa および timer です。

永続 IP/ラベル

```
clmgr add persistent_ip <persistent_IP> ¥
      NETWORK=<network> ¥
      [ {NETMASK=< 255.255.255.0 | PREFIX=1..128} ] ¥ ]
      [ NODE=<node>   ]
clmgr modify persistent_ip <persistent_label> ¥
      [ NAME=<new_persistent_label> ] ¥
      [ NETWORK=<new_network> ] ¥
      [ NETMASK=<node> 255.255.255.0 | PREFIX=1..128} ] ¥ ]
```

注: 基盤となるネットワークで別のプロトコル (IPv4 の場合に IPv6、または IPv6 の場合に IPv4) を使用している場合を除き、NETMASK または PREFIX に指定した値は無視されます。別のプロトコルを使用している場合は、NETMASK または PREFIX は必須です。

```
clmgr query persistent_ip [ <persistent_IP>[,<pIP#2>,...] ]
clmgr delete persistent_ip {<persistent_IP>[,<pIP#2>,...] |
                            ALL}
clmgr move persistent_ip <persistent_IP> ¥
                         INTERFACE=<new_interface>
```

注: *persistent_ip* のエイリアスは pe です。

サービス IP/ラベル

```
clmgr add service_ip <service_ip> ¥
      NETWORK=<network> ¥
      [ {NETMASK=<255.255.255.0 | PREFIX=1..128} ] ¥
      [ HWADDR=<new_hardware_address> ] ¥
      [ SITE=<new_site> ]
clmgr modify service_ip <service_ip> ¥
      [ NAME=<new_service_ip> ] ¥
      [ NETWORK=<new_network> ] ¥
      [ {NETMASK=<###.###.###.###> | PREFIX=1..128} ] ¥
      [ HWADDR=<new_hardware_address> ] ¥
      [ SITE=<new_site> ]
clmgr query service_ip [ <service_ip>[,<service_ip#2>,...] ]
clmgr delete service_ip {<service_ip>[,<service_ip#2>,,...] | ALL}
clmgr move service_ip <service_ip> ¥
      INTERFACE=<new_interface>
```

注: NETMASK/PREFIX 属性を指定しない場合、基礎となるネットワークのネットマスク値とプレフィックス値が使用されます。*service_ip* のエイリアスは si です。

アプリケーション・コントローラー

```
clmgr add application_controller <application_controller> ¥
      STARTSCRIPT="/path/to/start/script" ¥
      STOPSCRIPT ="/path/to/stop/script" ¥
      [ MONITORS=<monitor>[,<monitor#2>,...] ] \
      [ STARTUP_MODE={background|foreground}
clmgr modify application_controller <application_controller> ¥
      [ NAME=<new_application_controller_label> ] ¥
      [ STARTSCRIPT="/path/to/start/script" ] ¥
      [ STOPSCRIPT ="/path/to/stop/script" ] ¥
      [ MONITORS=<monitor>[,<monitor#2>,...] ] \
      [ STARTUP_MODE={background|foreground}
clmgr query application_controller [ <appctlr>[,<appctlr#2>,...] ]
clmgr delete application_controller {<appctlr>[,<appctlr#2>,...] | ¥
                                     ALL}
clmgr manage application_controller {suspend|resume} ¥
      <application_controller> ¥
      RESOURCE_GROUP=<resource_group>
clmgr manage application_controller {suspend|resume} ALL
```

注: *appctlr* という値は *application_controller* の省略語です。 *application_controller* のエイリアスは ac および app です。

アプリケーション・モニター

```
clmgr add application_monitor <monitor> ¥
      TYPE=Process ¥
      MODE={longrunning|startup|both} ¥
      PROCESSES="pmon1,dbmon,..."  \
```

```
      OWNER="<processes_owner_name>" ¥
      [ APPLICATIONS=<appctlr#1>[,<appctlr#2>,...] ] \
      [ STABILIZATION="1 .. 3600" ] ¥
      [ RESTARTCOUNT="0 .. 100" ] ¥
      [ FAILUREACTION={notify|fallover} ] ¥
      [ INSTANCECOUNT="1 .. 1024" ] ¥
      [ RESTARTINTERVAL="1 .. 3600" ] ¥
      [ NOTIFYMETHOD="</script/to/notify>" ] ¥
      [ CLEANUPMETHOD="</script/to/cleanup>" ] ¥
      [ RESTARTMETHOD="</script/to/restart>" ]

clmgr add application_monitor <monitor> ¥
      TYPE=Custom ¥
      MODE={longrunning|startup|both} ¥
      MONITORMETHOD="/script/to/monitor" ¥
      [ APPLICATIONS=<appctlr#1>[,<appctlr#2>,...] ] \
      [ STABILIZATION="1 .. 3600" ] ¥
      [ RESTARTCOUNT="0 .. 100" ] ¥
      [ FAILUREACTION={notify|fallover} ] ¥
      [ MONITORINTERVAL="1 .. 1024" ] ¥
      [ HUNGSIGNAL="1 .. 63" ] ¥
      [ RESTARTINTERVAL="1 .. 3600" ] ¥
      [ NOTIFYMETHOD="</script/to/notify>" ] ¥
      [ CLEANUPMETHOD="</script/to/cleanup>" ] ¥
      [ RESTARTMETHOD="</script/to/restart>" ]
```

注: STABILIZATION のデフォルトは 180 です。RESTARTCOUNT のデフォルトは 3 です。

```
clmgr modify application_monitor <monitor> ¥
      [ See the "add" action, above, for a list
        of supported modification attributes. ]
clmgr query application_monitor [ <monitor>[,<monitor#2>,...] ]
clmgr delete application_monitor {<monitor>[,<monitor#2>,...] | ALL}
```

注: *appctlr* という値は *application_controller* の省略語です。 *application_monitor* のエイリアスは am および mon です。

依存関係

```
# Temporal Dependency (parent ==> child)
clmgr add dependency ¥
      PARENT=<rg#1> ¥
      CHILD="<rg#2>[,<rg#2>,...]"
clmgr modify dependency <parent_child_dependency> ¥
      [ TYPE=PARENT_CHILD ] ¥
      [ PARENT=<rg#1> ] ¥
      [ CHILD="<rg#2>[,<rg#2>,...]" ]

# Temporal Dependency (start/stop after)
clmgr add dependency ¥
      {STOP|START}="<rg#2>[,<rg#2>,...]" ¥
      AFTER=<rg#1>
clmgr modify dependency ¥
      [ TYPE={STOP_AFTER|START_AFTER} ] ¥
      [ {STOP|START}="<rg#2>[,<rg#2>,...]" ] ¥
      [ AFTER=<rg#1> ]

# Location Dependency (colocation)
clmgr add dependency ¥
      SAME={NODE|SITE } ¥
      GROUPS="<rg1>,<rg2>[,<rg#n>,...]"
clmgr modify dependency <colocation_dependency> ¥
      [ TYPE={SAME_NODE|SAME_SITE} ] ¥
      GROUPS="<rg1>,<rg2>[,<rg#n>,...]"
```

```
# Location Dependency (anti-colocation)
clmgr add dependency ¥
      HIGH="<rg1>,<rg2>,..." \
      INTERMEDIATE="<rg3>,<rg4>,..." \
      LOW="<rg5>,<rg6>,..."
clmgr modify dependency <anti-colocation_dependency> ¥
      [ TYPE=DIFFERENT_NODES ] ¥
      [ HIGH="<rg1>,<rg2>,..." ] \
      [ INTERMEDIATE="<rg3>,<rg4>,..." ] \
      [ LOW="<rg5>,<rg6>,..." ]

# Acquisition/Release Order
clmgr add dependency ¥
      TYPE={ACQUIRE|RELEASE} ¥
      { SERIAL="{<rg1>,<rg2>,...|ALL}" |
      PARALLEL="{<rg1>,<rg2>,...|ALL}" }
clmgr modify dependency ¥
      TYPE={ACQUIRE|RELEASE} ¥
      { SERIAL="{<rg1>,<rg2>,...|ALL}" |
      PARALLEL="{<rg1>,<rg2>,...|ALL}" }

clmgr query dependency [ <dependency> ]
clmgr delete dependency {<dependency> | ALL} ¥
      [ TYPE={PARENT_CHILD|STOP_AFTER|START_AFTER| ¥
      SAME_NODE|SAME_SITE|DIFFERENT_NODES} ]
clmgr delete dependency RESOURCE_GROUP=<RESOURCE_GROUP>
```

注: *dependency* のエイリアスは de です。

テープ

```
clmgr add tape <tape> ¥
      DEVICE=<tape_device_name> ¥
      [ DESCRIPTION=<tape_device_description> ] ¥
      [ STARTSCRIPT="</script/to/start/tape/device>" ] ¥
      [ START_SYNCHRONOUSLY={no|yes} ] ¥
      [ STOPSCRIPT="</script/to/stop/tape/device>" ] ¥
      [ STOP_SYNCHRONOUSLY={no|yes} ]
clmgr modify tape <tape> ¥
      [ NAME=<new_tape_label> ] ¥
      [ DEVICE=<tape_device_name> ] ¥
      [ DESCRIPTION=<tape_device_description> ] ¥
      [ STARTSCRIPT="</script/to/start/tape/device>" ] ¥
      [ START_SYNCHRONOUSLY={no|yes} ] ¥
      [ STOPSCRIPT="</script/to/stop/tape/device>" ] ¥
      [ STOP_SYNCHRONOUSLY={no|yes} ]
clmgr query tape [ <tape>[,<tape#2>,...] ]
clmgr delete tape {<tape> | ALL}
```

注: *tape* のエイリアスは tp です。

ファイル・コレクション

```
clmgr add file_collection <file_collection> ¥
      FILES="/path/to/file1,/path/to/file2,..." \
      [ SYNC_WITH_CLUSTER={no|yes}  ] ¥
      [ SYNC_WHEN_CHANGED={no|yes} ] ¥
      [ DESCRIPTION="<file_collection_description>" ]
clmgr modify file_collection <file_collection> ¥
      [ NAME="<new_file_collection_label>" ] ¥
      [ ADD="/path/to/file1,/path/to/file2,..." ] \
      [ DELETE={"/path/to/file1,/path/to/file2,..."|ALL} ] ¥
      [ REPLACE={"/path/to/file1,/path/to/file2,..."|""} ] ¥
      [ SYNC_WITH_CLUSTER={no|yes}  ] ¥
      [ SYNC_WHEN_CHANGED={no|yes} ] ¥
```

```
      [ DESCRIPTION="<file_collection_description>" ]
clmgr query file_collection [ <file_collection>[,<fc#2>,...]]
clmgr delete file_collection {<file_collection>[,<fc#2>,...]|
                              ALL}
clmgr sync file_collection <file_collection>
```

注: REPLACE 属性は、既存のすべてのファイルを指定したセットで置き換えます。*file_collection* のエイリアスは fc および fi です。

Snapshot

```
clmgr add snapshot <snapshot> ¥
      DESCRIPTION="<snapshot_description>" ¥
      [ METHODS="method1,method2,..." ]
clmgr add snapshot <snapshot> TYPE="xml"
clmgr modify snapshot <snapshot> ¥
      [ NAME="<new_snapshot_label>" ] ¥
      [ DESCRIPTION="<snapshot_description>" ]
clmgr query snapshot [ <snapshot>[,<snapshot#2>,...] ]
clmgr view snapshot <snapshot> ¥
      [ TAIL=<number_of_trailing_lines> ] ¥
      [ HEAD=<number_of_leading_lines> ] ¥
      [ FILTER=<pattern>[,<pattern#2>,...] ] ¥
      [ DELIMITER=<alternate_pattern_delimiter> ] ¥
      [ CASE={insensitive|no|off|false} ]
clmgr delete snapshot {<snapshot>[,<snapshot#2>,...] |
                       ALL}
clmgr manage snapshot restore <snapshot> ¥
      [ CONFIGURE={yes|no} ] ¥
      [ FORCE={no|yes} ]
```

注: view アクションは、スナップショットの .info ファイルがあればその内容を表示します。*snapshot* のエイリアスは sn および ss です。

```
clmgr manage snapshot restore <snapshot> ¥
      NODES=<HOST>,<HOST#2> ¥
      REPOSITORIES=<DISK>[,<BACKUP>][:<DISK>[,<BACKUP>]] ¥
      [ CLUSTER_NAME=<NEW_CLUSTER_LABEL> ] ¥
      [ CONFIGURE={yes|no} ] ¥
      [ FORCE={no|yes} ]
```

注: REPOSITORIES オプションに対して、コロンの後に指定されたディスクは 2 番目のサイトに適用されます。リンク・クラスター・スナップショットを復元する場合は、REPOSITORIES オプションのコロンの後に指定されたディスクは 2 番目のサイトに適用されます。

メソッド

```
clmgr add method <method_label> ¥
      TYPE=snapshot ¥
      FILE=<executable_file> ¥
      [ DESCRIPTION=<description> ]
clmgr add method <method_label> ¥
      TYPE=verify ¥
      FILE=<executable_file> ¥
      [ SOURCE={script|library} ] ¥
      [ DESCRIPTION=<description> ]
clmgr modify method <method_label> ¥
      TYPE={snapshot|verify} ¥
      [ NAME=<new_method_label> ] ¥
      [ DESCRIPTION=<new_description> ] ¥
      [ FILE=<new_executable_file> ]
clmgr add method <method_label> ¥
      TYPE=notify ¥
      CONTACT=<number_to_dial_or_email_address> ¥
      EVENT=<event>[,<event#2>,...] ¥
```

```
      [ NODES=<node>[,<node#2>,...] ] ¥
      [ FILE=<message_file> ] ¥
      [ DESCRIPTION=<description> ] ¥
      [ RETRY=<retry_count> ] ¥
      [ TIMEOUT=<timeout> ]
```

注: NODES のデフォルトはローカル・ノードです。

```
clmgr modify method <method_label> ¥
      TYPE=notify ¥
      [ NAME=<new_method_label> ] ¥
      [ DESCRIPTION=<description> ] ¥
      [ FILE=<message_file> ] ¥
      [ CONTACT=<number_to_dial_or_email_address> ] ¥
      [ EVENT=<cluster_event_label> ] ¥
      [ NODES=<node>[,<node#2>,...] ] ¥
      [ RETRY=<retry_count> ] ¥
      [ TIMEOUT=<timeout> ]

clmgr query method [ <method>[,<method#2>,...] ] ¥
      [ TYPE={notify|snapshot|verify} ]
clmgr delete method {<method>[,<method#2>,...] | ALL} ¥
      [ TYPE={notify|snapshot|verify} ]
clmgr verify method <method>
```

注: verify アクションは notify メソッドに対してのみ適用できます。複数のメソッドが同じイベントを活用する場合にそのイベントを指定すると、両方のメソッドが呼び出されます。*method* のエイリアスは me です。

ログ

```
clmgr modify logs ALL DIRECTORY="<new_logs_directory>"
clmgr modify log {<log>|ALL} ¥
      [ DIRECTORY="{<new_log_directory>"|DEFAULT} ]
      [ FORMATTING={none|standard|low|high} ] ¥
      [ TRACE_LEVEL={low|high} ]
      [ REMOTE_FS={true|false} ]
clmgr query log [ <log>[,<log#2>,...] ]
clmgr view log [ {<log>|EVENTS} ] ¥
      [ TAIL=<number_of_trailing_lines> ] ¥
      [ HEAD=<number_of_leading_lines> ] ¥
      [ FILTER=<pattern>[,<pattern#2>,...] ] ¥
      [ DELIMITER=<alternate_pattern_delimiter> ] ¥
      [ CASE={insensitive|no|off|false} ]
clmgr manage logs collect ¥
      [ DIRECTORY="<directory_for_collection>" ] ¥
      [ NODES=<node>[,<node#2>,...] ] ¥
      [ RSCT_LOGS={yes|no} ] ¥
```

