CA Spectrum®

分散 SpectroSERVER 管理者ガイド

リリース 9.3

CA Technologies 製品リファレンス

このドキュメントでは、CA Spectrum ® について説明します。

CA への連絡先

テクニカル サポートの詳細については、弊社テクニカル サポートの Web サイト（http://www.ca.com/jp/support/）をご覧ください。

目次

第 1 章: 分散 SpectroSERVER の概要

このセクションには、以下のトピックが含まれています。

分散 SpectroSERVER について

分散 SpectroSERVER (DSS) は、大規模インフラストラクチャ全体に対して、管理の分散を可能にする強力なモデリング機能です。 インフラストラクチャは、地理的に整理することも、1 つの物理的な場所で複数のサーバにわたって整理することもできます。IT インフラストラクチャを管理する場合、DSS は CA Spectrum パフォーマンスを向上させることができます。 管理上のトラフィックでネットワーク負荷を分散し、リモート ワークステーションにタスクを委任することで、パフォーマンスが向上する可能性があります。

DSS を使用すると、インフラストラクチャの統合表示を作成することができます。これは、それぞれにローカルの SpectroSERVER を持つ複数のランドスケープから構成されます。 DSS 環境で、OneClick コンソールなど SpectroSERVER クライアントは、同時に複数の SpectroSERVER から情報にアクセスできます。

DSS の主な機能は、以下のとおりです。

分散表示

OneClick コンソールは、展開されたすべての SpectroSERVER に対して情報を表示します。 サマリ アラーム数は、各 SpectroSERVER に対して表示されます。 ネットワーク管理者は迅速にトラブル領域を見つけることができます。

ポーリング トラフィックのローカル分散

ネットワークをポーリングしている管理ワークステーションは、DSS で管理するデバイスに地理的に近くなります。 この設定では、ワイド エリア リンクのトラフィックが減少し、ローカル ネットワークの輻輳が回避されます。 遠距離からデバイスをポーリングする単一の SpectroSERVER よりも、複数の小型の SpectroSERVER が生成するトラフィックは少なくなります。

スケーラビリティ

ネットワークが拡張するにつれて、追加の専用 SpectroSERVER サーバとして、ローエンドのワークステーションを使用するコストが低下することがあります。1 台のワークステーションをアップグレードして、非分散型の大規模環境で 1 台の SpectroSERVER に格納するよりも、このようなソリューションをお勧めします。

フォールトトレランス

DSS は、SpectroSERVER 間のフォールト トレランスをサポートします。セカンダリ SpectroSERVER は、プライマリ SpectroSERVER に対する、フォールト トレランス用のバックアップまたはスタンバイとして提供できます。 プライマリ SpectroSERVER を実行しているワークステーションで障害が発生しても、ネットワーク管理は続行されます。

指定のランドスケープ用のデータベースをセカンダリ SpectroSERVER に再ロードするだけで、障害から保護することができます 障害が発生してプライマリ SpectroSERVER が無効になると、セカンダリ SpectroSERVER は自動的に処理を継承します。 CA Spectrum アプリケーションはすべて、自動的にセカンダリ SpectroSERVER を使用します。

詳細情報:

フォールト トレランスの確立 (P. 87)

ランドスケープ

単一の SpectroSERVER が管理するネットワーク ドメインは、ランドスケープとして知られています。 ランドスケープは、特定の SpectroSERVER に属するモデル、関連付け、属性値、アラーム、イベント、および統計から構成されます。 ネットワーク内のランドスケープはそれぞれ一意です。 一意のランドスケープ ハンドル（ID）により、各ランドスケープが識別されます。

OneClick では、ランドスケープ アイコンは各ランドスケープを表します。ランドスケープ アイコンは SpectroSERVER データベースを表します。ダブルクリック ゾーンおよびメニュー選択を介して、ランドスケープ アイコンにより、リモート ネットワーク モデルにアクセスできます。 またこのアイコンは、前述のリモート データベースでモデリングされるデバイスのアラーム情報のロールアップも表しています。

注: ローカルおよびリモートという用語は、特定の SpectroSERVER から見たランドスケープの指定に使用されます。

ランドスケープ マップ

CA Spectrum は、特定の DSS 環境を構成するすべてのランドスケープの「マップ」を自動的に維持します。 SpectroSERVER トポロジで変更が発生すると、内部ディスカバリ メカニズムが起動され、その環境内の他の SpectroSERVER のすべてのランドスケープ マップが更新されます。たとえば、ディスカバリにより、ネットワークに対して追加または削除された SpectroSERVER が検出されます。

ランドスケープ モデルはコンテナ モデルです。ランドスケープ モデルを使用すると、CA Spectrum ユーザ インターフェースにより、ランドスケープ マップ内の他の SpectroSERVER に接続できます。トポロジ、ロケーション、組織の 3 つの表示階層のいずれのレベルでも、ランドスケープ モデルを作成することができます。

モデリング カタログ

モデリング カタログは、特定の SpectroSERVER にインストールされる 1 セットのテンプレート（モデル タイプまたは関係など）です。 これらのテンプレートはモデルの作成に使用されます。 これらのテンプレートによって作成されるモデルは、その SpectroSERVER のランドスケープを構成します。

注: OneClick が起動すると、使用可能なモデル タイプのリストについて、デフォルト ランドスケープにクエリします。

DSS を使用して複数のランドスケープをモデリングする場合、各ランドスケープには同一のモデリング カタログを含める必要があります。1 つのランドスケープのモデリング カタログ内のすべてのモデルは、接続先の他のすべてのランドスケープのモデリング カタログにも存在する必要があります。 1 つの SpectroSERVER に新しい管理モジュールをインストールした場合、ランドスケープ マップ内の他のすべての SpectroSERVER にも、同じ管理モジュールをインストールする必要があります。

ユーザ モデル

ユーザ モデルには、個別のユーザ権限および他のユーザ データについての情報が含まれています。 この情報は、ランドスケープ内に保存されます。 初めて CA Spectrum をインストールする場合、デフォルト ユーザ モデルは CA Spectrum インストール中に［インストール所有者］フィールドで指定したユーザを表します。

注: 詳細については、「管理者ガイド」を参照してください。

DSS 環境ではランドスケープ マップ内のリモート SpectroSERVER に接続し、さまざまなランドスケープが相互に通信できるようにするには、すべてのランドスケープに同じユーザ モデルが存在する必要があります。 ユーザ モデルを追加または変更すると、ランドスケープと関連付けられた SpectroSERVER は、ランドスケープ マップ内の他のすべての SpectroSERVER にクエリを実行します。クエリにより、他のランドスケープ内のユーザ モデルがチェックされます。 ユーザ モデルが他のランドスケープに存在する場合、他のランドスケープ内のユーザ モデルは自動的に更新され、最初のユーザ モデルに対して行われた変更が反映されます。

注: ユーザ モデルを削除すると、そのランドスケープ内のユーザ モデルのみが削除されます。ランドスケープ マップ内の他の SpectroSERVER の重複するユーザ モデルは、手動で削除する必要があります。

SpectroSERVER (.vnmrc) リソース

SpectroSERVER リソースは、*<spectrum>*/SS/.vnmrc ファイルで定義されます。これらのリソース ファイルのデフォルト値の多くは、コード内に埋め込まれ、空白のままになっています。 しかし、これらのリソースで値を指定すると、コード内に記載されたデフォルト値は使用されません。

.vnmrc（Virtual Network Machine Runtime Configuration、仮想ネットワーク マシン ランタイム設定）ファイルのリソースは、SpectroSERVER のパス名とデフォルト設定を定義します。CA Spectrum システム ソフトウェアには、SpectroSERVER リソースのデフォルト設定を指定した、ランタイム設定ファイルが含まれています。

リソースの値を変更することによって、以下の変更を行うことができます。

- 一般的な SpectroSERVER リソースとディレクトリ パスを定義する
- イベント アーカイブを調整する
- ネーム サービス変数を指定する
- スレッド割り当てを調整する
- フォールト トレラント アラーム同期を制御する

これらのリソース エントリはすべて、「resource = resource_value」の形式で一覧表示されます。 .vnmrc ファイル内のリソースの多くで、リソース値は空白になっています。

特定のリソースに値が表示されない場合、CA Spectrum はデフォルト値を使用します。 .vnmrc ファイル内のリソースは、SpectroSERVER の起動時に有効になります。 SpectroSERVER 実行中にこのファイルを変更した場合、変更は SpectroSERVER の再起動時に有効になります。

例: .vnmrc リソースの定義

この例では、イベント ログ データベースに記録されるレコードの最大数を設定します。

```
max_event_records=5000
```

max_event_records

リソース名です。

5000

リソース値です。

一般的な SpectroSERVER（.vnmrc）リソース

一般的な SpectroSERVER（vnmrc）リソースは、多くの SpectroSERVER 機能を制御します。 一般的な SpectroSERVER（.vnmrc）リソースの説明は以下のとおりです。

comm_port

SpectroSERVER と通信するためにクライアント ユーザ インターフェースが使用する TCP ポートを指定します。 ポート番号はインストール中に定義されます。

コマンドの形式は以下のとおりです。

`comm_port=0xBEEF`

0xBEEF

デフォルトの TCP 接続ポート ソケットです。 このソケットは、/netinet/in.h ファイル内の IPPORT_USERRESERVED パラメータに割り当てられるポート番号より大きい、有効な任意の TCP ポートになる可能性があります。 ただし、ポートは 65535 未満（0xFFFF）である必要があります。

expiration_date

ユーザの CA Spectrum ライセンスが期限切れになる日付を示します。 インストール中に入力する CA Spectrum キーには有効期限日が含まれています。

コマンドの形式は以下のとおりです。

`expiration_date=mm/dd/yyyy`

パスワード

インストール中に入力された SpectroSERVER キーが自動的に設定されます。 インストール後、このリソースは変更しないでください。

コマンドの形式は以下のとおりです。

`password=password`

パスワードは CA によって提供されます。

snmp_comm_port

SNMP 要求を SpectroSERVER を介して送信できるポートを指定します。

コマンドの形式は以下のとおりです。

`snmp_comm_port=port#`

port#

0x400（1,024）～ 0xFFFF（65,535）の範囲の任意の符号なし 16 ビット整数に設定できます。 SNMP エージェントのインプリメンテーション（IBM Mainframe MVS システムなど）によっては、符号付き数値として扱われます。 その場合、このリソースは 0x400（1,024）～ 0x7FFF（32,768）の値に設定します。

デフォルト： 0xFFFF

snmp_trap_port_enabled

SpectroSERVER を SNMP トラップ ポートにバインドし、トラップをリスンします。 FALSE に設定すると、SpectroSERVER は SNMP トラップ ポートにバインドしません。

コマンドの形式は以下のとおりです。

`snmp_trap_port_enabled=TRUE`

デフォルト： True

vnm_file_path

データベース ファイルなど、SpectroSERVER の外部ファイルを格納するルート サブディレクトリを指定します。

コマンドの形式は以下のとおりです。

`vnm_file_path=<directory>/spectrum/SS/CsVendor`

/spectrum/SS/CsVendor

ルート サブディレクトリのファイル パス

tcp_buffer_size

LAN 上のトラフィックに合わせて、適切なバッファ サイズに変更します。 バッファ サイズが大きいと、スループットが向上します。 このリソースは、TCP バッファのバイト数を表す数値を受け入れます。 コマンドの形式は以下のとおりです。

```
tcp_buffer_size=<空白>
```

デフォルト： 未指定

制限： 8192 ～ 65536 バイトまで

注: 8 KB 未満の値は 8 KB に切り上げられます。 64 KB を越える値は 64 KB に設定されます。

resource_file_path

Ether Map など、VNM リソース ファイルのファイル パス

コマンドの形式は以下のとおりです。

```
resource_file_path=./CsResource
```

./CsResource

（オプション）VNM リソース ファイルのファイル パスです。 このパラメータはブランクのままにすることができます。

wait_active

サーバがすべてのモデルのロード後ただちに接続を受け入れるか、すべてのモデルがアクティブになるまで待つかを決定します。 *[Yes]* に設定すると、コントロール パネルのメッセージは、SpectroSERVER の起動中にアクティブにされたモデルの実行パーセントを表示します。

コマンドの形式は以下のとおりです。

```
wait_active=no
```

デフォルト： no

max_bind_retry_count

リスン ソケット バインド リトライの最大回数を指定します。

コマンドの形式は以下のとおりです。

```
max_bind_retry_count=50
```

デフォルト： 50

bind_retry_interval

リスン ソケット バインド再試行間の遅延を秒単位で指定します。

コマンドの形式は以下のとおりです。

```
bind_retry_interval=30
```

デフォルト： 30 秒

min_client_version

クライアント接続に対して SpectroSERVER が受け入れる最小の parm ブロック バージョンです。

コマンドの形式は以下のとおりです。

```
min_client_version=0
```

デフォルト： 最小クライアント接続 0

max_connections

SpectroSERVER が受け入れる最大接続数です。 接続は、他の SpectroSERVER、アーカイブ マネージャ、ロケーション サーバ（メインまたはローカル）など、クライアントからの場合もあれば、他のサーバからの場合もあります。

コマンドの形式は以下のとおりです。

```
max_connections=50 gensv000316 docs002958
```

デフォルト： 50 接続

クライアント接続の最大数を超過すると、接続は失敗することがあります。 SpectroSERVER の最大接続数を増やした場合、パフォーマンスを向上させるため、場合によって、ワークステーション上のオープン ファイル記述子の数を増やす必要があります。

handshake_timeout

接続ハンドシェーク中に初期 ID 情報を交換する時間を秒単位で設定します。

コマンドの形式は以下のとおりです。

```
handshake_timeout=40
```

デフォルト： 40 秒

vnm_message_timeout

vnm_message_timeout リソースは、VNM 間で送信されるメッセージのタイムアウトを秒単位で設定します。

コマンドの形式は以下のとおりです。

```
vnm_message_timeout=180
```

デフォルト： 180 秒

vnm_close_timeout

vnm_close_timeout リソースは、VNM 間で接続をクローズするタイムアウトを秒単位で設定します。

コマンドの形式は以下のとおりです。

```
vnm_close_timeout=180
```

デフォルト： 180 秒

connect_time_limit

VNM への接続のタイムアウトをミリ秒単位で設定します。

コマンドの形式は以下のとおりです。

```
connect_time_limit=5000
```

デフォルト： 5000 ミリ秒

rcpd_comm_port

リモート コピー プロセス デーモン（rcpd）のリスン ポートを指定します。 このリソースは、フォールト トレラント データベースのバックアップに使用されます。

コマンドの形式は以下のとおりです。

```
rcpd_comm_port=0xCAFE
```

デフォルト： 0xCAFE

procd_comm_port

VNM がプロセス デーモンへの接続に使用するポートを指定します。

コマンドの形式は以下のとおりです。

```
procd_comm_port=0xFEEB
```

デフォルト： 0xFEEB

snmp_trap_port

VNM がトラップを受信するポートを指定します。

コマンドの形式は以下のとおりです。

```
snmp_trap_port=162
```

デフォルト： 162

enable_traps_for_pingables

CA Spectrum が Ping 可能モデルで SNMP トラップを受信できるかどうかを指定します。 デフォルトでは、.vnmrc ファイルにこのリソースが含まれていません。 次の構文を使用して、ファイルに手動で追加します。

```
enable_traps_for_pingables = TRUE
```

デフォルト： TRUE

device_limit

デバイス制限適用時にモデリングできるデバイスの最大数を指定します。 値は、インストール中に入力される CA Spectrum キーから取得します。 この値を編集することはできません。 簡単なホスト、Ping 可能デバイス、またはノードのプロキシ エージェントではない管理対象ノードは、モデリングされたデバイスです。 デバイス制限が CA Spectrum インストールに適用されない場合、このリソースと関連付けられた値はありません （max_device も参照）。

コマンドの形式は以下のとおりです。

```
device_limit=<# of device models>
```

# of device models

IPAddress (0x000102bc) モデル タイプおよびデバイス (0x0001004b) モデル タイプから取得されたモデル数。

さらに、DIRIG_NODE アプリケーション、およびブロードバンド モデル MOTCBLMODEM、SAEXPLORER および DOCSISCM ら取得されたモデルがカウントされます。

ただし、以下は例外です。

GEN_HOST、PINGABLE、PATROL_PET、WINDOWS_PC。

max_device

その device_limit リソースが有効な CA Spectrum 製品に適用されます。デフォルトでは、モデル数が指定されたデバイス制限の 80% に到達すると、CA Spectrum は黄色のアラームを生成します。 モデル数が指定されたデバイス制限の 100% に到達すると、CA Spectrum は赤のアラームを生成します。 このリソースにより、実際のデバイス制限より小さい値を設定して、実際より前に警告を発生させることができます。

コマンドの形式は以下のとおりです。

```
max_devices=<# of device models>
```

# of device models

デバイス モデル タイプから取得した IP アドレスを持ち、かつ、指定した値が device_limit の値より下がったモデルを参照します。デフォルト値は空白です。

disable_redundancy_when_using_loopback

ループバックによってデバイスがモデリングされた場合、冗長性（0x11d2c）を無効にします。 このパラメータは、VNM モデルの use_loopback（0x12bb）属性が True に設定されている場合にのみ適用されます。

コマンドの形式は以下のとおりです。

```
disable_redundancy_when_using_loopback=False
```

デフォルトでは、このリソースは .vnmrc ファイル内にないので、自動的に False として評価されます。

注: 詳細については、「IT インフラストラクチャのモデリング/管理 - 管理者ガイド」を参照してください。

persistent_alarms_active

SpectroSERVER のシャット ダウン時、CA Spectrum がアラーム関連の情報を保持しないようにします。

コマンドの形式は以下のとおりです。

```
persistent_alarms_active=<False>
```

デフォルトでは、このリソースは .vnmrc ファイルに表示されません。SpectroSERVER がシャット ダウンし、再起動すると、CA Spectrum は自動的にアラーム関連の情報（トラブルシュータ割り当てまたはステータスなど）を保持します。 シャットダウンの前に存在していたアラームが保持されます。 これらのアラームは、「永続」アラームと見なされます。persistent_alarms_active=False を .vnmrc ファイルに追加すると、SpectroSERVER がシャットダウンしたときに、CA Spectrum がアラーム関連情報を保持しないようにすることができます。

注: アラーム関連の情報を保持するサードパーティ製のアプリケーションを統合しない限り、persistent_alarms_active リソースを追加しないことを強く推奨します。

このリソースを追加すると、以下の効果があります。

- 追加されたアラーム ステータスが失われる（過去のイベントとしてのみ記録される）。
- SpectroSERVER シャットダウン時に存在したアラームが、新しいタイム スタンプで再生成される。
- 再生成されたアラームがアラーム通知ツールに転送される。

unsupported_attr_poll_interval

外部属性が「noSuchName」エラーを返した場合に、その外部属性をポーリングするまで sp> が待機する時間を指定します。 このパラメータが .vnmrc ファイルで指定されていない場合、デフォルト値（12 時間）が使用されます。

コマンドの形式は以下のとおりです。

```
unsupported_attr_poll_interval=43200
```

デフォルト： 43200 秒（12 時間）

詳細情報:

リモート コピー プロセス デーモン（rcpd）のポート番号の設定 (P. 75)

