オラクル ホワイトペーパー
2010 年 4 月

Oracle Solaris コンテナで Oracle Database を実行するためのベストプラクティス

はじめに 1
Oracle Solaris コンテナ 2
Oracle Solaris ゾーン区分技術 2
Solaris リソースマネージャー 4
Oracle Solaris コンテナのライセンスモデル 7
Oracle Solaris コンテナの作成 8
要件 8
資源プールを使用可能にする 9
非大域ゾーンの作成 10
特別な条件 11
Oracle Solaris コンテナ上のデバイス 11
Oracle Solaris コンテナ上の UFS ファイルシステム 12
Oracle Solaris コンテナ上の Oracle Solaris ZFS ファイルシステム 13
System V 資源制御 16
Dynamic Intimate Shared Memory (DISM) 16
専用 CPU 17
IP インスタンス: 非大域ゾーンの LAN と VLAN の分離 18
Oracle Solaris コンテナの移動 18
ボリューム管理 19
CPU の可視性 19
まとめ 20
著者紹介 21
謝辞 21
参考文献 21
付録 A: Oracle Solaris コンテナを作成するスクリプト 23
README.txt 24
The setenv.sh ファイル 25

zone_cmd_template.txt ファイル 26
create_zone_cmg.pl スクリプト 26
create_container.sh スクリプト 27
付録 B: System V IPC カーネルパラメータの設定 30
Oracle インストールのための System V IPC パラメータの設定 33
資源制御コマンドを使用する System V IPC パラメータの設定 33

はじめに

Oracle Solaris オペレーティングシステムは Oracle Solaris コンテナに対応しています。Oracle Solaris コンテナは、単一の Oracle Solaris インスタンス内で、分離され、セキュリティ保護された実行環境を提供する仮想化テクノロジーです。Oracle Solaris コンテナを使用すると、管理者は個別にワークロードを管理したり、資源使用量を制御したり、IP ネットワークの分離を維持したりできます。これらの機能で、複数のアプリケーションや同じアプリケーションの複数のインスタンスを単一のシステムで安全に共存できるので、サーバー統合による経費節減が可能になります。

Oracle Database 9i R2 と 10g R2 のデータベースは、どちらも Oracle Solaris のコンテナ上で実行できることが認定されました。ライセンス契約では、制限付きの Oracle Solaris 10 コンテナをハードパーティションとみなしています。そのため、Oracle Solaris コンテナに設定する CPU やコアのみをライセンスできるので、柔軟性や、統合に有利な状況、また可能なかぎりの費用の節減を実現します。

このドキュメントでは次のトピックを取り上げます。

- 「Oracle Solaris コンテナ」(2 ページ目) では、Oracle Solaris コンテナの概要を示します。Oracle Solaris ゾーンと Solaris リソースマネージャーも説明します。
- 「Oracle Solaris コンテナのライセンスモデル」(7 ページ目) では、オラクルが対応するライセンスモデルについて説明します。
- 「Oracle Solaris コンテナの作成」(8 ページ目) では、Oracle Database を配備するのに適した Oracle Solaris コンテナに非大域ゾーンを作成し、構成する手順を示します。
- 「特別な条件」(11 ページ目) では、Oracle Solaris コンテナ上で Oracle Database を実行するガイドラインについて説明します。

注意–このドキュメントで説明する Oracle Solaris のすべての機能を利用するには、Oracle Solaris 10 10/08 (U6) 以降のリリースを使用する必要があります。このドキュメントでは、標準の Oracle Database についてのみ説明し、Oracle Real Application Cluster (RAC) については取り上げていません。Oracle Solaris コンテナテクノロジーで可能な、同一システム上の個別の Oracle Solaris コンテナに複数の Oracle Database インスタンスを統合する方法についても、このドキュメントでは説明しません。サーバー統合のための Oracle Solaris コンテナテクノロジーの使用方法について、詳細は参考文献 [1] と [3] を参照してください。

Oracle Solaris コンテナ

Oracle Solaris コンテナはオラクルのオペレーティングシステムレベルの仮想化テクノロジーで、完全で、分離され、セキュリティ保護された実行環境をアプリケーションに提供します。このテクノロジーは、柔軟なソフトウェア定義の境界を使用して、アプリケーションのコンポーネントをお互いに分離できます。Oracle Solaris コンテナは、アプリケーションが使用する資源についてきめ細かい制御ができるように設計され、特定のサービスレベルを維持しながら複数のアプリケーションを単一の Oracle Solaris 10 インスタンス上で動作させることができます (図 1 を参照)。

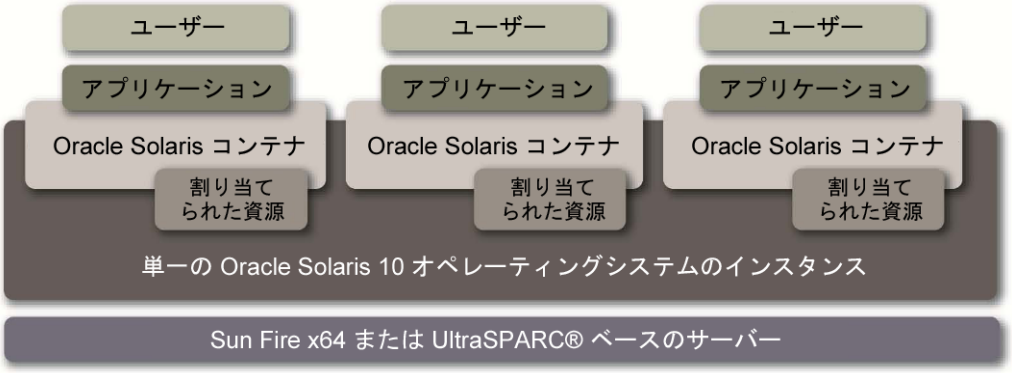

図 1. 単一 Oracle Solaris 10 インスタンス上の複数の Oracle Solaris コンテナ。

Oracle Solaris コンテナは、Solaris リソースマネージャー (SRM、Solaris Resource Manager) の機能と Oracle Solaris ゾーンソフトウェア区分技術を使用して、アプリケーションのワークロード間に資源境界を固定できる仮想化環境を実現します。Oracle Solaris コンテナのこの 2 つの主要コンポーネントについては、次のセクションで説明します。これらのテクノロジーの詳細は、文献 [2] と [4] を参照してください。

Oracle Solaris ゾーン区分技術

Oracle Solaris コンテナ環境のコンポーネント Oracle Solaris ゾーンは、オペレーティングシステムのサービスを仮想化し、分離されてセキュリティ保護された環境を、実行するアプリケーションに提供するソフトウェア区分技術です。Oracle Solaris ゾーンは、単一サーバー上の複数のアプリケーションを統合する環境に最適です。

ゾーンには、*大域ゾーン*と*非大域ゾーン*の 2 種類があります。ベースとなるオペレーティングシステム、つまりシステムのハードウェアによってブートされる Oracle Solaris インスタンスを大域ゾーンと呼びます。大域ゾーンは 1 つのシステムに 1 つで、システムのデフォルトゾーンであり、システム全体の管理制御に使われるゾーンでもあります。大域ゾーンの管理者は、1 つ以上の非大域ゾーンを作成できます。非大域ゾーンが作成されると、非大域ゾーンの管理者はそれぞれのゾーンを管理できますが、特権はそのゾーンに限定されます。

非大域ゾーンは、*疎ルート*と*完全ルート*の異なるルートファイルシステムモデルを使用して 2 種類作成することができます。

- *疎ルートモデル*–疎ルートゾーンモデルは、ルートパッケージのサブセットのみをインストールし、ほかのファイルのアクセス権を得るために読み取り専用のループバックファイルシステムを使用して、オブジェクトの共有を最適化します。このモデルでは、`/lib`、`/platform`、`/sbin`、`/usr` ディレクトリがループバックファイルシステムとしてデフォルトでマウントされます。このモデルの利点は、実行可能ファイルと共有ライブラリの効果的な共有によってパフォーマンスが改善されることと、ゾーン自身のディスクフットプリントが非常に小さいことです。
- *完全ルートモデル*–完全ルートゾーンモデルは、ゾーンのプライベートファイルシステムに必須パッケージと選択した任意のゾーンをインストールすることで、設定可能性を最大にします。このモデルの利点は、ゾーン管理者が管理するゾーンのファイルシステムのレイアウトをカスタマイズできることや、任意の別製品パッケージやオラクル以外のパッケージを追加できることなどです。

Oracle Solaris ゾーンは標準 Oracle Solaris インタフェースとアプリケーション環境を提供しますが、新しい ABI や API を必要としません。通常、アプリケーションを Oracle Solaris ゾーンに移植する必要はありません。

しかし、非大域ゾーンで実行するアプリケーションは、使用する Oracle Solaris インタフェースによっては、非大域ゾーンの動作を知る必要がある場合があります。特に次のような場合です。

- 1 つのゾーンで実行するすべてのプロセスが特権縮小セットを持つ場合。これは大域ゾーンで使用できる特権のサブセットです。この特権のセットは root ユーザーが使用できます。あるゾーンの非 root ユーザーはこれらの特権のサブセットを持ちます。デフォルトでは、非大域ゾーンの非 root ユーザーは、大域ゾーンの非 root ユーザーが使用できる特権とその非大域ゾーンで使用できる特権の「論理和」である特権を持ちます。

  非大域ゾーンで使用できない特権が必要なプロセスは、実行できないか、十分なパフォーマンスを得られない場合もわずかにあります。
- 管理者が、ゾーンが持つ特権を変更して、特権セットを縮小したり拡張したりできる場合。これによってゾーンで実行するアプリケーションに必要ない特権を削除してセキュリティを強化したり、アプリケーションの機能やパフォーマンスを向上させるためにゾーンに非デフォルト特権を提供することができます。`proc_lock_memory` 特権は、Dynamic Intimate Shared Memory (DISM) を使用する必要がありますが、ゾーンのデフォルトの特権セットに含まれるようになりました。
- 各非大域ゾーンがそれ自身の論理ネットワークとループバックインタフェースを持っている可能性がある場合。上層ストリームと論理インタフェース間の結合は、ストリームが同じゾーン内の論理インタフェースとの結合のみを確立できるように制限されています。同様に、論理インタフェースからのパケットは論理インタフェースと同じゾーン内の上層ストリームにしか転送できません。