注: DIRECTORY 属性に DEFAULT を指定すると、元のデフォルト PowerHA SystemMirror ディレクトリーの値が復元されます。

FORMATTING 属性は hacmp.out ログに対してのみ適用され、そのほかのすべてのログについては無視されます。FORMATTING 属性と TRACE_LEVEL 属性は hacmp.out ログと clstrmgr.debug ログに対してのみ適用され、そのほかのすべてのログについては無視されます。

ログ名の代わりに ALL を指定すると、指定された DIRECTORY および REMOTE_FS の変更はすべてのログに適用されます。

ログ名の代わりに EVENTS を指定すると、イベント要約レポートが表示されます。

ボリューム・グループ

```
clmgr add volume_group [ <vgname> ] ¥
      NODES="<node#1>,<node#2>[,...>]" ¥
      PHYSICAL_VOLUMES="<disk#1>[,<disk#2>,...]" ¥
      [ TYPE={original|big|scalable|legacy} ] ¥
      [ RESOURCE_GROUP=<RESOURCE_GROUP> ] ¥
      [ PPART_SIZE={1|2|4|8|16|32|64|128|256|512|1024} ] ¥
      [ MAJOR_NUMBER=## ] ¥
      [ CONCURRENT_ACCESS={false|true} ] ¥
      [ ACTIVATE_ON_RESTART={false|true} ] ¥
      [ QUORUM_NEEDED={true|false} ] ¥
      [ LTG_SIZE=### ] ¥
      [ MIGRATE_FAILED_DISKS={false|one|pool|remove} ] ¥
      [ MAX_PHYSICAL_PARTITIONS={32|64|128|256|512|768|1024} ] ¥
      [ MAX_LOGICAL_VOLUMES={256|512|1024|2048} ] ¥
      [ STRICT_MIRROR_POOLS={no|yes|super} ] ¥
      [ MIRROR_POOL_NAME="<mp_name>" ] ¥
      [ CRITICAL={false|true} ] ¥
      [ FAILUREACTION={halt|notify|fence|
                       stoprg|moverg} ] ¥
      [ NOTIFYMETHOD=</file/to/invoke> ]
```

注: ボリューム・グループのメジャー番号を設定すると、そのメジャー番号が現在使用可能でないノード上で、コマンドが正常に実行されない結果になる可能性があります。この設定は、すべてのノード上で共通に使用可能なメジャー番号を確認してから、変更するようにしてください。

```
clmgr modify volume_group <vgname> ¥
      [ ADD=<disk#n> [ MIRROR_POOL_NAME="<mp_name>" ] ] ¥
      [ REMOVE=<disk#n> ] ¥
      [ TYPE={big|scalable} ] ¥
      [ ENHANCED_CONCURRENT_MODE={false|true} ] ¥
      [ ACTIVATE_ON_RESTART={false|true} ] ¥
      [ QUORUM NEEDED={true|false} ] ¥
      [ LTG_SIZE=### ] ¥
      [ MIGRATE_FAILED_DISKS={false|one|pool|remove} ] ¥
      [ MAX_PHYSICAL_PARTITIONS={32|64|128|256|512|768|1024} ] ¥
      [ MAX_LOGICAL_VOLUMES={256|512|1024|2048} ] ¥
      [ STRICT_MIRROR_POOLS={off|on|super} ] ¥
      [ CRITICAL={false|true} ] ¥
      [ FAILUREACTION={halt|notify|fence|
                       stoprg|moverg} ] ¥
      [ NOTIFYMETHOD="</file/to/invoke>" ] ¥
      [ SCSIPR_ACTION={clear} ]
```

注: ENHANCED_CONCURRENT_MODE を false に設定すると、高速ディスク・テークオーバーが自動的に設定されます。

MAX_PHYSICAL_PARTITIONS、MAX_LOGICAL_VOLUMES、および MIRROR_POOL_NAME はスケーラブル・ボリューム・グループにのみ適用されます。

```
clmgr query volume_group [ <vg#1>[,<vg#2>,...] ]
clmgr delete volume_group
      {<volume_group> [,<vg#2>,...] | ALL }
clmgr discover volume_groups
```

注: *volume_group* のエイリアスは vg です。

論理ボリューム

```
clmgr add logical_volume [ <lvname> ] ¥
      VOLUME_GROUP=<vgname> ¥
      LOGICAL_PARTITIONS=## ¥
      [ DISKS="<disk#1>[,<disk#2>,...]" ] ¥
```

```
      [ TYPE={jfs|jfs2|sysdump|paging|
        jfslog|jfs2log|aio_cache|boot} ] ¥
      [ POSITION={outer_middle|outer_edge|center|
        inner_middle|inner_edge  } ] ¥
      [ PV_RANGE={minimum|maximum} ] ¥
      [ MAX_PVS_FOR_NEW_ALLOC=## ] ¥
      [ LPART_COPIES={1|2|3} ] ¥
      [ WRITE_CONSISTENCY={active|passive|off} ] ¥
      [ LPARTS_ON_SEPARATE_PVS={yes|no|superstrict} ] ¥
      [ RELOCATE={yes|no} ] ¥
      [ LABEL="<label>" ] ¥
      [ MAX_LPARTS=#### ] ¥
      [ BAD_BLOCK_RELOCATION={yes|no} ] ¥
      [ SCHEDULING_POLICY={parallel|sequential
        |parallel_sequential
        |parallel_round_robin} ] ¥
      [ VERIFY_WRITES={false|true} ] ¥
      [ ALLOCATION_MAP=<file> ] ¥
      [ STRIPE_SIZE={4K|8K|16K|32K|64K|128K|256K|512K|
        1M|2M|4M|8M|16M|32M|64M|128M} ] ¥
      [ SERIALIZE_IO={false|true} ] ¥
      [ FIRST_BLOCK_AVAILABLE={false|true} ] ¥
      [ FIRST_COPY_MIRROR_POOL=<mirror_pool> ] ¥
      [ SECOND_COPY_MIRROR_POOL=<mirror_pool> ] ¥
      [ THIRD_COPY_MIRROR_POOL=<mirror_pool> ] ¥
      [ GROUP=<group> ] ¥
      [ PERMISSIONS=<####> ] ¥
      [ NODE=<reference_node_in_vg> ]
```

注: STRIPE_SIZE は、LPARTS_ON_SEPARATE_PVS、PV_RANGE、または SCHEDULING_POLICY と一緒には使用しないでください。

```
clmgr query logical_volume [ <lvname>[,<LV#2>,...] ]
clmgr delete logical_volume {[ <lv#1>[,<LV#2>,...]  ] | ALL}
```

注: *logical_volume* のエイリアスは lv です。

ファイルシステム

```
clmgr add file_system <fsname> ¥
      VOLUME_GROUP=<group> ¥
      TYPE=enhanced ¥
      UNITS=### ¥
        [ SIZE_PER_UNIT={megabytes|gigabytes|512bytes} ] ¥
      [ PERMISSIONS={rw|ro} ]
      [ OPTIONS={nodev,nosuid,all} ] ¥
      [ BLOCK_SIZE={4096|512|1024|2048} ] ¥
      [ LV_FOR_LOG={ <lvname> | "INLINE" } ] ¥
        [ INLINE_LOG_SIZE=#### ] ¥
      [ EXT_ATTR_FORMAT={v1|v2} ] ¥
      [ ENABLE_QUOTA_MGMT={no|all|user|group} ] ¥
      [ ENABLE_EFS={false|true} ]
```

注:

1. *BLOCK_SIZE* はバイト数です。*LOG_SIZE* はメガバイト数です。
2. *LOG_SIZE* および *LV_FOR_LOG* は、*INLINE_LOG* を true に設定した場合にのみ使用できます。
3. 拡張ファイルシステムのサイズは 16 MB です。

```
clmgr add file_system <fsname> ¥
      TYPE=enhanced ¥
      LOGICAL_VOLUME=<logical_volume> ¥
```

```
      [ PERMISSIONS={rw|ro} ] ¥
      [ OPTIONS={nodev,nosuid,all} ] ¥
      [ BLOCK_SIZE={4096|512|1024|2048} ] ¥
      [ LV_FOR_LOG={ <lvname> | "INLINE" } ] ¥
      [ INLINE_LOG_SIZE=#### ] ¥
      [ EXT_ATTR_FORMAT={v1|v2} ] ¥
      [ ENABLE_QUOTA_MGMT={no|all|user|group} ] ¥
      [ ENABLE_EFS={false|true} ]
```

```
clmgr add file_system <fsname> ¥
      VOLUME_GROUP=<group> ¥
      TYPE={standard|compressed|large} ¥
      UNITS=### ¥
         [ SIZE_PER_UNIT={megabytes|gigabytes|512bytes} ] ¥
      [ PERMISSIONS={rw|ro} ] ¥
      [ OPTIONS={nodev|nosuid|all} ] ¥
      [ DISK_ACCOUNTING={false|true} ] ¥
      [ FRAGMENT_SIZE={4096|512|1024|2048} ] ¥
      [ BYTES_PER_INODE={4096|512|1024|2048|8192|
                         16384|32768|65536|131072} ] ¥
      [ ALLOC_GROUP_SIZE={8|16|32|64} ] ¥
      [ LV_FOR_LOG=<lvname> ]
```

注: FRAGMENT_SIZE は、標準ファイルシステムおよび圧縮ファイルシステムについてのみ有効です。

```
clmgr add file_system <fsname> ¥
      TYPE={standard|compressed|large} ¥
      LOGICAL_VOLUME=<logical_volume> ¥
      [ PERMISSIONS={rw|ro} ] ¥
      [ OPTIONS={nodev|nosuid|all} ] ¥
      [ DISK_ACCOUNTING={false|true} ] ¥
      [ FRAGMENT_SIZE={4096|512|1024|2048} ] ¥
      [ BYTES_PER_INODE={4096|512|1024|2048|8192|
                         16384|32768|65536|131072} ] ¥
      [ ALLOC_GROUP_SIZE={8|16|32|64} ] ¥
      [ LV_FOR_LOG=<lvname> ]

clmgr query file_system [ <fs#1>[,<fs#2>,...] ]
clmgr delete file_system { <fsname>[,<FS#2>,...] | ALL } ¥
      [ REMOVE_MOUNT_POINT={false|true} ]
```

注: *file_system* のエイリアスは fs です。

Physical volume

```
clmgr query physical_volume ¥
      [ <disk>[,<disk#2>,...] ] ¥
      [ NODES=<node>,<node#2>[,<node#3>,...] ] \
      [ TYPE={available|all|tiebreaker} ]
```

注: node には、ノード名あるいはネットワークで解決可能な名前 (ホスト名または IP アドレスなど) を指定できます。

disk はデバイス名 (hdisk0) または PVID (00c3a28ed9aa3512) のいずれかです。

```
clmgr modify physical_volume <disk_name_or_PVID> ¥
      NAME=<new_disk_name> ¥
      [ NODE=<reference_node> ] ¥
      [ ALL_NODES={false|true} ] ¥
      [ SCSIPR_ACTION={clear} ]
```

注: NODE 属性は、指定されたディスクを hdisk# といったデバイス名を使用して指定する場合に必要です。ディスクを PVID を使用して指定する場合は、NODE 属性を参照する必要はありません。

physical_volume のエイリアスは pv です。

ミラー・プール

```
clmgr add mirror_pool <pool_name> ¥
      VOLUME_GROUP=<vgname> ¥
      [ PHYSICAL_VOLUMES="<disk#1>[,<disk#2>,...]" ] \
      [ MODE={sync|async} ] ¥
      [ ASYNC_CACHE_LV=<lvname> ] ¥
      [ ASYNC_CACHE_HW_MARK=## ]

clmgr add mirror_pool <pool_name> ¥
      [ VOLUME_GROUP=<vgname> ] ¥
      PHYSICAL_VOLUMES="<disk>[,<disk#2>,...]"
```

注: *add* 操作を既存のミラー・プールに対して実行した場合、指定した物理ボリュームがそのミラー・プールに追加されます。

```
clmgr modify mirror_pool <pool_name> ¥
      [ VOLUME_GROUP=<vgname> ]¥
      [ NAME=<new_pool_name> ] ¥
      [ MODE={sync|async} ] ¥
      [ FORCE_SYNC={false|true} ] ¥
      [ ASYNC_CACHE_LV=<lvname> ] ¥
      [ ASYNC_CACHE_HW_MARK=## ]

clmgr query mirror_pool [ <pool_name>[,<pool#2>,...] ]
clmgr delete mirror_pool <pool_name>,[,<pool#2>,...]| ALL }¥
      [ VOLUME_GROUP=<vgname> ]
clmgr delete mirror_pool <pool_name> ¥
      [ VOLUME_GROUP=<vgname> ] ¥
      PHYSICAL_VOLUMES="<disk>[,<disk#2>,...]"
```

注: 削除操作に物理ボリュームを指定すると、ディスクのリストがミラー・プールから除去されます。すべてのディスクが除去された場合は、ミラー・プールが除去されます。

注: *mirror_pool* のエイリアスは mp および pool です。

EFS

```
clmgr add efs ¥
      MODE=ldap ¥
      [ PASSWORD=<password> ]
clmgr add efs ¥
      MODE=shared_fs ¥
      VOLUME_GROUP=<vgname> ¥
      SERVICE_IP=<service_ip> ¥
      [ PASSWORD=<password> ]

clmgr modify efs ¥
      MODE={ldap|shared_fs} ¥
      [ VOLUME_GROUP=<vgname> ] ¥
      [ SERVICE_IP=<service_ip> ] ¥
      [ PASSWORD=<password> ]

clmgr query efs
clmgr delete efs
```

レポート

```
clmgr view report [<report>] ¥
      [ FILE=<PATH_TO_NEW_FILE> ] ¥
      [ TYPE={text|html} ]
```

```
clmgr view report {nodeinfo|rginfo|lvinfo|
      fsinfo|vginfo|dependencies} ¥
      [ TARGETS=<target>[,<target#2>,...] ] ¥
      [ FILE=<PATH_TO_NEW_FILE> ] ¥
      [ TYPE={text|html} ]

clmgr view report cluster ¥
      TYPE=html ¥
      [ FILE=<PATH_TO_NEW_FILE> ] ¥
      [ COMPANY_NAME="<BRIEF_TITLE>" ] ¥
      [ COMPANY_LOGO="<RESOLVEABLE_FILE>" ]

clmgr view report availability ¥
      [ TARGETS=<appctlr>[,<appctlr#2>,...] ] ¥
      [ FILE=<PATH_TO_NEW_FILE> ] ¥
      [ TYPE={text|html} ] ¥
      [ BEGIN_TIME="YYYY:MM:DD" ] ¥
     [ END_TIME="YYYY:MM:DD" ]
```