イベント アーカイブ（.vnmrc）リソース

イベント アーカイブ（.vnmrc）リソースは、ログ記録用に SpectroSERVER がアーカイブ マネージャにイベントを送信するプロセスを制御します。 イベント アーカイブ（.vnmrc）リソースの説明は以下のとおりです。

max_event_records

イベント ログ データベースに保存できるレコードの最大数を指定します。

コマンドの形式は以下のとおりです。

```
max_event_records=# of Records
```

# of Record

イベント ログ レコードの数です。 最小値は、event_record_increment の値と等しくなります。 システム記憶容量は最大値を制限します。

デフォルト： 20,000

event_record_increment

レコード数が max_event_records 値を超えた場合、イベント ログ データベースから削除するレコードの数を指定します。

コマンドの形式は以下のとおりです。

```
event_record_increment=# of Records
```

# of Records

データベースから削除されるイベント ログ レコードの数。最小値は 100 です。

event_batch_max_size

1 バッチあたりに許容されるイベントの最大数を設定します。 バッチがいっぱいになると、イベント バッチはただちにアーカイブ マネージャに送信されます。

コマンドの形式は以下のとおりです。

```
event_batch_max_size=1000
```

デフォルト： 1000（イベントの最大数）

event_batch_timeout

イベント バッチがアーカイブ マネージャに送信される際の時間を秒単位で決定します。

コマンドの形式は以下のとおりです。

event_batch_timeout=*1*

デフォルト： *1* 秒

log_user_events

ユーザが開始したモデル属性値の書き込みそれぞれに対してイベントを生成するかどうかを制御します。 値を True にすると、VNM がイベントを生成します。

コマンドの形式は以下のとおりです。

log_user_events=*False*

デフォルト： False

use_log_queue

イベントが生成されたときに個別のキューにイベントを配置します。これらのイベントは、個別のスレッドにあるアーカイブ マネージャに送信されます。「TRUE」に設定すると、イベントは生成されたスレッドと同じスレッドにあるアーカイブ マネージャに送信されます。この状況では、アラーム作成が遅れる可能性があります。

コマンドの形式は以下のとおりです。

use_log_queue=<*空白*>

デフォルト： <*空白*>

ワーク スレッド(.vnmrc)リソース

SpectroSERVER はマルチスレッドのプロセスです。 通常の動作中、各サブシステムは多数のワーク スレッドを割り当てます。

SpectroSERVER は、一定期間再利用するワーク スレッドのプールを維持しています。 作業アクティビティが増加している期間中、サブシステムはそれぞれの上限までプールのスレッドを使用します。 作業スレッドの共通プールが使い果たされると、新しい作業スレッドが作成されます。プールは、増加したアクティビティを格納するため、拡大します。

サブシステムが必要としなくなったワーク スレッドは、後で使用できるように、共通プールに返されます。その後のニーズを満たすためにスレッドがプールから取得されることもあれば、プールが空の場合は、新しいスレッドが割り当てられることもあります。 指定された期間、未使用のままのスレッドはプールから削除され、そのリソースはシステムに返されます。 この処理を「エージング」といいます。

ワーク スレッド（.vnmrc）リソースの説明は以下のとおりです。

max_total_work_threads

すべての SpectroSERVER サブシステムに割り当てることができるワーク スレッドの最大数を指定します。各作業スレッドは処理リソースを消費し、メモリの重要なブロックを必要とします。 システム容量（メモリ サイズと速度）に基づいてその値を設定します。

このリソースの値が大きすぎると、SpectroSERVER でメモリ不足が定期的に発生することがあります。 このリソースの値が小さすぎると、SpectroSERVER の動作が遅くなることがあります。

このリソースの値を新たに設定または変更した場合に SpectroSERVER を再起動すると、CA Spectrum は新しい値を読み取ります。 読み込まれた値を使用して、VNM モデルの WorkThreadsMaxAvail 属性の値が更新されます。 WorkThreadsMaxAvail の値が更新されると、max_total_work_threads の値が .vnmrc ファイルから削除されます。

コマンドの形式は以下のとおりです。

```
max_total_work_threads=# of threads
```

VNM の WorkThreadsMaxAvail 属性のデフォルト値は 500 です。

work_thread_age

サブシステムが必要としなくなった作業スレッドは、作業スレッド プールに返されます。このリソースでは、ワーク スレッドを未使用のままプールに残しておくことができる期間（秒単位）を指定します。

このリソースを、サブシステム アクティビティと互換性のある値に設定します。ワーク スレッドを作成するには、大量の処理オーバーヘッドが必要です。 このリソースの値を低く設定しすぎると、作業スレッド要求に応じて新しいスレッドを頻繁に作成することになり、システム リソースに負担がかかります。 値を高くしすぎると、リソースが不必要にコミットされ続けることを意味します。

コマンドの形式は以下のとおりです。

```
work_thread_age =seconds
```

デフォルト： 60 秒

フォールトトレラント アラーム サービス（.vnmrc）のリソース

フォールト トレラント環境では、プライマリとセカンダリの SpectroSERVER 間でアラームを同期する必要があります。 サーバはアラーム情報を交換するために接続します。 初期接続試行が失敗した場合、以降の試行を実行できます。 フォールト トレラント アラーム サービスは、以下の設定を使って、アラーム同期を制御します。

ftasv_enabled

アラーム同期を有効にします。プライマリおよびセカンダリの両方のサーバに対して、.vnmrc ファイルにこのコマンドを含めます。

コマンドの形式は以下のとおりです。

```
ftasv_enabled=true
```

デフォルト： True

重要: 一方のサーバの ftasv_enabled を True に設定し、もう一方のサーバを False に設定することはできません。 両方のサーバに、同じ値を使用します。

ftasv_max_conn_retry_count

接続の試行が失敗した後、同期のために、プライマリ サーバがセカンダリ サーバへの接続を試行する回数を指定します。このパラメータは、プライマリ SpectroSERVER の .vnmrc ファイルで必要です。

注: セカンダリ サーバへの接続を試行する場合、プライマリ サーバはこのパラメータを使用します。セカンダリ サーバがプライマリへの接続を試行する場合、このパラメータは使われません。

コマンドの形式は以下のとおりです。

```
ftasv_max_conn_retry_count=# of retries
```

デフォルト： 4 回（セカンダリ サーバに関する同期試行は合計 5 回）。

ftasv_conn_retry_interval

プライマリ サーバがセカンダリ サーバとの同期を移行する間隔を秒数で指定します。 プライマリ SpectroSERVER の .vnmrc ファイルでこのパラメータを使用します。

注: セカンダリ サーバへの接続を試行する場合、プライマリ サーバはこのパラメータを使用します。セカンダリ サーバがプライマリへの接続を試行する場合、このパラメータは使用されません。

コマンドの形式は以下のとおりです。

```
ftasv_conn_retry_interval=# of seconds
```

デフォルト： 30 秒

ftasv_debug

アラーム同期アクティビティ用に、デバッグ出力を有効にします。 プライマリおよびセカンダリの両方のサーバに対して、.vnmrc ファイルにこのコマンドを含めます。 デバッグ出力は、各サーバの VNM.OUT ファイルに書き込まれます。 各メッセージは、「Fault Tolerant Alarm Service」という文字列で始まります。

コマンドの形式は以下のとおりです。

```
ftasv_debug=true
```

デフォルト： False

重要: プライマリ サーバによる接続試行時に、セカンダリ SpectroSERVER が動作していない場合、プライマリは同期試行を使い果たします。 その結果、プライマリ SpectroSERVER のスタートアップは遅れます。 ftasv_max_conn_retry_count と ftasv_conn_retry_interval に対してデフォルト設定を使用した場合の遅延は 2 分（30 秒間隔で 4 回の試行）です。 この遅延はモデルのアクティブ化の前に発生するため、回避不能です。 回避策は、プライマリの起動時にセカンダリ SpectroSERVER が動作していることを確認することです。 また潜在的な遅延を縮小するため、再試行回数または再試行間隔を減らすこともできます。

詳細情報:

フォールト トレランス (P. 77)
SpectroSERVER アラーム同期 (P. 82)

CA Spectrum ユーティリティをパックして別のコンピュータに移動する方法

packtool.pl スクリプトは CA Spectrum コマンド ライン インターフェース（CLI）、CA Spectrum SS Logger、AlarmNotifier、modelinggateway ツール、および sbgwimport ツールをパックして、別のコンピュータに転送することができます。このスクリプトを使用すると、ファイルの移動時に、ユーティリティとサポート ファイルの相対ディレクトリ構造を維持することができます。

注: どちらのコンピュータも同じオペレーティング システムで動作している必要があります。

packtool.pl スクリプトを実行する前に、以下の要件を考慮してください。

- packtool.pl スクリプトを実行するには、Solaris および Linux プラットフォームでは root ユーザ、Windows プラットフォームでは管理権限を持つユーザである必要があります。
- OneClick サーバだけがインストールされているコンピュータに CA Spectrum ユーティリティを転送する場合、以下の要件が適用されます。
  - 各コンピュータ上の OneClick サーバのバージョンおよびパッチ レベルは一致している必要があります。 各パッチをインストールした後、packtool.pl スクリプトを再実行し、再度ユーティリティを転送します。
  - ユーティリティの転送先のコンピュータのインストール ユーザは、ファイルをパックしたコンピュータのインストール ユーザと同じ必要があります。
  - ターゲット コンピュータにデバッグ パッチまたは PTF がインストールされる場合、ファイル転送後にデバッグ パッチまたは PTF を再インストールします。
- ユーティリティをパックするコンピュータでは、すべての CA Spectrum プロセスを停止します。

- tools_bundle ファイルを抽出する前に、ユーティリティを転送しているコンピュータ上の CA Spectrum プロセスをすべて停止します。
- tools_bundle ファイルを解凍するには、Solaris および Linux プラットフォームでは root ユーザ、Windows プラットフォームでは管理権限を持つユーザである必要があります。

重要: CA Spectrum ユーティリティを実行するには、有効な CA Spectrum ユーザとしてターゲット コンピュータにログオンする必要があります。そうでない場合、エラーが発生します。たとえば、Linux ワークステーションに「ルート」としてログオンし、CA Spectrum に一致するユーザ モデルが存在しない場合、「NO_USER」というエラーを受け取ります。

CA Spectrum ユーティリティをパックして別のコンピュータに転送するには、以下のステップを実行します。

1. packtool.pl スクリプトを使用して SPECTRUM ユーティリティをパックします。 (P. 28)
2. 別のコンピュータ上で tools_bundle ファイルを解凍します (P. 29)。

CA Spectrum ユーティリティのパック

CLI、SSLogger、AlarmNotifier、modelinggateway ツール、および sbgwimport ツールを別のコンピュータに移す前に、これらをバンドルするスクリプトを実行します。

次の手順に従ってください:

1. 環境変数 *<$SPECROOT>* が、ユーティリティをパックするコンピュータ上の SpectroSERVER インストール ディレクトリ パスに設定されていることを確認します。
2. packtool.pl スクリプトを実行して、ユーティリティおよびサポート ファイルをパックします。 packtool.pl スクリプトは *<$SPECROOT>*/SS-Tools ディレクトリにあります。

Bash シェルまたは他の UNIX シェルからスクリプトを実行するには、次のコマンドを入力します。

```
<$SPECROOT>/SS-Tools/packtool.pl [-no_notifier | -no_event_alarms]
[-f file_name]
```

<$SPECROOT>

CA Spectrum がインストールされている SpectroSERVER 上のディレクトリ構造です。

-no_notifier

AlarmNotifier をパックしないように指定します。

-no_event_alarms

AlarmNotifier EvFormat ファイルまたは PCause ファイルをパックしないように指定します。

-f *file_name*

実行可能ファイルの名前を指定します。

CA Spectrum ユーティリティとそのサポート ファイルを格納した linux_tools_bundle（Linux）、solaris_tools_bundle（Solaris）、または nt_tools_bundle.exe（Windows）という名前の実行可能ファイルが作成されます。

別のコンピュータへの CA Spectrum ユーティリティの移動

OneClick サーバを実行しているだけのコンピュータに CA Spectrum ユーティリティを移動することができます。

次の手順に従ってください:

1. CA Spectrum ユーティリティをパックします (P. 28)。
2. CA Spectrum ユーティリティを解凍するコンピュータ上で、ディレクトリを *<$SPECROOT>* に変更します。
3. バイナリ モードを使用して、linux_tools_bundle（Linux）、solaris_tools_bundle（Solaris）または nt_tools_bundle.exe（Windows）という名前の実行可能ファイルを現在のディレクトリに FTP 接続します。
4. DOS、Bash、または他の UNIX シェルから tools_bundle ファイルを解凍します。

   CA Spectrum ユーティリティとそのサポート ファイルが、適切なディレクトリ構造に解凍されます。

5. AlarmNotifier とイベント ファイルおよび pcause ファイルを移動した場合、<*$SPECROOT*>/SG-Support/CsResource/preferences/*.prf 内のパスにイベント ファイルと pcause ファイルの正しい場所が指定されていることを確認します。 また、ui= オプションと lhandle= オプションを確認します。
6. <*$SPECPATH*> 環境変数を、tools_bundle ファイルを解凍したディレクトリのパスに設定します。
   - Windows の場合は、以下の変数形式の値を使用してシステム環境変数 <*$SPECPATH*> を作成します。
     ```
     driveletter:¥PATH_TO_SPECTRUM
     ```
   - Solaris の場合は、以下の行を /opt/SPECTRUM/spectrum60.env ファイルに追加します。
     ```
     SPECPATH=PATH_TO_SPECTRUM
     ```
   - Linux の場合は、以下の行を /opt/SPECTRUM/spectrum80.env ファイルに追加します。
     ```
     SPECPATH=PATH_TO_SPECTRUM
     ```
7. Solaris および Linux プラットフォームの場合は、/opt/SPECTRUM ディレクトリにある spectrum*.env ファイル内で BES_LIC_DIR を PATH_TO_SPECTRUM/bin/VBNS/license として定義します。
8. Linux プラットフォームの場合は、/usr/bin/perl を <*$SPECROOT*>/bin にコピーします。
9. .hostrc ファイルにローカル ホスト名、およびユーティリティの転送元のコンピュータ名が含まれることを確認します。
10. メイン ロケーション サーバの .hostrc ファイルに tools_bundle ファイルを解凍するコンピュータのホスト名が含まれていることを確認します。
11. .LocalRegFile に正しいメイン ロケーション サーバが含まれていることを確認します。
12. CLI ユーティリティを使用するため、シェルに CLISESSID を設定します。

    Windows

    ```
    set CLISESSID=<NUMBER>
    ```

Solaris および Linux

```
export CLISESSID=<NUMBER>
```

NUMBER

シェルの一意の番号です。

13. Notifier/.alarmrc に SetScript、ClearScript および UpdateScript への正しいパスが設定されていることを確認します。 これらのスクリプトは、tools_bundle ファイルを抽出した Notifier ディレクトリに格納されています。
14. processd を再起動します。

    このコンピュータ上でユーティリティを実行できるようになりました。

CA Spectrum を実行していないコンピュータに CA Spectrum ユーティリティを移動することもできます。

次の手順に従ってください:

1. CA Spectrum ユーティリティをパックします。
2. CA Spectrum ユーティリティを解凍するコンピュータ上で、CA Spectrum ユーティリティとそのサポート ファイルを解凍するためのディレクトリを作成します。 たとえば、以下のディレクトリを作成します。

   Windows

   ```
   c:¥win32app¥spectrum
   ```

   Solaris および Linux

   ```
   /usr/spectrum
   ```

   注: 作業ディレクトリ（chdir）をこれらのディレクトリに変更してください。
3. バイナリ モードを使用して、FTP により、実行可能ファイルを現在のディレクトリに送信します。 ファイルの名前は、linux_tools_bundle（Linux）、solaris_tools_bundle（Solaris）、nt_tools_bundle.exe（Windows）のいずれかです。
4. DOS、Bash、または他の UNIX シェルから tools_bundle ファイルを実行します。

   CA Spectrum ユーティリティとそのサポート ファイルが、適切なディレクトリ構造に解凍されます。

5. PATH 変数が以下のとおりであることを確認します。

   Windows

   <*$SPECROOT*>/lib が PATH 変数に含まれていることを確認します。

   Solaris および Linux

   - /opt/SPECTRUM ディレクトリを作成します。
   - <*$SPECROOT*>/lib から /opt/SPECTRUM/lib へのリンクを作成します。
   - <*$SPECROOT*>/bin から /opt/SPECTRUM/bin へのリンクを作成します。

6. 以下のように、<*$SPECROOT*> 環境変数と <*$SPECPATH*> 環境変数を tools_bundle ファイルを解凍したディレクトリへのパスに設定します。

   - Windows の場合は、以下の変数形式の値を使用してシステム環境変数 <*$SPECROOT*> および <*$SPECPATH*> を作成します。

     ```
     driveletter:/PATH_TO_SPECTRUM
     driveletter:¥PATH_TO_SPECTRUM
     ```

   - Solaris の場合は、/opt/SPECTRUM/spectrum60.env を作成して、以下の行を追加します。

     ```
     SPECROOT=PATH_TO_SPECTRUM
     SPECPATH=PATH_TO_SPECTRUM
     ```

   - Linux の場合は、/opt/SPECTRUM/spectrum80.env を作成して、以下の行を追加します。

     ```
     SPECROOT=PATH_TO_SPECTRUM
     SPECPATH=PATH_TO_SPECTRUM
     ```

7. AlarmNotifier とイベント ファイルおよび pcause ファイルを移動した場合、<*$SPECROOT*>/SG-Support/CsResource/preferences/*.prf 内のパスにイベント ファイルと pcause ファイルの正しい場所が指定されていることを確認します。 また、ui= オプションと lhandle= オプションを確認します。

8. Solaris および Linux プラットフォームの場合は、/opt/SPECTRUM ディレクトリにある spectrum*.env ファイル内で BES_LIC_DIR を PATH_TO_spectrum/bin/VBNS/license として定義します。

9. .hostrc ファイルにローカル コンピュータのホスト名、およびユーティリティの転送元のコンピュータ名が含まれることを確認します。

10. .LocalRegFile に正しいメイン ロケーション サーバが含まれていることを確認します。

11. vnmsh/.vnmshrc に正しいメイン SpectroSERVER 名が含まれていることを確認します。
12. 以下のように、processd サービスをインストールします。

    Windows

    - SRAdmin がインストールされていない場合は、以下のようにインストールします。

      ```
      shell> cd %SPECROOT%¥Install-Tools¥sdic¥nt
      shell> .¥sradmin --install
      shell> .¥sradmin --start
      ```

    - 以下のように、processd サービスをインストールします。

      ```
      shell> cd %SPECROOT%¥lib¥SDPM
      shell> .¥processd.exe --install --username USERNAME --password PASSWORD
      shell> .¥processd.exe --start
      ```

      注: processd が起動しない場合は、コンピュータを再起動してください。

    Solaris および Linux

    - <*$SPECROOT*>/lib/SDPM/processd_init.sh を /etc/init.d/processd にコピーします。
    - <*$SPECROOT*>/lib/SDPM/processd.pl を /etc/init.d にコピーします。
    - /etc/init.d/processd から /etc/rc2.d/S99processd へのリンクを作成します。
    - /etc/init.d/processd start を使用して、processd を起動します。
13. メイン ロケーション サーバの .hostrc ファイルに tools_bundle ファイルを解凍するコンピュータのホスト名が含まれていることを確認します。
14. CLI ユーティリティを使用するため、シェルに CLISESSID を設定します。