- 各ゾーンに独自の IP 資源を持つことができる排他的 IP 特権を設定できる場合。これにより、IP 資源の完全な機能が得られ、大域ゾーンの IP に依存しません。特に排他的 IP ゾーンはゾーン自身のネットワークインタフェースと、経路指定テーブル、IPQoS 構成、IP フィルタ規則を管理できます。
- 制限されたデバイスのセットに非大域ゾーンがアクセスできる場合。通常、デバイスはシステムの共有資源です。そのため、セキュリティが脅かされないようにゾーン内に制限を設定します。
- 2 つのカテゴリの iSCSI ストレージにゾーンが対応している場合。大域ゾーンにマウントされ、iSCSI ストレージでバックアップされるディレクトリにゾーンをインストールできます。(Jeff Victor の「ネットワーク接続コンテナ」についての解説 *http://blogs.sun.com/JeffV/entry/ zoit_solaris_zones_on_iscsi* を参照してください。) あるいは、iSCSI ストレージは大域ゾーンにマウントすることができ、ファイルシステムのディレクトリはゾーンにループバックマウントすることができます。

Solaris リソースマネージャー

デフォルトでは、Oracle Solaris はシステムで実行しているすべてのワークロードが、すべてのシステム資源に公平にアクセスできます。この Oracle Solaris のデフォルトの動作は Solaris リソースマネージャーによって変更できますが、これは資源使用量の制御を実現する 1 つの方法です。

Solaris リソースマネージャー (SRM) は次のような機能を提供します。

- ワークロードを分類する機能。特定のワークロードに属するプロセスをシステムが識別します。
- ワークロードを測定する機能。ワークロードが実際に使用しているシステム資源の量を評価します。
- ワークロードを制御する機能。ワークロードが互いに干渉せず、事前に定義されたサービスレベル契約を満たすのに必要なシステム資源を取得します。

SRM は、次のような 3 種類のワークロード制御メカニズムを提供します。

- *制約メカニズム*。システム管理者はワークロードが消費できる資源を制限します。
- *スケジューリングメカニズム*。コミット不足またはコミット過剰なシナリオで、異なるワークロードのすべての資源要求を調整する割り当て決定します。
- *区分メカニズム*。事前に定義されたシステム資源が特定のワークロードに確実に割り当てます。

ワークロードの特定

Solaris リソースマネージャーは、プロジェクトとタスクの 2 つのレベルを使用し、次のようにワークロードを特定します。

- *プロジェクト*

  プロジェクトはワークロードの特定と分離を可能にする機能です。ワークロードは、いくつかのアプリケーションと、異なるグループとユーザーに属するプロセスで構成されている場合があります。プロジェクトによって提供される特定メカニズムは、1 つのワークロードのすべてのプロセスに「タグ」を提供します。この識別子は、プロジェ

クトのネームサービス・データベースによって複数のマシンで共有できます。データベースの場所は、/etc/nsswitch.conf ファイルのプロジェクト・データベースソースの定義に従い、ファイル、NIS、LDAP のいずれかとなります。プロジェクトに割り当てられた属性は、資源制御メカニズムがプロジェクト単位で資源管理のコンテキストを実現するために使用されます。

- *タスク*

  タスクは、ワークロードの特定で 2 番目の粒度レベルを実現します。タスクは、ワークロードコンポーネントを表す管理可能な対象にプロセスのグループを集めます。ログインするたびにプロジェクトに属する新しいタスクが作成され、ログインセッション中に開始されたすべてのプロセスはそのタスクに属します。プロジェクトとタスクの概念は、ps、pgrep、pkill、prstat、cron などのいくつかの管理コマンドに組み込まれています。

資源制御項目

ワークロードの資源使用量を制御するために、資源使用量を制限することができます。このような制限によって、ワークロードが特定の資源を過度に消費したり、ほかのワークロードに干渉したりするのを防止できます。Solaris リソースマネージャーは、資源使用量を制限する資源制御機能を提供します。

資源制御項目はそれぞれ、次の 3 つの値で定義されます。

- 特権レベル
- しきい値
- 特定のしきい値に関連付けられたアクション

特権レベルは資源の変更に必要な特権を示します。特権レベルは、次の 3 種類のいずれかです。

- *基本レベル*。呼び出し側プロセスの所有者が変更可能。
- *特権レベル*。特権的な呼び出し元 (スーパーユーザー) のみ変更可能。
- *システムレベル*。オペレーティングシステムのインスタンスが存続している期間は固定。

資源制御項目のしきい値は、アクションをトリガーできる実施ポイントになります。特定のしきい値に達すると、指定されたアクションが実行されます。大域アクションは、システム上のすべての資源制御項目の資源制御値が対象となります。ローカルアクションは、制御値を超過しようとするプロセスで実行されます。

ローカルアクションには、次の 3 種類があります。

- *なし*–しきい値より大きい値になっても資源要求に対してアクションを実行しません。
- *拒否*–しきい値より大きい値になると資源要求を拒否します。
- *信号*–資源制御値を超過すると大域信号メッセージアクションを有効にします。

たとえば、task.max-lwp=(privileged, 10, deny) は、そのタスクのすべてのプロセスで 10 より大きい軽量プロセスを拒否するように資源制御機能に指示します。

CPU とメモリーの管理

エンドユーザーは、SRM が提供する*公平配分スケジューラ (FSS*、*Fair Share Scheduler)*、*資源上限デーモン*、*CPU キャップ*、*専用 CPU 機能*を利用して、システム上のさまざまなワークロードで使用可能な CPU 資源と物理メモリーの消費を制御できます。

- *公平配分スケジューラ*

  Oracle Solaris のデフォルトのスケジューラは、すべてのプロセスが CPU 資源に等しくアクセスできるようにします。ただし、同じシステム上で複数のワークロードが実行している場合、1 つのワークロードが CPU 資源を独占することもできます。公平配分スケジューラは、ワークロードの重要度にもとづいて、CPU 資源にアクセスする優先順位を設定するメカニズムを提供します。

  FSS では、ワークロードの重要度はワークロードを表すプロジェクトにシステム管理者が割り当てるシェアの数によって表現されます。シェアはほかのプロジェクトに対するそのプロジェクトの相対的な重要度を定義します。プロジェクト A がプロジェクト B より 2 倍重要だとみなされる場合、プロジェクト A にはプロジェクト B の 2 倍のシェアが割り当てられるはずです。

  CPU 資源に対する競合がある場合、FSS は CPU 使用量のみを制限するように注意することが重要です。システム上にアクティブなプロジェクトが 1 つしかない場合、そのプロジェクトは、割り当てられたシェアの数にかかわらず、システムの CPU 資源を 100% 使用できます。

- *資源上限デーモン*

  資源上限デーモン (`rcapd`) は、定義した資源の上限によってプロジェクトが消費する物理メモリー量を制限するために使用されます。`rcapd` デーモンは、物理メモリー上限に設定されたプロジェクトのメモリー使用量を繰り返しサンプリングします。サンプリング間隔は管理者が指定します。システムの物理メモリー使用量の*弱い上限値*が上限実施のしきい値を超過し、その他の条件を満たす場合、デーモンはアクションを実行し、メモリー上限が設定されたプロジェクトのメモリー消費を上限以下に減らします。

  仮想メモリー (スワップ領域) も制限できます。これは*強い上限値*です。スワップ上限が設定されている Oracle Solaris コンテナで、プロセスが設定より多くの仮想メモリーを割当てようとすると失敗します。

  Oracle Database のプロセスや System Global Area (SGA) のスワッピングは妥当ではないので、Oracle Database で物理メモリーとスワップ制約を設定するのは適当ではないかもしれません。

  3 番目の新しいメモリー制限はロックメモリーです。これは、Oracle Solaris コンテナがロックダウンしたり、ページアウトするのを防ぐことができる物理メモリーの量です。デフォルトでは Oracle Solaris コンテナは `proc_lock_memory` 特権を持っているので、すべての Oracle Solaris コンテナにこの上限を設定するとよいでしょう。

- *CPU キャップ*

  CPU キャップは、プロジェクトやゾーンが消費できる CPU 資源の量に絶対的な細粒度の制限を提供します。CPU キャップは `zonecfg` 資源として提供され、プロジェクトとゾーン全体にわたる資源制御を実現します。

管理者はこの機能を使用して各ゾーンで CPU 使用量の上限値を制御することができます。これは、CPU に競合がある場合、最小の CPU 時間を特定のゾーンに保証する FSS と対照的です。

たとえば、次のコマンドでゾーンに CPU キャップを設定するとします。

```
zonecfg:myzone> add capped-cpu
zonecfg:myzone:capped-cpu> set ncpus=3.75
zonecfg:myzone:capped-cpu> end
```

ncpu パラメータは、ゾーンのユーザースレッドすべてが使用できる単一の CPU の割合を小数点数 (たとえば、.75) や混数 (整数と小数部、たとえば、3.25) で表します。ncpu 値 が 1 の場合、CPU の使用率は 100%、3.25 の場合 325%、.75 の場合 75% などとなります。上限が設定されたゾーン内のプロジェクトがプロジェクト自身の上限を持っているときは、最小値が優先されます。

- *専用 CPU*

  管理者はこの機能を使用して、ゾーンごとに指定された上限と下限内で、CPU をゾーンに動的に割り当てることができます。これは、CPU プールを作成してプールをゾーンに割り当てる必要をなくし、資源使用状況を改善し、管理をさらに単純にします。