注: 現在サポートされているレポートは、basic、 cluster、 status、 topology、 applications、 availability、 events、 nodeinfo、 rginfo、 networks、 vginfo、 lvinfo、 fsinfo、 dependencies、 および roha です。これらのレポートは、その一部は重複した情報を提供しますが、それぞれ独自の、固有な情報も提供します。

appctlr という値は application_controller の省略語です。

MM は 1 から 12 でなければなりません。DD は 1 から 31 でなければなりません。

BEGIN_TIME を指定しない場合、END_TIME の直前の 30 日間のレポートが生成されます。

END_TIME を指定しない場合、現在時刻がデフォルトとなります。

report のエイリアスは re です。

LDAP サーバー

以下の構文は、クラスターに 1 つ以上の LDAP サーバーを構成するために使用します。

```
clmgr add ldap_server <server>[,<server#2>,...] ¥
      ADMIN_DN=<admin_distinguished_name> ¥
      PASSWORD=<admin_password> ¥
      BASE_DN=<suffix_distinguished_name> ¥
      SSL_KEY=<full_path_to_key> ¥
      SSL_PASSWORD=<SSL_key_password> ¥
      VERSION=<version> ¥
      DB2_INSTANCE_PASSWORD=<password> ¥
      ENCRYPTION_SEED=<seed> ¥
      [ SCHEMA=<schema_type> ] ¥
      [ PORT={636|###} ]
```

注: *ldap_server* のエイリアスは ls です。

以下の構文は、クラスターに既に構成済みの 1 つ以上の LDAP サーバーを追加するために使用します。

```
clmgr add ldap_server <server>[,<server#2>,...] ¥
      ADMIN_DN=<admin_distinguished_name> ¥
      PASSWORD=<admin_password> ¥
      BASE_DN=<suffix_distinguished_name> ¥
      SSL_KEY=<full_path_to_key> ¥
      SSL_PASSWORD=<SSL_key_password> ¥
      [ PORT={636|###} ]
```

注: 複数のサーバーを指定した場合、それらは同じポート番号を共用するピアツーピア構成でなければなりません。

```
clmgr query ldap_server
clmgr delete ldap_server
```

LDAP クライアント

```
clmgr add ldap_client ¥
      SERVERS=<LDAP_server>[,<LDAP_server#2>]¥
      BIND_DN=<bind_distinguished_name> ¥
        PASSWORD=<LDAP_admin_password> ¥
      BASE_DN=<base_dn> ¥
      SSL_KEY=<full_path_to_key> ¥
        SSL_PASSWORD=<SSL_key_password> ¥
      [ PORT={636|###} ] ¥

clmgr query ldap_client
clmgr delete ldap_client
```

注: *ldap_client* のエイリアスは lc です。

ユーザー

```
clmgr add/modify user <user_name> ¥
      [ REGISTRY={local|ldap} ] ¥
      [ RESOURCE_GROUP=<resource_group> ] ¥
      [ ID=### ] ¥
      [ PRIMARY=<group> ] ¥
      [ PASSWORD="{<password>|}" ] ¥
      [ CHANGE_ON_NEXT_LOGIN={true|false} ] ¥
      [ GROUPS=<group#1>[,<group#2>,...] ] ¥
      [ ADMIN_GROUPS=<group#1>[,<group#2>,...] ] ¥
      [ ROLES=<role#1>[,<role#2>,...] ] ¥
      [ SWITCH_USER={true|false} ] ¥
      [ SU_GROUPS={ALL|<group#1>[,<group#2>,...]} ] ¥
      [ HOME=<full_directory_path> ] ¥
      [ SHELL=<defined_in_/etc/shells> ] ¥
      [ INFO=<user_information> ] ¥
      [ EXPIRATION=<MMDDhhmmyy> ] ¥
      [ LOCKED={false|true} ] ¥
      [ LOGIN={true|false} ] ¥
      [ REMOTE_LOGIN={true|false} ] ¥
      [ SCHEDULE=<range#1>[,<range#2>,...>] ] ¥
      [ MAX_FAILED_LOGINS={#|0} ] ¥
      [ AUTHENTICATION={compat|files|DCE|ldap} ] ¥
      [ ALLOWED_TTYS=<tty#1>[,<tty#2>,...] ] ¥
      [ DAYS_TO_WARN={#|0} ] ¥
      [ PASSWORD_VALIDATION_METHODS=<meth#1>[,<meth#2>,...]]¥
      [ PASSWORD_FILTERS=<filter#1>[,<filter#2>,...] ] ¥
      [ MIN_PASSWORDS=<number_of_passwords_before_reuse> ] ¥
      [ REUSE_TIME=<weeks_before_password_reuse> ] ¥
      [ LOCKOUT_DELAY=<weeks_btwn_expiration_and_lockout> ] ¥
      [ MAX_PASSWORD_AGE={0..52} ] ¥
      [ MIN_PASSWORD_LENGTH={0..8} ] ¥
      [ MIN_PASSWORD_ALPHAS={0..8} ] ¥
      [ MIN_PASSWORD_OTHERS={0..8} ] ¥
      [ MAX_PASSWORD_REPEATED_CHARS={0..52} ] ¥
      [ MIN_PASSWORD_DIFFERENT={0..8} ] ¥
      [ UMASK=#### ] ¥
      [ AUDIT_CLASSES=<class#1>[,<class#2>,...] ] ¥
      [ TRUSTED_PATH={nosak|on|notsh|always} ] ¥
      [ PRIMARY_AUTH={SYSTEM|.} ] ¥
      [ SECONDARY_AUTH={NONE|SYSTEM|<token>;<user>} ] ¥
      [ PROJECTS=<project#1>[,<project#2>,...] ] ¥
```

```
      [ KEYSTORE_ACCESS={file|none} ] ¥
          [ ADMIN_KEYSTORE_ACCESS={file|none} ] ¥
          [ KEYSTORE_MODE={admin|guard} ] ¥
          [ ALLOW_MODE_CHANGE={false|true} ] ¥
      [ KEYSTORE_ENCRYPTION={RSA_1024|RSA_2048|RSA_4096} ] ¥
          [ FILE_ENCRYPTION={AES_128_CBC|AES_128_EBC| ¥
                             AES_192_CBC|AES_192_ECB| ¥
                             AES_256_CBC|AES_256_ECB} ] ¥
      [ ALLOW_PASSWORD_CHANGE={no|yes} ]
```

注**:** INFO フィールドは、スペース、下線 (_)、およびハイフン (-) も含めて、英数字のみ受け入れます。

注**:** *add*操作の場合、*REGISTRY* はユーザーの作成先を示します。*modify* の場合は、指定したユーザーのどのインスタンスを変更するかを示します。

注**:** SCHEDULE は、ユーザーがこのシステムへのログインを許可される時間を定義します。SCHEDULE の値は、以下のようなコンマ区切りの項目のリストです。

```
* [!][MMdd[-MMdd]]:hhmm-hhmm
* [!]MMdd[-MMdd][:hhmm-hhmm]
* [!][w[-w]]:hhmm-hhmm
* [!]w[-w][:hhmm-hhmm]
```

*MM*は月を表す数字 (00 = 1 月、11 = 12 月)、*dd* は日付、*hh* は時刻 (00 から 23)、*mm* は分、*w* は曜日 (0 = 日曜日、6 = 土曜日) です。 感嘆符 (!) は、指定されたその時刻範囲ではアクセスが許可されていないことを示すために使用します。

MAX_FAILED_LOGINS、DAYS_TO_WARN、MIN_PASSWORDS、REUSE_TIME をゼロに設定すると、それぞれの機能を使用不可にできます。

LOCKOUT_DELAY を -1 に設定すると、以下の機能を使用不可にできます。

```
clmgr modify user {<user_name> | ALL_USERS} ¥
      ALLOW_PASSWORD_CHANGE={no|yes}
```

注**:** *ALLOW_PASSWORD_CHANGE* は、ユーザーが C-SPOC を使用してクラスター全体に対するパスワードを変更できるかどうかを示します。

```
clmgr query user TYPE={AVAILABLE|ALLOWED}
clmgr query user RESOURCE_GROUP=<resource_group>
clmgr query user <user_name> ¥
      [ RESOURCE_GROUP=<resource_group> ]
clmgr delete user <user_name> ¥
      [ RESOURCE_GROUP=<resource_group> ] ¥
      [ REMOVE_AUTH_INFO={true|false} ]
      [ REGISTRY={files |LDAP} ]
```

グループ

```
clmgr add group <group_name>
      [ REGISTRY={local(files)|LDAP} ]
      [ RESOURCE_GROUP=<resource_group> ] ¥
      [ ID=### ] ¥
      [ ADMINISTRATIVE={false|true} ] ¥
      [ USERS=<user#1>[,<user#2>,...] ] ¥
      [ ADMINS=<admin#1>[,<admin#2>,...] ] ¥
```

```
      [ PROJECTS=<project#1>[,<project#2>,...] ] ¥
      [ KEYSTORE_MODE={admin|guard} ] ¥
      [ KEYSTORE_ENCRYPTION={ RSA_1024|RSA_2048|RSA_4096} ] ¥
      [ KEYSTORE_ACCESS={file|none} ] ¥


clmgr modify group <group_name> ¥
      [ RESOURCE_GROUP=<resource_group> ] ¥
      [ ID=### ] ¥
      [ ADMINISTRATIVE={false|true} ] ¥
      [ USERS=<user#1>[,<user#2>,...] ] ¥
      [ ADMINS=<admin#1>[,<admin#2>,...] ] ¥
      [ PROJECTS=<project#1>[,<project#2>,...] ¥
      [ KEYSTORE_MODE={admin|guard} ] ¥
      [ KEYSTORE_ENCRYPTION={ RSA_1024|RSA_2048|RSA_4096} ] ¥
      [ KEYSTORE_ACCESS={file|none} ]
```

注: RG オプションはローカルで定義されたグループに必要です。 RG オプションを指定しないと、LDAP グループが存在すると想定されます。

```
clmgr query group RESOURCE_GROUP=<resource_group>
clmgr query group <group_name> ¥
      [ RESOURCE_GROUP=<resource_group> ]

clmgr delete group <group_name> ¥
      [ RESOURCE_GROUP=<resource_group> ] ¥
      [ REGISTRY={files|LDAP} ]
```

注: RG オプションはローカルで定義されたグループに必要です。*group* のエイリアスは gp です。

ストレージ・エージェント

```
clmgr add storage_agent <agent_name> ¥
      TYPE={ds8k_gm|xiv_rm} ¥
      ADDRESSES=<IP>[<IP#2>,...] ¥
      [ USER=<user_id> ] ¥
      [ PASSWORD=<password> ] ¥
      [ ATTRIBUTES=<NAME>@<VALUE>[,<NAME#2>@<VALUE#2>,...] ]
clmgr modify storage_agent <agent_name> ¥
      [ NAME=<new_agent_name> ] ¥
      [ ADDRESSES=<IP>[<IP#2>,...] ] ¥
      [ USER=<user_id> ] ¥
      [ PASSWORD=<password> ] ¥
      [ ATTRIBUTES=<NAME>@<VALUE>[,<NAME#2>@<VALUE#2>,...] ]
clmgr query storage_agent [ <agent>[,<agent#2>,...] ]
clmgr delete storage_agent {<agent>[,<agent#2>,...] | ALL}
```

注: *storage agent* のエイリアスは sta です。

ストレージ・システム

```
clmgr add storage_system <storage_system_name> ¥
      TYPE={ds8k_gm|xiv_rm} ¥
      SITE=<site> ¥
      AGENTS=<agent>[,<agent#2>,...] ¥
      VENDOR_ID=<identifier> ¥
      [ WWNN=<world_wide_node_name> ] ¥
      [ ATTRIBUTES=<NAME>@<VALUE>[,<NAME#2>@<VALUE#2>,...] ]
clmgr add storage_system <storage_system_name> ¥
      TYPE=ds8k_inband_mm ¥
      SITE=<site> ¥
      VENDOR_ID=<identifier> ¥
      [ WWNN=<world_wide_node_name> ] ¥
      [ ATTRIBUTES=<NAME>@<VALUE>[,<NAME#2>@<VALUE#2>,...] ]
```

```
clmgr add storage_system <storage_system_name> ¥
      TYPE=svc ¥
      SITE=<site> ¥
      ADDRESSES=<IP>[<IP#2>,...] ¥
      MASTER=<Master/Auxiliary> ¥
      PARTNER=<Remote Partner> ¥
      [ AGENTS=<agent>[,<agent#2>,...] ] ¥
      [ ATTRIBUTES=<NAME>@<VALUE>[,<NAME#2>@<VALUE#2>,...] ]

clmgr modify storage_system <storage_system_name> ¥
      [ NAME=<new_storage_system_name> ] ¥
      [ SITE=<site> ] ¥
      [ AGENTS=<agent>[,<agent#2>,...] ] ¥
      [ WWNN=<world_wide_node_name> ] ¥
      [ VENDOR_ID=<identifier> ] ¥
      [ ADDRESSES=<IP>[<IP#2>,...] ] ¥
      [ MASTER=<Master/Auxiliary> ] ¥
      [ PARTNER=<Remote Partner> ] ¥
      [ ATTRIBUTES=<NAME>@<VALUE>[,<NAME#2>@<VALUE#2>,...] ]

clmgr query storage_system [ <storage_system>[,<ss#2>,...] ]


clmgr -a VENDOR_ID query storage_system ¥
      TYPE={ds8k_gm|ds8k_inband_mm|xiv_rm}
```

注: 次の照会は、使用可能なベンダー ID をリストします。

```
clmgr delete storage_system {<storage_system>[,<ss#2>,...] | ALL}
```

注: *storage system* のエイリアスは sts です。

ミラー・ペア

```
clmgr add mirror_pair <mirror_pair_name> ¥
      FIRST_DISK=<disk_1> ¥
      SECOND_DISK=<disk_2>
clmgr modify mirror_pair <mirror_pair_name> ¥
      [ NAME=<new_mirror_pair_name> ] ¥
      [ FIRST_DISK=<disk_1> ] ¥
      [ SECOND_DISK=<disk_2> ]
clmgr query mirror_pair [ <mirror_pair>[,<mp#2>,...] ]
clmgr delete mirror_pair {<mirror_pair>[,<mp#2>,...] | ALL}
```