    Windows

    ```
    set CLISESSID=<NUMBER>
    ```

    Solaris および Linux

    ```
    export CLISESSID=<NUMBER>
    ```

    NUMBER

    シェルの一意の番号です。

15. Notifier/.alarmrc に SetScript、ClearScript および UpdateScript への正しいパスが設定されていることを確認します。 これらのスクリプトは、tools_bundle ファイルを抽出した Notifier ディレクトリに格納されています。
16. Windows プラットフォームで AlarmNotifier スクリプトを使用する場合は、http://www.cygwin.com から Cygwin をインストールします。 PATH 変数に Cygwin の bin ディレクトリが含まれていることを確認してください。

    このコンピュータ上でユーティリティを実行できるようになりました。

詳細情報:

CA Spectrum ユーティリティのパック (P. 28)

第 2 章: 分散 SpectroSERVER 環境の設定

このセクションには、以下のトピックが含まれています。

名前解決の要件

SpectroSERVER システムまたは OneClick Web サーバ システムは、SpectroSERVER のホスト名を IP アドレスに解決できる必要があります。

名前解決にはホスト ファイルを使用することをお勧めします。 この実習では、ネットワーク停止が SpectroSERVER 名前解決に影響しないことを確認します。

CA Spectrum と複数のインターフェース

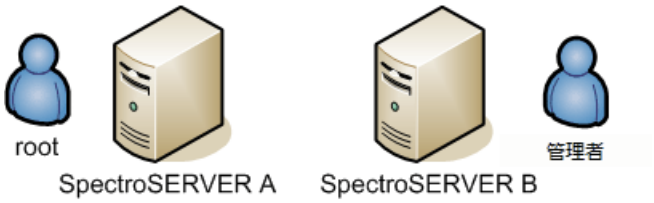
すべての CA Spectrum サーバは利用可能なすべてのインターフェースにバインドしてリスンし、完全修飾でないホスト名を使って自分をアドバタイズしています。 そのため、管理トポロジを設定する場合に、柔軟に対応できます。 接続を確立するため、CA Spectrum コンポーネント サーバは、接続が成功するまで、オペレーティング システムによって決定された順にすべてのインターフェースを試行します。

SpectroSERVER 間の通信

同じユーザ アカウントを使って、分散環境に SpectroSERVER をインストールして実行します。 このため、それ以上のユーザ モデル設定は必要ではありません。 別のユーザとしてインストールされ実行されている SpectroSERVER 間では通信できません。

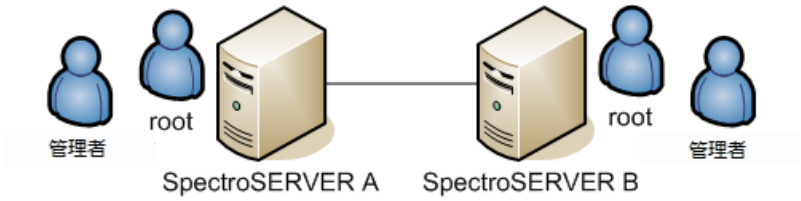
例: SpectroSERVER A と B が通信できない

この例では、SpectroSERVER A は「root」 として動作しています。 SpectroSERVER B は Administrator として実行されています。 要求がサーバ セキュリティを通り抜けることができないため、接続を確立できません。



例: SpectroSERVER と SpectroSERVER を接続するための設定

この例では、ユーザは SpectroSERVER B では「root」 としてログインし、SpectroSERVER A では「Administrator」としてログインしたことを示しています。 この設定では、2 台の SpectroSERVER が通信できます。



ポート競合の解決

別のアプリケーションとのポート競合を回避するため、CA Spectrum のプロセスとサービスが使用するデフォルト ポート番号を変更できます。

以下のテーブルでは、CA Spectrum のプロセスとサービス用のデフォルト ポートについて説明します。

注: 「リモート接続には使用しない」と記載されている場合は、この表のポートについて言及したものです。 一般的に、CA Spectrum がリモート接続に使用しないポートでは、ファイアウォール設定の問題はありません。

ポート（10 進数）	ポート（16 進数）	CA Spectrum コンポーネント	プロトコル	注
80	0x0050	OneClick/SRM	HTTP	なし
162	0x00A2	SpectroSERVER	UDP	SNMP トラップのリスンに使用するポート
3306	0x0CEA	SRM データ用 mysql ArchMgr（アーカイブ マネージャ）	TCP/ODBC/JDBC	Windows から Linux または Solaris に SRM データベースを移行する場合にのみリモート接続に使用
3307	0x0CEB	Solaris/Linux の BOXI 用 mysql	TCP	リモート接続には使用しない
6844	0x1ABC	SDC	TCP	なし
6400	0x1900	BOXI CMS	TCP	リモート接続には使用しない
7777	0x1E61	CLI vnmshd	TCP	「CLI デーモン（vnmshd） (P. 75)」を参照
14001	0x36B1	OneClick サーバ	TCP/CORBA	「OneClick Web サーバ ホスト (P. 69)」を参照
14002	0x36B2	SpectroSERVER	TCP/CORBA	「SpectroSERVER (P. 73)」を参照

ポート（10 進数）	ポート（16 進数）	CA Spectrum コンポーネント	プロトコル	注
14003	0x36B3	ArchMgr	TCP/CORBA	「アーカイブ マネージャ (P. 73)」を参照
14004	0x36B4	LocServ	TCP/CORBA	「ロケーション サーバ (P. 74)」を参照
14006	0x36B6	ネーミング サービス	TCP/CORBA	なし
31415	0x7AB7	SpectroSERVER による Telnet		なし
46517	0xB5B5	sradmin	TCP	リモート管理デーモン。 **注**: sradmin プロセスの詳細については、「インストール ガイド」を参照
47870	0xBAFE	ArchMgr	TCP/SSAPI	「アーカイブ マネージャ (P. 73)」を参照
48879	0xBEEF	SpectroSERVER	TCP/SSAPI	「SpectroSERVER (P. 73)」を参照
51966	0xCAFE	rcpd	TCP	「リモート コピー プロセス デーモン（rcpd） (P. 75)」を参照
56063	0xDAFF	LocServ	TCP	「ロケーション サーバ (P. 74)」を参照
61904	0xF1D0	processd	TCP	このポートは設定できません。
64222	0xFADE	TL1d	TCP/EPI	TL1 ゲートウェイ エージェント
65259	0xFEEB	processd	TCP	このポートは設定できません。

DSS 環境要件

複数の SpectroSERVER と複数のランドスケープのモデリングを有効にして、サーバを表すには、分散環境は以下の条件を満たす必要があります。

- ランドスケープはそれぞれ、モデル タイプとその関係の、同じモデリング カタログを含んでいる必要があります。 この複製により CA Spectrum のインテリジェンスに一貫性が保たれます。
- ランドスケープはそれぞれ、同じユーザ モデルを含む必要があります。これは［OneClick ユーザ］タブで確認できます。
- 異なるランドスケープでモデリングされたサブネット間の接続数は、最小限に抑えてください。
- 複数のランドスケープでモデリングされる同一デバイス数は制限してください。
- モデリングされたランドスケープごとに、一意のランドスケープ ハンドルを割り当ててください。その後、CA Spectrum は、DSS 環境内の他のランドスケープとそのランドスケープを識別することができます。

詳細情報:

ランドスケープ ハンドルの割り当て (P. 44)

ロケーション サーバ

SpectroSERVER をインストールすると、*ロケーション サーバ*も自動的にインストールされます。 このサーバは、ネットワーク上の他の CA Spectrum サービスを識別し見つけます。 CA Spectrum プロセスは、これらのサービスがどこで実行されているかを特定するため、ロケーション サーバを使用します。 processd がロケーション サーバを起動および停止します。

分散環境では、CA Spectrum はロケーション サーバを使用して SpectroSERVER ランドスケープ マップを維持し、クライアント アプリケーションに接続サービスを提供します。 CA Spectrum インストール中に、ランドスケープ マップ（または CA Spectrum ドメイン）内のロケーション サーバの 1 つを、*メイン ロケーション サーバ（MLS）*として指定します。

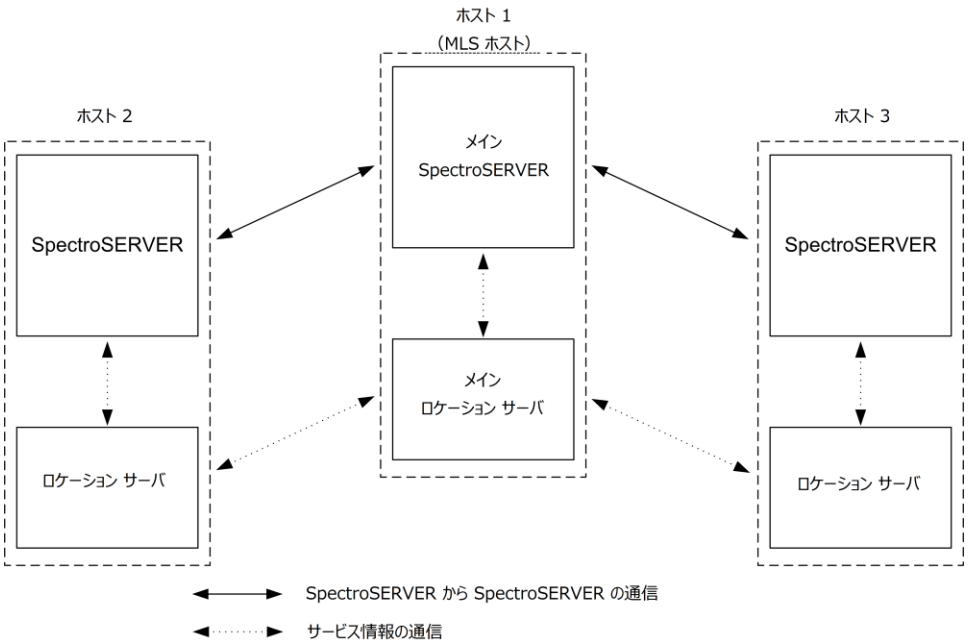
メイン ロケーション サーバを、信頼性が非常に高いコンピュータにインストールします。 このサーバは、ロケーション サーバ間にサービス通知を継承し、ロケーション サーバ間にサービスの場所を通信します。

以下の図では、Host 1 上のロケーション サーバが、この環境用のメイン ロケーション サーバとして設定されています。メイン ロケーション サーバが SpectroSERVER と同じホスト上で存在するため、その SpectroSERVER はメイン SpectroSERVER であると見なされます。

サーバ対サーバ通信はすべて、メイン ロケーション サーバ ホストを介してルーティングされます。 2 つの非 MLS SpectroSERVER は、メイン SpectroSERVER を介してのみ通信できます。

図内のホスト 2 上の SpectroSERVER は、ホスト 3 上の SpectroSERVER でモデリングされたリソースを使って、サービスを分散します。 リモート リソースに関する通信はすべて、メイン ロケーション サーバ ホストを介してルーティングされます。ホスト 2 およびホスト 3 上のサーバは直接通信しません。

重要: この図は、1 つの MLS と、複数の子 SpectroSERVER から構成される、2 層セットアップを反映しています。このような設定が推奨標準設定です。階層関係（多層展開）を持つ MLS から構成される、より複雑な設定も可能ですが、そのような設定は推奨されません。 多層展開は複雑さが増しますが、提供される機能が増えるわけではありません。



ロケーション サーバの対話方法

すべての CA Spectrum インストールには以下の 2 ファイルが含まれます。

- .LocalRegFile により、CA Spectrum プロセスはロケーション サーバを見つけることができます。
- LS/.locrc は、ローカルにインストールされたロケーション サーバ用の設定ファイルです。

これらのファイルにより、ロケーション サーバの階層と、各ロケーション サーバとメイン ロケーション サーバ間のインタラクションが管理されます。

分散 SpectroSERVER 展開では、ロケーション サーバは以下のように対話します。

1. SSAPI アプリケーションおよびサーバが .LocalRegFile を読み取り、使用するロケーション サーバを決定します。
2. 各ローカル ロケーション サーバは /LS/.locrc ファイルを読み取り、どのサーバがメイン ロケーション サーバか特定します。
3. ローカル ロケーション サーバはネットワーク サービスを検索するため、メイン ロケーション サーバに接続します。 メイン ロケーション サーバが利用不可能になると、メイン ロケーション サーバが使用可能になるまで、他のロケーション サーバはメモリ内にサービス情報を保持します。

メイン ロケーション サーバ接続

SpectroSERVER はそれぞれ、メイン ロケーション サーバとして指定されたサーバ上で実行されている SpectroSERVER に接続する必要があります。 この指定は <*$SPECROOT*>/LS/.locrc ファイルで指定します。

メイン ロケーション サーバへの接続では、以下の設定を必要とします。

- SpectroSERVER はそれぞれ、他方の SpectroSERVER ホストを接続させる必要があります。接続は、各サーバ上に .hostrc ファイル内の適切なエントリを必要とします。
- SpectroSERVER はそれぞれ、他方の SpectroSERVER を実行しているユーザ アカウント用に CA Spectrum ユーザ モデルを必要とします。

注: 主なランドスケープ（または .locrc ファイル内の MAIN_LOCATION_HOST_NAME の値）は、CA Spectrum でモデリングされます。 すでに他の場所でモデリングされていない限り、各 SpectroSERVER の VNM トポロジにはこのランドスケープが含まれます。 メイン ランドスケープへの接続が失われた場合、このモデルに関するアラームが生成されます。

新しいメイン ロケーション サーバの指定

メイン ロケーション サーバとして指定されるコンピュータを変更できます。

SpectroSERVER で、［ロケーション サーバ設定］ダイアログ ボックスまたは .locrc ファイルを使って、新しいメイン ロケーション サーバを指定します。

OneClick Web サーバでは、［OneClick 管理］ページの［CA Spectrum 構成］ページでメイン ロケーション サーバ情報を設定します。 このアクションを実行すると、OneClick Web の接続先であるメイン ロケーション サーバの名前が更新されます。

注: OneClick Web サーバを更新せずに SpectroSERVER 上のメイン ロケーション サーバを変更すると、CA Spectrum が正しく機能しません。 詳細については、「管理者ガイド」を参照してください。

例: SpectroSERVER 上での新しいメイン ロケーション サーバの指定

分散ネットワークは、Computer 1 ～ 4 で構成されていて、Computer 1 がメイン ロケーション サーバとして指定されています。 Computer 4 をメイン ロケーション サーバにするには、まず新しいメイン ロケーション サーバとして Computer 4 を再設定します。 次に、メイン ロケーション サーバとして Computer 4 をポイントするように、Computer 1、2、および 3 を再設定します。

次の手順に従ってください:

1. SpectroSERVER のコントロール パネルを表示します。
2. ［設定］メニューから［ロケーション サーバ］を選択します。

   ［ロケーション サーバの設定］ダイアログ ボックスが表示されます。
3. Computer 4 をポイントするように、ロケーション サーバの特性領域にある［メイン LS ホスト］フィールドを変更します。

注: <*$SPECROOT*>/LS ディレクトリ内の .locrc ファイルを編集して、メイン ロケーション サーバを変更することもできます。.locrc ファイル内の以下のエントリによって、メイン ロケーション サーバを識別します。

```
MAIN_LOCATION_HOST_NAME=Computer 4
```

詳細情報:

フォールト トレランスの確立 (P. 87)

ランドスケープ マップの整合性の維持

CA Spectrum ユーザは一般的に、実稼働環境とテスト環境を個別にメンテナンスします。 テスト環境の SpectroSERVER がメイン ロケーション サーバとして実稼働環境の SpectroSERVER に接続しないようにすることで、ランドスケープ マップの整合性を保護します。逆もまた当てはまります。

ランドスケープ ハンドル

分散展開内の SpectroSERVER は、*ランドスケープ ハンドル*によって各ランドスケープを識別します。 ランドスケープはそれぞれ一意のランドスケープ ハンドルが必要です。ランドスケープ ハンドルは、4 ～ 16,376 の範囲にある、4 で割り切れる数です（16 進数の形式では、ハンドルは、下位の 20 ビットを 0 に設定した、0x100000 ～ 0xffe00000 の範囲の数です）。エンコードされたランドスケープ ハンドルが、そのランドスケープに関連付けられたすべてのビューの上部に表示されます。

注: ランドスケープ ハンドルには、連続する番号スキームを使用することを推奨します。 ビルド番号またはサブネット IP アドレスの一部など、ランドスケープの重要な機能をランドスケープ ハンドルに関連付けることができます。 ただし、ユーザのエントリはランドスケープ ハンドルの 12 の最上位ビットにエンコードされます。 そのため、適切なランドスケープ機能と無関係に見える場合があります。

ランドスケープ ハンドルの割り当て

CA Spectrum のインストール時に、SpectroSERVER の［検証］ダイアログ ボックス、または lh_set ユーティリティを使用して、ランドスケープ ハンドルを割り当てることができます。

重要: SpectroSERVER を初めて実行する前に、lh_set ユーティリティを実行してください。そうでない場合、CA Spectrum はデフォルトのランドスケープ ハンドルを割り当てますが、CA Spectrum が割り当てるランドスケープ ハンドルは毎回同じです。 その結果、複数のランドスケープを設定した場合、重複したランドスケープ ハンドルが作成される可能性があります。このようなランドスケープには、同じアプリケーションから同時にアクセスすることはできません。

次の手順に従ってください:

1. SS ディレクトリに移動します。
2. 以下のコマンドを入力します。

   ```
   ../SS-Tools/lh_set <landscape handle>
   ```

   新しいランドスケープ ハンドルは、10 進法または 16 進法で指定できます。10 進法を使用する場合は、lh_set ユーティリティが入力を 16 進のランドスケープ ハンドルに変換します。

注: 詳細については、「インストール ガイド」を参照してください。

詳細情報:

DSS 環境要件 (P. 39)

ランドスケープ ハンドルの変更

SpectroSERVER を起動した場合、SpectroSERVER データベースのランドスケープ ハンドルの変更プロセスは、より多くの手順を必要とします。 ランドスケープ ハンドルは、SpectroSERVER データベース内の各モデルのモデル ハンドルに含まれています。 このため、ランドスケープ ハンドルを変更するには、すべてのモデル ハンドルを古いランドスケープ ハンドルから新しいランドスケープ ハンドルに変換する必要があります。

起動時に自動作成されたすべてのモデルのランドスケープ ハンドルを変更し、複数のリソース ファイル内のランドスケープ ハンドルを更新します。

次の手順に従ってください:

1. モデリング ゲートウェイ ツールを使用して、ランドスケープ ハンドルを変更する SpectroSERVER からモデルをエクスポートします。

2. SpectroSERVER をシャットダウンします。
3. SSdbload ユーティリティを使用して、データベースを初期化します。

   注: 詳細については、「データベース管理ガイド」を参照してください。
4. lh_set ユーティリティを使用して、新しいランドスケープ ハンドルを割り当てます。
5. SpectroSERVER を起動します。
6. モデリング ゲートウェイ ツールを使用して、モデルを SpectroSERVER にインポートします。

注: 詳細については、「Modeling Gateway Toolkit ガイド」を参照してください。

プロセス デーモン（processd）

CA Spectrum は、プロセス デーモン（processd）という、プロセスの開始および追跡デーモンを使用して、DSS 環境の複数のサーバ上のプロセスを制御できます。CA Spectrum コントロール パネルなどのアプリケーションが要求を実行すると、このデーモンはプロセスを開始します。 またインストール チケットを使用して、システム起動時にプロセスを自動的に開始するように processd を設定することもできます。 インストール チケットは、予期せずに停止した重要なプロセスを自動的に再起動することもできます。

