Oracle® VM Server for SPARC 3.4 開発者ガイド

Part No: E71834
2016 年 8 月

ORACLE

目次

このドキュメントの使用法

- **概要** – Oracle VM Server for SPARC テンプレート、XML インタフェース、Logical Domains Manager の検出および仮想ドメイン情報 API について説明した詳細情報と手順を提供します。
- **対象読者** – SPARC サーバー上の仮想化を管理するシステム管理者
- **必要な知識** – これらのサーバーのシステム管理者は、UNIX システムおよび Oracle Solaris オペレーティングシステム (Oracle Solaris OS) の実践的な知識を持っている必要があります

製品ドキュメントライブラリ

この製品および関連製品のドキュメントとリソースは http://www.oracle.com/technetwork/documentation/vm-sparc-194287.html で入手可能です。

フィードバック

このドキュメントに関するフィードバックを http://www.oracle.com/goto/docfeedback からお聞かせください。

♦ ♦ ♦ 第 1 章

Oracle VM Server for SPARC テンプレート、XML ファイル、およびプログラムの開発

Oracle VM Server for SPARC テンプレート、XML ファイル、およびプログラムの開発

このガイドは、次の Oracle VM Server for SPARC 機能に関する情報を提供します。

- **Oracle VM Server for SPARC テンプレートユーティリティー。**これらのユーティリティーを使用して、ドメイン構成を配備するために使用できるテンプレートを作成します。第2章「Oracle VM Server for SPARC テンプレートの使用」を参照してください。
- **Oracle VM Server for SPARC XML インタフェース。**この XML インタフェースを使用して、論理ドメインの構成を記述します。第3章「Logical Domains Manager での XML インタフェースの使用」を参照してください。
- **Logical Domains Manager の検出 API。**この API を使用して、どのドメインが Logical Domains Manager を実行しているかを検出できるプログラムを記述します。第4章「Logical Domains Manager の検出」を参照してください。
- **仮想ドメイン情報 API。**この API を使用して、仮想ドメインに関する情報を提供するプログラムを記述します。第5章「仮想ドメイン情報コマンドおよび API の使用」を参照してください。

注記 - この本で説明されている機能は、『Oracle VM Server for SPARC 3.4 インストールガイド』に記載されているすべてのサポートされるシステムソフトウェアおよびハードウェアプラットフォームで使用できます。ただし、一部の機能は、サポートされているシステムソフトウェアおよびハードウェアプラットフォームのサブセット上でしか使用できません。このような例外については、『Oracle VM Server for SPARC 3.4リリースノート』 の 「このリリースの最新情報」および What's New in Oracle VM Server for SPARC Software (`http://www.oracle.com/technetwork/server-storage/vm/documentation/sparc-whatsnew-330281.html`) を参照してください。

♦ ♦ ♦ 第 2 章

2 Oracle VM Server for SPARC テンプレートの使用

この章では、次の項目について説明します。

Oracle VM Server for SPARC テンプレートについて

Oracle VM Server for SPARC テンプレートコマンドを使用すると、SPARC システム用の Oracle VM Server for SPARC テンプレートを作成、配備、および構成できます。これらのコマンドは、ディスクイメージファイル、およびアーカイブ (`.ova`) に含まれているプロパティーの XML 記述子を含む OVF テンプレート仕様に基づいています。

これらのコマンドは、Oracle VM Server for SPARC システムの制御ドメインで実行します。コマンド行での実行に加えて、これらのコマンドは完全に自動化された方法で、またはより大きなワークフローの一部としてほかのプログラムから実行できます。

これらのテンプレートを使用して、ゲストドメイン、I/O ドメイン、およびルートドメインだけを配備および構成できます。ただし、テンプレートを使用して、`primary` ドメインでもあるルートドメインまたは I/O ドメインを配備および構成したり、サービスドメインを展開および構成したりすることはできません。

Oracle VM Server for SPARC テンプレートコマンドは次のとおりです。

- `ovmtadm` – `ovmtcreate`、`ovmtconfig`、および `ovmtdeploy` コマンドを起動することによってテンプレートを作成、構成、配備、および削除できるようにします。`ovmtadm(1M)` のマニュアルページを参照してください。
- `ovmtconfig` – プロパティーの名前と値のペアをターゲットドメイン内のアプリケーションやプロセス (Oracle Solaris OS 構成メカニズムや初回ブートスクリプト