  たとえば、次のコマンドでゾーンに専用 CPU を設定するとします。

```
zonecfg:myzone> add dedicated-cpu
zonecfg:myzone:dedicated-cpu> set ncpus=8-12
zonecfg:myzone:dedicated-cpu> end
```

  この例では、ゾーンがブートすると、システムは大域ゾーンの CPU を利用して一時的な専用プールをこのゾーンに作成します。ゾーンに CPU がさらに必要で、利用できる CPU があれば、システムは指定された制限内でゾーンに CPU を割り当てます。

Oracle Solaris コンテナの CPU とメモリー管理のそのほかの例は、文献 [9] (「New Zones Features」、Jeff Victor 著) を参照してください。

注意 - Oracle ユーザーは Oracle ソフトウェアが対応する動的な cpu_count の変更について現在の状況を確認するようにしてください。Oracle ソフトウェアのリリース時に、動的な cpu_count の変更を使用すると正しく動作しないバグがいくつか存在しました。

Oracle Solaris コンテナのライセンスモデル

オラクルは、上限付き Oracle Solaris 10 コンテナをハードパーティションと呼ばれるライセンス付与対象とみなすようになりました。Oracle Solaris 10 環境で Oracle Database を実行するオラクルのお客様は、上限付き Oracle Solaris コンテナの CPU またはコアのみのライセンスを受けることができます。

オラクルのライセンスポリシーでは、ハードパーティションは「内蔵型サーバーのように動作するサーバーの物理サブセット」と定義されています (詳細は参考文献 [2] を参照)。次の例 (図 2) は、Oracle Solaris 10 で Oracle Solaris コンテナテクノロジーを使用して、8 プロセッサシステムを 3 プロセッサのサブシステムに分割する方法を示しています。

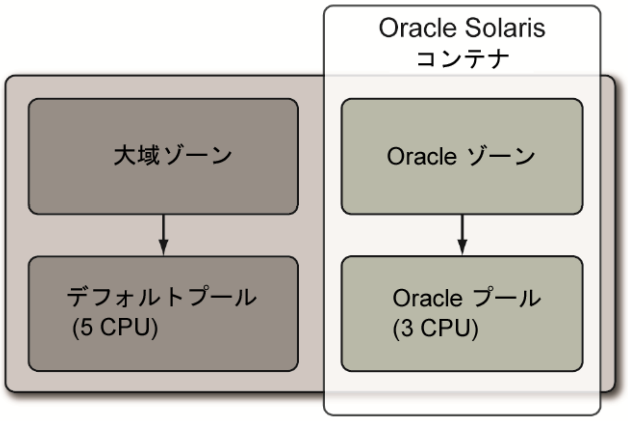

図 2. Oracle Database インスタンスを実行するために構成された Oracle Solaris コンテナの例

オラクルが設定したライセンス要件に合う Oracle Solaris 10 コンテナを作成するには、システム管理者が、必要な数の CPU またはコアを持つ資源プールを作成し、この資源プールにゾーンを結合する必要があります。あるいは、管理者は指定された CPU の上限値で動的なプールを使用するゾーンを設定してもかまいません。ライセンスは、このプールの CPU またはコア数によって変わります。

Oracle Solaris コンテナの作成

このセクションでは、Oracle Database のインストールと実行に適した Oracle Solaris コンテナを作成する手順を示します。23 ページの「付録 A: Oracle Solaris コンテナを作成するスクリプト」に掲載されているスクリプト例は、ここで紹介する手順に従っていて、Oracle Solaris コンテナの便利な作成方法を提供します。

要件

1. Oracle Solaris コンテナのルートディレクトリを格納するファイルシステムに 6G バイト以上の物理ディスク容量の空きがあることを確認します。6G バイトは、Oracle Solaris コンテナを作成し、Oracle Database バイナリをインストールするのに必要な容量です。この例では疎ルートゾーンを使用します。
2. Oracle Solaris コンテナ用の仮想インタフェースを起動するために使用される物理インタフェースを特定します。共通インタフェースには、たとえば `ce0`、`bge0`、`hme0` などがあります。大域ゾーンで利用可能な物理インタフェースを検索するには、次のコマンドを実行します。

```
# /usr/sbin/ifconfig -a
```

あるいは、ゾーンの排他的 IP を設定するには、次に示すような dladm コマンドの出力に一覧表示されるインタフェースを使用することを検討します。これらの出力は、ifconfig -a コマンドの出力では表示されません。

```
# /usr/sbin/dladm show-link
```

例では hme1 または e1000g1 が表示される可能性があります。

3. Oracle Solaris コンテナの IP アドレスとホスト名を取得します。
   a. 排他的 IP が使用されていない場合、この IP アドレスは前の手順で選択された物理インタフェースに割り当てられた IP と同じサブネット上にあるようにします。
   b. 排他的 IP が使用されている場合、どの IP アドレスでも選択できます。(このアドレスは大域ゾーンの IP と同じサブネットに属している必要はありません。)
4. 排他的 IP が使用されていなければ、/etc/nsswitch.conf ファイルで使用されるデータベースによって、大域ゾーン内で Oracle Solaris コンテナの、IP アドレスのネットマスクを解決できることを確認します。解決できない場合は、IP アドレスが属しているサブネットに必要な、ネットマスクを持つ大域ゾーン内で /etc/netmasks ファイルを更新します。

CPU をプールや使用中の専用 CPU にパーティション分割する場合、Oracle Solaris コンテナ用に予約される CPU の数を決定します。デフォルトプールで使用可能な CPU の数を調べるには、poolstat コマンドを実行します。デフォルトプールが、使用可能な CPU の数を示します。デフォルトプールには常に 1 つ以上の CPU があるように注意してください。システムにプールが設定されていない場合は、psrinfo を使用して使用可能な CPU をすべて検索してください。

資源プールを使用可能にする

資源プールと動的資源プールは、Oracle Solaris サービス管理機能 (SMF、Service Management Facility) に統合されています。動的資源プールは資源プールサービスとは別に有効にできます。

動的資源プールサービスの障害管理資源識別子 (FMRI、Fault Management Resource Identifier) は svc:/system/pools/dynamic です。資源プールサービスの FMRI は svc:/system/pools です。

資源プール機能を有効にするには、次のコマンドを使用します。

```
# svcadm enable svc:/system/pools:default
# svcadm enable svc:/system/pools/dynamic:default
```

資源プールの状態を確認するには、次のコマンドを使用します。

```
# svcs -a | grep pool
STATE STIME FMRI
online 11:21:40 svc:/system/pools:default online
11:21:45 svc:/system/pools/dynamic:default
```

非大域ゾーンの作成

資源プールサービスが使用可能になると、非大域ゾーンを作成して資源プールに結合できます。Oracle Database をインストールして実行する目的では、非大域ゾーンは完全ルートモデルと疎ルートモデルのどちらにもすることができますが、特定の要件と競合しない限り、ゾーンのディスクフットプリントを非常に小さくできるので、疎ルートモデルの使用をお勧めします。

Oracle Solaris の非大域ゾーンは、次の手順で作成できます。

1. root ユーザーとして非大域ゾーンのルートディレクトリを格納するディレクトリ (たとえば、/zones/myzone) を作成し、アクセス権を 700 に設定します。このディレクトリの名前はゾーンの名前と一致させるようにしてください (この例では myzone)。このディレクトリは Oracle Solaris ZFS (Zettabyte File System) ファイルシステムに作成することもできます (たとえば rpool/myzone または rpool/zones/myzone)。
2. 特別な手順が追加されなければ、Oracle Solaris コンテナ内の /usr ディレクトリはループバックファイルシステム (lofs) になります。つまり、Oracle Solaris コンテナは、Oracle Solaris 大域コンテナの /usr をコンテナ自身のファイルシステムツリーの /usr に、読み取り専用モードでマウントします。デフォルトでは、Oracle インストーラは /usr/local ディレクトリにファイルを作成するように root ユーザーに要求します。/usr は Oracle Solaris コンテナ内では読み取り専用なので、root ユーザーがコンテナ内に要求されたファイルを作成できるように、この例では /usr/local に特別なマウントポイントを作成します。/usr/local ディレクトリが Oracle Solaris 大域コンテナ内に存在するかどうかを確認し、存在しない場合は作成します。

3. Oracle Solaris 大域コンテナでディレクトリを作成し、/usr/local をマウントします。ディレクトリは /opt/ZONE_NAME/local に作成することをお勧めします。

4. 付録 A (25 ページの「setenv.sh ファイル」) の setenv.sh ファイルを使用して、ゾーンを作成するための設定ファイルを作成し、資源に結合します。変数 ZONE_DIR、ZONE_NAME、IP_TYPE、NET_IP、NET_PHYSICAL に適切な値を設定します。zone_cmd_template.txt ファイルでは、create コマンドが疎ルートを作成するために使用されます。このコマンドを create -b に置き換えると、完全ルートが作成されます。ゾーンは、NUM_CPUS_MIN と NUM_CPUS_MAX 制限内で動的 CPU プールで作成されます。排他的 IP を新しいゾーンに設定したい場合は、IP_TYPE=EXCLUSIVE を設定します。ゾーンに対するスケジューリングクラスと max-shm-memory もこのファイル内で設定することができます。

5. 次のコマンドを root ユーザーとして実行してゾーンを作成します (27 ページの付録 A「create_container.sh スクリプト」を参照)。

```
# sh ./create_container.sh
```

6. 次のコマンドで Oracle Solaris コンテナの構成が終了します。

```
# zlogin -C ZONE_NAME
```

特別な条件

このセクションでは、Oracle Solaris コンテナ内部で Oracle Database を実行するときの特別な条件について説明します。

Oracle Solaris コンテナ上のデバイス

セキュリティと分離が脅かされないことを保証するために、非大域ゾーンではデバイスに対し、次のようないくつかの制限が課せられます。

- 非大域ゾーンでアクセスできるのは、デフォルトでは /dev/null、/dev/zero、/dev/poll、/dev/random、/dev/tcp などの、主に擬似デバイスで構成されるデバイスの限られたセットのみ。
- dtrace、kmem、ksyms のようなシステムデータを公開するデバイスは非大域ゾーンでは使用できない。
- デフォルトでは、非大域ゾーンは物理デバイスにもアクセスできない。