CA Spectrum インストール プログラムは、Solaris システム（root として実行する必要があります）、および Windows システム（LocalSystem アカウントとして実行されます）上で自動起動するように processd を設定します。

processd を使ったプロセスの開始と監視は、アプリケーションが該当するアクションを要求した場合に限り、発生します。 通常、processd はバックグラウンドで動作します。

プロセスが起動しない場合、または正しく機能していない場合、processd はエラー メッセージを *<$SPECROOT>*¥lib¥SDPM¥processd_log ファイルに書き込みます。 エラー メッセージには、問題を識別するための情報が含まれています。

再起動時、processd は processd_log.bak ファイルを作成して古いエラー メッセージを保存し、processd_log ファイルに新しいエラー メッセージを追加します。

重要: このセクションに記載されている手順を実行する前に、CA Spectrum、分散ネットワーク、およびネットワーク設定について熟知する必要があります。

詳細情報:

チケット ファイルのインストール (P. 50)

Windows 環境と Solaris 環境の processd の相違

Windows と Solaris プラットフォームではネイティブ セキュリティ アーキテクチャが異なっているため、processd デーモンの動作もわずかに異なります。

- Windows 環境でリモート接続からプロセスの開始を要求した場合、［Windows サービスの構成］ダイアログ ボックスで指定されたユーザとして、プロセスが開始されます。Solaris で、要求を実行しているユーザとしてプロセスを開始します。
- ユーザ ログアウト後（またはユーザのログイン前）に実行を継続する必要があるサーバ プロセスについては、Windows 環境では［SERVERPROCESS］フィールドが使われます。ログアウト時に Windows がこれらのプロセスをシャットダウンするのを防ぐために、これらのプロセスは、［Windows サービスの構成］ダイアログ ボックスで指定したユーザとして開始されます。 Solaris では、これらのプロセスは、インストール チケットで指定された TICKETUSER として開始されます。

詳細情報:

チケット ファイルのインストール (P. 50)

processd の Windows パスワードの変更

Windows でのインストール中、［Windows サービスの構成］ダイアログ ボックスにユーザ名とパスワードを入力するように求められます。processd デーモンは、指定されたユーザとして CA Spectrum プロセスを開始するために、このユーザ名とパスワードを使用します。 ユーザがログインしていない場合、セキュリティ情報を提供することでこれらのプロセスを実行できます。

注: 詳細については、「インストール ガイド」を参照してください。

［Windows サービスの構成］ダイアログ ボックスで指定したパスワードを変更したり、有効期限切れになった場合、新しいパスワードで processd を更新する必要があります。

重要: パスワードに感嘆符 (!) などの特殊機能が含まれている場合、バックスラッシュ (¥) を使って、その文字をエスケープ処理してください。または、コマンド プロンプトからパスワードを変更することもできます。

次の手順に従ってください:

1. ローカル Administrators グループのメンバとしてログインします。

   processd のユーザ名とパスワードを変更するには、ユーザは、CA Spectrum ユーザ グループのメンバおよび管理である必要があります。

2. コマンド プロンプトを開きます。
3. /lib/SDPM ディレクトリに移動します。

4. 以下のコマンドを入力します。

   ```
   processd --install --username user --password newpassword
   ```

   user

   ユーザ名を指定します。

   newpassword

   このユーザ名の新しいパスワードを指定します。

   ユーザ名またはパスワードにシェルが誤って解釈する可能性がある特殊文字が含まれる場合、ユーザ名またはパスワードを引用符で囲みます。 たとえば、ユーザ名にはバックスラッシュ（¥）区切り記号と共に、ドメイン名とユーザが含まれます。 パスワードにはアスタリスク（*）が含まれています。 これらの文字は両方とも特殊文字です。 そのため、以下の例のように、ユーザ名とパスワードは引用符で囲みます。

   ```
   processd --install --username “DOMAIN¥JSmith” --password “283EJ*”
   ```

   注: Windows ユーザ名のパスワードをが変更したにもかかわらず、processd のサービス パスワードを更新しない場合、processd は開始しません。 Windows のイベント ログには、以下のイベントが生成されます。

   - 現在のパスワードでのログオン試行が以下のエラーにより失敗しました：
     ログオン失敗： ユーザ名が不明またはパスワードが間違っています。
   - CA Spectrum プロセス デーモン サービスの起動が以下のエラーにより失敗しました；
     ログオン エラーのためサービスは開始しませんでした。

チケット ファイルのインストール

システム起動時にプロセスを開始するように processd を設定できます。また、プロセス終了時に、プロセスを再起動するように選択することもできます。 *インストール チケット*というファイルは、この機能をサポートします。

インストール チケットは以下の 2 つのディレクトリ内のファイルを使用します。

- <*Spectrum Installation Directory*>/lib/SDPM/partslist
- <*Spectrum Installation Directory*>/lib/SDPM/runtime

partslist ディレクトリには、個々のインストール チケット ファイルが格納されます。

runtime ディレクトリには、*<PID>*.rt の形式で、エンコードされたファイルが格納されます。*<PID>* は、動作しているプロセスのプロセス ID です。

注: SDPM は CA Spectrum 分散プロセス マネージャ（CA Spectrum Distributed Process Manager）を意味します。

partslist ディレクトリにインストール チケットを追加できます。新しいインストール チケットは、特定のフォーマット ルールに従う必要があります。partslist ディレクトリの新しいまたは変更したインストール チケットを識別するには、processd を再起動します。 これらのファイルは processd の起動時に読み取られ、将来使用できるようにキャッシュ メモリに保存されます。

インストール チケット ファイルは、設定情報を格納しているプロセスを参照するという命名規則に従います。 インストール チケット ファイル名は、*<PARTNAME>*.jdb という形式になります。 *<PARTNAME>* 変数は、processd が制御するプロセスを識別する内部キーです。

インストール チケットには以下の定義フィールドが使用されます。 その形式は *<fieldname>;<value>* です。ここで *<fieldname>* は以下に示す名前のいずれかです。 *<value>* は対応するフィールド名の定義を説明する文字列です。 引用符はサポートされません。

PARTNAME

スペースで区切らないマルチ文字文字列を使って、特定のプロセス/アプリケーションを識別します。 このアプリケーションのインストール チケットのファイル名は、*<PARTNAME>*.idb という形式です。 *<PARTNAME>* はプロセス名です。

APPNAME

アプリケーションの名前をマルチ文字文字列として定義します。

WORKPATH

アプリケーションをどこで実行するか指定します。 以下に説明する ARGV フィールドの一部として使用する前に、このフィールドに値を設定する必要があります。

LOGNAMEPATH

アプリケーションからの出力用ログ ファイルを指定します。 このフィールドは、*<$SPECROOT>* または *<$WORKPATH>* で始める必要があります。 スペースは使用できません。

ADMINPRIVS

Purism で提供するインストール チケットでのみ使用するように予約されています。ユーザが作成するインストール チケットではコメントアウトします。

AUTORESTART

プロセスの終了時に、プロセスを自動的に再起動するかどうかを示します。インストール チケットにこのフィールドが含まれていない場合、自動再起動はデフォルトで無効になります。 Y、y、N、n の値を使用できます。

AUTOBOOTSTART

processd の起動時に、プロセスを起動するかどうかを示します。 インストール チケットにこのフィールドが含まれていない場合、この機能はデフォルトで無効になります。 Y、y、N、n の値を使用できます。

STATEBASED

プロセスの開始時に、プロセスが複数の状態を持つことがあるかどうかを示します。通常、プロセスは通信の準備ができると、「準備完了」チケットを送信します。 このフィールドは、Purism で提供するインストール チケットのみで使用するように予約されています。インストール チケットにこのフィールドが含まれていない場合、デフォルトで無効になります。 Y、y、N、n の値を使用できます。

NUMPROCS

1 つのプラットフォーム上で実行できる、プロセスのインスタンス数を指定します。 数値を使用できます。

RETRYTIMEOUT

障害発生後、processd がアプリケーションの再起動を試みる秒数を指定します。

TICKETUSER

AUTOBOOTSTART フィールドや AUTORESTART フィールドが設定されている場合、プロセスの実行を許可されたユーザのユーザ名を定義します。インストール チケットにこれらのフィールドが含まれている場合にのみ、このフィールドは必要です。 Windows ではすべてのプロセスが processd インストール ユーザとして実行されるのため、このフィールドは適用できません。

RETRYMAX

指定された RETRYTIMEOUT 期間内に、processd がアプリケーションの再起動を試行する回数を指定します。

STARTPRIORITY

他のプロセスに対する、プロセスの相対的な起動優先度を示します。使用する値は、以下のとおりです。

- スタンドアロン プロセスの場合は 10
- スタンドアロン プロセスに依存するプロセスの場合は 20
- SpectroSERVER に依存するプロセスの場合は 30

SERVERPROCESS

ユーザのログアウト後もプロセスの実行を継続するかどうかを processd に示します。このフィールドが有効な場合、プロセスは必ず、processd サービスの実行を許可されたユーザとして開始されます。 このフィールドは、Windows 環境でのみ適用されます。 以下のプロセスのデフォルト値は yes です。

- アーカイブ マネージャ（ARCHMGR.idb）
- ロケーション サーバ（LOCSERV.idb）
- SpectroSERVER（SS.idb）

SERVICE

チケットが Windows サービスを表すかどうかを示します。processd は Service Control Manager から Windows サービスを管理できます。

ENV

アプリケーション環境に 1 つ以上の変数を追加します。 値と、行を終了するセミコロンの間で、<CSPATHSEP> マクロを使用すると、複数の値が個別の行にリスト表示されます。

ARGV

プロセスの引数リストを定義します。このリストには実行可能パスと、任意の数の引数（スペースは使用可能）が含まれます。

詳細情報:

プロセス デーモン（processd） (P. 46)
Windows 環境と Solaris 環境の processd の相違 (P. 47)

例: インストール チケット

インストール チケットは、以下の構文規則を適用します。

- パウンド記号（#）で始まる行は、コメントとして処理されます。
- すべてのフィールド名と、各フィールド名の後のすべての値の後ろにセミコロン（;）を記述する必要があります。
- 2 番目のセミコロンの後ろはすべて無視されます。

- <CSEXE>、<CSBAT>、<CSCMD>、および <CSPATHSEP> の 4 つのマクロが使用できます。プラットフォームに基づいて、適切な定義に置換されます。構文解析の競合を避けるため、実際のパス セパレータの代わりに <CSPATHSEP> を使用してください。

以下の例に、有効なインストール チケットを示します。

```
# Processd Install Ticket for SpectroSERVER Daemon.
PARTNAME;SS;
APPNAME;SpectroSERVER Daemon;
WORKPATH;$SPECROOT/SS;
LOGNAMEPATH;$WORKPATH/VNM.OUT;
ADMINPRIVS;y;
#AUTORESTART;N;
#AUTOBOOTSTART;N;
STATEBASED;y;
NUMPROCS;3; // unlimited
RETRYTIMEOUT;0; // seconds
#TICKETUSER;$USER;
RETRYMAX;0;     // retries
STARTPRIORITY;20;
SERVERPROCESS;Y;
#ENV;<var>=<value>;
ARGV;$SPECROOT/SS/SpectroSERVER<CSEXE>;
```

以下の例では、別の有効なインストール チケットを示します。

```
# Processd Install Ticket for Visibroker Naming Service
PARTNAME;NAMINGSERVICE;
APPNAME;Visibroker Naming Service;
WORKPATH;$SPECROOT/bin/VBNS;
LOGNAMEPATH;$WORKPATH/NAMINGSERVICE.OUT;
ADMINPRIVS;y;
AUTORESTART;y;
AUTOBOOTSTART;y;
#STATEBASED;N;
NUMPROCS;1; // one per host
RETRYTIMEOUT;600; // 10 minutes
#TICKETUSER;$USER;
RETRYMAX;5;    // 5 retries
STARTPRIORITY;10;
SERVERPROCESS;Y;
#ENV;<var>=<value>;
ENV;CLASSPATH=$SPECROOT/lib/vbjorb.jar<CSPATHSEP>;
ENV;CLASSPATH=$SPECROOT/lib/vbsec.jar<CSPATHSEP>;
ENV;CLASSPATH=$SPECROOT/lib/lm.jar<CSPATHSEP>;
ENV;CLASSPATH=$SPECROOT/lib/sanct6.jar<CSPATHSEP>;
ARGV;
$SPECROOT/bin/JavaApps/bin/nameserv<CSEXE>
-DORBpropStorage=
$SPECROOT/.corbarc
-Dvbroker.orb.admDir=
$SPECROOT/bin/VBNS
-Dborland.enterprise.licenseDir=
$SPECROOT/bin/VBNS/license
-Dborland.enterprise.licenseDefaultDir=
$SPECROOT/bin/VBNS/license
-Djava.endorsed.dirs=
$SPECROOT/lib/endorsed
-Dorg.omg.CORBA.ORBClass=
com.inprise.vbroker.orb.ORB
-Dorg.omg.CORBA.ORBSingletonClass=
com.inprise.vbroker.orb.ORB
com.inprise.vbroker.naming.ExtFactory;
```

新しいインストール チケット プロセスの起動

インストール チケット ファイルを追加した場合、チケットで指定されたプロセスを開始する再起動オプションを使用できます。 再起動オプションを使用すると、processd と processd が監視しているすべてのプロセスを終了して再起動する必要がなくなります。

Solaris では、以下の手順に従ってください。

1. root としてログインします。
2. /lib/SDPM ディレクトリに移動します。
3. 以下のコマンドを入力します。

   ```
   processd.pl restart
   ```

   プロセスが起動します。

Windows では、以下の手順に従ってください。

1. CA Spectrum ユーザ グループのメンバとしてログインします。
2. コマンド プロンプトを開きます。
3. /lib/SDPM ディレクトリに移動します。
4. 以下のコマンドを入力します。

   ```
   perl processd.pl restart
   ```

   プロセスが起動します。

processd の停止と再起動

lib/SDPM/runtime ディレクトリが破損した疑いがある場合は、processd を停止して再起動する必要があります。

Solaris 上での 次の手順に従ってください:

1. root としてログインします。
2. /lib/SDPM ディレクトリに移動します。
3. 以下のコマンドを入力します。

   ```
   processd.pl stop
   ```

4. CA Spectrum プロセスがすべてシャット ダウンしていることを確認します（そうでない場合、SpectroSERVER に関する問題が発生する可能性があります）。

5. /lib/SDPM/runtime ディレクトリのすべてのエントリを削除します。
6. 以下のコマンドを入力して、processd を再起動します。

   ```
   processd.pl start
   ```

Windows での 次の手順に従ってください:

1. CA Spectrum ユーザ グループのメンバとしてログインします。
2. コマンド プロンプトを開きます。
3. /lib/SDPM ディレクトリに移動します。
4. 以下のコマンドを入力します。

   ```
   perl processd.pl stop (または start)
   ```

注: Windows では、processd.pl *<start/stop>* コマンドを実行しても、NT サービスとして動作している CA Spectrum プロセス（MySQL や VisiBroker など）が停止および起動します。

詳細情報:

Solaris 環境での SpectroSERVER のシャットダウン (P. 62)

インストール中の userconf プロセスの動作方法

userconf プロセスは processd をバックグラウンドで実行します。 Userconf では、インストール中およびインストール後にユーザが CA Spectrum にログインすると、CA Spectrum のユーザ固有の設定を実行できます。

userconf プロセスは、CA Spectrum インストール中に以下のタスクを実行します。

1. userconf -install %SPECROOT% を実行して、以下の変更を行います。
   - SPECROOT および SPECPATH をシステム環境に追加する。
   - $SPECPATH¥lib を $PATH システム変数に追加する。
   - インストール ディレクトリを変更した場合は、古い $SPECPATH¥lib エントリをパスから削除する。
   - %SPECPATH%¥lib をパスから削除する（存在する場合）。
   - （引数を指定せずに）ユーザがログインするたびに、userconf が実行されるように、レジストリを変更します。

2. userconf -start を実行して processd を起動します。 -start フラグを使用すると、ユーザが CA Spectrum ユーザ グループのメンバであることを確認せずに、processd が起動されます。

   注: インストール プログラムは、現在のユーザを CA Spectrum ユーザ グループに追加します。 ただし、ユーザがログアウトして再度ログインするまで変更は有効になりません。-start フラグを追加すると、ユーザが CA Spectrum ユーザ グループのメンバであることを確認せずに、processd が起動します。 その後、ユーザのログアウトと再ログインを求められることがなく、インストールが継続します。

3. 初めて Exceed をインストールする（CA Spectrum のインストール後）場合は、userconf -restart を実行して processd を停止してから再起動します。 これによって、processd は、Exceed インストールの一部である PATH 環境の変更を更新することができます。

ユーザのログイン時の userconf の動作方法

インストールにより、レジストリ Run キーに userconf プロセスが追加されます。 ユーザのログイン時、プロセスが自動的に開始され、ユーザが CA Spectrum ユーザ グループのメンバかどうかを確認します。 userconf プロセスは、以下のルールを適用します。

- ユーザが CA Spectrum ユーザ グループのメンバである場合、userconf は cygwin と Exceed を確認し、processd を起動する前に必要に応じて再設定を行います。 Microsoft VC++ がインストールされている場合、CA Spectrum で使用できるよう、userconf によって適切に設定されます。
- ユーザがユーザ グループのメンバでない場合、CA Spectrum はインストールされているが、ユーザが CA Spectrum ユーザ グループにいないことを通知するメッセージが表示されます。

  CA Spectrum を使用するユーザを有効にするには、ユーザ アカウントを CA Spectrum ユーザ グループに追加します。 次に、ユーザのログアウトを実行してから、再ログインする必要があります。

  CA Spectrum を使用しないユーザは、ユーザが Spectrum を実行しないチェック ボックスをオンにできます。 このオプションを使用すると、ユーザのログイン時に userconf は実行を継続しますが、プロセスは非通知で終了します。

  注: メッセージを再表示するには、以下のコマンドを実行します。

  ```
  userconf -install %SPECROOT%
  ```

ネットワーク分割

すでに設定されているネットワークをモデリングする場合、ネットワーク分割は重大ではありません。 ただし、複数のランドスケープを作成しており、ディスカバリを使用してトポロジ階層を作成する場合は、ディスカバリをサポートするメソッドを使用します。

目的は、ランドスケープ内のモデルの重複と、ランドスケープ間の接続数を最小限に抑えることです。障害分離を促進するには、2 つの IP サブネットを接続するルータなど、いくつかのモデルを複製します。

ネットワーク分割については、以下のオプションがあります。

- IP アドレス境界によって定義されるパーティション（サブネット）を作成します。 IP アドレス範囲は、ディスカバリによる検索を制限する最も便利な手段です。
- IP アドレス境界によりランドスケープを分離することができない場合は、各ランドスケープ内のデバイスを区別するために一意のコミュニティ名を使用します。 このメソッドはより多くの作業を必要とします。各デバイスのコミュニティ名テーブルに、このような名前を追加します。
- 複数の IP サブネット アドレスを含む別のパラメータ（タイム ゾーンまたは状態など）に従って、パーティションを作成します。 ほとんどの場合、手動でネットワークをモデリングする必要があります。 ただし、ランドスケープ間で維持する接続数を最小限に抑え、モデリング ルールを遵守します。 接続数が極端に多くなると、DSS の利点が損なわれることがあります。