など) に転送することによってドメイン上の構成アクションを実行します。プロパティーの名前と値のペアは、`ovmtprop` コマンドを使用して設定されます。

また、このコマンドを使用すると、ドメインの ZFS ファイルシステムをバックマウントして、制御ドメインがこれらのファイルシステム上で直接コマンドを実行するようにもできます。この方法には、ファイルをコピーしたり、Oracle Solaris OS パッケージをインストールおよびアップグレードしたり、構成アクションを実行したりする機能が含まれています。`ovmtconfig(1M)` のマニュアルページを参照してください。

- `ovmtcreate` – 既存の Oracle VM Server for SPARC ドメインからテンプレートを作成します。`ovmtcreate(1M)` のマニュアルページを参照してください。
- `ovmtdeploy` – Oracle VM Server for SPARC テンプレートからドメインを作成します。`ovmtdeploy(1M)` のマニュアルページを参照してください。
- `ovmtlibrary` – ファイルを構成し、データベース内の情報を格納、取得、および編集することによって、Oracle VM Server for SPARC テンプレートのためのデータベースおよびファイルシステムベースのリポジトリを管理します。`ovmtlibrary(1M)` のマニュアルページを参照してください。
- `ovmtprop` – テンプレートから配備されたドメイン内で Oracle Solaris OS プロパティーを表示および設定できるようにします。このプロパティーは、名前と値のペアとして指定されます。このコマンドは構成アクションや、プロパティーによって起動される可能性のあるアクションを実行するために、ほかのスクリプトやプログラムから呼び出されます。`ovmtprop(1M)` のマニュアルページを参照してください。

Oracle VM Server for SPARC テンプレートの使用を開始する方法については、次のブログを参照してください。

- テンプレートの配備 (`https://blogs.oracle.com/cmt/en/entry/getting_started_with_ovm_templates`)
- テンプレートの作成 (`https://blogs.oracle.com/cmt/en/entry/getting_started_with_ovm_templates1`)
- Oracle SuperCluster でのテンプレートの使用 (`https://blogs.oracle.com/cmt/en/entry/getting_started_with_ovm_templates2`)

Oracle VM Server for SPARC テンプレートユーティリティーのインストール

制御ドメインに Oracle VM Server for SPARC テンプレートユーティリティーのソフトウェアパッケージをインストールするには、まず Oracle VM Server for SPARC 3.4 ソフトウェアをダウンロードして制御ドメインにインストールする必要があります。『Oracle VM Server for SPARC 3.4 インストールガイド』 の 第 2 章,「ソフトウェアのインストール」を参照してください。

次に、Oracle VM Server for SPARC テンプレートユーティリティーのソフトウェアパッケージ `ovmtutils` をインストールします。

```
# pkg install -v pkg:/system/ldoms/ovmtutils
```

Oracle VM Server for SPARC テンプレートのライフサイクル

このセクションでは、テンプレート作成プロセスの各段階、実行されるアクション、および Oracle VM Server for SPARC テンプレートユーティリティーを使用してこのプロセスを支援する方法について説明します。

注記 - アプリケーションや初回ブートスクリプトを使用したテンプレートの作成および開発は繰り返しのプロセスです。ソースコード管理システムを使用してスクリプトとプロパティーを管理することにより、構成のすべての側面を同期させるように注意してください。

次に、テンプレート作成プロセスの各段階、実行されるアクション、および Oracle VM Server for SPARC テンプレートユーティリティーを使用してこのプロセスを支援する方法について説明します。

1. **テンプレートの作成。**事前に作成された汎用のテンプレートを使用できますが、既存のドメインからカスタムテンプレートを作成できます。このドメインには、必要なすべてのオペレーティングシステムコンポーネント、アプリケーションソフトウェア、およびその他のユーティリティーが完全にインストールされている必要があります。

   この環境は通常、環境を完成させるために必要なアクションがごく少数になるように、できるだけ完全に構成されます。メモリー、仮想 CPU、仮想ネットワーク、ディスクなどのすべてのドメイン設定に、必要とする配備が反映されているようにしてください。

   この段階では、1 つまたは複数の「初回ブート」スクリプトを作成します。これらのスクリプトは、指定したプロパティーに基づいて最終的な構成を実行する環境に含めます。これらのプロパティーは必ず、テンプレートごとの README ファイルに記録し、説明を加えるようにしてください。