注: *mirror_pair* のエイリアスは mip です。

ミラー・グループ

```
: HyperSwap user mirror groups
  clmgr add mirror_group <mirror_group_name> ¥
        TYPE=ds8k_inband_mm ¥
        MG_TYPE=user ¥
        VOLUME_GROUPS=<volume_group>[,<vg#2>,...] ¥
        DISKS=<raw_disk>[,<disk#2>,...] ¥
        [ HYPERSWAP_ENABLED={no|yes} ] ¥
        [ CONSISTENT={yes|no} ] ¥
        [ UNPLANNED_HS_TIMEOUT=## ] ¥
        [ HYPERSWAP_PRIORITY={medium|high} ] ¥
        [ RECOVERY={manual|auto} ] ¥
        [ RESYNC={manual|auto} ] ¥
        [ ATTRIBUTES=<NAME>@<VALUE>[,<NAME#2>@<VALUE#2>,...] ]

 clmgr modify mirror_group <mirror_group_name> ¥
       [ NAME=<new_mirror_group_name> ] ¥
       [ VOLUME_GROUPS=<volume_group>[,<vg#2>,...] ] ¥
       [ DISKS=<raw_disk>[,<disk#2>,...] ] ¥
       [ STORAGE_SYSTEMS=<storage_system>[,<ss#2>,...] ] ¥
```

```
      [ HYPERSWAP_ENABLED={no|yes} ] ¥
      [ CONSISTENT={yes|no} ] ¥
      [ UNPLANNED_HS_TIMEOUT=## ] ¥
      [ HYPERSWAP_PRIORITY={medium|high} ] ¥
      [ RECOVERY={manual|auto} ] ¥
      [ RESYNC={manual|auto} ] ¥
      [ ATTRIBUTES=<NAME>@<VALUE>[,<NAME#2>@<VALUE#2>,...] ]

: HyperSwap system mirror groups
  clmgr add mirror_group <mirror_group_name> ¥
        TYPE=ds8k_inband_mm ¥
        MG_TYPE=system ¥
        VOLUME_GROUPS=<volume_group>[,<vg#2>,...] ¥
        DISKS=<raw_disk>[,<disk#2>,...] ¥
        NODE=<node> ¥
        HYPERSWAP_ENABLED={no|yes} ¥
        [ CONSISTENT={yes|no} ] ¥
        [ UNPLANNED_HS_TIMEOUT=## ] ¥
        [ HYPERSWAP_PRIORITY={medium|high} ] ¥
        [ ATTRIBUTES=<NAME>@<VALUE>[,<NAME#2>@<VALUE#2>,...] ]

  clmgr modify mirror_group <mirror_group_name> ¥
        [ NAME=<new_mirror_group_name> ] ¥
        [ VOLUME_GROUPS=<volume_group>[,<vg#2>,...] ] ¥
        [ DISKS=<raw_disk>[,<disk#2>,...] ] ¥
        [ NODE=<node> ] ¥
        [ STORAGE_SYSTEMS=<storage_system>[,<ss#2>,...] ] ¥
        [ HYPERSWAP_ENABLED={no|yes} ] ¥
        [ CONSISTENT={yes|no} ] ¥
        [ UNPLANNED_HS_TIMEOUT=## ] ¥
        [ HYPERSWAP_PRIORITY={medium|high} ] ¥
        [ ATTRIBUTES=<NAME>@<VALUE>[,<NAME#2>@<VALUE#2>,...] ]

: HyperSwap repository mirror groups
  clmgr add mirror_group <mirror_group_name> ¥
        TYPE=ds8k_inband_mm ¥
        MG_TYPE=repository ¥
        SITE=<site> ¥
        NON_HS_DISK=<Non-HyperSwap_disk> ¥
        HS_DISK=<HyperSwap_disk> ¥
        [ HYPERSWAP_ENABLED={no|yes} ] ¥
        [ CONSISTENT={yes|no} ] ¥
        [ UNPLANNED_HS_TIMEOUT=## ] ¥
        [ HYPERSWAP_PRIORITY={medium|high} ] ¥
        [ RESYNC={manual|auto} ] ¥
        [ ATTRIBUTES=<NAME>@<VALUE>[,<NAME#2>@<VALUE#2>,...] ]

  clmgr modify mirror_group <mirror_group_name> ¥
        [ NAME=<new_mirror_group_name> ] ¥
        [ SITE=<node> ] ¥
        [ NON_HS_DISK=<non-HyperSwap_disk> ] ¥
        [ HS_DISK=<HyperSwap_disk> ] ¥
        [ STORAGE_SYSTEMS=<storage_system>[,<ss#2>,...] ] ¥
        [ HYPERSWAP_ENABLED={no|yes} ] ¥
        [ CONSISTENT={yes|no} ] ¥
        [ UNPLANNED_HS_TIMEOUT=## ] ¥
        [ HYPERSWAP_PRIORITY={medium|high} ] ¥
        [ RESYNC={manual|auto} ] ¥
        [ ATTRIBUTES=<NAME>@<VALUE>[,<NAME#2>@<VALUE#2>,...] ]

: DS8000 Global Mirror and XIV mirror groups
  clmgr add mirror_group <mirror_group_name> ¥
        TYPE={ds8k_gm|xiv_rm} ¥
        MODE={sync|async} ¥
        RECOVERY={auto|manual} ¥
        [ STORAGE_SYSTEMS=<storage_system>[,<ss#2>,...] ] ¥
        [ VENDOR_ID=<vendor_specific_identifier> ] ¥
```

```
       [ ATTRIBUTES=<NAME>@<VALUE>[,<NAME#2>@<VALUE#2>,...] ]

  clmgr modify mirror_group <mirror_group_name> ¥
       [ NAME=<new_mirror_group_name> ] ¥
       [ MODE={sync|async} ] ¥
       [ RECOVERY={auto|manual} ] ¥
       [ STORAGE_SYSTEMS=<storage_system>[,<ss#2>,...] ] ¥
       [ VENDOR_ID=<vendor_specific_identifier> ] ¥
       [ ATTRIBUTES=<NAME>@<VALUE>[,<NAME#2>@<VALUE#2>,...] ]

: SVC mirror groups
  clmgr add mirror_group <mirror_group_name> ¥
       TYPE=svc ¥
       STORAGE_SYSTEMS=<MASTER_SVC>,<AUXILIARY_SVC> ¥
       MIRROR_PAIRS=<mirror_pair>[,<mirror_pair#2>,...] ] ¥
       [ MODE={sync|async} ] ¥
       [ RECOVERY={auto|manual} ]

  clmgr modify mirror_group <mirror_group_name> ¥
       [ NAME=<new_mirror_group_name> ] ¥
       [ STORAGE_SYSTEMS=<MASTER_SVC>,<AUXILIARY_SVC> ] ¥
       [ MIRROR_PAIRS=<mirror_pair>[,<mirror_pair#2>,...] ] ¥
       [ MODE={sync|async} ] ¥
       [ RECOVERY={auto|manual} ]

: Hitachi mirror groups
  clmgr add mirror_group <mirror_group_name> ¥
       TYPE=hitachi ¥
       VENDOR_ID=<device_group> ¥
       HORCM_INSTANCE=<instance> ¥
       [ MODE={sync|async} ] ¥
       [ RECOVERY={auto|manual} ] ¥
       [ HORCM_TIMEOUT=### ] ¥
       [ PAIR_EVENT_TIMEOUT=### ]

  clmgr modify mirror_group <mirror_group_name> ¥
       [ NAME=<new_mirror_group_name> ] ¥
       [ VENDOR_ID=<device_group> ] ¥
       [ HORCM_INSTANCE=<instance> ] ¥
       [ MODE={sync|async} ] ¥
       [ RECOVERY={auto|manual} ] ¥
       [ HORCM_TIMEOUT=### ] ¥
       [ PAIR_EVENT_TIMEOUT=### ]

: EMC mirror groups
  clmgr add mirror_group <mirror_group_name> ¥
       TYPE=emc ¥
       [ MG_TYPE={composite|device} ] ¥
       [ MODE={sync|async} ] ¥
       [ RECOVERY={auto|manual} ] ¥
       [ CONSISTENT={yes|no} ] ¥
       [ VENDOR_ID=<vendor_specific_identifier> ]

  clmgr modify mirror_group <mirror_group_name> ¥
      [ NAME=<new_mirror_group_name> ] ¥
      [ MG_TYPE={composite|device} ] ¥
      [ MODE={sync|async} ] ¥
      [ RECOVERY={auto|manual} ] ¥
      [ CONSISTENT={yes|no} ] ¥
      [ VENDOR_ID=<device_group> ]

: HyperSwap mirror groups
  clmgr {swap|view} mirror_group <mirror_group_name>[,<mg#2>,...] ¥
       [ NODE=<node_name> ]
  clmgr {swap|view} mirror_group ¥
       NODES=<node_name>[,<node#2>,...] ¥
       [ SYSTEM_GROUPS={yes|no} ]
```

```
  clmgr {swap|view} mirror_group ¥
        SITES=<site_name>[,<site#2>] ¥
        [ SYSTEM_GROUPS={yes|no} ] ¥
        [ REPOSITORY_GROUP={yes|no} ]
```

注: swap 属性と view 属性は、DS シリーズ・インバンド (HyperSwap®) の場合のみ有効です。

```
clmgr manage mirror_group refresh
      <mirror_group_name>[,<mg#2>,...] ¥
      [ NODE=<node_name> ]
clmgr manage mirror_group refresh ¥
      NODES=<node_name>[,<node#2>,...] ¥
      [ SYSTEM_GROUPS={yes|no} ]
clmgr manage mirror_group refresh ¥
      SITES=<site_name>[,<site#2>] ¥
      [ SYSTEM_GROUPS={yes|no} ] ¥
      [ REPOSITORY_GROUP={yes|no} ]

: All mirror groups
  clmgr query mirror_group [ <mirror_group>[,<mg#2>,...] ]
  clmgr delete mirror_group {<mirror_group>[,<mg#2>,...] | ALL}
```

注: *mirror_group* のエイリアスは mig です。

イベント

```
cl clmgr add event <EVENT_NAME> ¥
         FILE=<EXECUTABLE_FILE> ¥
         [ DESCRIPTION=<EVENT_DESCRIPTION> ]
clmgr modify event <EVENT_NAME> ¥
         [ NAME=<NEW_EVENT_NAME> ] ¥
         [ FILE=<EXECUTABLE_FILE> ] ¥
         [ DESCRIPTION=<EVENT_DESCRIPTION> ]
clmgr modify event <BULTIN EVENT_NAME> ¥
         [ COMMAND=<COMMAND_OR_FILE> ] ¥
         [ NOTIFY_COMMAND=<COMMAND_OR_FILE> ] ¥
         [ RECOVERY_COMMAND=<COMMAND_OR_FILE> ] ¥
           [ RECOVERY_COUNTER=# ] ¥
         [ PRE_EVENT_COMMAND=<CUSTOM_EVENT> ] ¥
         [ POST_EVENT_COMMAND=<CUSTOM_EVENT> ]
clmgr query event [ <EVENT_NAME>[,<EVENT_NAME#2>,...] ]
         [ TYPE={CUSTOM|PREDEFINED|ALL} ]
clmgr delete event { <EVENT_NAME>[,<EVENT_NAME#2>,...] | ALL }
```

注: *event* のエイリアスは ev です。

HMC

```
clmgr add hmc <HMC> ¥
            [ TIMEOUT=<###> ] ¥
            [ RETRY_COUNT=<###> ] ¥
            [ RETRY_DELAY=<###> ] ¥
            [ NODES=<node>[,<node#2>,...] ] ¥
            [ SITES=<site>[,<site#2>,...] ] \
            [ CHECK_HMC=<Yes|No> ]
clmgr modify hmc <HMC> ¥
            [ TIMEOUT=<###> ] ¥
            [ RETRY_COUNT=<###> ] ¥
            [ RETRY_DELAY=<###> ] ¥
            [ NODES=<node>[,<node#2>,...] ] ¥
            [ SITES=<site>[,<site#2>,...] ] \
            [ CHECK_HMC=<Yes|No> ]


clmgr query hmc [<HMC>[,<HMC#2>,...]]
clmgr delete hmc {<HMC> | ALL}
```

注: **clmgr delete** の例では、指定ノードに関連付けられた指定の HMC またはすべての HMC を除去します。ノードを指定しなかった場合は、すべてのノードが除去されます。

CoD

```
clmgr add cod <APPCTRL> ¥
      [ USE_DESIRED="Yes|No"> ] ¥
      [ OPTIMAL_MEM=#.## ]  ¥
      [ OPTIMAL_CPU=# ]  ¥
      [ OPTIMAL_PU=#.## ] ¥
      [ OPTIMAL_VP=# ]

clmgr modify cod <APPCTRL> ¥
      [ USE_DESIRED="Yes|No"> ] ¥
      [ OPTIMAL_MEM=#.## ]  ¥
      [ OPTIMAL_CPU=# ]  ¥
      [ OPTIMAL_PU=#.## ] ¥
      [ OPTIMAL_VP=# ]
```

注:

1. このコマンドを使用して、アプリケーション・コントローラーの実行に必要な最適レベルのリソースをプロビジョンできます。
2. USE_DESIRED=1 を設定した場合は、アプリケーション・コントローラーに最適なレベルのリソースを提供する、LPAR プロファイルの望ましいレベルが使用されます。
3. USE_DESIRED=0 を指定した場合は、より正確に、OPTIMAL_MEM、OPTIMAL_CPU、OPTIMAL_PU、および OPTIMAL_VP の各値を使用してアプリケーション・コントローラーで必要なリソースのレベルを構成できます。
4. アプリケーション・コントローラーのリソースのレベルをプロビジョニングすることにより、PowerHA SystemMirror は、アプリケーション・コントローラーに最適なレベルのリソースを提供する操作 (DLPAR、On/Off CoD、EPCoD) を実行できます。
5. **clmgr verify cluster** コマンドでクラスターを検査することにより、プロビジョニングのレベルを確認できます。
6. *cod* のエイリアスは *roha*、*dlpar*、および *cuod* です。

```
clmgr query cod [<APPCTRL> ]
clmgr delete cod {<APPCTRL> | ALL}
```

例

以下の例では、**clmgr** コマンドのクラス属性には大/小文字の区別はありません。例えば、以下のコマンドの場合、NODES 属性は NODES、nodes、または Nodes のいずれでも構いません。

```
clmgr create cluster clMain NODES=nodeA,nodeB
```

1. 以下の例では、nodeA と nodeB という名前の 2 つのノードが含まれている PowerHA SystemMirror Standard Edition for AIX クラスターを作成します。クラスター名は haCL で、hdisk5 という名前のリポジトリー・ディスクがあります。この環境では、クラスターに既定のマルチキャスト・アドレス 229.9.3.17 を使用する必要があります。

```
clmgr create cluster haCL NODES=nodeA,nodeB ¥
      REPOSITORY=hdisk5 ¥
      CLUSTER_IP=229.9.3.17
clmgr sync cluster
```

注: この例で CLUSTER_IP 属性が必要なのは、単に環境でマルチキャスト・アドレスが必要であるからというだけの理由です。マルチキャスト・アドレスが指定されていない場合は、その時点で使用中のアドレスに基づいて、システムがアドレスを選択します。

2. 以下の例では、デフォルト・ポリシーを使用して標準 (非コンカレント) リソース・グループを作成します。リソース・グループの名前は db2RG で、access1 という名前のサービス IP アドレスと、db2Controller という名前のアプリケーション・コントローラーが含まれています。このリソース・グループは、vg1 と vg2 という 2 つの非コンカレント・ボリューム・グループを管理します。

```
clmgr add resource_group db2RG SERVICE_IP=access1 ¥
      APPLICATIONS=db2Controller ¥
      VOLUME_GROUP=vg1,vg2
clmgr sync cluster
```

3. 以下のコマンドを使用して、クラスター内部のさまざまなオブジェクトの状況を確認できます。

```
clmgr -a STATE query cluster
clmgr -a STATE query node nodeA
clmgr -a STATE query resource_group rg1
```

注:

- STATE クラスは、クラスター全体に対する論理的なワースト・ケース集計を返します。例えば、4 ノード・クラスター内の 1 つのノードにエラーが発生した場合、クラスター全体に対して返される状況はエラーとして報告されます。
- このコマンドの実行から返される値は、標準の ATTR=VALUE 形式です。例えば、クラスターがオフラインの場合、返される値は STATE=OFFLINE になります。
- **-a** フラグを使用して、複数の属性を一度に取得できます。例えば、以下のコマンドを実行すると、クラスターの名前と状態の両方を取得できます。

  ```
  clmgr -a STATE,NAME query cluster
  ```

4. すべてのアクション、クラス、および属性は、明示的に指定されたエイリアス、または固有のものとして識別できる最小数の文字に短縮できます。以下の例では、完全形のコマンドを表示し、その下に同じコマンドの省略形を示します。

- ```
  clmgr query resource_group
  clmgr q rg
  ```
- ```
  clmgr modify node mynode PERSISTENT_IP=myIP NETWORK=myNet
  clmgr mod node mynode pe=myIP netw=myNet
  ```
- ```
  clmgr online node nodeA
  clmgr start node nodeA
  ```