注: 詳細については、「IT インフラストラクチャのモデリング/管理 - 管理者ガイド」を参照してください。

分散環境の重複モデル

DSS 環境で、CA Spectrum は「ホーム」モデルとして指定することにより、重複モデル（複数のランドスケープに存在するユーザ モデルなど）を追跡します。 ホーム モデルは、「root」メイン ロケーション サーバ（MLS）を使って、ホスト上で実行されている SpectroSERVER 上に存在します。

重要：DSS 環境で root MLS を変更した場合、MLS の「ホーム」モデルと一致するように、すべての重複モデルの状態が更新されます。 これは、DSS 環境においては、MLS が最終権限を持っているためです。

CA Spectrum は、ホーム モデルを使用して、すべての重複モデルの情報を同期します。 ホーム モデルでない重複モデルに対して行われた修正要求は、ホーム モデルのランドスケープに中継されます。 その後ホーム モデルは、他のすべての重複モデルに要求を分配します。

ホーム ランドスケープへの接続失敗エラー

編集中のモデルのホーム ランドスケープに接続できない場合、編集操作が失敗することがあります。 OneClick は、SpectroSERVER からの以下のエラーなど、関係エラーをレポートします。

編集中のモデルが複数のランドスケープに存在し、そのホーム ランドスケープにアクセスできないため、操作に失敗しました。

この種のエラーは、「root」 MLS 上で実行されている SpectroSERVER 上のホーム モデルがアクセス不能であることを示している可能性があります。分散環境で重複モデルが追加または削除された場合、ホーム ランドスケープを通じてルーティングされます。その SpectroSERVER がダウンしているか、接続できない場合、関係は失敗し、エラーが生成されます。

以下のいずれかの条件下では、ホーム ランドスケープに接続できません。

- ホーム SpectroSERVER、または重複モデルとホーム SpectroSERVER 間の中間 SpectroSERVER が動作していない場合。
- ネットワークの問題によって、SpectroSERVER にアクセスできない場合。
- セキュリティ上の問題により、SpectroSERVER が通信できない場合。

ホスト リソース設定ファイル（.hostrc）

.hostrc ファイルは、各ローカル SpectroSERVER へのクライアント アプリケーションのアクセスを制限します。 クライアント アプリケーションは、.hostrc ファイルがそのアプリケーションのアクセスを許可するよう設定されていない限り、ローカル ランドスケープに接続することができません。 .hostrc ファイルは、テキスト エディタを使って編集できます。

注: 詳細については、「管理者ガイド」を参照してください。

デフォルトでは、.hostrc ファイルはローカル ホスト名を使用して初期インストールされ、すべてのリモート アクセスを制限します。 ホスト名を個別に追加すると、そのサーバとの接続が可能になります。 「+」記号を追加することで、すべてのリモート サーバに対して、無制限のアクセスが可能になります。 .hostrc ファイルにマシン名を追加すると、個々のリモート サーバからのクライアント アクセスが有効になります。

CA Spectrum はホスト名または IP アドレスによって、複数のレイヤにホスト セキュリティを実装します。 .hostrc ファイルには、ホスト名をリストして、ホスト名に関連付けられたすべての IP アドレスを含めるようにすることをお勧めします。 接続用に IP アドレスを使用する場合、ホスト名によって開始される接続試行は必ずしも成功するとは限りません。

DSS 環境のタイム ゾーン

複数のタイム ゾーンにまたがる DSS システムでは、1 つの SpectroSERVER 上で発生するすべてのアクティビティは、スケジュールされたイベントを含め、そのサーバのローカル時間で発生します。 OneClick で作成され、モデリングされたデバイスに適用されたすべてのスケジュールは、SpectroSERVER またはデバイス ランドスケープのローカル時間に従って開始および終了します。

注: 詳細については、「オペレータ ガイド」を参照してください。

SpectroSERVER のシャットダウン

SpectroSERVER を実行しているサーバをシャットダウンする前に、CA Spectrum コントロール パネルを使用して、SpectroSERVER をシャットダウンすることをお勧めします。 ただし、SpectroSERVER をシャットダウンする方法として、オペレーティング システム シャットダウン手順を使用する場合は、processd が停止するための期間を増加させます。

Solaris 環境での SpectroSERVER のシャットダウン

Solaris 環境ではデフォルトで、processd は停止する前に、すべてのサブプロセスがシャットダウンするまで 20 秒待ちます。SpectroSERVER のシャットダウン時間が長くなると、processd が停止する前の待機時間が長くなります。 デフォルト値を変更するには、PROCESSD_SHUTDOWN_TIMEOUT 環境変数を使用します。 この変数と適切な値（ミリ秒単位）を *<$SPECROOT>*/spectrum60.env ファイルに追加する必要があります。

たとえば、processd がすべてのサブプロセスのシャットダウンを 60 秒待つようにするには、以下の行を spectrum60.env ファイルに追加します。

```
PROCESSD_SHUTDOWN_TIMEOUT=60000
```

次に、変更を有効にするため、processd の停止と再起動 (P. 56)を行います。

Windows 環境での SpectroSERVER のシャットダウン

Windows 環境では、processd は停止する前に、すべてのサブプロセスがシャットダウンするまで待ちます。 ただし、Windows レジストリ設定 WaitToKillServiceTimeout は、Windows のシャットダウンが開始されてからすべてのサービスが停止するまでの Windows の待ち時間を設定します。 WaitToKillServiceTimeout で指定された時間の経過後もサービス（processd など）がまだ動作している場合、Windows はそのサービスを終了させます。 デフォルト値は 20 秒です。 この値は、システム シャットダウン時に SpectroSERVER が完全にシャットダウンするために十分な時間とは限りません。 タイムアウト値を増加させることができます。

次の手順に従ってください:

1. レジストリ エディタを開きます。
2. HKEY_LOCAL_MACHINE/SYSTEM/CurrentControlSet/Control に移動します。

3. Ctrl キーを押します。
4. 右のペインで WaitToKillServiceTimeout の値をダブルクリックします。
5. ウィンドウで、600,000 ミリ秒（10 分）以上の値に変更します。
6. ［OK］をクリックします。
7. Windows サーバを再起動して、変更を有効にします。

ランドスケープ マップ エントリ タイムアウトの設定

デフォルトでは、各ランドスケープ エントリは、ランドスケープ マップで無期限に維持されます。 ロケーション サーバ設定ファイル（.locrc）を使用すると、ランドスケープ エントリのタイムアウト発生後、ランドスケープ マップからエントリが自動削除される時間を指定できます。

注: 以前の CA Spectrum リリースでは、ランドスケープ エントリは 1 時間後にタイムアウトになり、ランドスケープ マップから自動削除されました。 CA Spectrum 9.2.2 以降、ランドスケープ エントリはデフォルトではタイムアウトになりません。

重要: ランドスケープ マップからエントリを自動削除するため、タイムアウト値を使用することは推奨されません。 タイムアウト値はデフォルトのままにしてください。 ランドスケープ マップからエントリを削除するには、必要に応じて MapUpdate コマンドを使用します。詳細については、「データベース管理ユーザガイド」を参照してください。

次の手順に従ってください:

1. <*$SPECROOT*>/LS ディレクトリに移動します。
2. .locrc ファイルに以下のエントリを更新または追加します。

   MET_INTERVAL=*time_out_value*

   time_out_value

   ランドスケープ エントリのタイムアウト発生後、ランドスケープ マップからエントリを自動削除する時間をミリ秒で示します。

   デフォルト： 0 ミリ秒（タイムアウトなし）

既存の DSS セットアップからのランドスケープ マッピングに関する問題

問題の状況：

別のインストールからランドスケープ マップをコピーして、個別の分散環境をセットアップしようとした場合に、ランドスケープ マップに関する問題が発生しました。 プライマリがダウンし、ユーザ セキュリティに関する問題が発生した後、セカンダリ SpectroSERVER への切り替えで障害が発生しました。

解決方法:

ランドスケープ マップをコピーするには、以前の DSS セットアップへの参照をすべて削除します。 ランドスケープ マップには、以前の DSS セットアップへの参照を含めないでください。 以下の手順に従って、既存の DSS セットアップからランドスケープをマッピングしてください。

次の手順に従ってください:

1. 以下のように SpectroSERVER を分断します。
   a. 各 SpectroSERVER をスタンドアロン サーバにします。

      ロケーション サーバは、インストール先のローカル サーバをポイントしています。

   b. ランドスケープ マップが SpectroSERVER ごとに異なっていることを確認します。

      注: 既存の DSS セットアップでは、ランドスケープ マップはすべてのサーバを含む単一マップでした。
2. SpectroSERVER データベースを保存します。

   ランドスケープ マップはクリーンです。
3. 元の MLS をリストアするため、これらの SpectroSERVER の MLS を変更します。

   注: 元の DSS 設定は変更しないでください。
4. 異なるホスト上で各 SpectroSERVER のデータベースを再ロードします。
5. 新しい DSS セットアップで MLS として SpectroSERVER を選択し、他の SpectroSERVER がそれをポイントするようにします。

   ユーザ モデル内の dupModeList が適切に更新されます。

6. ランドスケープ マップにセカンダリ エントリが存在する場合は、そのエントリを削除します。
7. セカンダリ SpectroSERVER エントリをセットアップします。
8. 新しい DSS 内のプライマリ SpectroSERVER からデータベースをロードします。

   ランドスケープは、既存の DSS セットアップからマッピングされます。

詳細情報:

分散 SpectroSERVER 環境の設定 (P. 35)

古いランドスケープのページに関する問題

症状:

当社の実稼働 CA Spectrum 環境は単一の分散インストールで構成されていました。 環境には複数の SpectroSERVER および OneClick サーバがあり、それらはすべて同じメイン ロケーション サーバを使用していました。

分離開発環境を作成するために、実稼働環境から 2 つの SpectroSERVER および 1 つの OneClick サーバを削除することに決定しました。新しいロケーション サーバを参照するために、開発用 SpectroSERVER および OneClick サーバを設定しました。 各環境内のサーバがそれぞれのロケーション サーバへのアクセス権のみを持つように、.hostrc ファイルを更新しました。さらに、実稼働環境のランドスケープ マップを更新し、開発ランドスケープを削除しました。

ただし、開発用 OneClick サーバ上の監視対象ランドスケープのリストを見ると、まだ古い実稼働ランドスケープがすべて表示されます。 それらは、.hostrc ファイルへの変更のために "権限なし" としてリスト表示されます。

解決方法:

ご説明のセットアップでは、現在 2 つの異なる環境があります。 ただし、開発ランドスケープ マップは古くなった実稼働ランドスケープをキャッシュしています。 古くなったランドスケープを実稼働環境から削除しましたが、開発環境からそのランドスケープを削除しませんでした。 そのため、古くなったランドスケープを開発環境から手動で削除する必要があります。

古くなったランドスケープの削除に関して 2 つのオプションがあります。

解決方法:

古くなったランドスケープを開発ランドスケープ マップから削除するために、以下のコマンドを使用できます。

`MapUpdate -remove` *ランドスケープ ハンドル*

解決方法:

すべての SpectroSERVER プロセスを再起動できます。 詳細については、「processd の停止と再起動 (P. 56)」を参照してください。 次に、OneClick サーバを再起動します。 OneClick サーバ管理の詳細については、「管理者ガイド」を参照してください。

サーバを再起動すると、問題が解決されます。

第 3 章: 分散 SpectroSERVER 環境におけるファイアウォール経由の通信

このセクションには、以下のトピックが含まれています。

ファイアウォールを経由した通信

多くのネットワーク環境で、ファイアウォールを経由した通信が使用されている場合があります。 分散環境では、ファイアウォールがユーザの展開に影響を与える可能性があります。

注: 分散 CA Spectrum インストールを開始する前に、ファイアウォールを設定して、ポート 46517 （ポート sradmin）上のトラフィックを有効にします。

CA Spectrum コンポーネント間の通信を有効にするためのオプションは、展開するファイアウォールの種類によって決まります。 コストや必要なセキュリティのレベルなどの他の要因も影響します。 その中には、仮想プライベート ネットワーク（VPN）、ノード間トンネルまたはルート、ファイアウォールを通してパケットを渡す前にパケットをカプセル化するプロキシなどがあります。

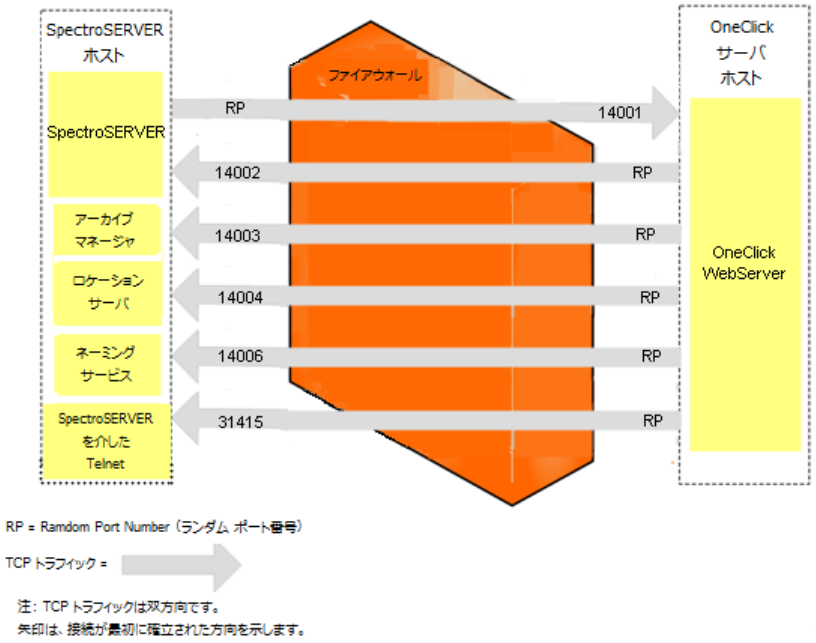
CA Spectrum と OneClick のトラフィックがファイアウォールを通過するための戦略を練るため、ネットワークのファイアウォール管理者と共同で作業することをお勧めします。

ファイアウォールを経由した SpectroSERVER と OneClick Web サーバの通信

OneClick クライアントに表示するデータを収集するため、OneClick Web サーバは SpectroSERVER ホスト システム上のプロセスと通信します。 OneClick Web サーバは通常、この通信を開始します。

OneClick Web サーバは、特定の SpectroSERVER ホスト側 TCP ポートへの接続を確立します。 OneClick Web サーバは要求の送信および応答の受信に、これらのポートを使用します。 ただし、OneClick は単一のリスン ポート（デフォルト 14001）を使用します。 SpectroSERVER は、そのポートへの接続を開始します。 その結果、多くの場合、ファイアウォール設定の変更が必要になります。 SpectroSERVER は、双方向 IIOP（Internet Inter-ORB Protocol）を使用して、CORBA クライアントと通信します。

以下の図は、OneClick Web サーバが SpectroSERVER と通信するために必要な IP 接続を示しています。 TCP を使用する場合は常に、ランダム ポートから特定の固定ポートに対して接続を確立します。



OneClick のデフォルトのポートとファイアウォール

OneClick が HTTP 通信に使用するデフォルトポートはポート 80 です。OneClick Web サーバがポート 80 以外のポートを使用するように設定した場合、ファイアウォールもこのトラフィックを許可するよう設定する必要があります。詳細については、「*OneClick 管理ガイド（5166）*」および「*インストール ガイド（5136）*」を参照してください。

注: Windows XP SP2 プラットフォーム上の OneClick ユーザが Windows ファイアウォールを有効のままにしていると、OneClick コンソール クライアントの実行に関して問題が発生することがあります。 Windows ファイアウォールの設定の詳細については、http://support.microsoft.com にある Microsoft ナレッジ ベースの記事 842242 を参照してください。

HTTP リスン ポート

OneClick が HTTP 通信に使用するデフォルトポートはポート 80 です。ポート 80 以外のポートを使用するように OneClick Web サーバを設定した場合、このトラフィックを許可するようにファイアウォールも設定する必要があります。

注: 詳細については、「管理者ガイド」と「インストール ガイド」を参照してください。

Windows XP SP2 プラットフォーム上の OneClick ユーザが Windows ファイアウォールを有効のままにしていると、OneClick コンソールの実行に関して問題が発生することがあります。

注: Windows ファイアウォールの設定の詳細については、http://support.microsoft.com にある Microsoft ナレッジ ベースの記事 842242 を参照してください。

CORBA リスン ポート

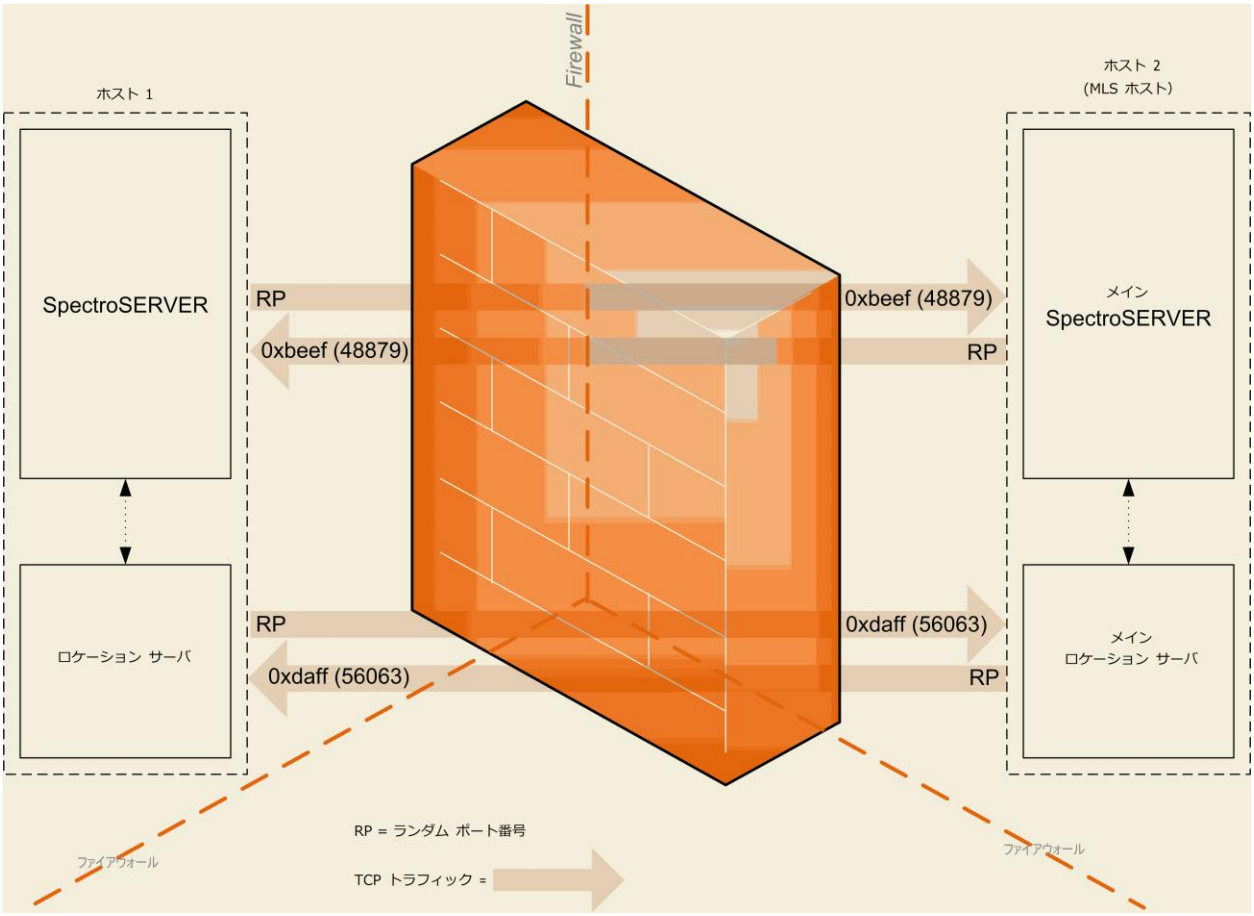
デフォルトの CORBA OneClick サーバ リスン ポートの値は、<*$SPECROOT*>/tomcat/webapps/spectrum/META-INF/context.xml にあります。

```
vbroker.se.iiop_tp.scm.iiop_tp.listener.port=<new port number>
```