   注記 - いずれかの初回ブートスクリプトがドメイン変数にアクセスする場合は、ゲストドメインに `ovmtprop` ユーティリティーがインストールされていることを確認してください。

   ---

2. **テンプレートの作成。**テンプレートを作成する前に、ソースドメインの環境が確実に未構成の状態であるようにして、規定されたアクション (初回ブートスクリプトの一部である場合が多い) によってあとで構成できるようにします。

たとえば、次の手順を実行します。

- アプリケーション固有の構成をすべて削除し、あとで再作成されるようにします。
- 構成ファイルのデフォルト値を使用します。
- ルートファイルシステム以外の zpool をすべてエクスポートして、新しいドメインで認識できるようにします。
- テンプレートからの配備のあとの最初のブート時に構成プロパティーを受け入れる準備ができるように、オペレーティングシステムを構成解除された状態に戻します。次のコマンドを実行することにより、サイトレベルのカスタマイズをすべて削除し、構成解除して、オペレーティングシステムを停止します。

```
# sysconfig unconfigure -g system -s --include-site-profile --destructive
```

これらの手順を実行したあと、`ovmtcreate` コマンドを実行してドメインからテンプレートを作成します。

注記 - このコマンドを実行し、配備時にプロパティーが指定されない場合、システムは次回ブート時に Oracle Solaris 対話型インストーラを実行します。

---

3. **テンプレートの名前の指定。**テンプレートを識別するには、次の形式などの整合性のある表記規則を使用します。

   technology`.`*OS*`.`*application*`.`*architecture*`.`*build*`.ova`

   たとえば、次のテンプレート名は、SPARC プラットフォーム上で Oracle Solaris 11.2 OS のビルド 2 を実行し、WebLogic サーバーのバージョン 12.1.2 を実行するドメインに対するものです: `OVM_S11.2_WLS12.1.2_SPARC_B2.ova`
4. **テンプレートの配布。**このテンプレートは、`.ova` の拡張子を持つ 1 つのファイルです。このファイルには、配備に必要な圧縮されたディスクイメージとメタデータが含まれています。このテンプレートにはまた、ペイロードファイルチェックサムのマニフェストファイルも含まれています。これは、配布以降に内容が変更されていないことを検証するための全体的なアーカイブチェックサムと結合されている場合があります。

   Web ベースのサービスを使用してテンプレートを配布することも、テンプレートを複製する代わりに中央リポジトリを保持することもできます。
5. **テンプレートの配備。**テンプレートにはソースドメインによって認識されるシステムの側面のみが取得されるため、テンプレートの配備をサポートするにはどのサービスが存在している必要があるかを理解する必要があります。

   必要なサービスには次のものが含まれます。

   - テンプレートからの仮想ネットワークが接続される可能性のある適切なインタフェースへの 1 つまたは複数の仮想スイッチ
   - 仮想ディスクサービス

- コンソールサービス
- テンプレートの要件に対応できる十分な仮想 CPU およびメモリー

`ovmtdeploy` ユーティリティーはこれらの設定の多くをオーバーライドできますが、テンプレートで提供される最小値がベースライン要件を表します。

`ovmtdeploy` ユーティリティーを使用すると、仮想ディスクを自動的に抽出し、圧縮解除して配備ディレクトリにコピーしたり、テンプレートに記述されているさまざまな仮想デバイスを構築したりできます。

この時点でドメインを起動できますが、ドメインが完全に機能するには、ドメインコンソールを使用して手動の構成手順をいくつか実行することが必要になる可能性があります。

6. **ドメインの自動的な構成。**テンプレートによって作成されたドメインの構成は、いくつかのタイプのアクションで構成されます。たとえば、初回ブートスクリプトに構成対象の情報を提供するために、プロパティーの名前と値のペアを指定することがあります。また、ドメインのファイルシステムに対して構成ファイルのコピーなどのアクションを実行するために、仮想ディスクを制御ドメインにバックマウントすることもあります。

   `ovmtconfig` ユーティリティーはこれらのドメイン構成アクティビティーを自動化し、1 つまたは複数のコマンドスクリプトやプロパティーファイルを指定することにより、実行するアクションやドメインを構成するために使用するプロパティーを指定できるようにします。

   Oracle Solaris OS を構成するために、`ovmtconfig` ユーティリティーはドメインのルートファイルシステムをバックマウントし、指定された構成スクリプトやプロパティーから `sc_profile.xml` ファイルを作成します。このプロファイルにより、Oracle Solaris OS は最初のブート時に自身を構成できます。