大域ゾーン管理者は、非大域ゾーンで物理デバイスを使用可能にできます。物理デバイスを使用可能にしてもシステムのセキュリティが脅かされないようにすることは、主に次の 2 つの理由から、管理者の責任です。

- 物理デバイスを複数のゾーンに配置すると、ゾーン間にコバートチャネル (covert channel) が作成される場合がある。
- このようなデバイスを使用する大域ゾーンのアプリケーションでは、データの漏洩や非大域ゾーンによるデータの破壊が発生する恐れがある。

大域ゾーン管理者は、zonecfg の add device サブコマンドを使用して、非大域ゾーンにデバイスを追加できます。たとえばブロックデバイス /dev/dsk/c1t1d0s0 を非大域ゾーン my-zone に追加する場合、管理者は次のコマンドを実行します。

```
# zonecfg -z my-zone
zonecfg:my-zone> add device
zonecfg:my-zone:device> set match=/dev/dsk/c1t1d0s0
zonecfg:my-zone:device> end
zonecfg:my-zone> exit
# zoneadm -z my-zone reboot
```

/dev/dsk/c1t1d0 のすべてのスライスは、match コマンドで /dev/dsk/c1t1d0* を使用して、非大域ゾーンに追加することもできます。キャラクタデバイス (raw デバイス) やそのほかすべての種類のデバイスに同じ手順を使用できます。

Oracle インストール CD を使用して非大域ゾーンに Oracle Database をインストールする場合は、非大域ゾーンで CD-ROM デバイスを認識できるようにします。このための推奨される方法の一つは、大域ゾーンから /cdrom ディレクトリをループバックマウントすることです。/cdrom はほぼシステム全体にわたる共有デバイスなので、大域ゾーンから非大域ゾーンに物理デバイスをエクスポートする代替えの方法は推奨されません。さらに、CD-ROM デバイスのエクスポートも、大域ゾーンのボリューム管理デーモン vold を停止したり無効にしたりする必要があるため、推奨されません。非大域ゾーンから CD-ROM デバイスにアクセスする方法の詳細は、文献 [7] を参照してください。

Oracle Solaris コンテナ上の UFS ファイルシステム

各ゾーンには独自のファイルシステム階層があり、ルートはゾーンルートと呼ばれるディレクトリです。ゾーン内のプロセスは、ゾーンルートの下のファイルシステム階層部分にあるファイルのみにアクセスできます。

ここでは、非大域ゾーンに UFS ファイルシステムをマウントする次のような 3 種類の方法を、例を示して説明します。ループバックファイルシステムを使用してマウントする方法、UFS ファイルシステムとしてマウントする方法、大域ゾーンからデバイスをエクスポートする方法。

方法 1: ループバックファイルシステム (lofs) を使用してマウントする

大域ゾーンでファイルシステムを作成し、ループバックファイルシステム (lofs) として非大域ゾーンにマウントします。この方法は、大域ゾーンと非大域ゾーンの間でファイルシステムを共有するために使用します。

1. 大域ゾーン管理者としてログインします。
2. 次のように大域ゾーンにファイルシステムを作成します。

```
global# newfs /dev/rdsk/c1t0d0s0
```

3. 次のコマンドで大域ゾーンにファイルシステムをマウントします。

```
global# mount /dev/dsk/c1t0d0s0 /mystuff
```

4. 次のように非大域ゾーンに lofs 型のファイルシステムを追加します。

```
global# zonecfg -z my-zone
zonecfg:my-zone> add fs
zonecfg:my-zone:fs> set dir=/usr/mystuff
zonecfg:my-zone:fs> set special=/mystuff
zonecfg:my-zone:fs> set type=lofs
zonecfg:my-zone:fs> end
```

方法 2: UFS ファイルシステムとしてマウントする

大域ゾーンでファイルシステムを作成し、UFS ファイルシステムとして非大域ゾーンにマウントします。

1. 大域ゾーン管理者としてログインします。
2. 次のコマンドで大域ゾーンにファイルシステムを作成します。

```
global# newfs /dev/rdsk/c1t0d0s0
```

3. 次のように非大域ゾーンに ufs 型のファイルシステムを追加します。

```
global# zonecfg -z my-zone
zonecfg:my-zone> add fs
zonecfg:my-zone:fs> set dir=/usr/mystuff
zonecfg:my-zone:fs> set special=/dev/dsk/c1t0d0s0
zonecfg:my-zone:fs> set raw=/dev/rdsk/c1t0d0s0
zonecfg:my-zone:fs> set type=ufs
zonecfg:my-zone:fs> end
```

方法 3: 大域ゾーンからデバイスをエクスポートする

大域ゾーンから非大域ゾーンにデバイスをエクスポートし、非大域ゾーンからマウントします。この方法を使用して作成されたファイルシステムはゾーン間で共有できません。

1. 大域ゾーン管理者としてログインします。
2. 次のように非大域ゾーンに raw デバイスをエクスポートします。

```
global# zonecfg -z my-zone
zonecfg:my-zone> add device
zonecfg:my-zone:device> set match=/dev/rdsk/c1t0d0s0
zonecfg:my-zone:device> end
zonecfg:my-zone> add device
zonecfg:my-zone:device> set match=/dev/dsk/c1t0d0s0
zonecfg:my-zone:device> end
```

3. 非大域ゾーンで root ユーザーとしてログインします。
4. 次のコマンドで非大域ゾーンにファイルシステムを作成します。

```
my-zone# newfs /dev/rdsk/c1t0d0s0
```

5. 次のように非大域ゾーンにファイルシステムをマウントします。

```
my-zone# mount /dev/dsk/c1t0d0s0 /usr/mystuff
```

Oracle Solaris コンテナ上の Oracle Solaris ZFS ファイルシステム

Oracle Solaris ゾーン区分技術は Oracle Solaris ZFS コンポーネントに対応し、Oracle Solaris ZFS ファイルシステムやストレージプールをゾーンに追加することなどができます。このセクションでは Oracle Solaris ZFS ファイルシステムを追加したり、非大域ゾーンにデータベースを委任する方法について説明します。詳細は「Open Solaris ZFS 管理ガイド」(文献 [10]) を参照してください。

単一の Oracle Solaris ZFS ファイルシステムの追加

この例では、非大域ゾーンに Oracle Solaris ZFS ファイルシステムを追加する方法を示します。

1. 次のコマンドを実行して、大域ゾーンから従来のファイルシステムにマウントポイントを設定します。

```
global# zfs set mountpoint=legacy tank/home
```

2. 大域ゾーンから、従来のファイルシステムにマウントポイントが設定されているのを確認します。次のコマンドを使用してマウントポイントを確認し、従来のファイルシステムに設定されているかどうか確認します。

```
global# zfs get mountpoint tank/home
```

3. 次のコマンドを使用して Oracle Solaris ZFS ファイルシステムをゾーンに追加します。

```
global# zonecfg -z myzone
zonecfg:myzone> add fs
zonecfg:myzone:fs> set type=zfs
zonecfg:myzone:fs> set dir=/export/share
zonecfg:myzone:fs> set special=tank/home
zonecfg:myzone:fs> end
```

この構文で、Oracle Solaris ZFS ファイルシステムの `tank/home` が `myzone` ゾーンに追加され、`/export/share` にマウントされます。ゾーン管理者は、この構成でファイルシステム内のファイルを作成したり削除したりできます。しかし、ファイルシステムを異なる場所に再マウントすることも、ファイルシステム上で圧縮や読み取り専用にするなどのプロパティーを変更することもできません。ファイルシステムのプロパティーの設定と制御は大域ゾーン管理者が行います。

完全な Oracle Solaris ZFS データセットの委任

ストレージ管理をゾーンに委任するのが主な目的であれば、完全な Oracle Solaris ZFS データセットを非大域ゾーンに委任できます。次の例では、大域ゾーン管理者が Oracle Solaris ZFS ファイルシステムを非大域ゾーンに委任します。

1. まず、次のコマンドで大域ゾーンに Oracle Solaris ZFS ファイルシステムを作成します。

```
# zfs create distr/vol1
```

2. 次に、ZFS ファイルシステムをデータセットとして次のようにゾーンに追加します。

```
# zonecfg -z myzone
zonecfg:myzone> add dataset
zonecfg:myzone:dataset> set name=distr/vol1
zonecfg:myzone:dataset> end
zonecfg:myzone> verify
zonecfg:myzone> commit
zonecfg:myzone> end
```

3. 最後に、Oracle Solaris ZFS ファイルシステムがゾーンで有効になるように、ゾーンを再起動します。再起動すると、Oracle Solaris ZFS ファイルシステムがゾーン内部で次のように見えるようになります。

```
# zfs list
NAME USED AVAIL REFER MOUNTPOINT
distr 48,9G 18,3G 30,5K /distr
distr/vol1 24,5K 18,3G 24,5K /distr/vol1
```

この例では、distr/vol1 ディレクトリがインポートされたので、distr/vol1 ディレクトリは distr ディレクトリレベルでは管理できません。しかし、次の例のように追加のファイルシステムを distr/vol1 ディレクトリの下に作成することは可能です。

```
# zfs create distr/vol2
cannot create 'distr/vol2': permission denied
# zfs create distr/vol1/vol2
```

前述の単一ファイルシステムを追加する例と異なり、この例では Oracle Solaris ZFS ファイルシステムの /distr/vol1 が myzone ゾーン内で見えるようになります。ゾーン管理者は、ファイルシステムプロパティーの設定、子ディレクトリの作成、スナップショットやクローンの作成、そのほか Oracle Solaris ZFS ファイルシステム階層全体を制御することができます。これによって、Oracle Solaris ZFS ファイルシステム (および、ゾーンで作成された下位のすべてのファイルシステム) の管理を非大域ゾーンに委任することができます。