リモート SpectroSERVER とファイアウォール

以下の図は、2 つのリモート SpectroSERVER がファイアウォールを経由して通信する場合に必要な IP 接続を示したものです。 TCP を使用する場合は常に、ランダム ポートから特定の固定ポートに対して接続を確立します。

重要: メイン ロケーション サーバ（MLS）でない SpectroSERVER 間の通信はすべて、MLS-SpectroSERVER を介してルーティングされます。その結果、各 SpectroSERVER は MLS-SpectroSERVER とだけ直接通信します。

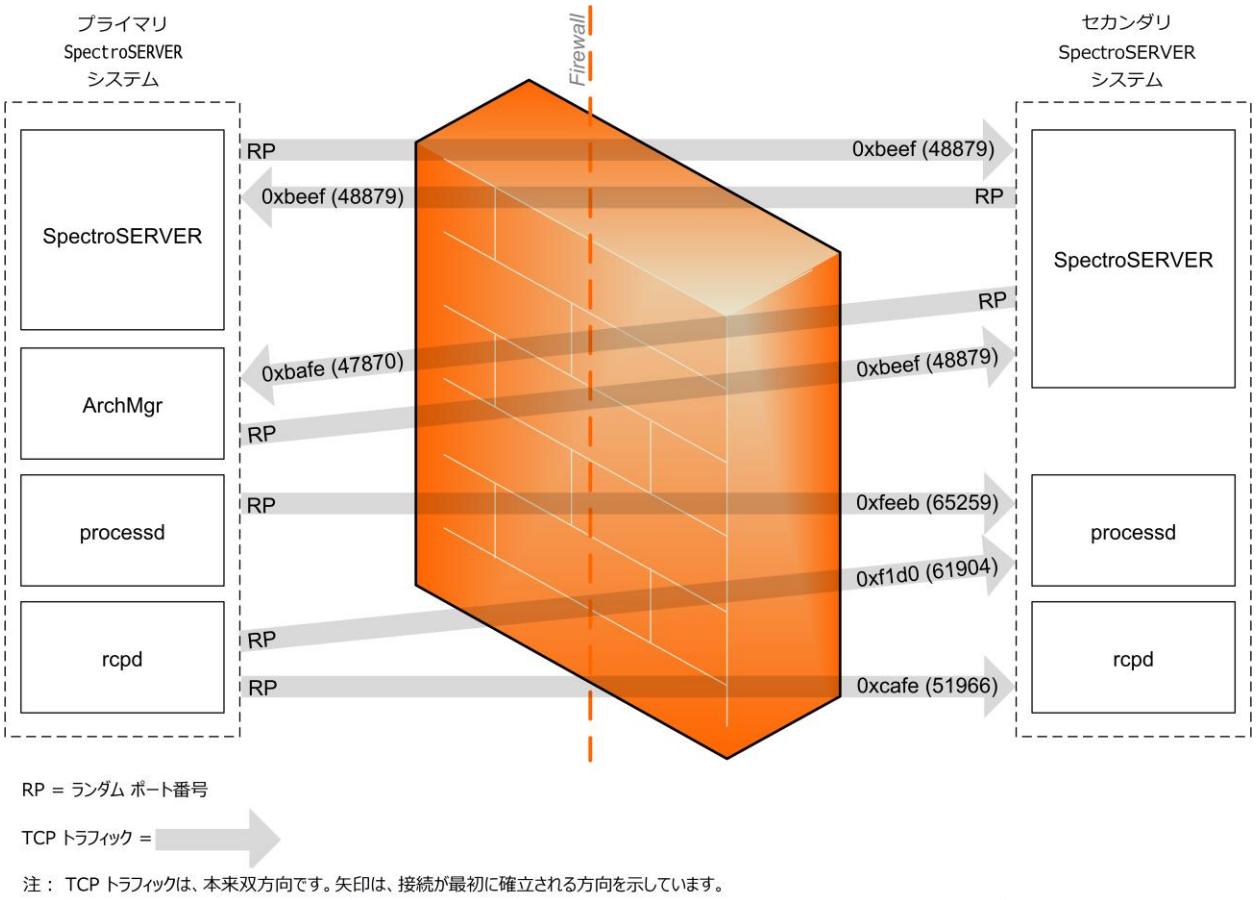


詳細情報:

ロケーション サーバ (P. 39)

ファイアウォールを経由したプライマリ SpectroSERVER とセカンダリ SpectroSERVER の通信

以下の図は、フォールト トレラント環境においてプライマリ SpectroSERVER とセカンダリ SpectroSERVER がファイアウォール経由で通信を行う場合に必要な IP 接続を示したものです。 TCP を使用する場合は常に、ランダム ポートから特定の固定ポートに対して接続を確立します。



RP = ランダム ポート番号

TCP トラフィック =

注： TCP トラフィックは、本来双方向です。矢印は、接続が最初に確立される方向を示しています。

NAT ファイアウォール環境用の CA Spectrum 設定ファイル

一貫した内部 IP アドレス指定方式を使用し、インターネットへの IP アドレス変換を単一ノードで処理するネットワークを構築する目的で、ネットワーク アドレス変換（NAT）が使われています。 CA Spectrum は NAT をサポートします。 プライベート IP アドレス ドメインで CA Spectrum を展開し、NAT ファイアウォールの外側のクライアントへの接続を保持することができます。そのため、DSS 環境には、このようなファイアウォールによって分断されている複数ドメインを含めることができます。

NAT 環境の唯一の要件は、クライアントがサーバを名前によって解決できるということです。NAT のプライベート側では、ホスト名をプライベート側の IP アドレスに解決する必要があります。 NAT のパブリック側では、ホスト名をパブリック側の IP アドレスに解決する必要があります。

デフォルトのポート設定

プロセスがファイアウォール環境で正しく動作するように、CA Spectrum プロセスが使用するデフォルトのポートまたはソケット番号を変更できます。

以下のプロセスのポート番号またはソケット番号を変更することができます。

- OneClick Web サーバ（P. 69）
- SpectroSERVER（P. 73）
- アーカイブ マネージャ（P. 73）
- ロケーション サーバ（P. 74）
- ネーミング サービス（P. 74）
- リモート コピー プロセス デーモン（rcpd）（P. 75）
- CLI デーモン（vnmshd）（P. 75）

注: リモート管理デーモン、sradmin、および SpectroSERVER 経由の Telnet のポートは変更できません。

SpectroSERVER ポート番号の変更

SpectroSERVER が CORBA 要求に使用するポート番号は変更可能です。

テキスト エディタを使って .vnmrc ファイルを編集し、新しいポート番号を反映することができます。 このファイルは、<*$SPECROOT*>/SS/ に格納されています。 以下のコマンドを入力します。

```
orb_args=-Dvbroker.se.iiop_tp.scm.iiop_tp.listener.port=<新しいポート番号>
```

詳細情報:

ポート競合の解決 (P. 37)

アーカイブ マネージャのポート番号およびソケット番号の変更

アーカイブ マネージャが特定の要求に使用するポート番号およびソケット番号を変更することができます。

テキスト エディタを使って .configrc ファイルを編集し、新しいポート番号を反映することができます。 このファイルは、<*$SPECROOT*>/SS/DDM に格納されています。 以下のコマンドを入力します。

```
orb_args=-Dvbroker.se.iiop_tp.scm.iiop_tp.listener.port=<新しいポート番号>
```

アーカイブ マネージャが VNM クライアントおよび SSAPI クライアントからの要求のリスンに使用するソケット番号を変更するには、テキスト エディタを使用して、.configrc ファイルを編集します。 このファイルは、<*$SPECROOT*>/SS/DDM に格納されています。 以下の変数を変更します。

```
ARCH_MGR_SOCKET_NUMBER=<新しいポート番号>
```

詳細情報:

ポート競合の解決 (P. 37)

ロケーション サーバのポート番号およびソケット番号の変更

ロケーション サーバが特定の要求に使用するポート番号およびソケット番号を変更することができます。

ロケーション サーバが CORBA 要求に使用するポート番号を変更するには、テキスト エディタを使用して、新しいポート番号を反映するように .locrc ファイルを編集します。 このファイルは、*<$SPECROOT>*/LS に格納されています。 以下のコマンドを入力します。

```
orb_args=-Dvbroker.se.iiop_tp.scm.iiop_tp.listener.port=<新しいポート番号>
```

ロケーション サーバが VNM および SSAPI の要求に使用するソケット番号を変更するには、テキスト エディタを使用して、.locrc ファイルを編集します。 以下の変数を変更します。

```
LOC_SERVER_SOCKET_NUMBER=<新しいポート番号>
```

詳細情報:

ポート競合の解決 (P. 37)

Visibroker ネーミング サービスのポート番号の変更

Visibroker ネーミング サービスが使用するポート番号を変更できます。

注: デフォルトのポート番号は 14006 です。

次の手順に従ってください:

1. ［マイ コンピュータ］を右クリックして［プロパティ］を選択します。

   ［システムのプロパティ］ダイアログ ボックスが表示されます。

2. ［詳細］タブをクリックし、次に［環境変数］ボタンをクリックします。
3. NAMING_SERVICE_PORT を選択して、新しいポート番号を反映するよう、以下のとおり編集します。

   ```
   NAMING_SERVICE_PORT=<新しいポート番号>
   ```

Solaris で Visibroker ネーミング サービスのポート番号を変更するには、spectrum60.env ファイルで環境変数を設定します。 このログ ファイルは、/opt/SPECTRUM ディレクトリにあります。 以下の変数を変更します。

NAMING_SERVICE_PORT=<*新しいポート番号*>

リモート コピー プロセス デーモン（rcpd）のポート番号の設定

リモート コピー プロセス デーモン（rcpd）のポート番号は、.vnmrc リソース rcpd_comm_port (P. 14) から設定可能です。

CLI デーモン（vnmshd）のポート番号の設定

CLI デーモン（vnmshd）のポート番号は、vsh_tcp_port パラメータから設定可能です。

注: vsh_tcp_port パラメータについては、「*Command Line Interface User Guide （0664）*」を参照してください。

第 4 章: フォールトトレランス

このセクションには、以下のトピックが含まれています。

フォールトトレランスについて

フォールト トレランスを実現するには、指定されたランドスケープを管理する、複数の SpectroSERVER を必要とします。 対象のランドスケープ用のデータベースのコピーは、SpectroSERVER ごとにロードされます。 ただし、アクティブなコピーは常に 1 つだけです。アクティブなデータベースを持った SpectroSERVER は、*プライマリ SpectroSERVER* と呼ばれます。 非アクティブ データベースは、スタンバイ SpectroSERVER（セカンダリ SpectroSERVER）上で実行されます。また、3 番目の SpectroSERVER に、データベースの別の非アクティブ コピーをインストールすることもできます。

プライマリ SpectroSERVER で障害が発生した場合、セカンダリ SpectroSERVER 上のデータベースがアクティブになり、セカンダリ SpectroSERVER はネットワークの管理を開始します。 プライマリ SpectroSERVER に接続されているアプリケーションは自動的にセカンダリ SpectroSERVER に切り替えられます。 プライマリ SpectroSERVER がサービスに復帰すると、アプリケーションは自動的にプライマリ SpectroSERVER に切り替わり、セカンダリ SpectroSERVER は再度非アクティブになります。

注: アプリケーションがセカンダリ SpectroSERVER から実行されている場合、すべてのアプリケーションが、全機能を実行できるとは限りません。フォールト トレラント環境を設定する主な目的は、CA Spectrum の完全なコピーを作成することではなく、ネットワークの監視が途切れないようにすることです。

詳細情報:

フォールト トレラント設定のトラップ ディレクタ (P. 103)

フォールトトレラント環境の SpectroSERVER の優先順位

プライマリ、セカンダリ、および第 3 の SpectroSERVER が同じランドスケープを管理できるようにするには、これらのサーバがすべて同じランドスケープ ハンドルとモデリング カタログを持つ必要があります。 サーバは優先順位の値で、相互に区別されます。 1 番小さな値は、プライマリ SpectroSERVER を示します。 SpectroSERVER は、10 というデフォルトの優先順位値でインストールされます。 セカンダリ サーバとして SpectroSERVER を指定するには、これより大きな優先順位番号（20 など）を割り当てます。 同様に、第 3 の SpectroSERVER には、30 など、セカンダリより大きい優先順位を割り当てます。

最初にフォールト トレラント環境を設定する際、SSdbload ユーティリティ (P. 87)を使用してデータベースのコピーをスタンバイ SpectroSERVER にロードするときに、優先順位値を割り当てることができます。

後から優先順位値を変更するには、［ロード済みランドスケープ］サブビューを使用することができます。 ナビゲーション パネルでローカル ランドスケープを選択し、次に［コンポーネント詳細］パネルの［情報］タブを選択することで、このサブビューにアクセスします。

注:［ロード済みランドスケープ］サブビューは、［SpectroSERVER コントロール］サブビューとは異なります。ナビゲーション パネルで VNM を選択し、次に［コンポーネント詳細］パネルの［情報］タブを選択することで、［SpectroSERVER コントロール］サブビューにアクセスします。

詳細情報:

フォールト トレランスの確立 (P. 87)

データ同期

フォールト トレラント CA Spectrum 環境では、アクティブなデータベースは常に 1 つだけです。 そのため、新しいモデル、およびアクティブなデータベースの属性値への変更を反映するため、他のデータベースを定期的に更新する必要があります。 このようなデータの同期は CA Spectrum オンライン バックアップ機能によって実行されます。 必要に応じて、または定期的なスケジュール間隔で、オンライン バックアップを実行できます。プライマリ SpectroSERVER に対してオンライン バックアップを実行すると、現在のデータベースのバックアップ コピーが作成されます。 オンライン バックアップを実行すると、指定した各セカンダリ SpectroSERVER にコピーを自動的にロードされます。

DSS 環境と同様に、フォールト トレラント環境の各 SpectroSERVER には、同じモデリング カタログをインストールする必要があります。 オンライン バックアップでは、現在のモデリング カタログがコピーされます。 ただし、すべての .i ファイル、または個別の管理モジュールに関連付けられている他のエレメントがすべてコピーされるとは限りません。 そのため、プライマリ SpectroSERVER に新しい管理モジュールをインストールする場合は、セカンダリ SpectroSERVER にも同じ新規管理モジュールをインストールする必要があります。

注: 詳細については、「データベース管理ガイド」を参照してください。

セカンダリ SpectroSERVER がステータス情報を得るため、プライマリ SpectroSERVER をポーリングする場合、*<$SPECROOT>*/custom/Events ディレクトリで定義されている EventDisp ファイルと Alertmap ファイルは、フォールト トレラント サーバに伝達されます。

セカンダリ SpectroSERVER が再起動しない場合のアラームの生成

プライマリ SpectroSERVER のデータベースがセカンダリ SpectroSERVER と同期すると、「セカンダリ サーバとの接続損失」イベント（0x00010c0e）とアラームが生成されます。 セカンダリ SpectroSERVER は、プライマリ SpectroSERVER から新しいデータベースをロードするために停止しました。

セカンダリ SpectroSERVER が再起動しない場合にのみアラームが生成されるように、このアラームの処理ルールを設定できます。

EventPair ルールでは、「セカンダリ サーバとの接続損失」イベントが発生し、指定した時間内に「セカンダリ サーバへの接続確立」（0x00010c0f）イベントが発生しない場合、新規イベントを生成することを指定できます。次に、この新しいイベントが、セカンダリ SpectroSERVER がまだ停止していることを示すイベントとアラームを作成するように指定できます。

次の手順に従ってください:

1. EventDisp ファイルをテキスト エディタで開きます。

   注: EventDisp ファイルは *<$SPECROOT>*/SS/CsVendor/Cabletron ディレクトリにあります。

2. 「0x00010c0e E 50 A 2, 0x00010c0e」という行を探して、以下のように変更します。

   ```
   0x00010c0e R Aprisma.EventPair, 0x00010c0f,
       <numberofsecondstowait><generatedeventcode>
   ```

   <generatedeventcode>

   <numberofsecondstowait> で指定した時間内にセカンダリ SpectroSERVER が起動しない場合に生成されるイベント コードです。

3. EventDisp ファイルに以下の行を追加します。

   ```
   <generatedeventcode>E 50 A 2, <generatedalarmcode>
   ```

   <generatedeventcode>

   セカンダリ SpectroSERVER が起動しなかった場合に手順 2 で生成されたイベント コードです。「E 50」は、イベントがログ記録され、重大度の値が 50 であることを示します。「A 2」は、重大アラームが生成されたことを示します。*<generatedalarmcode>* は、このイベントに基づいて生成されるアラーム コードです。

4. このアラームの Probable Cause ファイルを作成し、データ同期後にセカンダリ SpectroSERVER との接続が再確立されていないことを示します。

注: 詳細については、「Event Configuration User Guide」を参照してください。

セカンダリ SpectroSERVER の準備状況レベル

セカンダリ SpectroSERVER は、3 レベルのいずれかの準備状況レベルにあると考えられます。 準備状況は、サーバ設定とステータスによって決まります。 準備状況レベルは、以下のように定義されます。

ホット

セカンダリ SpectroSERVER は稼働中で、すでにポーリングを行っているため、プライマリ SpectroSERVER の障害時にただちに処理を継承できます。 セカンダリ SpectroSERVER をこの準備状況レベルに設定するには、「secondary_polling=yes」という行を .vnmrc ファイルに追加します。 このステートメントにより、プライマリ SpectroSERVER との接続ステータスにかかわらず、スタンバイ サーバの起動時にポーリングとトラップの処理が常に開始されます。

ウォーム

セカンダリ SpectroSERVER は稼働中ですが、サーバが完全動作状態になるまで、少し時間がかかる可能性があります。 セカンダリ SpectroSERVER は、プライマリ SpectroSERVER との接続が失われるまで、ポーリングを開始しないように設定されています。 たとえば、.vnmrc ファイルに secondary_polling エントリが設定されていない、またはこのエントリが no に設定されている場合などです。

.vnmrc ファイル内に secondary_polling エントリがない、またはこのエントリが no に設定されている場合、セカンダリ SpectroSERVER はスタンバイ モードのときにはトラップを処理しません。