7. **最初の構成。**Oracle Solaris OS が正常に構成され、最初のブートが終了したら、インストールされているアプリケーションをすべて構成する必要があります。構成フェーズ中に、`ovmtconfig` ユーティリティーは、次のいずれかの方法を使用して配備対象のドメインに構成情報を渡します。

   - **ドメイン変数 –**ローカルプロパティーファイルに加えて、制御ドメインで `ovmtconfig` ユーティリティーを実行することにより、あとでゲストドメインで `ovmtprop` ユーティリティーが使用できるドメイン変数を設定できます。この方法を使用すると、構成が完了したあとに初回ブートスクリプトがこれらのプロパティーに直接アクセスし、ゲストドメインに直接構成情報を提供できます。

     たとえば、ゲストで `ovmtprop` を実行するスーパーバイザースクリプトを使用し、制御ドメインから `ovmtconfig -v` を実行することによって、ネットワークにアクセスできない構成の側面の変更を自動化できます。

   - **直接のアクション –** `ovmtconfig` ユーティリティーは、ゲストドメインのファイルシステムを制御ドメインにバックマウントし、ファイルおよびファイルシステムに対して直接のアクションを実行します。これらのアクションには、構成ファイルの作成やシステムバイナリのコピーが含まれることがあります。

これらのアクションは、ovmtconfig ユーティリティーに指定したスクリプトに記述されています。コマンドアクションを実行するスクリプトは、/opt/ovmtutils/share/scripts ディレクトリで見つけることができます。

注記 - これらのアクションは制御ドメインに影響を与える可能性があるため、このようなアクションには通常、ゲストドメイン内で実行されるように設計された初回ブートプロセスは含まれません。

実行するコマンドを指定するには、ovmtconfig -c コマンドを使用します。

注意 - ドメインにパスワードなどの機密情報を渡すために、暗号化されていないプロパティーを使用しないでください。Oracle Solaris OS を構成するために使用されるもの以外のプロパティーは、ldm 変数として平文でドメインに渡されます。これらのプロパティー値は、ldm コマンドの実行を承認されている制御ドメイン上のユーザー、および配備されたドメインにログインしているユーザーに表示されます。

この時点で、このドメインは完全に構成され、動作可能になります。

Oracle VM Server for SPARC テンプレートの例

このセクションでは、次の Oracle VM Server for SPARC テンプレートタスクの例を示します。

- 例1「Oracle VM Server for SPARC テンプレートの作成」
- 例2「Oracle VM Server for SPARC テンプレートのプロパティーの構成」
- 例3「Oracle VM Server for SPARC テンプレート の配備」
- 例4「Oracle VM Server for SPARC テンプレートライブラリの管理」

例 1 Oracle VM Server for SPARC テンプレートの作成

次の ovmtcreate コマンドは、ovmtcreate_example という名前の ldg1 ドメインに基づいたテンプレートを作成します。結果として得られるテンプレート名には .ova の接尾辞が付くことに注意してください。

```
# ovmtcreate -d ldg1 -o ovmtcreate_example
...

STAGE 1 - EXAMINING SYSTEM AND ENVIRONMENT
-------------------------------------------
Performing platform & prerequisite checks
Checking user permissions
```

```
Checking for required packages
Checking for required services
Checking directory permissions

STAGE 2 - ANALYZING DOMAIN
--------------------------
Retrieving and processing attributes
Checking domain state
Getting domain resource settings
Discovering network topology
Discovering disk topology

STAGE 3 - ARCHIVE CREATION
--------------------------
Checking destination and current directory capacity
Compressing disk image
Creating XML configuration
Calculating manifest checksums
Creating archive file
Checking archive

PROCESS COMPLETED
-----------------
Started: Tue Aug 18 15:29:14 PDT 2015
Completed: Tue Aug 18 15:41:25 PDT 2015
Elapsed time: 0:12:11
```

例 2 Oracle VM Server for SPARC テンプレートのプロパティーの構成

`ovmtconfig` および `ovmtprop` ユーティリティーを使用すると、それぞれ Oracle VM Server for SPARC テンプレートのプロパティー値と Oracle Solaris OS のプロパティー値を指定できます。