注: これらのアクション、クラス、および属性の短縮化は、クラスターで対話式に **clmgr** コマンドを使用する場合に使用するためのものです。これらの省略形はスクリプト内で使用できますが、読みやすいコードを提供するものではないので、スクリプト内部では使用しないようにしてください。

5. **clmgr** コマンドについてのヘルプ情報はコマンド・ラインから入手できます。実行するコマンドの一部が不明な場合は、わかっている部分だけを入力すると、ヘルプ情報が表示されます。例えば、コマンドの一部として有効でないオブジェクトまたは値を指定すると、有効なオブジェクトまたは値のみについてのヘルプ情報が表示されます。以下のコマンドを例として実行して、コマンド・ラインから表示されるヘルプ情報の違いを見てください。

```
clmgr
clmgr view
clmgr view report
clmgr view report -h
```

注: **-h** フラグを使用できるのは、特定の操作のすべての有効なオプションのリストを要求するオブジェクト・クラスまたは一組のオプション・ペアの後のみです。このフラグは、**clmgr** コマンドのフラグの中で、clmgr コマンドの直後に配置する必要のない唯一のフラグです。
```

以下の例では、clmgr コマンドの共通の使用シナリオについて説明します。例はすべて、テスト済みです。値は、お客様の環境に有効な値に置き換えてください。以下のタスクがシナリオのベースになっており、それについての詳細な説明があります。

- クラスターの作成
- リソース・グループの作成
- 現在の状況の確認
- すべての属性および設定の表示
- 何らかのフィルターまたは基準に基づくオブジェクトの表示
- clmgr コマンドをもう少し使いやすくする
- clmgr コマンドの簡易ヘルプの取得

例: 標準クラスターの作成

詳細:

このクラスターは、2 つのノードを持ち、関連したサイトを持たない標準クラスターです。クラスター名は DB2_cluster で、ノードの名前は DBPrimary および DBBackup です。リポジトリー・ディスクは、hdisk5 という名前のディスク上で作成されます。

例:

1. `clmgr create cluster DB2_cluster NODES=DBPrimary,DBBackup ¥`
   `REPOSITORY=hdisk5`
2. `clmgr sync cluster`

コメント:

- リポジトリー・ディスクは、**clmgr** コマンドを実行するノード上で解決されます。リポジトリー・ディスクは PVID または UUID のフォーマットで指定できます。
- ハートビート・タイプが指定されていません。このため、クラスターはデフォルトのユニキャスト通信を使用します。
- **clmgr** コマンドに大/小文字の区別はありません。リポジトリー属性は、REPOSITORY、Repository、または repository のように指定できます。

例: 拡張クラスターの作成

詳細:

このクラスターは、Oracle_cluster という名前の拡張クラスターです。クラスターには、Ora1、Ora2、Ora3、および Ora4 の 4 つのノードがあります。クラスターには、Ora_Primary および Ora_Secondary の 2 つのサイトがあります。サイト Ora_Primary は、ノード Ora1 および Ora2 を管理します。サイト Ora_Secondary は、ノード Ora3 および Ora4 を管理します。リポジトリー・ディスクは、hdisk5 という名前のディスク上で作成されます。クラスターは、ハートビート・タイプとしてマルチキャスト通信を使用します。

例:

1. `clmgr create cluster Oracle_cluster ¥`
   `NODES=Ora1,Ora2,Ora3,Ora4 ¥`
   `TYPE=SC ¥`
   `REPOSITORY=hdisk5 ¥`
   `HEARTBEAT_TYPE=multicast`
2. `clmgr add site Ora_Primary NODES=Ora1,Ora2`

3. clmgr add site Ora_Secondary NODES=Ora3,Ora4
4. clmgr sync cluster

コメント:

リポジトリー・ディスクは、**clmgr** コマンドを実行するノード上で解決されます。リポジトリー・ディスクは PVID または UUID のフォーマットで指定できます。

例: リンク・クラスターの作成

詳細:

このクラスターは、SAP-cluster という名前のリンク・クラスターです。クラスターには、SAP-A1、SAP-A2、SAP-B1、および SAP-B2 の 4 つのノードがあります。クラスターには、SAP_Active および SAP_Backup の 2 つのサイトがあります。サイト SAP_Active は、ノード SAP-A1 および SAP-A2 を管理します。サイト SAP_Backup は、ノード SAP-B1 および SAP-B2 を管理します。 SAP_Active サイト上のリポジトリー・ディスクの名前は hdisk5 です。 SAP_Backup サイト上のリポジトリー・ディスクの名前は hdisk11 です。クラスターは、ハートビート・タイプとしてユニキャスト通信を使用します。

例:

1. clmgr create cluster SAP-cluster ¥
   NODES=SAP-A1,SAP-A2,SAP-B1,SAP-B2 ¥
   TYPE=LC ¥
   HEARTBEAT_TYPE=unicast
2. clmgr add site SAP_Active NODES=SAP-A1,SAP-A2 REPOSITORY=hdisk5
3. clmgr add site SAP_Backup NODES=SAP-B1,SAP-B2 REPOSITORY=hdisk11
4. clmgr sync cluster

コメント:

- リンク・クラスター上では、それぞれのサイトがリポジトリー・ディスクを使用する必要があります。サイトごとにリポジトリー・ディスクを指定する必要があります。
- リポジトリー・ディスクは、**clmgr** コマンドが通信できる最初のノード上で解決されます。リンク・クラスターの場合、**clmgr** コマンドはサイトごとに最初に定義されているノードとの通信を試行します。この例では、hdisk5 リポジトリー・ディスクは SAP-A1 ノード上で解決され、hdisk11 リポジトリーは SAP-B1 ノード上で解決されます。
- リポジトリー・ディスクは PVID または UUID のフォーマットで指定できます。

例: リソース・グループの作成

詳細:

このリソース・グループは、デフォルト・ポリシーを使用する標準 (非コンカレント) リソース・グループで、名前は db2RG です。このリソース・グループには、access1 という名前のサービス IP アドレスと、db2Controller という名前のアプリケーション・コントローラーが含まれます。さらに、このリソース・グループは、vg1 と vg2 という名前の 2 つのボリューム・グループ (どちらも非コンカレント) を管理します。

例:

- clmgr add resource_group db2RG SERVICE_IP=access1 ¥
  APPLICATIONS=db2Controller ¥
  VOLUME_GROUP=vg1,vg2
- clmgr sync cluster

例: 現在の状況の確認

詳細:

非常に多くの場合、特定のオブジェクトが正確にどの状態になっているかを知り、適切なアクションが取れるようにすることが重要です。これは、clmgr を使用して照会アクションを実行することで可能になります。

例:

- clmgr -a STATE query cluster
- clmgr -a STATE query site siteA
- clmgr -a STATE query node nodeA
- clmgr -a STATE query resource_group rg1

コメント:

- サイト・クラスとクラスター・クラスの両方の場合は、返される STATE はメンバー・ノードの論理的なワースト・ケース集計です。例えば 4 ノード・クラスターの場合、1 つのノードでエラーが検出された場合でも、クラスター全体の状況が ERROR として報告されます。
- 返される値は、標準の ATTR=VALUE 形式 (例えば STATE=OFFLINE) になります。値のみが必要な場合は、その他のいくつかのフラグを **-a** と組み合わせると、希望する結果を得ることができます。**-cSa** という組み合わせフラグを使用すると、VALUE のみ (OFFLINE など) が返されます。これは、一度に 1 つの値についてのみ有効です。
- **-a** フラグを使用して複数の属性を一度に取得することができます。例えば、**-a NAME,STATE** のようにします。また、**-a** フラグでは大/小文字の区別はなく (-a Name,state でも同じ)、ワイルドカード (-a N*) をサポートしています。

例: すべての属性および設定の表示

詳細:

PowerHA SystemMirror は、セットアップと十分なテストが完了した後は、問題が発生するか何らかの保守が必要になるまでは一般にもうアクティブに対話することのない製品です。問題の発生時や保守の際には、クラスターの内容およびすべての設定を表示できることが必要になります。これは、clmgr で、query アクションを使用し、必要に応じて特定の形式 (コロン区切りまたは XML) を要求して行うことができます。以下のコマンド例ではリソース・グループを使用していますが、原理はすべてのオブジェクト・クラスで同じです。

例:

- clmgr query resource_group
- clmgr query resource_group rg1,rg2
- clmgr -c query resource_group rg1,rg2
- clmgr -x query resource_group rg1,rg2
- clmgr -v query resource_group
- clmgr -cv query resource_group
- clmgr -xv query resource_group

コメント:

- query アクションでターゲット・オブジェクトを指定していない場合、および詳細フラグ **-v** を使用していない場合は、単純なオブジェクト・リストが表示されます。

- query アクションで 1 つ以上のターゲット・オブジェクトを指定した場合は、これらのオブジェクトの既知の属性または設定がすべて表示されます。この場合、**-v** フラグがオーバーライドされます。
- **-v** フラグを query アクションで使用すると、指定のクラスのすべての既知のオブジェクトの既知の属性または設定がすべて表示されます。
- 詳細な属性または設定が表示される場合、デフォルトでは ATTR=VALUE 形式で、1 行に 1 つずつ表示されます。**-c** を指定した場合は、コロン区切り形式で、すべての値が 1 行に表示されます。 **-x** を指定した場合は、すべての属性および値が単純な XML 形式で表示されます。

例: 何らかのフィルターまたは基準に基づくオブジェクトの表示

詳細:

リソース・グループなどの特定のクラスに多数のオブジェクトを定義したり、特定のクラス内に多数の設定を定義したりすることは珍しくありません。そのため、本当に必要な情報を見つけるのが困難になる場合があります。幸い、clmgr には query アクションでフィルター基準を指定する機能があるため、この問題を解決できます。

例:

- `clmgr query file_collection FILE="*rhosts*"`
- `clmgr query resource_group CURRENT_NODE=`get_local_nodename``

コメント:

- 最初の例では、特定の値または設定が含まれているオブジェクト (この場合は、rhosts という名前のファイルを含むファイル・コレクション) を検出する簡単な方法を示します。ここではワイルドカード文字がサポートされていることに注意してください。
- 2 番目の例では、動的値と一致するオブジェクトの検出方法として好適な実施例を示します。このケースでは、例はローカル・ノード上で現在実行中のすべてのリソース・グループのリストの取得方法を示しています。
- このフィルタリング機能を **-a** フラグと組み合わせて使用すると、非常に強力で柔軟性の高いデータ・リトリーブを提供することができます。

例: clmgr をもう少し使いやすくする

詳細:

clmgr には大/小文字の区別が必要なものは何もないので、もどかしいタイプ入力のミスをなくすのに役立ちます。さらに、すべてのアクション、クラス、属性、またはオプションは、明示的に指定されたエイリアス (online に代わり start、resource_group に代わり rg など)、または固有のものとして識別できる最小数の文字に短縮することができます。以下のコマンドのペアは、機能的に同じものです。

例:

- ```
  clmgr query resource_group
  clmgr q rg
  ```
- ```
  clmgr modify node mynode PERSISTENT_IP=myIP NETWORK=myNet
  clmgr mod node mynode pe=myIP netw=net_ether_0
  ```
- ```
  clmgr online node nodeA
  clmgr start node nodeA
  ```

コメント:
  ```

アクションおよびクラスの短縮は、clmgr を端末内部で対話式に使用する場合のためのものです。これらの省略形はスクリプトでも使用できますが、スクリプトではアクションとクラスの両方に完全な名前を使用することを強くお勧めします。そうすることによって、信頼性と保守可能性がより高いコードを作成できます。

例: clmgr の簡易ヘルプの取得

詳細:

clmgr のヘルプはオンラインでいつでも利用可能です。しかし、Web ブラウザーを起動するのは時には不便であり、実際的ではない場合や、不可能な場合さえあります。そのため、clmgr では、必要なヘルプをすぐに取得できるように、できる限り組み込みヘルプを提供しています。提供されるヘルプのタイプの 1 つに、既知のオブジェクトまたは値のセットからのオブジェクトまたは値が必要な場合のヘルプがあります。無効なオブジェクトまたは値を指定すると、該当のエラー・メッセージが表示されるだけでなく、その操作に有効なオブジェクトまたは値のリストも表示されます。これは、なかなか解決しないタイピング・エラーを克服するのに非常に役立ちます。必要なアクション、クラス、またはオブジェクトが不明な場合も、clmgr から別のヘルプを利用できます。わかっている部分を入力するだけで、次にくる可能性のあるすべての値が clmgr から表示されます。値の 1 つを選択して次に進みさえすればいいのです。以下のコマンドを実行して、clmgr に用意されているヘルプの表示例を見てください。

例:

- clmgr
- clmgr view
- clmgr view report
- clmgr view report -h

コメント:

コマンド・ラインで、オブジェクト・クラスまたは何らかのオプション・ペアのセットの後に **-h** フラグを指定すると、その特定の操作に対するすべての有効なオプションのリストを要求できます。このフラグは、**clmgr** コマンド内で、**clmgr** コマンド自体の直後に置く必要がない唯一のフラグです。

関連情報:

リソース・グループ依存関係

---

clpasswd コマンド

目的

クラスターまたはリソース・グループ内のすべてのノード上で現在のユーザー・パスワードを変更します。

構文

clpasswd [-g resource group] *user*

説明

クラスター・パスワード (**clpasswd**) ユーティリティーを使用すると、ユーザーは、PowerHA SystemMirror 管理者による指定に従ってクラスター内またはリソース・グループ内のすべてのノード上の自分のパスワードを 1 つのノードから変更できます。ユーザーがクラスター・ノード全体でパスワードを変更できるように、PowerHA SystemMirror 管理者は root 権限を持たないユーザーを、パスワード変更を許可されるユーザーのリストに事前に追加しておきます。

このクラスター・パスワード・ユーティリティーは、SMIT 高速パス **cl_passwd** の AIX パスワード・ユーティリティーを置換することもできます。

次の表は、ユーザーのパスワードがユーザーの権限に基づいて変更され、パスワード・ユーティリティーがアクティブである状態を示しています。

ユーザー許可レベル	システム・パスワード・ユーティリティーが **clpasswd** にリンクされ、**/bin/passwd** が呼び出される場合	システム・パスワード・ユーティリティーがアクティブな場合
クラスター全体でパスワードの変更を許可されたユーザー	パスワードはすべてのクラスター・ノードで変更されます。	パスワードがすべてのクラスター・ノードで変更される。
クラスター全体でパスワードの変更を許可されていないユーザー	パスワードはローカル・ノードでのみ変更される。	パスワードは変更されない。

フラグ

-g ユーザーがパスワードを変更できるリソース・グループの名前を指定します。 パスワードは指定されたリソース・グループの各ノードで変更されます。

ユーザー
パスワードを変更しているユーザーのユーザー名

例

```
clpasswd -g rg1 myusername
```

---

clRGinfo コマンド

目的

指定された 1 つ以上のリソース・グループのロケーションと状態を表示するレポートを作成します。

構文

```
clRGinfo [-h][-v][-s|-c][-t][-p][-a][-m][i][resgroup1] [resgroup2]...
```

説明

クラスター・サービスがローカル・ノード上で実行されていない場合、この **clRGinfo** コマンドはクラスター・サービスがアクティブであるノードを識別し、アクティブ・クラスター・マネージャーからリソース・グループ情報を取得します。リソース・グループを指定せずにこのコマンドを使用すると、構成されたすべてのリソース・グループに関する情報が表示されます。