コールド

セカンダリ SpectroSERVER が稼働しておらず、プライマリ SpectroSERVER の障害時に起動する必要があります。 この場合、セカンダリ SpectroSERVER がセカンダリ ポーリング用に設定されるかどうかは関係ありません。

詳細情報:

フォールト トレランスの確立 (P. 87)
フォールト トレランス設定の検証 (P. 91)

SpectroSERVER アラーム同期

プライマリおよびセカンダリの SpectroSERVER は、アラーム情報を共有するため、GAS（Global Alarm Service）接続を使用します。SpectroSERVER は、アラームを同期するためにアラーム情報を使用します。 この同期により、アラームの重複を防止できます。

フォールト トレラント アラーム同期用オプションには、サービスおよびデバッグ ログ記録の有効化などがあります。 .vnmrc ファイル内の設定によって、これらのオプションを制御します。 詳細については、「フォールト トレラント アラーム サービス（.vnmrc） リソース (P. 25)」を参照してください。

セカンダリ SpectroSERVER が設定された VHM およびシャーシ デバイスを展開している場合、アラーム同期でいくつかの予期しない動作が発生します。プライマリ SpectroSERVER が停止した場合、プライマリ SpectroSERVER で相関されたアラームはセカンダリ SpectroSERVER では相関されません。代わりに、異なる条件または症状ごとにのアラームが表示されます。 プライマリ SpectroSERVER がオンラインに戻ると、アラーム相関が適切に実行されます。 ただし、シャーシ デバイスで相関はまだ実行されません。

相関が保持されないのは、セカンダリ SpectroSERVER 上のアラームがイベントから作成されるのではなく生成されるためです。

アラーム同期のプロセスは、同期がプライマリ SpectroSERVER からセカンダリ SpectroSERVER か、セカンダリからプライマリかに応じて異なります。以下のセクションでは、これらの各シナリオについて説明します。

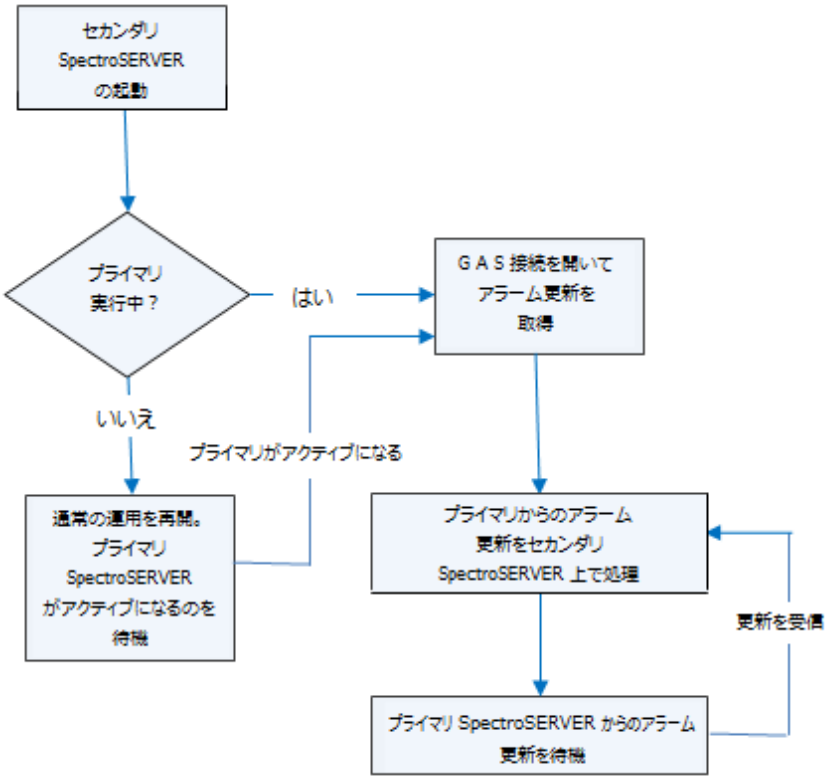
- プライマリからセカンダリ SpectroSERVER への同期（P. 82）
- セカンダリからプライマリ SpectroSERVER への同期（P. 85）

プライマリからセカンダリ SpectroSERVER への同期

SpectroSERVER がセカンダリ SpectroSERVER として実行されている場合、プライマリ SpectroSERVER への GAS 接続を開こうとします。接続が成功した場合、セカンダリ SpectroSERVER はプライマリ SpectroSERVER からアラーム更新を受信するように登録します。 セカンダリ サーバはプライマリへの接続を維持し、プライマリ サーバ上でアラーム更新が発生すると、その情報を受信し続けます。 そうでない場合、プライマリ SpectroSERVER がアクティブになるまで、セカンダリ SpectroSERVER は 1 分間に 1 回、接続を試行します。

注: アラーム同期目的で、セカンダリ SpectroSERVER がプライマリ SpectroSERVER との接続を試行するには、プライマリ サーバとセカンダリ サーバの両方で、フォールト トレラント アラーム サービスを有効にする必要があります。 この機能を制御するには、.vnmrc ファイルで ftasv_enabled パラメータを使用します。詳細については、「フォールト トレラント アラーム サービス（.vnmrc） リソース (P. 25)」を参照してください。

以下の図は、セカンダリ SpectroSERVER がプライマリ SpectroSERVER からのアラーム更新をどのように処理するかを示しています。



- 新しいプライマリ SpectroSERVER アラームは、セカンダリ SpectroSERVER に追加され、「古い」としてマーキングされます。 新しいアラームと既存アラームが同等の場合、新しいアラームによって既存アラームが置き換えられます。 アラームが同等かどうかは、以下のパラメータを確認することで決定できます。
  - Unique Alarm（ID）
  - Model Handle
  - Probable Cause
  - Alarm Discriminators

  ID が同じ場合、2 つのアラームは同等であると考えられます。 同じモデルに関するアラーム（つまり Model Handle が同じ場合）で、Probable Cause も同じ場合、Alarm Discriminator が異なっていない限り、アラームは同等であると考えられます。

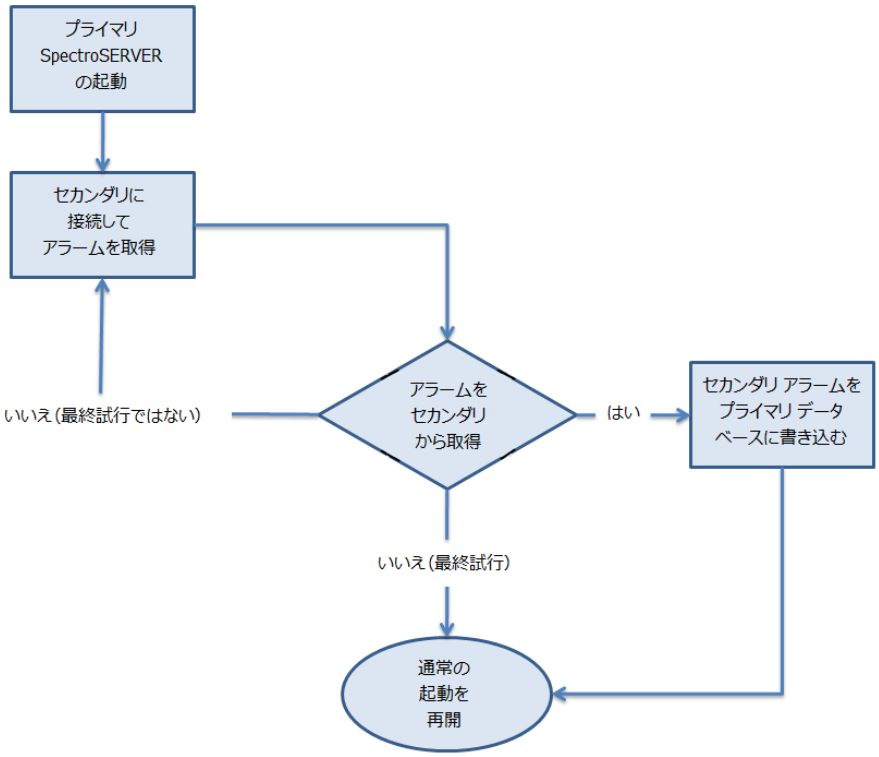
- プライマリ SpectroSERVER からのアラーム属性更新は、セカンダリ SpectroSERVER 上の同じアラームに適用されます。 アラームがプライマリ SpectroSERVER 上でクリアされると、セカンダリ SpectroSERVER 上でもクリアされます。
- isManaged 属性と isNotHibernating 属性は、メンテナンス モード アラーム同期用に更新されます。

セカンダリからプライマリ SpectroSERVER への同期

障害または計画的メンテナンスの後にプライマリ SpectroSERVER がオンラインに復帰すると、セカンダリ SpectroSERVER に接続し、利用可能なアラームをすべて取得します。最初の接続試行が失敗した場合、プライマリ サーバは同期のため、接続試行を複数回実行できます。 プライマリ SpectroSERVER の .vnmrc ファイル内にある ftasv_max_conn_retry_count および ftasv_conn_retry_interval パラメータによって、試行の回数と頻度を制御します。 詳細については、「フォールト トレラント アラーム サービス（.vnmrc） リソース (P. 25)」を参照してください。

重要: プライマリ サーバによる接続試行時に、セカンダリ SpectroSERVER が動作していない場合、プライマリは同期試行を使い果たします。 その結果、プライマリ起動時に遅延が発生します。 この遅延はモデルのアクティブ化の前に発生するため、回避不能です。 プライマリ サーバの起動時に、セカンダリ サーバが動作していることを確認することで、このような状況を回避します。 また潜在的な遅延を縮小するため、再試行回数または再試行間隔を減らすこともできます。 または、.vnmrc ファイルで ftasv_enabled パラメータを使用して、フォールト トレラント アラーム サービスを無効にできます。

以下の図は、プライマリ SpectroSERVER がセカンダリ SpectroSERVER への GAS 接続をどのように開き、すべてのアラームを取得するかを示しています。



接続確立後:

- プライマリ SpectroSERVER はアラーム リストを取得した後、データベースにアラームを書き込みます。 プライマリ SpectroSERVER のスタートアップは続行し、これらのアラームはデータベースから読み取られます。
- セカンダリ SpectroSERVER からのアラームは、取得時にプライマリ データベースに格納されているアラームと置き換えられます。
- データベースから読み取られたアラームはすべて、正しいモデル上で処理され、アサートされます。

- プライマリ SpectroSERVER のダウン中、セカンダリ SpectroSERVER で生成された新しいアラームごとに、セカンダリ「アラーム セット」イベント（0x10714）が生成されます。
- プライマリ SpectroSERVER のダウン中、セカンダリ SpectroSERVER でクリアされたアラームごとに、セカンダリ アラーム クリア イベント（0x10715）が生成されます。

注: 青と灰色のアラームに対して、同期は発生しません。 WA_Link モデルに対してではなく、茶色のアラームに対して、同期が発生します。WA_Link モデル例外は、茶色のアラームにのみ適用されます。

同期失敗後:

- プライマリ SpectroSERVER がセカンダリへの接続を試行している場合にアラーム同期が失敗すると、プライマリは起動しますが、イベントおよびクリティカル アラーム（0x10c35）はプライマリ LocalLscpe モデル上で生成されます。 イベントとアラームには、各試行の失敗理由が含まれています。 アラーム同期が失敗した後、プライマリ SpectroSERVER の実行を継続するには、オンライン バックアップを実行して、プライマリとセカンダリ SpectroSERVER データベースを同期します。

フォールトトレランスの確立

CA Spectrum の初期インストール時、任意のモデルを作成する前に、フォールト トレラント環境をセットアップできます。 または、CA Spectrum をインストールした後、フォールト トレラント環境をセットアップすることができます。

以下の手順では、2 台の SpectroSERVER（プライマリとセカンダリ）をセットアップする方法について説明します。 また同じステップを実行して、3 番目の SpectroSERVER をセットアップすることもできます。 ただし 3 番目の SpectroSERVER には、セカンダリ SpectroSERVER より大きな優先順位を割り当てます。

注: Southbound Gateway を統合した環境でフォールト トレランスを確立するには、「Southbound Gateway Toolkit Guide」を参照してください。

次の手順に従ってください:

1. プライマリ SpectroSERVER とセカンダリ SpectroSERVER の両方に、モデリング カタログが同じ CA Spectrum の同一バージョンをインストールします。 各サーバは同じランドスケープ ハンドルを必要とします。
2. プライマリとセカンダリの両方の SpectroSERVER の .hostrc ファイルに、SpectroSERVER の相互アクセスを許可するエントリがあることを確認します。

   注: プライマリ SpectroSERVER の .hostrc ファイルでセカンダリ SpectroSERVER の安全なユーザが指定されていて、セカンダリ SpectroSERVER が Windows 環境で実行されている場合、安全なユーザリストにユーザ SYSTEM を含めてください。
3. セカンダリ SpectroSERVER サーバ上の .locrc ファイル内の MAIN_LOCATION_HOST_NAME パラメータが、プライマリ SpectroSERVER の .locrc ファイル内の同じシステム名をポイントしていることを確認します。 そうでない場合、同期は失敗します。
4. 各 SpectroSERVER を実行しているユーザが同じになるよう、プライマリ SpectroSERVER とセカンダリ SpectroSERVER を設定します。ユーザが同じでない場合、オンライン バックアップ実行後、セカンダリ SpectroSERVER で障害が発生するか、正しく動作しなくなります。

5. オンライン バックアップを実行して、プライマリ SpectroSERVER データベースのコピーを作成します。 または、SpectroSERVER がシャットダウンしている場合、（モデリング カタログと新規モデルを保存するため）-cm 引数を指定して SSdbsave ユーティリティを使用します。

   注: 詳細については、「データベース管理ガイド」を参照してください。

6. 作成した保存ファイルがセカンダリ SpectroSERVER をホストするサーバで利用できることを確認します。 必要に応じて、ファイルをサーバにコピーします。

7. SpectroSERVER がシャットダウンしているセカンダリ サーバ上で、CA Spectrum SS ディレクトリに移動し、以下のコマンドを使用して、保存ファイルをロードします。

   ```
   ../SS-Tools/SSdbload -il -add precedence savefile
   ```

   precedence

   プライマリ サーバのデフォルト値 10 より大きな値を指定します（20 を推奨します）。

   savefile

   以前作成した保存ファイル名を指定します。

8. （オプション）セカンダリ SpectroSERVER が「ホット」バックアップ (P. 81)として機能するように、.vnmrc ファイルに「secondary_polling=yes」という行を追加します。

9. プライマリ SpectroSERVER が起動していない場合は起動します。

10. セカンダリ SpectroSERVER を起動します。

11. 設定を確認するには、view 引数を付けて MapUpdate コマンドを使用し、現在のランドスケープ マップを表示します。

    注: 詳細については、「データベース管理ガイド」を参照してください。

    プライマリ SpectroSERVER で障害が発生した場合、セカンダリ SpectroSERVER が自動的に処理を継承できるようになりました。 セカンダリのポーリングをアクティブ化している場合は、セカンダリ SpectroSERVER が即座に使用可能になります。 そうでない場合、サーバがプライマリ SpectroSERVER との接続損失を検出すると、ただちにポーリングが開始されます。

サービスがプライマリ SpectroSERVER からセカンダリ SpectroSERVER に切り替わると、［接続ステータス］アイコン  が黄色で表示されます。ランドスケープ内のすべてのサーバの接続ステータスを表示するには、［接続ステータス］アイコンをクリックします。 ［接続ステータス］ダイアログ ボックスで、ランドスケープ内の各サーバの［接続ステータス］アイコンが黄色で表示され、「切り替え済み」状態を示します。

プライマリ SpectroSERVER がオンラインに復帰すると、セカンダリ SpectroSERVER はポーリングを停止します（secondary_polling を yes に設定していない場合）。 すべてのアプリケーションは再度、プライマリ SpectroSERVER に切り替わります。 ただし、セカンダリ SpectroSERVER がアクティブ中にセカンダリ SpectroSERVER に対して行った編集は、プライマリ SpectroSERVER に自動的にレプリケート *されません*。これらの変更は、プライマリ SpectroSERVER に手動で実行してください。

プライマリ SpectroSERVER を再起動すると、すべてのモデルがロードされた後、すべてのモデルがアクティブ化される前に、接続が受け入れられます。 モデルのアクティブ化には、少し時間がかかることがあります。 プライマリ SpectroSERVER が再起動されると、セカンダリ SpectroSERVER がただちにポーリングを停止するため、ネットワーク管理範囲内のギャップが発生する場合があります。

この状況を回避するため、wait_active リソースを yes に設定するように、プライマリ SpectroSERVER の .vnmrc ファイルを編集します。このパラメータにより、サーバはすべてのモデルがアクティブになるまで待機してから接続を受け入れます。また CA Spectrum コントロール パネル内のメッセージ エリアは、アクティブにされるモデルのパーセンテージを動的に表示します。 SpectroSERVER の起動に時間がかかるように思われることがあります。 ただし、モデルがすべてアクティブにされると、SpectroSERVER はネットワークを管理する準備ができたことになります。

セカンダリ SpectroSERVER でも、wait_active リソースを yes に設定することができます。プライマリ SpectroSERVER の計画済みのシャットダウン中に、CA Spectrum コントロール パネルで、セカンダリ SpectroSERVER が処理を継承する準備ができていることを確認できます。

注: 詳細については、「データベース管理ガイド」を参照してください。

詳細情報:

分散 SpectroSERVER について (P. 9)
新しいメイン ロケーション サーバの指定 (P. 42)
フォールト トレラント環境の SpectroSERVER の優先順位 (P. 78)

フォールトトレランス設定の検証

分散 SpectroSERVER 展開でフォールト トレランスをセットアップした後、OneClick サーバにプライマリおよびセカンダリ SpectroSERVER へのアクセス権があることを確認します。 両方のサーバに接続されていない場合、OneClick サーバはセカンダリ SpectroSERVER へのフェールオーバを実行できません。

次の手順に従ってください:

1. ［OneClick 管理］（ランドスケープ Web ページ）にアクセスします。
2. セカンダリのステータス列を確認します。 OneClick がセカンダリ SpectroSERVER と接続を確立したことを確認します。

   またステータスは、フォールト トレランスがフェールオーバの準備ができているかどうかを示します。

   フォールト トレランス設定が検証されます。

フォールトトレランスのテスト

初期インストールでは、プライマリ SpectroSERVER がアクセス権が持つすべてのデバイスに対してセカンダリ SpectroSERVER がアクセス権を所有しているとは限らないことがあります。 この状況では、セカンダリ SpectroSERVER は誤ったアラームを生成します。 誤った警報を回避するには、フォールト トレランスをテストすることで、セカンダリ SpectroSERVER がネットワーク デバイスを管理できることを確認します。

注: 新しいデバイスをプライマリ SpectroSERVER に追加した場合は、フォールト トレランスをテストしてください。

次の手順に従ってください:

1. プライマリ SpectroSERVER とセカンダリ SpectroSERVER の両方が稼働している状態で、プライマリ SpectroSERVER を停止します。

   ［接続ステータス］アイコン  が黄色で表示され、「切り替え済み」状態を示します。

   赤色のコネクタは、OneClick サーバがセカンダリ SpectroSERVER に接続できなかったことを示します。

2. セカンダリ SpectroSERVER を実行するまで、15 ～ 20 分待ちます。
3. 以下の条件を確認します。
   - ［接続ステータス］アイコンが赤色で表示されていない。
   - すべてのデバイス モデルおよび Ping 可能モデルの SNMP 接続または ICMP 接続が失われていない。