- 次の `ovmtconfig` コマンドは、`ldg1` ドメインのファイルシステムに対して直接、構成操作を実行します。

  `-c` オプションは、プロパティー値を設定するための `/opt/ovmtutils/share/scripts/ovmt_s11_scprofile.sh` スクリプトを指定します。`-p` オプションは、`com.oracle.solaris.network.ipaddr` および `com.oracle.solaris.system.computer-name` プロパティーの特定の値を指定します。

  ```
  # ovmtconfig -v -d ldg1 -f -s \
  -c /opt/ovmtutils/share/scripts/ovmt_s11_scprofile.sh \
  -p com.oracle.solaris.network.ipaddr.0=10.153.118.211,\
  com.oracle.solaris.system.computer-name=system1
  ...

  STAGE 1/7 - EXAMINING SYSTEM AND ENVIRONMENT
  --------------------------------
  Checking operating system
  Checking platform type
  Checking user permissions
  Checking packages
  Checking host domain name
  ```

```
Checking host domain type
Checking services

STAGE 2/7 - PROCESSING COMMAND LINE OPTIONS
------------------------------
Parsing individual properties
Creating consolidated properties list

STAGE 3/7 - ANALYZING TARGET DOMAIN
-----------------------
Stopping domain ldg1
Analyzing domain disk topology for domain ldg1
Discovering 1 volumes for vDisks
Examining 1 backend devices
unbinding domain ldg1
Creating 1 virtual disks for back mount
Created virtual disk 0

STAGE 4/7 - PERFORMING BACKMOUNT
--------------------
Finding Solaris device for vdisks
Importing zpools for 1 Solaris devices
Detected conflicting zpool name, attempting rename
Getting boot file system for properties in 1 zpool
Setting properties in 1 zpools
Mounting ZFS file systems
Mounting ZFS found in zpool rpool_1

STAGE 5/7 - PERFORMING ACTIONS ON TARGET DOMAIN
-----------------------------------

STAGE 6/7 - UNMOUNTING AND RESTORING DOMAIN STATE
-------------------------------------
Rolling back commands DEBUG [20150819-07:02:42]: Rolling back 8 /usr/sbin/zfs unmount -f
 rpool_1/ROOT/solaris/var
completed

STAGE 7/7 - SETTING TARGET DOMAIN ENVIRONMENT
--------------------------------
Checking 2 properties to set as domain variables
Process completed
```

- 次の `ovmtprop` コマンドは、Oracle Solaris OS のプロパティー値を指定します。

```
primary# ovmtprop set-prop com.oracle.solaris.system.computer-name=test ldg1
```

`com.oracle.solaris.system.computer-name` プロパティーの値が `test` であることを確認するには、`ldm list -l` コマンドを使用します。

```
primary# ldm list -l ldg1
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  NORM  UPTIME
ldg1             active     -n----  5000    8     8G       0.0%  0.0%  23h 23m
..
VARIABLES
    auto-boot?=true
    boot-file=-k
    pm_boot_policy=disabled=0;ttfc=2000;ttmr=0;

VMAPI TO GUEST
    com.oracle.solaris.fmri.count=0
    com.oracle.solaris.system.computer-name=test
```

例 3 Oracle VM Server for SPARC テンプレート の配備

次の `ovmtdeploy` コマンドは、`/export/ovmtdeploy` ディレクトリ内の `ovmtcreate_example.ova` Oracle VM Server for SPARC テンプレートを使用して、`ldg1` という名前のドメインを作成します。

```
# ovmtdeploy -d ldg1 -o /export/ovmtdeploy ovmtcreate_example.ova
...

STAGE 1 - EXAMINING SYSTEM AND ENVIRONMENT
------------------------------------------
Checking user privilege
Performing platform & prerequisite checks
Checking for required services
Named resourced available

STAGE 2 - ANALYZING ARCHIVE & RESOURCE REQUIREMENTS
---------------------------------------------------
Checking .ova format and contents
Validating archive configuration
Checking sufficient resources present
WARNING: Virtual switch primary-vsw0 already exists

STAGE 3 - EXTRACTING ARCHIVE
----------------------------
Extracting archive
Validating checksums
Decompressing disk image(s)

STAGE 4 - DEPLOYING DOMAIN
--------------------------
Creating domain and adding resources
Validating deployment
Domain created:
```

`ldm list` の出力は、`ldg1` という名前の新しいドメインが作成されたことを示しています。

```
# ldm list
NAME    STATE  FLAGS  CONS VCPU MEMORY UTIL NORM UPTIME
primary active -n-cv- UART 8    40G    1.4% 1.1% 6d 2h 18m
ldg1    active -n---- 5000 8    8G     41%  38%  28s
```