コマンドの出力には、リソース・グループのグローバル状態と、ローカル・ノード上のリソース・グループの特別状態の両方が表示されます。

リソース・グループの 1 次インスタンスは、次のいずれかの状態になります。

オンライン
このリソース・グループのすべてのリソースがアクティブです。

エラー
PowerHA SystemMirror がこのリソース・グループを処理しているときにエラーが発生しました。

管理外
: unmanage オプションを指定してクラスター・サービスが停止されました。

オフライン
: リソース・グループがアクティブではありません。

クラスター・イベントの進行中に、リソース・グループは次のいずれかの推移的状態になります。

獲得中
: リソース・グループのリソースをアクティブ化しています。

解放中
: リソース・グループのリソースを解放しています。

一時エラー
: 回復可能エラーが起こりました。

クラスターがサイトと複製リソースを使用する場合、複製リソースを含むリソース・グループには、複製エンドポイントを管理する 1 次インスタンスと 2 次インスタンスが存在します。 **clRGinfo** コマンドは、リソース・グループの 2 次インスタンスの次に示す状態を表示します。

オンライン **2** 次
: このリソース・グループのすべての 2 次リソースがアクティブです。

エラー **2** 次
: PowerHA SystemMirror がリソース・グループの 2 次リソースを処理しているときにエラーが発生しました。

管理外 **2** 次
: unmanage オプションを指定してクラスター・サービスが停止されました。

オフライン **2** 次
: リソース・グループの 2 次インスタンスがアクティブではありません。

獲得中 **2** 次
: リソース・グループの 2 次リソースをアクティブ化しています。

解放中 **2** 次
: リソース・グループの 2 次リソースを解放しています。

一時エラー **2** 次
: PowerHA SystemMirror がリソース・グループの 2 次リソースを処理しているときに、リカバリー可能エラーが発生しました。

他のリソース・グループとの関係に応じたリソース・グループの自動配置と管理を可能にする依存関係を使用して、リソース・グループを構成できます。 **clRGinfo** コマンドは、親子関係を持つリソース・グループ、およびロケーション依存関係を持つリソース・グループについて、次に示す状態を表示します。

親のオフラインのためにオフライン
: 親リソース・グループがアクティブでないために、子リソース・グループがアクティブではありません。

フォールオーバーのためにオフライン
: フォールオーバーが発生したため、リソース・グループがアクティブではありません。

ノードの欠落のためにオフライン
: クラスター内のノードに対してリソース・グループが指定されていません。

ターゲットのオフラインのためにオフライン
別のリソース・グループとの関係に関与しているリソース・グループが非アクティブであり、構成された依存関係により、このリソース・グループがアクティブであってはならないと定められています。

フラグ

-a リソース・グループの現在のロケーション、およびクラスター・イベント後の移動先を表示します。このフラグは、イベント前およびイベント後のスクリプトに使用します (特に、従属リソース・グループが存在する PowerHA SystemMirror クラスターで)。PowerHA SystemMirror が依存リソース・グループを処理するとき、複数のリソース・グループを **rg_move** イベントで同時に移動できます。

-c 出力をコロン区切り形式で表示します。

-h 使用法のメッセージを表示します。

-i 管理者が指示したあらゆるオンライン操作またはオフライン操作を表示します。

-m アプリケーションの状況を表示します。

-p リソース・グループの優先順位オーバーライド・ロケーション情報を表示します。

-s 出力をコロン区切り形式で表示します。

-t ローカル・ノード上で現在アクティブになっている遅延タイマー情報、すべての遅延フォールバック・タイマー、および整定タイマーを表示します。

注**:** このフラグは、クラスター・マネージャーがローカル・ノード上でアクティブになっている場合にのみ使用できます。

-v 詳細出力を表示します。

例

1. 次の例は、フラグ・パラメーターを指定せずに **clRGinfo** コマンドを実行した場合のレポートを示しています。

```
# clRGinfo
-----------------------------------------------------------------------------
Group Name     State                          Node
-----------------------------------------------------------------------------
VS_DATA_RG     ONLINE                         powerha53
               ONLINE                         powerha54
               ONLINE                         powerha63
               ONLINE                         powerha64

VS_REDO_RG     ONLINE                         powerha53
               ONLINE                         powerha54
               ONLINE                         powerha63
               ONLINE                         powerha64

RG1            ONLINE                         powerha53
               OFFLINE                        powerha54
               ACQUIRING                      powerha63
               OFFLINE                        powerha64
```

2. 次の例は、サイトを含むクラスター内で **clRGinfo** コマンドを実行した場合のレポートを示しています。

```
# clRGinfo
-----------------------------------------------------------------------------
Group Name     State                          Node
-----------------------------------------------------------------------------
OASTRG         ONLINE                         als018022@site1
```

```
                ONLINE SECONDARY            alm194200@site2

VOTERG          ONLINE                      als018022@site1
                ONLINE SECONDARY            alm194200@site2
```

3. 次の例は、**clRGinfo -m** コマンドを実行した場合のレポートを示しています。

```
$ /usr/es/sbin/cluster/utilities/clRGinfo -m

Group Name     State           Application state        Node
-----------------------------------------------------------------
Group1         ONLINE          ONLINE MONITORED         merry

Application state could be any one of the below possible values:
    OFFLINE
    ONLINE FAILED
    ONLINE FAILOVER
    ONLINE MONITORED
    ONLINE NOT MONITORED
    ONLINE MONITOR FAILED
    ONLINE MONITOR SUSPENDED
```

4. 次の例は、**clRGinfo -i** コマンドを実行した場合のレポートを示しています。

```
$ /usr/es/sbin/cluster/utilities/clRGinfo -i
-------------------------------------------------------------------------------
Group Name      State           Application state                       Node
-------------------------------------------------------------------------------
Rg1             ONLINE          ONLINE STOPPED BY ADMINISTRATOR         node1
INSTANCE_SAP    OFFLINE                                                 node2

Application state could be any one of the below possible values:
    OFFLINE
    ONLINE FAILED
    ONLINE FAILOVER
    ONLINE MONITORED
    ONLINE NOT MONITORED
    ONLINE MONITOR FAILED
    ONLINE MONITOR SUSPENDED
    ONLINE STOPPED BY ADMINISTRATOR
    OFFLINE STARTED BY ADMINISTRATOR
```

clRGmove コマンド

目的

ユーザー要求 rg_move イベントを実行して、リソース・グループをオフラインまたはオンラインにするか、リソース・グループをあるノードから別のノードに移動します。

構文

```
clRGmove -g <groupname> -n <nodename> | -x -n <sitename> |-r | -a [-m | -u | -d] [-i] [-s true | false]
```

説明

clRGmove を使用して、リソース・グループのロケーションを手動で制御できます。

非コンカレント・リソース・グループに対して、以下のいずれかのアクションを実行できます。

- リソース・グループをオンライン・ノードまたはオンライン 2 次ノードからオフラインにします。
- リソース・グループを特定ノードに対してオンラインまたはオンライン 2 次にします。
- リソース・グループを現行ホスティング・ノードから新しいロケーションに移動します。

コンカレント・リソース・グループに対して、以下のいずれかのアクションを実行できます。

- リソース・グループをグループ・ノード・リスト内のすべてのノードからオフラインにします。
- リソース・グループをグループ・ノード・リスト内の 1 つのノードからオフラインにします。
- リソース・グループをグループ・ノード・リスト内のすべてのノードでオンラインにします。
- リソース・グループをグループ・ノード・リスト内の 1 つのノードでオンラインにします。

優先順位オーバーライド・ロケーション

優先順位オーバーライド・ロケーションは、リソース・グループのその他すべてのノード・ポリシーおよび可能性のあるロケーションをオーバーライドします。

非コンカレント・リソース・グループの優先順位オーバーライド・ロケーションは以下のようになります。

- `-r` フラグではなく `-n` フラグを使用して明示的に宛先を指定する非コンカレント・リソース・グループの移動ではすべて、宛先は優先順位オーバーライド・ロケーションになります。優先順位オーバーライド・ロケーションは、リソース・グループを再び手動で移動する際に、`-n` フラグではなく `-r` フラグをロケーションに明示的に使用するまで存続します。
- リソース・グループをオフラインにすると、そのリソース・グループは、手動でオンラインに戻すまでオフラインのままになります。ノードを指定するために `-n` フラグを使用して手動でリソース・グループをオンラインに戻した場合、そのノードは優先順位オーバーライド・ロケーションになります。 `-r` フラグを使用してリソース・グループをオンラインに戻した場合は、アクティブな最高優先順位ノードが使用され、優先順位オーバーライド・ロケーションはリソース・グループから除去されます。

コンカレント・リソース・グループの優先順位オーバーライド・ロケーションは以下のようになります。

- すべてのノードでコンカレント・リソース・グループをオフラインにすると、優先順位オーバーライド・ロケーションは、リソース・グループ内のすべてのノードで *OFFLINE* 状態になります。コンカレント・リソース・グループを 1 つのノードでのみオフラインにすると、そのノード上のリソース・グループの *OFFLINE* 状態が優先順位オーバーライド・ロケーション・リストに追加されます。
- すべてのノードでコンカレント・リソース・グループをオンラインにすると、優先順位オーバーライド・ロケーションは、リソース・グループ内のすべてのノードで除去されます。コンカレント・リソース・グループを 1 つのノードでのみオンラインにすると、そのノード上のリソース・グループの *OFFLINE* 状態が優先順位オーバーライド・ロケーション・リストから除去されます。

すべてのリソース・グループの移動で、以下のいずれかの移動を使用できます。

非永続移動
クラスター内のすべてのノードがオフラインになるまで続きます。クラスター全体がオフラインになるとすぐに、優先順位オーバーライド・ロケーションは無視され、クラスターがオンラインに戻るとリソース・グループは通常の動作を再開します。

永続移動
クラスター・リブート後も続きます。クラスターがオンラインに戻ったとき、優先順位オーバーライド・ロケーションは存続します。

制限

clRGmove コマンドには以下の制限事項があります。

- 一度にオンラインまたはオフラインにできるリソース・グループは 1 つのみです。
- コマンド・ラインで複数のリソース・グループを移動する場合は、要求が理にかなったものでなければなりません。そのため、リソース・グループの移動には SMIT インターフェースを使用することをお勧

めします。これにより、管理エラーの可能性がなくなります。 SMIT インターフェースを使用してリソース・グループを移動するには、コマンド・ラインで smit cspoc と入力し、「リソース・グループおよびアプリケーション」を選択します。

フラグ

-a このフラグはコンカレント・リソース・グループにのみ使用できます。このフラグを使用して、リソース・グループ内のすべてのノードでリソース・グループをオンラインまたはオフラインにします。単一ノード上のコンカレント・リソース・グループをオンラインまたはオフラインにするには、-n フラグを使用します。

-d リソース・グループをオフラインにします。このフラグを -u フラグまたは -m フラグと共に使用することはできません。

-g 移動するリソース・グループの名前を以下の形式で指定します。

-g <groupname>
単一リソース・グループ名を指定します。

-g "groupname1,groupname2,..."
複数リソース・グループ名のコンマ区切りリストを指定します。

-i リソース・グループが正常に移動された後で **clRGinfo** コマンドを実行します。

-m リソース・グループを別のノードに移動します。このフラグは -u フラグまたは -d フラグとは併用できません。このフラグは、複数のオンライン・リソース・グループを一度に 1 つずつ別のノードに移動する場合に使用します。

-n <nodename>
移動するか、オンラインにするか、またはオフラインにするリソース・グループを含むノードの名前。このフラグは -r フラグまたは -a フラグとは併用できません。ノード名の前に * 文字がある場合、そのノードは、このリソース・グループの最高優先順位ノードになるように構成されており、リソース・グループは別のノードに移動されています。 * 文字で識別されるノード内のリソース・グループを移動する場合、移動によってリソース・グループのオリジナルの構成が変更されます。

-n <sitename>
サイトを越えて移動するリソース・グループを含むサイトの名前。このフラグは -x フラグと併用する必要があります。サイト名の前に * 文字がある場合、そのサイトは、このリソース・グループの最高優先順位サイトになるように構成されており、リソース・グループは別のサイトに移動されています。 * 文字で識別されるサイト内のリソース・グループを移動する場合、移動によってリソース・グループのオリジナルの構成が変更されます。

-r このフラグは非コンカレント・リソース・グループにのみ使用できます。リソース・グループを移動する宛先ノードに、使用可能な最高優先順位ノードを使用します。このフラグは、移動するリソース・グループの優先順位オーバーライド・ロケーション属性を除去します。このフラグを使用できるのは、非コンカレント・リソース・グループをオンラインにする場合、または非コンカレント・リソース・グループを別のノードに移動する場合のみです。このフラグは -n フラグまたは -a フラグとは併用できません。

-s true | false
リソース・グループの 1 次インスタンスまたは 2 次インスタンスでのアクションを指定します (サイトが定義されている場合)。このフラグは、リソース・グループの 1 次インスタンスまたは 2 次インスタンスを、オフラインにするか、オンラインにするか、または同じサイト内の別のノードに移動する場合に使用します。このフラグは、-r、-d、-u、および -m の各フラグと併用できます。

-s true
リソース・グループの 2 次インスタンスでのアクションを指定します。

-s flase
リソース・グループの 1 次インスタンスでのアクションを指定します。

-u リソース・グループをオンラインにします。このフラグは **-d** フラグまたは **-m** フラグとは併用できません。

-x このフラグを使用すると、リソース・グループをサイトを越えて移動できます。このフラグは **-n** <sitename> フラグと併用する必要があります。

例

1. オフラインの非コンカレント・リソース・グループを nodeB という名前のノードでオンラインにするには、次のように入力します。

   ```
   clRGmove -g rgA -n nodeB -u
   ```

2. オンラインの非コンカレント・リソース・グループを nodeB という名前の別のノードに移動するには、次のように入力します。

   ```
   clRGmove -g rgA -n nodeB -m
   ```

3. 複数のオンラインの非コンカレント・リソース・グループを nodeB という名前の別のノードに移動するには、次のように入力します。

   ```
   clRGmove -g "rgA,rgB,rgC" -n nodeB -m
   ```

4. オンラインの非コンカレント・リソース・グループを nodeB という名前のノードでオフラインにするには、次のように入力します。

   ```
   clRGmove -g rgA -n nodeB -d
   ```

5. オンラインの非コンカレント・リソース・グループを、別の rg_move イベントによる以前の構成設定が除去される、アクティブな最高優先順位ノードに移動するには、次のように入力します。

   ```
   clRGmove -g *rgA -m -r
   ```

6. オンラインのコンカレント・リソース・グループを nodeB という名前の 1 つのノードでオフラインにするには、次のように入力します。

   ```
   clRGmove -g rgA -n nodeB -d
   ```

7. オンラインのコンカレント・リソース・グループをすべてのノードでオフラインにするには、次のように入力します。

   ```
   clRGmove -g rgA -a -d
   ```

8. オフラインのコンカレント・リソース・グループを nodeB という名前の 1 つのノードでオンラインにするには、次のように入力します。

   ```
   clRGmove -g rgA -n nodeB -u
   ```

9. オフラインのコンカレント・リソース・グループをすべてのノードでオンラインにするには、次のように入力します。

   ```
   clRGmove -g rgA -a -u
   ```

10. リソース・グループを site2 という名前のサイトに移動するには、次のように入力します。

    ```
    clRGmove -s false -x -g rgA -n site2
    ```

関連資料:

49 ページの『clmgr コマンド』

clruncmd コマンド

目的

クラスター・マネージャーを通常操作に復元します。

構文

```
clruncmd nodename
```

注**:** nodename はクラスター・サービスがアクティブであるクラスター・ノードの名前です。

説明

clruncmd コマンドは、指定したノードのクラスター・マネージャーに、イベント・スクリプト障害の発生後にイベント処理を再開するように指示します。 **clruncmd** コマンドは必ず、障害の原因を手動で訂正してから実行してください。イベント・スクリプト障害の発生後、残りの失敗したイベントはスキップされ、イベント処理はイベント・キュー内の次のイベントから再開されます。イベント障害の発生後にスキップされたアクションはすべて、手動で実行する必要があります。

例

node1 という名前のノードに対して通常操作に戻すようにクラスター・マネージャーに指示するには、次のように入力します。

```
clruncmd node1
```

関連資料:

49 ページの『clmgr コマンド』

clshowres コマンド

目的

クラスターまたはノードのリソース・グループ情報を表示します。

構文

```
clshowres [-g group ] [-n nodename ] [-d odmdir ]
```

フラグ

-g group
表示するリソース・グループの名前。

-n nodename
指定したノードからリソースの構成データベースを検索します。

-d odmdir
ODM オブジェクト・リポジトリーのディレクトリーとして、デフォルトの **/etc/objrepos** ではなく、*odmdir* を指定します。

例

1. クラスターのすべてのリソース・グループ情報をリストするには、以下のコマンドを実行します。

   ```
   clshowres
   ```

2. clam ノードのリソース・グループ情報をリストするには、以下のコマンドを実行します。

```
clshowres -n clam
```

clshowsrv コマンド

目的

PowerHA SystemMirror サブシステムの状況を表示します。

構文

```
clshowsrv { -a | -v | subsystem ...}
```

説明

clshowsrv コマンドは、PowerHA SystemMirror サブシステムの状況を表示します。状況には、サブシステム名、グループ名、プロセス ID、および状況が含まれます。デーモンの状況は、System Resource Controller (SRC) サブシステムに反映されるあらゆる状況 (アクティブ、作動不能、停止の警告など) になります。

フラグ

-a すべての PowerHA SystemMirror デーモンを表示します。

subsystem
指定した PowerHA SystemMirror サブシステムの状況を表示します。このフラグの有効な値は clstrmgrES、clinfoES、および clcomd です。複数のサブシステムを指定する場合は、エントリーをスペースで区切る必要があります。

-v すべての RSCT、PowerHA SystemMirror、およびオプションの PowerHA SystemMirror デーモンを表示します。

例

1. すべての PowerHA SystemMirror および RSCT サブシステムの状況を表示するには、次のように入力します。

   ```
   clshowsrv -v
   ```

   このコマンドによって、以下の例のような出力情報が表示されます。

   ```
   Local node: "hadev11" ("hadev11.aus.stglabs.ibm.com", "hadev11.aus.stglabs.ibm.com")
            Cluster services status:   "OFFLINE" ("ST_INIT")
            Remote communications:     "UP"
            Cluster-Aware AIX status:  "UP"

   Remote node: "hadev12" ("hadev12.aus.stglabs.ibm.com", "hadev12")
            Cluster services status:   "OFFLINE" ("ST_INIT")
            Remote communications:     "UP"
            Cluster-Aware AIX status:  "UP"

   Status of the RSCT subsystems used by PowerHA SystemMirror:
   Subsystem         Group            PID          Status
    cthags           cthags           9371848      active
    ctrmc            rsct             11862036     active

   Status of the PowerHA SystemMirror subsystems:
   Subsystem         Group            PID          Status
    clstrmgrES       cluster          12124406     active

   Status of the CAA subsystems:
   ```

```
Subsystem         Group            PID          Status
 clconfd          caa              10420354     active
 clcomd           caa              8912916      active
```

2. すべての PowerHA SystemMirror サブシステムの状況を表示するには、次のように入力します。

   ```
   clshowsrv -a
   ```

3. clstrmgr サブシステムの状況を表示するには、次のように入力します。

   ```
   clshowsrv clstrmgrES
   ```

4. clstrmgr サブシステムおよび clinfo サブシステムの状況を表示するには、次のように入力します。

   ```
   clshowsrv clstrmgrES clinfo
   ```

関連資料:

49 ページの『clmgr コマンド』

---

clsnapshot コマンド

目的

クラスター・スナップショットを作成します。スナップショットは、PowerHA SystemMirror クラスター構成データおよび状態情報を含む ASCII ファイルのセットです。

構文

```
clsnapshot [-a] [-c] [-C] [-d description] [-e] [-f true|false] [-g] [-h]
[-i] [-l] [-m methodlist] -n filename [-N filename] [-o odmdir]
[-q] [-r] [-R] [s] [-t]
```

説明

clsnapshot コマンドは、2 つのファイルを作成、変更、または除去します。ファイル拡張子 .odm で識別される最初のファイルには、現行 PowerHA SystemMirror ODM クラス・オブジェクトが含まれています。簡潔な説明をこのファイルに書き込むことができます。拡張子 .info を持つ 2 番目のファイルには、PowerHA SystemMirror クラスターのトラブルシューティングに役立つ情報が含まれています。

clsnapshot コマンドは、ノード固有の情報を取得するためにすべての構成済みノードで実行されます。

clsnapshot コマンドを使用して、スナップショットを現行クラスター・ハードウェアに適用できます。構成情報をクラスター・ノードに対して同期化するには、その前に検査ユーティリティーを実行し、合格していなければなりません。 **-f** フラグを使用すれば、検査ルーチンが失敗した場合でも、スナップショットを強制的に適用できます。

注**:** 環境変数 *SNAPSHOTPATH* に、スナップショット・ファイルへのパスが含まれています。デフォルトでは、このパスは /usr/es/sbin/cluster/snapshots になります。

フラグ

-a クラスター・スナップショットを適用します。

-c クラスター・スナップショットを作成します。

-C スナップショットの適用時にアクティブ・クラスター・リソースをリフレッシュしません。

-d text
説明をスナップショットに追加します。

-e クラスター・ログをスナップショットに保存します。ログをスナップショットに保存すると、スナップショットのファイル・サイズが大幅に増加する可能性があります。

-f true|false
検査が失敗した場合にスナップショットを強制適用します。

-g スナップショットを保持する一時 ODM を生成します。

-h スナップショットの使用法を表示します。

-i .info 拡張子を持つファイルを生成します。

-l スナップショット・ファイルをリストします。

-m methodlist
methodlist ファイルにリストされている各カスタム・スナップショット・メソッドを実行します。

-n file
スナップショットの名前を指定します。

-N file
スナップショットの新規の名前を指定します。

-o odmdir
PowerHA SystemMirror ODM クラスの ODM ディレクトリー (ODMDIR) を指定します。

-r スナップショットを除去します。

-R スナップショットを置き換えます。

-s スナップショットを表示します。

-t クラスター・オプションをリセットします。

関連資料:
49 ページの『clmgr コマンド』

---

clsnapshotinfo コマンド

目的

特定の PowerHA SystemMirror クラスター構成情報を検索および表示します。

構文

```
clsnapshotinfo [-m <METHOD> [<METHOD#2> ...]]
```

説明

clsnapshotinfo コマンドは、PowerHA SystemMirror クラスターに関する情報を収集するために PowerHA SystemMirror コマンドおよび AIX コマンドを実行します。 **clsnapshotinfo** コマンドは、コマンドが実行されるノードからのみ情報を収集します。コマンドからの出力は STDOUT に書き込まれます。 **clsnapshotinfo** コマンドが **clsnapshot** コマンドから実行 (自動的に行われます) されると、クラスター内のすべてのノードから情報が収集され、出力が .info 拡張子を持つスナップショット・ファイルに格納されます。

クラスターに関する情報を可能な限り多く収集するには、**clsnapshotinfo** コマンドを **clsnapshot** コマンドの一部として実行することをお勧めします。

フラグ

-m 1 つ以上のカスタム・スナップショット・メソッドを指定します。これらのメソッドからの出力は、**clsnapshotinfo** コマンドによって収集される全データの一部分です。

関連資料:

49 ページの『clmgr コマンド』

clstat コマンド (ASCII モードおよび X Windows モード)

注**:** このトピックには、**clstat** コマンドの ASCII モードおよび X Windows モードに関する情報があります。

ASCII モード

目的

クラスター状況モニター (ASCII モード) です。

構文

```
clstat [-c cluster ID | -n cluster name] [-i] [-r seconds] [-a] [-o][-s]
```

フラグ

-c cluster id
指定された ID を持つクラスターのみに関するクラスター情報を表示します。指定されたクラスターが使用可能ではない場合、clstat はクラスターが見つかるかプログラムが取り消されるまで、クラスターを検索し続けます。 -i オプションを使用している場合は指定できません。

-i 対話モードで ASCII **clstat** を実行します。最初に、システムからアクセス可能なすべてのクラスターのリストが表示されます。ユーザーは、詳細情報を表示する対象のクラスターを選択する必要があります。詳細表示から多数の機能を使用できます。

-n name
指定した名前を持つクラスターに関するクラスター情報を表示します。 -i オプションを使用している場合は指定できません。

-r seconds
指定した秒数でクラスター状況表示を更新します。デフォルトは 1 秒です。ただし、クラスターの状態に変化がなければ、表示は更新されません。

-a **clstat** を ASCII モードで表示します。

-o クラスターの状態の単一スナップショットを作成して終了します。 このフラグは、**cron** ジョブの中から **clstat** を実行するときに使用できます。**-a** とともに実行する必要があります。 **-i** オプションと **-r** オプションを無視します。

-s サービス・ラベルおよびその状態 (アップまたはダウン) を表示します。

X Windows モード

目的

クラスター状況モニター (X Windows モード) です。

構文

```
clstat [-a] [-c id | -n name ] [-r tenths-of-seconds ][-s]
```

フラグ

-a ASCII モードで **clstat** を実行します。

-c id
指定された ID を持つクラスターのみに関するクラスター情報を表示します。指定されたクラスターが使用可能ではない場合、**clstat** はクラスターが見つかるかプログラムが取り消されるまで、クラスターを検索し続けます。 **-n** オプションを使用している場合は指定できません。

-n name
指定した名前を持つクラスターのクラスター情報だけを表示します。

-r tenths-of-seconds
clstat ユーティリティーが表示を更新する間隔。グラフィカル・インターフェースを使用する場合、この値は、0.1 秒単位で解釈されます。デフォルトでは、 **clstat** は表示を 0.10 秒ごとに更新します。

-s サービス・ラベルおよびその状態 (アップまたはダウン) を表示します。

例

1. mycluster クラスターに関するクラスター情報を表示するには、以下のコマンドを実行します。

   ```
   clstat -n mycluster
   ```

2. 対話モードで ASCII clstat を実行し、マルチクラスター・モニターを可能にします。

   ```
   clstat -i
   ```

   以下は、X Window システム画面上のボタンです。

   戻る 前のクラスターを表示します。

   次へ 次のクラスターを表示します。

   Name:Id
   リフレッシュ・バー。このバーをクリックすると **clstat** がただちにリフレッシュします。

   終了 アプリケーションを終了します。

   ヘルプ
   ポップアップ・ヘルプ・ウィンドウに **clstat** のマニュアル・ページが表示されます。

---

clstop コマンド

目的

クラスター・サブシステムを停止します。

構文

```
clstop { -f | -g | -gr } [-s] [-y] [ -N | -R | -B ]
```

説明

clclstop は、ローカル・ノード上のクラスター・サービスを停止し、指定したフラグに従ってアクティブ・リソース・グループを処理します。このコマンドは、オプションで、/etc/inittab ファイル内のエントリーによってリブート時の自動開始を解除します。

フラグ

-f シャットダウンを強制します。クラスター・デーモンは、ローカル・プロシージャーを実行することなく終了します。

-g テークオーバーなしで正常終了シャットダウンを行います。

-gr
正常終了シャットダウンを行います。リソースはこのノードによって解放され、他のノードによってテークオーバーされます。デーモンは正常に終了し、ノードはそのリソースを解放します。これらのリソースはテークオーバーされます。テークオーバーを伴う正常終了シャットダウンの場合は、ノード・リストを指定する必要があります。

-s サイレント・シャットダウンを実行します。このフラグは、**wall** コマンドによるシャットダウン・メッセージのブロードキャストを行いません。デフォルト設定では、ブロードキャストが行われます。

-y クラスター・ノードをシャットダウンする前に、オペレーターに確認を求めません。このフラグがデフォルトです。

-B 即時に、および以降のシステム再始動時に停止します。

-N 即時にシャットダウンします。

-R その後のシステム再始動時に停止し、/etc/inittab ファイル内のエントリーを除去します。

注: /etc/rc.shutdown ファイルは、シャットダウン・コマンドの実行中に実行されるコマンドを含むオプション・ファイルです。

例

1. クラスター・プロセスが停止する前に、警告メッセージをユーザーに送信することなく、正常終了オプションを使用し、リソースを解放することによってクラスター・ノードをシャットダウンするには、次のように入力します。

   ```
   clstop -gr -s -y
   ```

2. クラスター・プロセスが停止する前に、警告メッセージをユーザーにブロードキャストし、すべてのクラスター・ノード上のクラスターを強制的に即時シャットダウンする (リソースは解放しません) には、次のように入力します。

   ```
   clstop -f -y
   ```

3. クラスター・プロセスが停止する前に、ユーザーに警告メッセージをブロードキャストし、正常終了オプションを使用し、リソースを解放することによってクラスター・ノードをシャットダウンするには、次のように入力します。

   ```
   clstop -gr -y
   ```

関連資料:

49 ページの『clmgr コマンド』

---

cltopinfo コマンド

目的

完全なトポロジー情報 (クラスター名、ネットワークの合計数、欠落したハートビートおよびクラスター内に構成されたノードの合計数) を表示します。 各ノードごとに、構成されたネットワークをすべて表示します。 各ネットワークごとに、構成されたインターフェースをすべて表示します。 また、定義されたすべてのリソース・グループを表示します。

構文

```
cltopinfo [-c] [-i] [-n] [-w]
```

フラグ

-c クラスター名およびセキュリティー・モード (標準または拡張) を表示します。

-i クラスター内に構成されたすべてのインターフェースを表示します。この情報に含まれるのは、インターフェース・ラベル、接続されているネットワーク (適切な場合)、IP アドレス、ネットマスク、ノード名、およびデバイス名です。

-n クラスター内に構成されたすべてのノードを表示します。各ノードごとに、定義されたすべてのネットワークをリストします。ネットワークごとに、サービス IP ラベル・エイリアス (定義されている場合) に定義されているすべてのインターフェースおよび配布設定をリストします。

-w クラスター内に構成されたすべてのネットワークを表示します。各ネットワークごとに、そのネットワークに接続されたすべてのノードをリストします。 ノードごとに、サービス IP ラベル・エイリアス (定義されている場合) に定義されているすべてのインターフェースおよび配布設定をリストします。