Oracle Solaris ZFS データセットがゾーンに委任されると、データセットプロパティーは zoned=on に設定されます。これによって、マウントポイントを含むほとんどの Oracle Solaris ZFS プロパティーが大域ゾーンで使えなくなります。マウントポイントがどのような特別な値に設定されていても、実際には mounpoint=legacy に設定されていることになります。

Oracle Solaris ZFS キャッシュの上限

大域ゾーンでの Oracle Solaris ZFS キャッシュの上限は、/etc/system ファイルに zfs_arc_max パラメータを指定することで設定できます。たとえば次の例では、Oracle Solaris ZFS キャッシュの上限は 8G バイト (16 進数で 0x200000000) になります。

```
set zfs:zfs_arc_max = 0x200000000
```

使用するキャッシュに Oracle Solaris ZFS が独自の上限を設定する場合、管理者はキャッシュ使用量を監視し、明示的な上限を設定するように選択できます。ブログ記事、「Monitoring ZFS Statistic on the system loaded with Oracle database」(*http://blogs.sun.com/pomah/entry/monitoring_zfs_statistic*) に、キャッシュ使用量の監視に使用できるスクリプトが掲載されています。キャッシュの上限は、物理メモリーサイズの値が Oracle Database のすべてのインスタンスに対する SGA の総和より少なくなるように計画してください。

注–次のサイト「ZFS Evil Tuning Guide」で、Oracle Solaris ZFS キャッシュの上限やそのほかのチューニングのベストプラクティスに関する追加情報を参照できます。http://www.solarisinternals.com/wiki/index.php/ZFS_Evil_Tuning_Guide

System V 資源制御

公平配分スケジューラ (FSS、Fair Share Scheduler) は、重要度をもとにゾーンで利用可能な CPU 資源の割り当てを制御するために使用されます。

```
zonecfg:myzone> set scheduling-class=FSS
```

各ゾーンにシェアの設定を計画する場合、`set cpu-shares` コマンドを使用してシェアを指定できます。次に例を示します。

```
zonecfg:myzone> set cpu-shares=10
```

CPU に競合がある場合、このゾーンのプロセスは割り当てより多いシェアと CPU サイクルを得られます。

さらに、必要な共有メモリーセグメントの最大サイズを Oracle Database のインスタンスに設定できます。次に例を示します。

```
zonecfg:myzone> set max-shm-memory=4G
```

次のコマンドで FSS を大域的に設定し (特定の構成要件がこの設定に不適切でない場合)、大域ゾーンを再起動するようにしてください。

```
global# dispadmin -d FSS
```

この場合、スケジューリングクラスは大域ゾーンから継承されるので、各ゾーンで FSS を明示的に設定する (`set scheduling-class=FSS`) 必要はありません。

Dynamic Intimate Shared Memory (DISM)

DISM は動的にサイズ変更可能な共有メモリーを提供します。DISM セグメントを使用するプロセスは、プロセスの実行中にメモリーセグメントの一部をロックしたりロック解除したりできます。そうすることによって、アプリケーションは動的に物理メモリーをシステムに追加する調整をしたり、削除する準備をしたりできます。

デフォルトで、Oracle Database のインスタンスは Oracle Solaris 上の標準 System V 共有メモリーの代わりに Intimate Shared Memory (ISM) を使用します。共有メモリーセグメントが ISM セグメントに作られると、共有メモリーセグメントは大規模ページを使用してマップされ、そのセグメントのメモリーはロックされます (つまり、ページアウトできません)。これはプロセスコンテキスト切り替えによるオーバーヘッドを大幅に削減し、データベースのパフォーマンスを負荷時に線形に向上します。Oracle Database と DISM の詳細は、参考文献 [5] を参照してください。

ISM は標準 System V 共有メモリーに比べ、明らかにメリットがあります。しかし、ISM セグメントはサイズ変更ができません。ISM データベースのバッファーキャッシュのサイズを変更するには、データベースを停止して再起動する必要があります。DISM はこの制約を解消して、動的にサイズ変更可能な共有メモリーを提供します。DISM は Oracle Database 9*i* から Oracle Database で対応しています。Oracle Database は、`init.ora` で `sga_max_size` パラメータによって設定された SGA の最大サイズが、それらのコンポーネント (つまり、`db_cache_size`、`shared_pool_size`、そのほかのより小さな SGA コンポーネント) の合計より大きい場合、ISM の代わりに DISM を使用します。

ISM では、カーネルはメモリーページをロックしたりロック解除したりします。しかし DISM では、メモリーページのロックとロック解除は Oracle Database のプロセス `ora_dism_$`*`ORACLE_SID`* によって行われます。メモリーをロックするためには、プロセスは `proc_lock_memory` 特権が必要です。この特権は、Oracle Solaris 10 の非大域ゾーンでデフォルトで使用可能になりました。

次の例では、Oracle Database を DISM 用に構成します。

```
SQL> alter system set sga_max_size=1000M scope=spfile;
SQL> alter system set sga_target=600M scope=spfile;
SQL> shutdown immediate
SQL> startup
$ ps -ef|grep ism
root 28976 5865 0 10:54:00 ? 0:17 ora_dism_mydb
```

専用 CPU

専用 CPU 機能を導入すると、システム管理者は非常に簡単に、また、さらに効果的な方法で CPU プールを管理できるようになります。実際、システム管理者は CPU プールの作成に気を配る必要もありません。もちろん管理者は、CPU がいくつ使用可能で、どのように使用されるか知る必要はあります。専用 CPU 機能を使用すると、管理者は新しいゾーンを作成すると同時に CPU を割り当てることができ、ゾーンのプロパティーの変更で割り当てられた CPU を管理することができます。

たとえば、次のコマンドは `myzone` ゾーンで使用する CPU の範囲を指定します。

```
zonecfg:myzone> add dedicated-cpu
zonecfg:myzone:dedicated-cpu> set ncpus=8-12
zonecfg:myzone:dedicated-cpu> end
```

IP インスタンス: 非大域ゾーンの LAN と VLAN の分離

ゾーンに排他的 IP インスタンスが割り当てられているか、IP 層の構成と状態を大域ゾーンと共有しているかどうかによって、IP ネットワークを二通りの方法で構成できるようになりました。IP の種類は zonecfg コマンドを使用して設定されます。

デフォルトの IP は共有 IP です。共有 IP のゾーンでは、大域ゾーンと同じ VLAN または LAN に接続して IP 層を共有します。

排他的 IP ゾーンでは IP レベルの完全な機能を使用できます。ゾーンをネットワーク上の IP 層で分離する場合は、そのゾーンに排他的 IP を割り当てることができます。排他的 IP ゾーンを使用すると、別の VLAN や LAN の、異なるサブネット上で通信するアプリケーションを統合することができます。

大域ゾーンを含むほかのゾーンは、ネットワークインタフェースを使用しないようにしてください。

次の例では、myzone ゾーンが物理インタフェース e1000g1 を使用して排他的 IP を使用するように構成します。

```
zonecfg:myzone> set ip-type=exclusive
zonecfg:myzone> add net
zonecfg:myzone:net> set physical=e1000g1
zonecfg:myzone:net> end
```

Oracle Solaris コンテナの移動

Oracle Solaris コンテナは、1 つのサーバーから別のサーバーへ移動したり移行したりできます。この機能を利用すると、管理者はすばやくゾーンをプロビジョニングしたり、要件の変更を満たすために必要に応じてワークロードを再配置することができます。

Oracle Solaris コンテナを移動するには、ゾーンを停止します。さらに、新旧 2 つのシステムは Oracle Solaris のパッチレベルに互換性があるようにし、デバイスの場所は、そのデバイスが構成時に使用された場合、同じ場所にあるようにします。

基本的な手順は次のようになります。

1. 元のシステム上に Oracle Solaris コンテナを作成します。
2. Oracle Solaris コンテナを切り離します。
3. アーカイブを作成します。
4. アーカイブを新しいシステムに移動し、展開します。
5. Oracle Solaris コンテナを新しいシステムに接続します。

Oracle Solaris コンテナを移動する手順は、Oracle Solaris コンテナの専有ファイルを保持しているファイルシステムの種類に一部依存します。たとえば、Oracle Solaris コンテナは、ルートファイルシステムに常駐できますし、Oracle Solaris ZFS ファイルシステムにも常駐できます。また、Oracle Solaris ボリュームマネージャーのメタデバイス上に構築された UFS ファイルシステムに常駐することもできます。この 3 種類のファイルシステムの場合は、手順の詳細について参考文献 [8] を参照してください。

ボリューム管理

オラクル以外のボリュームマネージャー製品を含めて、ボリューム管理にはさまざまなオプションが利用できるため、Oracle Solaris コンテナ上でのユーザビリティーは完全に一般化できません。このドキュメントの執筆時点では、非大域ゾーンからボリュームマネージャーをインストールしたり管理することはできません。現時点では、システムの大域ゾーンからボリュームマネージャーソフトウェアをインストールし、管理するようにしてください。デバイス、ファイルシステム、ボリュームは、大域ゾーンで作成したあとで `zonecfg` サブコマンドを使用して非大域ゾーンで使用可能にできます。

たとえば、Oracle Solaris ボリュームマネージャー (SVM、Solaris Volume Manager) は大域ゾーンからインストールし、管理するようにしてください。

ストレージが大域ゾーンから適切な方法で構成されると、メタデバイスは非大域ゾーンで使用可能にできるか、マウントされたファイルシステムで使用できます。

注–オラクル以外のボリューム管理ソフトウェアに関する、技術的な問題の最新の対応状況と公開された回避方法についてはすべて、オラクル以外のサポート Web サイトにお問い合わせください。