例 4 Oracle VM Server for SPARC テンプレートライブラリの管理

ovmtlibrary コマンドは、ファイルを構成し、データベース内の情報を格納、取得、および編集することによって、Oracle VM Server for SPARC テンプレート のための データベースおよびファイルシステムベースのリポジトリを管理します。

- 次のコマンドは、export/user1/ovmtlibrary_example 内にテンプレートライブラリを作成します。

```
# ovmtlibrary -c init -l /export/user1/ovmtlibrary_example
...

Init complete
```

- 次のコマンドは、sol-11_2-ovm-2-sparc.ova テンプレートを export/user1/ovmtlibrary_example ライブラリ内に格納します。

```
# ovmtlibrary -c store -d "ovmtlibrary example" -o http://system1.example.com/s11.2/templates/
sol-11_2-ovm-2-sparc.ova -l /export/user1/ovmtlibrary_example
...

Templates present in path "/export/user1/ovmtlibrary_example"

event id is 2

*******************************************************************************
converted 'http://system1.example.com/s11.2/templates/sol-11_2-ovm-2-sparc.ova' (646) ->
'http://system1.example.com/s11.2/templates/sol-11_2-ovm-2-sparc.ova' (UTF-8)
--2015-08-18 16:37:17-- http://system1.example.com/s11.2/templates/sol-11_2-ovm-2-
sparc.ova
Resolving system1.example.com (system1.example.com)... 10.134.127.18
Connecting to system1.example.com (system1.example.com)|10.134.127.18|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1018341888 (971M) [text/plain]
Saving to: '/export/user1/ovmtlibrary_example/repository/templates/1/1/sol-11_2-ovm-2-
sparc.ova'

/export/user1/ovmtlibrary_example/repo 100%
[=============================================================================>]
 971.17M 6.05MB/s in 5m 37s
2015-08-18 16:42:55 (2.88 MB/s) - '/export/user1/ovmtlibrary_example/repository/
templates/1/1/sol-11_2-ovm-2-sparc.ova' saved
[1018341888/1018341888]
```

```
********************************************************************************
Download complete
Extracting the ova file...
Extract complete
Decompress file System.img.gz
Store complete
```

- 次のコマンドは、export/user1/ovmtlibrary_example ライブラリの内容を一覧表示します。

```
# ovmtlibrary -c list -l /export/user1/ovmtlibrary_example
...

Templates present in path "/export/user1/ovmtlibrary_example"

ID Name                         Version Description         Date
--------------------------------------------------------------------------------
1  sol-11_2-ovm-2-sparc         1       ovmtlibrary example 2015-08-18
```

- 次のコマンドは、export/user1/ovmtlibrary_example ライブラリの詳細な一覧を表示します。

```
# ovmtlibrary -c list -i 1 -o -l /export/user1/ovmtlibrary_example
...

Templates present in path "/export/user1/ovmtlibrary_example"

ID Name                             Type Path                        Size(bytes)
--------------------------------------------------------------------------------
1  sol-11_2-ovm-2-sparc.ova         ova /export/user1/ovmtlibrary_example/repository/
templates/1/1/sol-11_2-ovm-2-sparc.ova 1018341888
2  sol-11_2-ovm-sparc.ovf           ovf /export/user1/ovmtlibrary_example/repository/
templates/1/1/sol-11_2-ovm-sparc.ovf 3532
3  sol-11_2-ovm-sparc.mf            mf /export/user1/ovmtlibrary_example/repository/
templates/1/1/sol-11_2-ovm-sparc.mf 137
4  System.img                       img /export/user1/ovmtlibrary_example/repository/
templates/1/1/System.img 21474836480
```

- 次のコマンドは、export/user1/ovmtlibrary_example ライブラリからテンプレート ID 1 を削除します。

```
# ovmtlibrary -c delete -i 1 -l /export/user1/ovmtlibrary_example
```

♦ ♦ ♦ 第 3 章

3 Logical Domains Manager での XML インタフェースの使用

この章では、外部ユーザープログラムが Oracle VM Server for SPARC ソフトウェアと直接対話できる eXtensibleMarkup Language (XML) の通信メカニズムについて説明します。ここで取り上げる基本事項は、次のとおりです。