例 1

クラスター内で定義されたすべてのノードとネットワーク (ノード coffey1 および lee1) を表示するには、次の **cltopinfo** コマンドを使用します。次のクラスターは、IPv4 アドレスと IPv6 アドレスで構成されています。出力は次のようになります。

```
Cluster Name: hacmp_full_ipv6
Cluster Connection Authentication Mode: Standard
Cluster Message Authentication Mode: None
Cluster Message Encryption: None
Use Persistent Labels for Communication: No
There are 2 node(s) and 2 network(s) defined

NODE coffey1:
   Network net_ether_01
      service_ipv4_2   1.8.4.2
      service_ipv6_1   fe80::c862:67ff:fe58:5646
      coffey1_boot3   1.4.6.4
      coffey1_boot1   1.2.4.4
   Network net_ether_02
      service_ipv4_32   1.8.4.4
      service_ipv6_31  fe80::c862:67ff:fe58:5846
     coffey1_boot_v6  fe80::c872:67ff:fe59:8647
     coffey1_boot_v6  fe80::c872:678f:fe95:8683
NODE lee1:
   Network net_ether_01
      service_ipv4_2   1.8.4.2
      service_ipv6_1   fe80::c862:67ff:fe58:5646
      lee1_boot1   1.2.4.3
      lee1_boot3   1.4.6.3
   Network net_ether_02
      service_ipv4_32   1.8.4.4
      service_ipv6_31   fe80::c862:67ff:fe58:5846
      lee1_boot_v6   fe80::c672:fe56:fe82:2345
      lee1_boot_v6   fe80::fe34:3456:f873:f345

Resource Group RG1
   Startup Policy    Online On Home Node Only
   Fallover Policy    Fallover To Next Priority Node In The List
   Fallback Policy    Fallback To Higher Priority Node In The List
   Participating Nodes    coffey1 lee1
   Service IP Label       service_ipv4_1
   Service IP Label       service_ipv4_31
```

```
Resource Group RG2
   Startup Policy    Online On Home Node Only
   Fallover Policy    Fallover To Next Priority Node In The List
   Fallback Policy    Fallback To Higher Priority Node In The List
   Participating Nodes    lee1 coffey1
   Service IP Label       service_ipv4_2
   Service IP Label       service_ipv4_32
```

例 2

クラスター名と現在のセキュリティー・モードを表示するには、次の **cltopinfo** コマンドを使用します。出力は次のようになります。

```
# cltopinfo -c

 Cluster Name: c10
 Cluster Connection Authentication Mode: Standard
 Cluster Message Authentication Mode: None
 Cluster Message Encryption: None
 Use Persistent Labels for Communication: No
```

例 3

クラスター内に定義されたすべてのノードを表示するには、次の **cltopinfo** コマンドを使用します。次のクラスターは、IPv4 アドレスと IPv6 アドレスで構成されています。出力は次のようになります。

```
# cltopinfo -n

 NODE abby:
      Network net_ether_01
    abby_en1stby    192.168.121.7
    abby_en0boot    192.168.120.7
      Network net_ether_02
abby_boot1_v6  fe80::c872:67ff:fe59:8647
abby_boot2_v6  fe80::c872:678f:fe95:8683
      Network net_rs232_01
      Network net_rs232_02
    abby_tty0_01    /dev/tty0

 NODE polly:
      Network net_ether_01
    polly_en0boot    192.168.120.9
    polly_en1stby    192.168.121.9
    polly_en2boot    192.168.122.9
      Network net_ether_02
polly_boot1_v6  fe80::c672:fe56:fe82:2345
polly_boot2_v6  fe80::fe34:3456:f873:f345
      Network net_rs232_01
      Network net_rs232_02
    polly_tty0_01    /dev/tty0
```

例 4

クラスター内に定義されたすべてのネットワークを表示するには、次の **cltopinfo** コマンドを使用します。次のクラスターは、IPv4 アドレスと IPv6 アドレスで構成されています。出力は次のようになります。

```
# cltopinfo -w

 Network net_ether_01
      NODE abby:
    abby_en1stby    192.168.121.7
    abby_en0boot    192.168.120.7
      NODE polly:
```

```
    polly_en0boot    192.168.120.9
    polly_en1stby    192.168.121.9
    polly_en2boot    192.168.122.9

 Network net_ether_02
      NODE abby:
abby_boot1_v6  fe80::c872:67ff:fe59:8647
abby_boot2_v6  fe80::c872:678f:fe95:8683
      NODE polly:
polly_boot1_v6  fe80::c672:fe56:fe82:2345
polly_boot2_v6  fe80::fe34:3456:f873:f345

 Network net_rs232_01
      NODE abby:
      NODE polly:

 Network net_rs232_02
      NODE abby:
    abby_tty0_01    /dev/tty0
      NODE polly:
    polly_tty0_01    /dev/tty0
```

例 5

クラスター内に定義されたすべてのインターフェースを表示するには、次の **cltopinfo** コマンドを使用します。出力は次のようになります。

```
# cltopinfo -i
IP Label NetworkType   Node  Address If   Netmask   Pefixlenth
====================   ====  ======= ==== =======   ==========
abby_en1stby    net_ether_01  ether  abby  192.168.121.7  en2  255.255.255.0
abby_en0boot    net_ether_01  ether  abby  192.168.120.7  en1  255.255.255.0
abby_boot1_v6   net_ether_02  ether  abby  fe80::c872     en3           64
abby_boot2_v6   net_ether_02  ether  abby  fe80::c672     en4           64
abby_tty0_01    net_rs232_02  rs232  abby  /dev/tty0      tty0
polly_en0boot   net_ether_01  ether  polly 192.168.120.9  en1  255.255.255.0
polly_en1stby   net_ether_01  ether  polly 192.168.121.9  en2  255.255.255.0
polly_en2boot   net_ether_01  ether  polly 192.168.122.9  en3  255.255.255.0
polly_boot1_v6  net_ether_02  ether  polly fe80::c072     en4           64
polly_boot2_v6  net_ether_02  ether  polly fe80::c172     en5           64
polly_tty0_01   net_rs232_02  rs232  polly /dev/tty0      tty0
```

---

clvaryonvg コマンド

目的

ボリューム・グループをオンに変更します。

構文

```
clvaryonvg [-F] [-f] [-n] [-p] [-s] [-o] <vg>
```

説明

clvaryonvg コマンドは、AIX オペレーティング・システムの一部である **varyonvg** コマンドに代わるものとして設計されています。このコマンドは、AIX **varyonvg** コマンドを呼び出す前に、ボリューム・グループに対していくつかの検査を実行し、ボリューム・グループに何らかの変更が行われたかどうかを判別します。ボリューム・グループが最後にローカル側でオンに変更された後に変更が行われた場合、ボリューム・グループはエクスポートされ、オンに変更される前にインポートされます。この処理により、ボリューム・グループの内容の整合したビューがすべてのノードにあるかが検査されます。

ボリューム・グループ更新中にシステム障害が発生した場合、ボリューム・グループはノードから不可視になる可能性があります。システム障害に対して保護するために、以下のメカニズムが実装されています。

- ボリューム・グループをエクスポートする前に、*<VG>*.replay というファイルが /usr/es/sbin/cluster/etc/vg ディレクトリーに作成されます。 *VG* はボリューム・グループの名前です。このファイルは、ボリューム・グループがノードから不可視になった場合、またはボリューム・グループが存在しない場合にボリューム・グループを復元するための一連のコマンドを含むシェル・スクリプトです。ボリューム・グループが存在しない場合は、次回に **clvaryonvg** コマンドを使用すると、*<VG>*.replay ファイル内のコマンドが自動的に実行されます。
- replay ファイルによって問題が修正されない場合は、hacmp.out ファイル内のメッセージを参照できます。これらのメッセージには、ボリューム・グループの手動による復元方法についての説明があります。 hacmp.out ファイルのメッセージは /usr/es/sbin/cluster/etc/vg/*<VG>*.desc ファイルにもあります。 *VG* はボリューム・グループの名前です。失敗した replay ファイルのコピーは、/var/tmp ディレクトリーに入れられます。

フラグ

-f フラグを **importvg** コマンドまたは **varyonvg** コマンドに渡します。

-F ボリューム・グループに対して **exportvg** コマンドまたは **importvg** コマンドを実行し、タイム・スタンプを無視することによって、強制的に更新を行います。

-n ボリューム・グループ内の不整合物理区画の同期化を無効にします。このフラグは **varyonvg** コマンドに渡されます。

-p すべての物理ボリュームが、**clvaryonvg** コマンドを使用する対象として有効でなければならないことを指定します。

-s ボリューム・グループをシステム管理モードでのみ使用可能にします。

-o 完了後にボリューム・グループをオフに変更したままにします。完了処理では、すべての整合性検査が実行され、必要であれば **importvg** コマンドおよび **exportvg** コマンドが実行されます。

例

1. *vg03* というラベルのボリューム・グループを活動化するには、次のように入力します。

   ```
   clvaryonvg vg03
   ```

2. *vg03* というラベルのボリューム・グループでノードの情報を強制的に更新するには、次のように入力します。

   ```
   clvaryonvg -F vg03
   ```

---

get_local_nodename コマンド

目的

ローカル・ノードの名前を取り出します。

構文

```
get_local_nodename
```

説明

ローカル・ノードの名前を表示します。

例

ローカル・ノードの名前を表示するには、次のように入力します。

```
get_local_nodename
```

halevel コマンド

目的

システムにインストールされている PowerHA SystemMirror のバージョン、リリース、モディフィケーション、およびサービス・パック・レベルを表示します。

構文

```
halevel [-h|-?] [-s] [-x]
```

説明

このコマンドを PowerHA SystemMirror クライアントから実行した場合、コマンドは正しく機能しません。このコマンドは PowerHA SystemMirror サーバー・ノードから実行する必要があります。

フラグ

-h | -?
ヘルプ情報を表示します。

-s サービス・パック・レベルを表示します。

-x デバッグをオンにします (ksh set -x)。

例

1. PowerHA SystemMirror のバージョン、リリース、およびモディフィケーションのレベルを表示するには、次のように入力します。

   ```
   halevel
   ```

2. PowerHA SystemMirror のバージョン、リリース、モディフィケーション、およびサービス・パックのレベルを表示するには、次のように入力します。

   ```
   halevel -s
   ```

3. すべてのクラスター・ノード上の PowerHA SystemMirror のバージョン、リリース、モディフィケーション、およびサービス・パックのレベルを表示するには、次のように入力します。

   ```
   /usr/es/sbin/cluster/cspoc/cli_on_cluster -S halevel -s
   ```

rc.cluster コマンド

目的

rc.cluster コマンドを使用して、オペレーティング・システム環境をセットアップし、すべてのクラスター・ノードでクラスター・デーモンを始動します。

注**:** 特定のフラグに関連する引数は、フラグの直後に指定しなければなりません。 PowerHA SystemMirror

構文

```
rc.cluster [-boot] [b] [-i | -I] [-N | -R | -B] [-M | -A] [-r] [-v] [-x] [-C interactive|yes]
```

フラグ

-boot
IPAT が使用可能な場合、サービス・ネットワーク・インターフェースを、そのサービス・アダプターのブート・アドレスに置かれるように構成します。

-i クラスター情報 (**clinfoES**) デーモンをデフォルト・オプションで開始します。

-I トラップを有効にして、クラスター情報 (**clinfoES**) デーモンを始動します。

-b 始動をブロードキャストします。

-N 即座にデーモン (**inittab** ファイルの変更なし) を始動します。

-R PowerHA SystemMirror デーモンを、システム再始動時にのみ始動します。 PowerHA SystemMirror 起動コマンドが **inittab** ファイルに追加されます。

-B 即座にデーモンを始動し、PowerHA SystemMirror エントリーを **inittab** ファイルに追加します。

-C 問題が発生した場合の修正アクションに使用するモードを指定します。問題を自動的に修正するには、**yes** を指定してください。修正アクションのそれぞれが実行される前にプロンプトが出されるようにするには、**interactive** を指定してください。

-M 手動リソース獲得モードでクラスター・サービスを始動します。このオプションは、リソース・グループを手動でオンラインにする場合に使用します。

-A 自動リソース獲得モードでクラスター・サービスを始動します。 このオプションは、クラスターの起動時にリソース・グループを自動的にオンラインにする場合に使用します。これはデフォルト・オプションです。

-r 強制終了したあとで、クラスター・リソースを再獲得します。クラスターが強制終了されているときにいずれかのクラスター・リソース (IP ラベル、ディスク、アプリケーション) の状態を変更した場合は、このオプションを使用します。

-v 起動時 (auto ver sync) に検査エラーを無視します。

-x NFS クロスマウントを活動化します。

例

clinfo サービスを実行している状態でクラスターを始動し、イベントをブロードキャストするには、次のコマンドを実行します。

```
rc.cluster -boot -N -i
```

特記事項

本書は米国が提供する製品およびサービスについて作成したものです。

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒*103-8510*
東京都中央区日本橋箱崎町*19*番*21*号
日本アイ・ビー・エム株式会社
法務・知的財産
知的財産権ライセンス渉外

IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態で提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。 IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

記載されている性能データとお客様事例は、例として示す目的でのみ提供されています。実際の結果は特定の構成や稼働条件によって異なります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。 IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。 IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

IBM の将来の方向または意向に関する記述は、予告なしに変更または撤回される場合があり、単に目標を示しているものです。

表示されている IBM の価格は IBM が小売り価格として提示しているもので、現行価格であり、通知なしに変更されるものです。卸価格は、異なる場合があります。

本書はプランニング目的としてのみ記述されています。記述内容は製品が使用可能になる前に変更になる場合があります。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、類似する個人や企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。これらのサンプル・プログラムは特定物として現存するままの状態で提供されるものであり、いかなる保証も提供されません。 IBM は、お客様の当該サンプル・プログラムの使用から生ずるいかなる損害に対しても一切の責任を負いません。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生した創作物には、次のように、著作権表示を入れていただく必要があります。

© (お客様の会社名) (西暦年).

このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。

© Copyright IBM Corp. _年を入れる_.

プライバシー・ポリシーに関する考慮事項

サービス・ソリューションとしてのソフトウェアも含めた IBM ソフトウェア製品 (「ソフトウェア・オファリング」) では、製品の使用に関する情報の収集、エンド・ユーザーの使用感の向上、エンド・ユーザーとの対話またはその他の目的のために、Cookie はじめさまざまなテクノロジーを使用することがあります。多くの場合、ソフトウェア・オファリングにより個人情報が収集されることはありません。 IBM の「ソフトウェア・オファリング」の一部には、個人情報を収集できる機能を持つものがあります。ご使用の「ソフトウェア・オファリング」が、これらのCookie およびそれに類するテクノロジーを通じてお客様による個人情報の収集を可能にする場合、以下の具体的事項を確認ください。

この「ソフトウェア・オファリング」は、Cookie もしくはその他のテクノロジーを使用して個人情報を収集することはありません。

この「ソフトウェア・オファリング」が Cookie およびさまざまなテクノロジーを使用してエンド・ユーザーから個人を特定できる情報を収集する機能を提供する場合、お客様は、このような情報を収集するにあたって適用される法律、ガイドライン等を遵守する必要があります。これには、エンドユーザーへの通知や同意の要求も含まれますがそれらには限られません。

このような目的での Cookie などの各種テクノロジーの使用について詳しくは、『IBM オンラインでのプライバシー・ステートメントのハイライト』(http://www.ibm.com/privacy/jp/ja/)、『IBM オンラインでのプライバシー・ステートメント』(http://www.ibm.com/privacy/details/jp/ja/) の『クッキー、ウェブ・ビーコン、その他のテクノロジー』というタイトルのセクション、および『IBM Software Products and Software-as-a-Service Privacy Statement』(http://www.ibm.com/software/info/product-privacy) を参照してください。

商標

IBM、IBM ロゴおよび ibm.com は、世界の多くの国で登録された International Business Machines Corp. の商標です。他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。現時点での IBM の商標リストについては、http://www.ibm.com/legal/copytrade.shtml をご覧ください。

UNIX は、The Open Group の米国およびその他の国における登録商標です。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

C

R

Printed in Japan

日本アイ・ビー・エム株式会社
〒103-8510 東京都中央区日本橋箱崎町19-21