CPU の可視性

ゾーンが資源プールに結合されている場合、ユーザーは CPU の情報を表示する Oracle Solaris コンテナ内のシステムの仮想化ビューを要求できます。このような場合、ゾーンが認識するのは、結合している資源プールに関連付けられた CPU のみです。

デフォルトで Oracle Solaris コンテナは `pool_default` プールに結合しますが、そのプールは大域ゾーンのプロセスが使用するのと同じプロセッサのセットです。

Oracle Database アプリケーションは主に、`_SC_NPROCESSORS_ONLN` 引数とともに `pset_info(2)` と `sysconf(3c)` を呼び出し、使用可能な CPU の数を取得します。この数にもとづいて、Oracle Database は内部資源のサイズを決定し、たとえば、並列クエリーや読み取り数などの異なる目的に対するスレッドを作成します。これらの呼び出しは、関連付けられた `pset` を持つ資源プールにゾーンが結合されている場合は、期待値、つまり資源プール内の CPU 数を返します。表 1 に、Oracle Solaris で修正され、このシナリオで期待値を返すインタフェースを示します。

表 1. コンテナを認識する CPU 関連の Oracle Solaris 10 インタフェースの一覧

インタフェース	型
p_online(2)	システムコール
processor_bind(2)	システムコール
processor_info(2)	システムコール
pset_list(2)	システムコール
pset_info(2)	システムコール
pset_getattr(2)	システムコール
p_online(2)	システムコール

表 1. コンテナを認識する CPU 関連の Oracle Solaris 10 インタフェースの一覧

pset_getloadavg(3c)	システムコール
getloadavg(3c)	システムコール
sysconf(3c)	システムコール
p_online(2)	システムコール
_SC_NPROCESSORS_CONF	sysconf(3c) の引数
_SC_NPROCESSORS_ONLN	sysconf(3c) の引数
pbind(1M)	コマンド
psrset(1M)	コマンド
psrinfo(1M)	コマンド
mpstat(1M)	コマンド
vmstat(1M)	コマンド
iostat(1M)	コマンド
sar(1M)	コマンド

これらのインタフェースに加えて、`psrinfo(1M)` や `mpstat(1M)` などのツールが特定のカーネル統計情報 (`kstats`) をよく使用し、システムに関する情報を取得します。`kstats` のすべての利用者には、ゾーンに結合されているプール内の `pset` の情報のみが表示されます。

まとめ

Oracle Solaris コンテナは、単一の Oracle Solaris インスタンス上で複数のアプリケーションを管理する、非常に柔軟で、セキュリティ保護された方法を提供します。Oracle Solaris コンテナはソフトウェア区分技術を使用してオペレーティングシステムを仮想化し、分離され、セキュリティ保護された実行環境をアプリケーションに提供します。

Solaris リソースマネージャーは、メモリーや CPU 使用量を制限するなど、資源使用量を制御するために使用され、ワークロードが必要なシステム資源を得られるようにします。Oracle Solaris コンテナを利用することで、複数のアプリケーションや同じアプリケーションの複数のインスタンスも、単一システムでセキュリティ保護された状態で共存できるので、サーバーを統合して経費節減できる可能性があります。

Oracle Database 9i R2 と 10g R2 は、Oracle Solaris コンテナ上での実行が認定されました。このドキュメントでは、RAC 以外の Oracle Database を実行するのに適した Oracle Solaris コンテナで、非大域ゾーンを作成する、ステップ・バイ・ステップの手順を提供しました。さらに、Oracle Solaris コンテナ内で Oracle Database を実行する場合の特別な条件について説明しました。

著者紹介

Ritu Kamboj は、Sun Microsystems の ISV 技術部門のオープンソースチームのスタッフエンジニアです。Kamboj は、データベースの設計、パフォーマンス、高可用性の専門技術によるソフトウェア開発に、12 年以上の経験があります。Sybase、Oracle、MySQL データベースの広範囲にわたって開発を行ってきました。最近では主に Solaris プラットフォーム上で MySQL の実行を最適にするための仕事をしています。

Roman Ivanov は 2006 年 1 月にサンに入社しました。Ivanov は ISV 開発部門に所属し、独立系ソフトウェアベンダーがサンのテクノロジーを導入し、サンのハードウェアパフォーマンスを向上させる支援を行っています。ブログは *http://blogs.sun.com/pomah/* で閲覧できます。

謝辞

このドキュメントの執筆にあたり、有益な助言をいただいた Alain Chéreau、Jeff Victor、Fernando Castano の各氏に感謝します。

参考文献

[1] 『Consolidating Applications with Solaris 10 Containers』、Sun Microsystems、2004。
http://www.sun.com/datacenter/consolidation/solaris10_whitepaper.pdf

[2] *Oracle Solaris 10 System Administrator Collection* -『Solaris のシステム管理 (Solaris コンテナ : 資源管理と Solaris ゾーン)』。
http://docs.sun.com/app/docs/doc/817-1592

[3] 『Solaris Containers−What They Are and How to Use Them』、Menno Lageman 著、オラクルの *Sun BluePrints OnLine*、2005。
http://www.sun.com/blueprints/0505/819-2679.html

[4] BigAdmin システム管理者向け情報、Solaris - ゾーン。
http://www.sun.com/bigadmin/content/zones

[5] 『Dynamic Reconfiguration and Oracle 9i Dynamically Resizable SGA』、Erik Vanden Meersch、Kristien Hens 著、*Sun BluePrints OnLine*、2004。
http://www.sun.com/blueprints/0104/817-5209.pdf

[6] Oracle のパーティションに関するドキュメント。
http://www.oracle.com/corporate/pricing/partitioning.pdf

[7] 『アプリケーションのゾーン対応』、Paul Lovvik、Joseph Balenzano 著、オラクルの Sun Developer Connection - Solaris 10 Technical Article、2005。
http://developers.sun.com/solaris/articles/application_in_zone.html

[8] 『How to Move a Solaris Container』、Jeff Victor 著。
http://www.sun.com/software/solaris/howtoguides/moving_containers.jsp

[9] 「New Zones Features」、Jeff Victor 著。
http://blogs.sun.com/JeffV/entry/new_zones_features

[10] 『Oracle Solaris ZFS Administration Guide』、Sun Microsystems。
http://opensolaris.org/os/community/zfs/docs/zfsadmin.pdf

付録 A: Oracle Solaris コンテナを作成するスクリプト

この付録に掲載されたスクリプトを使用すると、Oracle Database の RAC 以外のインスタンスをインストールし実行するのに適した Oracle Solaris コンテナを作成できます。これらのスクリプトを使用しなくても Oracle Solaris コンテナを作成することは可能です。スクリプトはサンプルコードとして提供しているので、固有の要件や制約事項に合わせて修正するようにしてください。

この例では、ユーザーが指定するルートディレクトリ、IP アドレス、物理インタフェースを使用して、疎ルートゾーンが作成されます。ユーザーは排他的 IP を選択して大域ゾーンに依存しないようにすることもできます。/usr/local は、Oracle のユーティリティーのいくつかをインストールするのに必要なデフォルトディレクトリなので、このディレクトリへの特別なマウントポイントは /opt/zone_name/local に作成され、Oracle インストールを容易にします。これらのスクリプトを使用するには、すべてのファイルを同じディレクトリに保存し、次の手順に従います。

1. 次の変数が適切な値になるように setenv.sh ファイルを編集します。
   - ZONE_NAME: ゾーンのホスト名
   - ZONE_DIR: ゾーンのルートディレクトリ名または Oracle Solaris ZFS プール名
   - NET_IP: ゾーンの IP アドレス。排他的 IP を使用する場合は設定する必要はありません
   - IP_TYPE: 排他的 IP を設定する場合
   - NET_PHYSICAL: ゾーンの仮想インタフェースを作成する物理インタフェース
   - NUM_CPUS_MAX: ゾーンの最大 CPU 数
   - NUM_CPUS_MIN: ゾーンの最小 CPU 数
   - SCHEDULING_CLASS=FSS: ゾーンのデフォルトスケジューリングに FSS を使用する場合
   - MAX_SHM_MEMORY=4GB: 最大共有メモリーセグメントを指定する場合
2. 次の変数が適切な値になるように setenv.sh ファイルを編集します。

```
# ./create_container.sh
```

3. 次のコマンドを大域ゾーンで実行して、この Oracle Solaris コンテナを構成します。zone_name は適切なゾーン名に置き換えます。

```
# zlogin -C zone_name
```

スクリプトを構成するファイルは次のセクションで掲載し、説明します。

README.txt

このファイルは、スクリプトを使用する場合の Oracle Solaris コンテナの作成方法を説明しています。raw デバイスへのゾーンアクセスの許可や資源プールの削除など、一般的な操作の実行方法に関するヒントもあります。

```
The scripts in this directory can be used to create a container
suitable for installing and running non-RAC instances of Oracle database. These
scripts do not represent the only way in which you can create an appropriate container
for Oracle; depending on your requirements and constrains you can modify these scripts
to fit your needs.
1) creating a container for Oracle
A sparse root zone will be created with the root directory, IP and interface provided
by the user. A special mount point for /usr/local will be created in
/opt/<zone_name>/local to facilitate the oracle installation, since /usr/local is
the default directory for the installation of some of the oracle utilities. To use
these scripts follow these steps :
a) edit the file setenv.sh with appropriate values for:
- ZONE_NAME: hostname for the zone
- ZONE_DIR: directory for the root directory of the zone
- NET_IP: IP for the zone
- NET_PHYSICAL: physical interface in which the virtual interface for the zone will
be created
- NUM_CPUS_MAX: maximum number of CPUs for the zone
- NUM_CPUS_MIN: minimum number of CPUs for the zone
- IP_TYPE: You may wish to have exclusive IP
- SCHEDULING_CLASS=FSS: To have FSS as default scheduling for the zone
- MAX_SHM_MEMORY=4GB: To specify Maximum Shared memory segment
b) from the global container run ./create_container.sh
c) Once the container has been created run "zlogin -C <zone_name>" from the global
container to finish configuring the zone.
2) giving your container access to raw devices
If you need to give your container access to a raw device follow this example once
the container has been created (these commands must be issued from the global
container):
zonecfg -z my-zone
zonecfg:my-zone> add device
zonecfg:my-zone:device> set match=/dev/rdsk/c3t40d0s0
zonecfg:my-zone:device> end
zonecfg:my-zone> exit
zonecfg -z my-zone halt
zonecfg -z my-zone boot
3) giving your container access to a file system
If you need to give your container access to a file system created in the global
container follow this example once the non-global container has been created:
global# newfs /dev/rdsk/c1t0d0s0
global# zonecfg -z my-zone
```

```
zoncfg:my-zone> add fs
zoncfg:my-zone> set dir=/usr/mystuff
zoncfg:my-zone> set special=/dev/dsk/c1t0d0s0
zoncfg:my-zone> set raw=/dev/rdsk/c1t0d0s0
zoncfg:my-zone> set type=ufs
zoncfg:my-zone> end
zonecfg -z my-zone halt
zonecfg -z my-zone boot
4) to uninstall and delete a previously created zone use these commands:
zoneadm -z $ZONE_NAME halt
zoneadm -z $ZONE_NAME uninstall -F
zonecfg -z $ZONE_NAME delete -F
```