     この接続が失われた場合、セカンダリ SpectroSERVER がデバイスにアクセス権があることを確認します。 該当する場合、ネットワークに問い合わせて、この問題を解決します。
   - CA Spectrum は、接続状態を確立したすべてのデバイスを管理します。 任意のデバイス モデルからデバイス接続または管理接続の損失アラームがないかチェックして、ステータスを確認します。
4. プライマリ SpectroSERVER を再起動します。

   ［接続ステータス］アイコンが緑色で表示され、正常な接続状態を示します。

プライマリ アーカイブ マネージャおよびプライマリ SpectroSERVER の再起動

フォールト トレラント SpectroSERVER 環境でイベントまたは統計データの損失を防ぐには、アーカイブ マネージャを 1 つだけ動作させます。 このプライマリ アーカイブ マネージャを、プライマリ SpectroSERVER ホストにインストールします。 このような構成では、障害が発生してもデータが失われることはありません。

2 つの失敗シナリオが考えられます。

- プライマリ SpectroSERVER が停止します。 その後、セカンダリ SpectroSERVER は、プライマリ SpectroSERVER をホストするサーバ上で実行されているプライマリ アーカイブ マネージャへ、イベントおよび統計情報を転送します。プライマリ SpectroSERVER が再起動した場合、イベントおよび統計データの損失はありません。
- プライマリ SpectroSERVER およびプライマリ アーカイブ マネージャを実行しているコンピュータが、動作を完全に停止します。 その後、プライマリ SpectroSERVER コンピュータがオンラインに復帰するまで、セカンダリ SpectroSERVER は、イベントおよび統計データをデータベースにキャッシュします。

サーバがダウンした場合、またはプライマリ SpectroSERVER が作動を停止した場合、プライマリ アーカイブ マネージャとプライマリ SpectroSERVER の両方を再起動します。

次の手順に従ってください:

1. プライマリ SpectroSERVER で CA Spectrum コントロール パネルを起動し、［制御］メニューから［アーカイブ マネージャの開始］を選択します。

   プライマリ アーカイブ マネージャが再度動作可能になったら、セカンダリ SpectroSERVER によってキャッシュされたイベントと統計データをプライマリ アーカイブ マネージャに転送します。

   重要: データ損失を回避するため、セカンダリ SpectroSERVER がキャッシュされたデータを転送する前に、プライマリ SpectroSERVER を起動しないでください。

2. セカンダリ SpectroSERVER でイベントおよび統計ログ情報ビューにアクセスします。
   a. ［VNM］アイコンをハイライトします。
   b. アイコンのサブ表示メニューから［設定］を選択します。
   c. ランドスケープ設定ビュー内のイベント/統計ログをクリックします。
3. イベントと統計ログ情報ビューを監視します。

   ［ローカルに格納されたイベント数］フィールドと［ローカルに格納されたレコード数］フィールドにそれぞれゼロが表示された場合、セカンダリ SpectroSERVER のデータベースにキャッシュされたイベントと統計は、プライマリ アーカイブ マネージャに転送されています。
4. プライマリ SpectroSERVER を再起動します。

プライマリ SpectroSERVER およびセカンダリ SpectroSERVER のホスト名の変更

フォールト トレラント環境内の SpectroSERVER は、ホスト名に関連付けられている優先順位値を使って、相互の関係を認識します。 そのため、フォールト トレラント関係を維持するには、プライマリ SpectroSERVER のホスト名を変更する場合に SSdbsave と SSdbload を使用する必要があります。

次の手順に従ってください:

1. -cm オプションを付けて SSdbsave を使用し、データベースを保存します。
2. ホスト名を変更します。
3. 最初の手順で作成した保存ファイルを使って、データベースを再ロードします。 -il オプションおよび -replace オプションを指定して、SSdbload を実行します。

   ```
   SSdbload -il -replace precedence savefile
   ```

   このコマンドは、プライマリ SpectroSERVER を指定する優先順位値（10）を使って、データベースに新規ホスト名を関連付けます。

   オンライン バックアップの実行結果として、次回データベースが同期されると、ホスト名の変更がウォーム スタンバイまたはホット スタンバイの SpectroSERVER に伝達されます。

ただし、ホスト名の変更により、プライマリ SpectroSERVER を動作していることを、スタンバイ SpectroSERVER が検出できなくなります。その結果、ウォーム スタンバイとして設定されている SpectroSERVER はポーリングを開始します。

4. これを防ぐには、-il オプションと -replace オプションを付けて SSdbload を使用し、ウォーム スタンバイ上で保存ファイルをロードして、スタンバイとして指定する、より大きい優先順位値（20 など）を指定します。

これで、セカンダリ SpectroSERVER のホスト名を変更できます。

次の手順に従ってください:

1. -cm オプションを付けて SSdbsave を使用し、データベースを保存します。
2. ホスト名を変更します。
3. 最初の手順で作成した保存ファイルを使って、データベースを再ロードします。 -il オプションおよび -replace オプションを指定して、SSdbload を実行します。

   ```
   SSdbload -il -replace precedence savefile
   ```

   このコマンドは、セカンダリ SpectroSERVER を指定する優先順位値（20）を使って、データベースに新規ホスト名を関連付けます。

   セカンダリ SpectroSERVER を再起動する場合、サーバはプライマリ SpectroSERVER に新しいホスト名と優先順位を伝えます。

注: 詳細については、「データベース管理ガイド」を参照してください。

プライマリとセカンダリの SpectroSERVER 間の切り替えの監視方法

ウォッチを使用して、フォールト トレラント環境のステータスを監視することができます。プライマリまたはセカンダリの SpectroSERVER の引き継ぎ準備ができたら、アラートを生成するウォッチを作成します。

次の手順に従ってください:

1. VNM モデルに対して、PercentInitialized（0x11da6）属性を監視するウォッチを作成します。

   この属性の値が 100% になると、SpectroSERVER が初期化され、引き継ぎ準備が完了します。

2. 以下の式が真として評価された場合にイベントまたはアラームを生成するか、スクリプトを実行するよう、ウォッチを設定します。

   ```
   PercentInitialized == 100
   ```

3. アクティブなポーリング対象のウォッチとして、ウォッチをセットアップします。

4. ウォッチを伝達するため、プライマリ SpectroSERVER とセカンダリ SpectroSERVER を同期します。

5. ウォッチ式で、Model_Name （0x1006e）属性の値を指定します。 セカンダリ SpectroSERVER の引き継ぎ準備ができている場合に限り、この属性はユーザに通知します。

   たとえば、以下のウォッチ式が真として評価された場合、セカンダリ SpectroSERVER *<sec_server>* の引き継ぎ準備は完了しています。

   ```
   (PercentInitalized == 100) & (Model_Name= <sec_server>)
   ```

   注: 詳細については、「ウォッチ ユーザ ガイド」を参照してください。

6. false イベントまたはアラームの可能性を制限するために、セカンダリ SpectroSERVER 上の .vnmrc ファイルに次の行を追加します。

   ```
   is_secondary = TRUE
   ```

   この設定では、セカンダリ SpectroSERVER がプライマリ SpectroSERVER を継承することを CA Spectrum によって決定しない限り、セカンダリ SpectroSERVER はイベントをドロップします。

セカンダリ SpectroSERVER のステータスの監視方法

セカンダリ SpectroSERVER サーバがプライマリ SpectroSERVER として動作している場合に、アラートを生成するウォッチを作成することができます。

次の手順に従ってください:

1. VNM モデルに対して、セカンダリ SpectroSERVER の PausePolling 属性（0x11b63）の値を監視するウォッチを作成します。

   セカンダリ SpectroSERVER でこの属性が FALSE に設定されている場合、セカンダリ SpectroSERVER はポーリングを行い、プライマリ SpectroSERVER として動作しています。

   たとえば、以下のウォッチ式が真として評価された場合、セカンダリ SpectroSERVER *<sec_server>* はプライマリ SpectroSERVER として動作しています。

   ```
   !PausePolling & (Model_Name == <sec_server>)
   ```

2. アクティブなポーリング対象のウォッチとして、ウォッチをセットアップします。

3. ウォッチを伝達するため、プライマリ SpectroSERVER とセカンダリ SpectroSERVER を同期します。

   注: 詳細については、「ウォッチ ユーザ ガイド」を参照してください。

4. false イベントまたはアラームの可能性を制限するために、セカンダリ SpectroSERVER 上の .vnmrc ファイルに次の行を追加します。

   ```
   is_secondary = TRUE
   ```

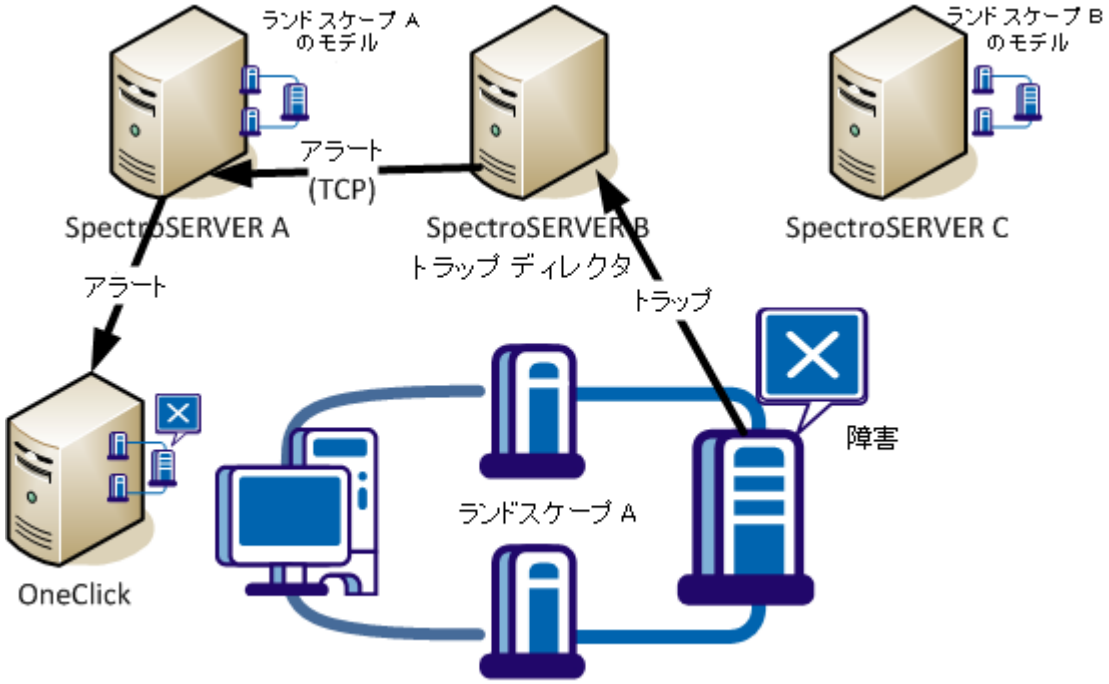
   この設定では、セカンダリ SpectroSERVER がプライマリ SpectroSERVER を継承することを CA Spectrum によって決定しない限り、セカンダリ SpectroSERVER はイベントをドロップします。

第 5 章: トラップ ディレクタの使用

このセクションには、以下のトピックが含まれています。

トラップ ディレクタ

トラップ ディレクタは、分散 SpectroSERVER 環境で有効にできる SpectroSERVER 機能です。 トラップ ディレクタはトラップを受信し、トラップ ディレクタ ホスト サーバ上でそれらを処理します。 その後、トラップ ディレクタは、トラップの情報を使用してアラートを作成します。このアラートは、TCP 接続を介して SpectroSERVER へ転送されます。 そのため、主なトラップ ディレクタ メカニズムは、追加の処理のために SpectroSERVER がリモート ホスト上のモデルに送信する CA Spectrum 内部アラートです。 以下の図では、トラップ ディレクタのアーキテクチャを示します。



トラップ ディレクタを使用するには、トラップ ディレクタ サーバとして 1 つの SpectroSERVER を指定します。 そのサーバ上でトラップ ディレクタを有効にします (P. 104)。 次に、リモート ランドスケープでモデル化されるデバイスからのトラップ用のネットワーク管理ステーション（NMS）受信者としてそのサーバを設定します。

トラップ ディレクタは暗号化された SNMPv3 トラップをサポートします。 CA Spectrum は、ユーザの SNMPv3 プロファイルを使って、受信した暗号化 SNMPv3 トラップの復号化を試行します。

SpectroSERVER の使用可能な SNMPv3 プロファイルは、不明な SNMPv3 トラップ用認証情報テンプレートとして扱われます。 暗号化された SNMPv3 トラップを受信すると、トラップを復号化するために、SNMPv3 プロファイルが使われます。

注: トラップ ディレクタのランドスケープに有効な認証情報を持つ SNMPv3 プロファイルが存在する場合、トラップの転送元として機能している SpectroSERVER は SNMPv3 トラップを処理することができます。

トラップとメモリの使用率

トラップ ディレクタが有効な場合は、SpectroSERVER プロセスのメモリ使用率を監視します。 多くのトラップ（毎秒 100 以上）が複数のリモート ランドスケープに転送される場合、SpectroSERVER がすべての受信トラップを転送できない状況が発生する可能性があります。 その後、トラップは以降の処理のためキューに格納されます。 状況が引き続き発生する場合、キューが使用するメモリは SpectroSERVER プロセスが利用できるメモリ リソースを使い果たしてしまう可能性があります。 その後、十分なメモリがないため、SpectroSERVER は予期せずシャット ダウンします。

プロセス メモリ使用率に加えて、VNM モデルの以下の属性を綿密に監視します。

`alert_remote_fwd_queue_length`（属性 ID： `0x130c3`）

一般的に、キュー サイズが 1000 エレメントを超える大きさになった場合、環境でトラップ ディレクタを有効にすることはお勧めしません。

トラップ データトラフィックの統合

ネットワークの成長に伴い、追加のランドスケープ間でモデルを分配することが求められるため、元のランドスケープ ホストをこのようなモデルのトラップ送信先として保持することができます。 このため、関連するデバイス上で新しいトラップ送信先を再設定する必要がありません。 反対に、複数のトラップ送信先を統合し、トラップの受信とルーティングを行う単一の SpectroSERVER を指定することができます。 負荷分散が重要な要因である場合、複数のサーバでトラップ ディレクタを有効にできます。この場合、各サーバでは、そのサーバにトラップを送信するよう設定されたデバイス群からのトラップを処理します。

トラップ ディレクタは、最新のモデル アドレス キャッシュをメンテナンスします。このキャッシュにより、トラップ ディレクタはモデルとトラップの送信元を一致させることができます。 トラップ ディレクタは、一致を使って、リモート ランドスケープ上の送信先モデルへトラップを転送することができます。 キャッシュ情報を最新にしておくことで、トラップ ディレクタは、モデルの場所と関係なく、CA Spectrum によるモデル用イベントの生成を可能にします。

注: 新しいランドスケープを分散環境に追加した場合、トラップ ディレクタのパフォーマンスが影響を受けることがあります。 多数の新規モデルに対して多数のトラップを処理する場合も、パフォーマンスが影響を受ける可能性があります。 このような状況では、トラップ通知遅延が増加する場合があります。

トラップ ディレクタがアドレス キャッシュを更新する方法

トラップ ディレクタは、分散環境にあるモデルの IP アドレスとロケーションのキャッシュを保持します。 アドレス キャッシュは、アラートの転送先を決定するためのインデックスの役割を果たします。 モデルについては、ランドスケープへの追加、ランドスケープからの削除、ランドスケープ間の移動が定期的に実行されることがあります。そのため、トラップ ディレクタは、コンテンツを最新状態に維持するため、定期的にキャッシュを更新します。 指定可能な保存期間（またはエージング）しきい値に適合するレコードが削除されます。 トラップ ディレクタはまた、モデル IP アドレスからトラップを受信すると、キャッシュ内で使用できないモデル IP アドレスに対して、ランドスケープのクロス検索を実行します。

以下の手順では、トラップ ディレクタがトラップの送信先モデルをどのように決定するかについて説明します。

1. トラップ ディレクタ サーバはトラップを受信します。
2. トラップ ディレクタは、トラップに含まれていた IP アドレスと、キャッシュ レコード内の IP アドレスを比較します。
3. トラップ ディレクタが一致を見つけた場合、リモート ランドスケープ上のモデルへアラートを転送します。

   そうでない場合、トラップ ディレクタは、送信先モデルを決定し、アラートを転送するために以下のタスクを実行します。

   - トラップ ディレクタは、一致する IP アドレスがないか既知のランドスケープを検索します。
   - リモート ランドスケープ上の IP アドレスに関連付けられる送信先モデルが検出された場合、トラップ ディレクタはモデル情報を使ってキャッシュを更新します。 トラップ ディレクタはモデルへアラートを転送します。
4. ランドスケープのクロス検索によって一致するアドレスが検出されなかったためにトラップがドロップされた場合、CA Spectrum は、トラップ ディレクタ サーバ上の VNM モデルに関するイベントを生成します。このイベントは、送信先モデルが見つからなかったことを示します。

   たとえば、モデルが削除されている場合、一致は見つかりません。

フォールト トレラント設定のトラップ ディレクタ

フォールト トレラントのトラップ ディレクタ設定を実装する場合、トラップをプライマリ SpectroSERVER とセカンダリ SpectroSERVER の両方に転送するようデバイスを設定する必要があります。セカンダリ サーバは、プライマリ サーバの障害を検出した場合にのみトラップをルーティングします。 セカンダリ SpectroSERVER は、プライマリのバックアップまたは同期処理中にトラップ ディレクタの設定を受信します。

詳細情報

フォールト トレランスについて (P. 77)

トラップ ストーム設定

トラップ ストームを処理するため、トラップ ディレクタは、モデリングされたデバイス上で設定されているトラップ ストームの設定を使用します。 トラップ ディレクタがトラップ ストームを検出すると、リモート ランドスケープ上のモデルへのアラーム転送が停止されます。 またトラップ ディレクタは、モデルに対するトラップ ストーム アラームのアサートも行います

トラップ ストームは、CA Spectrum でモデリングされていないデバイスから発生することもあります。 トラップ ディレクタはこのようなストームに対して、トラップ ディレクタ サーバ上の VNM モデル用に設定されたトラップ ストーム処理設定を使用します。

トラップ ディレクタの有効化および無効化

SpectroSERVER 上でトラップ ディレクタを有効または無効にすることができます。

注: OneClick 属性エディタまたは CA Spectrum コマンド ライン インターフェースを使用して、サーバの VNM モデルの属性値を設定します。

次の手順に従ってください:

1. VNM モデルの［情報］タブで、［トラップ管理］サブビューを展開します。
2. ［トラップ ディレクタの有効化］フィールドで［設定］をクリックし、必要に応じ［有効］を選択します。

   トラップ ディレクタが有効になります。
3. （オプション）トラップ ディレクタを無効にするには、［トラップ ディレクタの有効化］フィールドで［設定］をクリックし、［無効］を選択します。

   トラップ ディレクタが無効になります。

キャッシュ レコード保存期間の定義

トラップ ディレクタのトラップ キャッシュを破棄する頻度をコントロールできます。 キャッシュ レコードの保存期間を定義できます。

キャッシュ レコード保存期間を定義するには、以下の属性の保存期間を定義します。

`trap_cache_age_out_minutes（0x12ad5）`

デフォルト： 180（分）