The setenv.sh ファイル

setenv.sh ファイルは、Oracle Solaris コンテナの作成に使用するパラメータの定義に使用されます。

```
#!/usr/bin/sh
# host name for the zone
ZONE_NAME=myzone
# directory where to place root dir for the zone
# or ZFS pool
#ZONE_DIR=/zones
ZONE_DIR=rpool
#IP for the zone (make sure netmask can be resolved for this IP according to
# the databases defined in nsswitch.conf)
# or use Exclusive-IP with its own IP stack
#NET_IP=129.146.182.199
IP_TYPE=EXCLUSIVE
#interface used by the zone
NET_PHYSICAL=e1000g1
#min and max CPUs for the dynamic pool bound to the zone
NUM_CPUS_MIN=8
NUM_CPUS_MAX=12
SCHEDULING_CLASS=FSS
MAX_SHM_MEMORY=4G YSICAL=e1000g1
# do not make changes beyond this point
export ZONE_NAME ZONE_DIR NET_IP NET_PHYSICAL
export NUM_CPUS_MIN NUM_CPUS_MAX
export MOUNT_ZFS SCHEDULING_CLASS MAX_SHM_MEMORY IP_TYPE
```

zone_cmd_template.txt ファイル

zone_cmd_template.txt ファイルは、ゾーンを作成するコマンドのテンプレートセットを含んでいます。ユーザーが指定する値で文字列をいくつか置き換えると、このファイルでゾーンを作成できます。

```
create
set zonepath=MOUNTPOINT
set autoboot=true
SCHEDULING_CLASS
MAX_SHM_MEMORY
IP_TYPE
add net
NET_IP
set physical=NET_PHYSICAL
end
add fs
set dir=/usr/local
set special=/opt/ZONE_NAME/local
set type=lofs
end
MOUNT_ZFS
add dedicated-cpu
set ncpus=NUM_CPUS_MIN-NUM_CPUS_MAX
end
verify
commit
```

create_zone_cmg.pl スクリプト

このスクリプトは、perl ユーティリティーを使用してゾーンを作成するコマンドファイルを作成します。そして、ゾーンのコマンドテンプレートファイルのユーザーが指定するパラメータを置き換えます。このスクリプトは create_container.sh によって呼び出されます。

```
#!/usr/bin/perl
# Copyright (c) 2005,2008 Sun Microsystems, Inc. All Rights Reserved.
#
# SAMPLE CODE
# SUN MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT
# THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESS
# OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
# IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS
# FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.
# SUN SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED
# BY LICENSEE AS A RESULT OF USING, MODIFYING OR
# DISTRIBUTING THIS SOFTWARE OR ITS DERIVATIVES.

while (<>){
   s/MOUNTPOINT/$ENV{'MOUNTPOINT'}/;
   s/NET_IP/$ENV{'NET_IP'}/;
   s/IP_TYPE/$ENV{'IP_TYPE'}/;
```

```
   s/MOUNT_ZFS/$ENV{'MOUNT_ZFS'}/;
   s/ZONE_NAME/$ENV{'ZONE_NAME'}/;
   s/NET_PHYSICAL/$ENV{'NET_PHYSICAL'}/;
   s/NUM_CPUS_MIN/$ENV{'NUM_CPUS_MIN'}/;
   s/NUM_CPUS_MAX/$ENV{'NUM_CPUS_MAX'}/;
   s/SCHEDULING_CLASS/$ENV{'SCHEDULING_CLASS'}/;
   s/MAX_SHM_MEMORY/$ENV{'MAX_SHM_MEMORY'}/;

   print;
}
```

create_container.sh スクリプト

メインのスクリプトです。setenv.sh ファイルで指定されたパラメータを使用して Oracle Solaris コンテナを作成します。

```
#!/usr/bin/ksh
# Copyright (c) 2005,2008 Sun Microsystems, Inc. All Rights Reserved.
#
# SAMPLE CODE
# SUN MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT
# THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESS
# OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
# IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS
# FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.
# SUN SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED
# BY LICENSEE AS A RESULT OF USING, MODIFYING OR
# DISTRIBUTING THIS SOFTWARE OR ITS DERIVATIVES.
# script to create a container to run oracle RDBMS.
# to use this script follow the instructions in the README.txt file
# located in this directory.
. ./setenv.sh
#zone already exists?
zonecfg -z $ZONE_NAME info > /tmp/z.$$ 2>&1
cat /tmp/z.$$ | grep "No such zone" > /dev/null 2>&1
if [ $? -eq 1 ]
then
   echo "ERROR: zone $ZONE_NAME already exists. IF you want to remove it do:"
   echo "use zoneadm -z $ZONE_NAME halt"
   echo "use zoneadm -z $ZONE_NAME uninstall -F"
   echo "use zonecfg -z $ZONE_NAME delete -F"
   exit 1
fi
rm -rf /tmp/z.$$ > /dev/null 2>&1
# 1)............................... validate setenv.sh values
if [ `expr ${ZONE_DIR} : '¥(.¥)'` = '/' ]; then
   echo Using standard directory for the zone
```

```
   #zone path exists?
   if [ ! -d $ZONE_DIR/$ZONE_NAME ]
   then
       mkdir -p $ZONE_DIR/$ZONE_NAME
       if [ $? = 1 ]
       then
           echo ERROR: could not create root directory
           exit 1
       fi
   fi
   MOUNTPOINT=$ZONE_DIR/$ZONE_NAME
else
   echo Using ZFS for the zone
   MOUNTPOINT=`zfs get -H -o value mountpoint $ZONE_DIR/$ZONE_NAME 2>/dev/null`
   if [ x$MOUNTPOINT = 'x' ]; then
       echo zfs not exists. I will create $ZONE_DIR/$ZONE_NAME for you.
       zfs create $ZONE_DIR/$ZONE_NAME
       if [ $? -ne 0 ]; then
           echo Failed to create $ZONE_DIR/$ZONE_NAME
           exit 1
       fi
       MOUNTPOINT=`zfs get -H -o value mountpoint $ZONE_DIR/$ZONE_NAME
2>/dev/null`
   else
       echo $ZONE_DIR/$ZONE_NAME filesystem already exists
   fi
   if [ $MOUNTPOINT = 'legacy' ]; then
       echo Legacy mounted ZFS is not supported. Exiting.
       exit 1
   fi
fi
if [ x`ls $MOUNTPOINT` != 'x' ]; then
   echo ERROR:Directory $MOUNTPOINT is not empty. exiting.
   exit 1
fi
export MOUNTPOINT
chmod 700 $MOUNTPOINT
# Enabling Resource Pools
svcadm enable svc:/system/pools:default
svcadm enable svc:/system/pools/dynamic:default
#/usr/local directory exists?
if [ ! -d /usr/local ]
then
   mkdir /usr/local
fi
#special mnt point for /usr/local exists?
if [ ! -d /opt/$ZONE_NAME/local ]
```

```
then
   mkdir -p /opt/$ZONE_NAME/local
fi

# 2)............................... pool creation
# this section does not exist anymore since dynamic CPU pools is used
if [ `expr $NUM_CPUS_MIN ¥* 2` -le $NUM_CPUS_MAX ]; then
   echo WARNING: Having NUM_CPUS_MIN less then a half of NUM_CPUS_MAX is not
recommended by Oracle.
   echo Oracle bugs: 6309685 and 6309691 and ...
   echo You may adjust values before running Oracle or continue at your own risk
fi
if [ -n "$SCHEDULING_CLASS" ]; then
   SCHEDULING_CLASS="set scheduling-class=¥"$SCHEDULING_CLASS¥""
fi
if [ -n "$MAX_SHM_MEMORY" ]; then
   MAX_SHM_MEMORY="set max-shm-memory=$MAX_SHM_MEMORY"
fi if [ x"$IP_TYPE" = "xEXCLUSIVE" ]; then
   NET_IP=" "
   IP_TYPE="set ip-type=exclusive"
else
   NET_IP="set address=$NET_IP"
fi

# 3)............................... mounting zfs inside a zone
MOUNT_ZFS=' '
if [ -n "$MOUNT_ZFS" ]; then
   echo WARNING: mounting ZFS before the zone is started is not supported
   echo ignoring MOUNT_ZFS settings
   echo
   #MOUNT_ZFS=' '
   TMP_MOUNT=' '
   for z in $MOUNT_ZFS; do
       echo Changing mountpoint to legacy for $z
       zfs set mountpoint=legacy `echo $z|awk -F= '{print $1}'`
       TMP_MOUNT=${TMP_MOUNT}`echo $z|awk -F= '{print "¥nadd fs¥nset type=zfs¥nset
dir="$2"¥nset special="$1"¥nend¥n"}'`
   done
   MOUNT_Z FS=" $TMP_MOUNT "
fi

# 4)............................... zone creation
perl ./create_zone_cmd.pl < zone_cmd_template.txt > /tmp/zone_commands.txt
zonecfg -z $ZONE_NAME -f /tmp/zone_commands.txt
echo $ZONE_NAME was configured with this information:
echo ------------------------------------------------------
zonecfg -z $ZONE_NAME info
```

```
echo ------------------------------------------------------------
zoneadm -z $ZONE_NAME install
zoneadm -z $ZONE_NAME boot
echo "to finish configuring your container please run: zlogin -C $ZONE_NAME"
```

付録 B: System V IPC カーネルパラメータの設定

Oracle Solaris 10 OS 以前は、主に共有メモリー、メッセージキュー、セマフォーで構成される System V IPC 資源は、/etc/system ファイルで設定されていました。この実装には、次のようなデメリットがありました。

- 管理メカニズムとして /etc/system に依存しているため、再構成するのに再起動が必要。
- /etc/system でパラメータを設定したときの単純なスペルミスが、追跡が難しい構成エラーの原因となる場合があった。
- 従来の実装で使用されていたアルゴリズムは、サイズを変更できないデータ構造を想定していた。
- すべてのユーザーに資源を許可しないで、1 人のユーザーに追加資源を割り当てる方法がなかった。これは、資源の量が常に固定されていたので、1 人のユーザーの追加割り当てによって、ほかのユーザーのパフォーマンスに必要な割り当てができない可能性があることが明らかだったためです。
- パラメータの値を監視する適当な方法がなかった。
- 特定の調整可能なパラメータのデフォルト値が小さすぎた。

Oracle Solaris 10 OS では、これらの制限事項がすべて解決されています。Oracle Solaris 10 の System V IPC 実装では、/etc/system ファイルを変更する必要はなくなりました。その代わりに資源制御機能を使用するため、次のようなメリットがあります。

- /etc/system ファイルを変更せずに、Oracle Database インスタンスをインストールして起動できるようになった (ほとんどの場合、資源制御項目も変更する必要がない)。
- システムを再起動しないで、プロセス単位またはプロジェクト単位で (制限対象の資源による)、System V IPC 機能の使用を制限できるようになった。
- これらの制限は割り当てに直接影響を与えない。制限は、システムにすぐには何も影響を与えず、可能な限り大きくできます。(ただし、ユーザーは制限なしに資源を割り当てられるようになるため、システムに影響を与える場合があります。)
- 内部実装が管理者に公開されなくなったので、構成タスクが簡単になった。
- 資源制御項目が少なくなり、調整可能なパラメータは以前より詳細でわかりやすい名前になった。
- prctl(1) や getrctl(2) などの共通資源制御インタフェースを使用して、制限の設定を監視できる。

- 共有メモリーは、セグメント単位ではなくプロジェクト単位で割り当てられる総容量をもとに制限される。つまり、管理者は、ユーザーが多数のセグメントや大きなセグメントを割り当てられるようにできますが、多数の大きなセグメントは作成させません。
- 資源制御項目は管理メカニズムであるため、project(4) を使用して構成を持続的にしたり、ネットワーク経由で構成できる。

Oracle Solaris 10 では、次のような変更が加えられました。

- メッセージヘッダーを動的に割り当てられる。以前は、すべてのメッセージヘッダーがモジュールのロード時に割り当てられていました。
- セマフォー配列が動的に割り当てられる。以前は、セマフォー配列が seminfo_semmns でサイズ指定された vmem アリーナで割り当てられていたため、断片化によって割り当てが失敗する可能性がありました。
- セマフォー取り消し構造体が、プロセス単位とセマフォー配列単位で動的に割り当てられる。数の制限はなく、対応するセマフォー配列と常に同じ大きさである。以前は、数に限りがあるプロセス単位の取り消し構造体で、モジュールのロード時に割り当てられていました。さらに、取り消し構造体はすべて固定された同じサイズでした。プロセスが取り消し構造体を割り当てられない場合や、プロセスの取り消し構造体が一杯になる場合もありました。
- セマフォー取り消し構造体は、符号付き整数として取り消し値を維持するため、セマフォーの値が大きすぎて取り消せないことがない。
- すべての機能は、サイズが固定された名前空間でオブジェクトを割り当てるために使用され、モジュールのロード時に割り当てられていた。すべての機能の名前空間のサイズが変更できるようになり、需要が高まると増加できるようになった。

これらの変更の結果、次の関連パラメータが削除されました (表 2 を参照)。Oracle Solaris システムの /etc/system ファイルにこれらのパラメータを記述しても無視されます。

表 2. Oracle Solaris 10 で不要になったシステムパラメータ

パラメータ名	概説
semsys:seminfo_semmns	システム上の System V セマフォーの最大数
semsys:seminfo_semmnu	System V セマフォーシステムが対応する取り消し構造体の総数
semsys:seminfo_semmap	セマフォーマップ内のエントリー数
semsys:seminfo_semvmx	セマフォーに設定できる最大値
semsys:seminfo_semaem	取り消し構造体内のセマフォー値に設定できる最大値
semsys:seminfo_semusz	取り消し構造体のサイズ
shmsys:shminfo_shmseg	プロセス当たりのセグメント数
shmsys:shminfo_shmmin	共有メモリーセグメントの最小サイズ

表 2. Oracle Solaris 10 で不要になったシステムパラメータ

パラメータ名	概説
msgsys:msginfo_msgmap	共有メモリーセグメントの最小サイズ
msgsys:msginfo_msgssz	メッセージセグメントのサイズ
msgsys:msginfo_msgseg	メッセージセグメントの最大数
msgsys:msginfo_msgmax	System V メッセージの最大サイズ

前述のとおり、/etc/system ファイル内の多数のパラメータが不要になったので削除されました。残りのパラメータにはより妥当なデフォルト値を設定し、より多くのアプリケーションがパラメータを設定しなくても、購入したままの状態で動作できるようになりました。

表 3 では、/etc/system ファイルに残ったパラメ-タのデフォルト値を示します。

表 3. Oracle Solaris 10 のシステムパラメータのデフォルト値

資源制御項目	廃止された 調整可能な変数	以前の デフォルト値	新しい デフォルト値
process.max-msg-qbytes	msginfo_msgmnb	4096	65536
process.max-msg-messages	msginfo_msgtql	40	8192
process.max-sem-ops	seminfo_semopm	10	512
process.max-sem-nsems	seminfo_semmsl	25	512
project.max-shm-memory	shminfo_shmmax	0x800000	物理メモリーの 1/4
project.max-shm-ids	shminfo_shmmni	100	128
project.max-msg-ids	msginfo_msgmni	50	128
project.max-sem-ids	seminfo_semmni	10	128

Oracle インストールのための System V IPC パラメータの設定

表 4 に、『Oracle インストールガイド』で推奨されている /etc/system パラメータの値と対応する Oracle Solaris 資源制御項目を示します。

表 4. Oracle 10g を実行する場合のシステムパラメータの推奨値

パラメータ	オラクルの推奨値	Oracle Solaris 10 での必要の有無	資源制御項目	デフォルト値
SEMMNI (semsys:seminfo_semmni)	100	要	project.max-sem-ids	128
SEMMNS (semsys:seminfo_semmns)	1024	不要	なし	なし
SEMMSL (semsys:seminfo_semmsl)	256	要	process.max-sem-nsems	512
SHMMAX (shymsys:shminfo_shmmax)	4294967295	要	project.max-shm-memory	物理メモリーの 1/4
SHMMIN (shmsys:shminfo_shmmin)	1	不要	なし	なし
SHMMNI (shmsys:shminfo_shmmni)	100	要	project.max-shm-ids	128
SHMSEG (shmsys:shminfo_shmseg)	10	不要	なし	なし

デフォルト値はオラクルの推奨値より高いので、設定が必要な可能性がある資源制御項目は project.max-shm-memory と process.max-sem-nsems のみです。project.max-shm-memory を設定する場合、すべての Oracle Database インスタンスの SGA の合計より高くします。Oracle プロセス数を非常に多くしたい場合は (init.ora 内のプロセスパラメータで指定)、process.max-sem-nsems を高くするようにします。process.max-sem-nsems の値は、プロセスパラメータで指定する値より高くするようにしてください。

次のセクションでは、資源制御を使用して特定の値を設定するプロセスの詳細を説明します。

資源制御コマンドを使用する System V IPC パラメータの設定

prctl コマンドを使用すると、プロセス、タスク、プロジェクトを実行する資源制御項目の値を表示したり変更したりできます。prctl コマンドを -n オプション付きで呼び出して、特定の資源制御項目の値を表示します。次のコマンドは、指定されたプロセスの max-file-descriptor 資源制御項目の値を表示します。

```
# prctl -n process.max-file-descriptor pid
```

次のコマンドは、group.dba プロジェクトの project.cpu-shares の値を更新します。

```
# prctl -n project.cpu-shares -v 10 -r -i project group.dba
```

次のコマンドは、SHMMAX 値を 32G バイトに、SEMMSL 値を 4096 に設定して、再起動しても持続する Oracle プロジェクトを作成します。

```
# projadd -c "Oracle project" group.dba
# projmod -sK "project.max-shm-memory=(privileged,32G,deny)" group.dba
# projmod -sK "process.max-sem-nsems=(privileged,4096,deny)" group.dba
```

Oracle Solaris コンテナで実行する
Oracle Databases のベストプラクティス
2010 年 4 月
著者: Ritu Kamboj、Roman Ivanov

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

海外からのお問い合わせ窓口:
電話: +1.650.506.7000
ファクシミリ: +1.650.506.7200
oracle.com

Oracle is committed to developing practices and products that help protect the environment

SOFTWARE. HARDWARE. COMPLETE.