XML トランスポート

外部プログラムは、eXtensible Messaging and Presence Protocol (XMPP – RFC 3920) を使用して、Logical Domains Manager と通信できます。XMPP は、ローカル接続とリモート接続の両方でサポートされており、デフォルトで有効です。XML インタフェースは、TLS (Transport Layer Security) プロトコルのバージョン 1.2 のみをサポートしています。

リモート接続を無効にするには、`ldmd/xmpp_enabled` SMF プロパティーを `false` に設定し、Logical Domains Manager を再起動します。

```
# svccfg -s ldom/ldmd setprop ldmd/xmpp_enabled=false
# svcadm refresh ldmd
# svcadm restart ldmd
```

注記 - XMPP サーバーを無効にすると、ドメインの移行およびメモリーの動的再構成も防止されます。

XMPP サーバー

Logical Domains Manager は、数多くの利用可能な XMPP クライアントアプリケーションおよびライブラリと通信できる XMPP サーバーを実装しています。Logical Domains Manager は次のセキュリティーメカニズムを使用しています。

- クライアントと自身の間の通信チャネルをセキュリティー保護するための Transport Layer Security。
- 認証用の Simple Authentication and Security Layer (SASL)。唯一サポートされている SASL メカニズムは `PLAIN` です。モニタリング操作や管理操作を可能にするには、サーバーが承認できるようにユーザー名およびパスワードをサーバーに送信する必要があります。

ローカル接続

Logical Domains Manager は、ユーザークライアントが LDoms Manager 自身と同じドメインで動作しているかどうかを検出し、同じドメインである場合はこのクライアントとの間で最小限の XMPP ハンドシェークを行います。具体的には、TLS を介したセキュアチャネルの設定後の SASL 認証手順がスキップされます。認証および承認は、クライアントインタフェースを実装しているプロセスの資格に基づいて行われます。

クライアントは、フル XMPP クライアントを実装することも、単に `libxml2` Simple API for XML (SAX) パーサーなどのストリーミング XML パーサーを実行することも選択できます。いずれの場合も、クライアントは XMPP ハンドシェークを TLS ネゴシエーションまで処理する必要があります。必要な手順については、XMPP の仕様を参照してください。

XML プロトコル

通信の初期化が完了すると、次に Oracle VM Server for SPARC 定義の XML メッセージが送信されます。XML メッセージの一般的な 2 つのタイプは次のとおりです。

- `<LDM_interface>` タグを使用するリクエストメッセージと応答メッセージ。このタイプの XML メッセージは、コマンドの伝達と、Logical Domains Manager からの結果の取得に使用されます。これはコマンド行インタフェース (CLI) を使用したコマンドの実行に類似しています。このタグは、イベントの登録および登録解除にも使用されます。
- `<LDM_event>` タグを使用するイベントメッセージ。このタイプの XML メッセージは、Logical Domains Manager によって送信されたイベントを非同期に報告するために使用されます。

要求メッセージと応答メッセージ

Oracle VM Server for SPARC の XML インタフェースには、次の異なる 2 つの形式があります。

- Logical Domains Manager にコマンドを送信するための形式。
- 受信メッセージのステータスおよびこのメッセージ内で要求されているアクションに基づいて Logical Domains Manager が応答するための形式。

2 つの形式の XML 構造の多くは共通していますが、両者の違いをよく理解するために、ここでは別々に取り扱います。

要求メッセージ

Logical Domains Manager への受信 XML 要求には、もっとも基本的なレベルで、1 つのオブジェクトで動作する 1 つのコマンドの記述が含まれています。要求が複雑になると、1 つのコマンドで複数のコマンドと複数のオブジェクトを処理できます。次の例に、基本的な XML コマンドの構造を示します。

例 5 1 つのオブジェクトで動作する 1 つのコマンドの形式

```
<LDM_interface version="1.3">
  <cmd>
    <action>Place command here</action>
    <options>Place options for certain commands here</options>
    <arguments>Place arguments for certain commands here</arguments>
    <data version="3.0">
      <Envelope>
        <References/>
        <!-- Note a <Section> section can be here instead of <Content>
        <Content xsi:type="ovf:VirtualSystem_Type" id="Domain name">
          <Section xsi:type="ovf:ResourceAllocationSection_type">
            <Item>
              <rasd:OtherResourceType>LDom Resource Type</rasd:OtherResourceType>
              <gprop:GenericProperty
              key="Property name">Property Value</gprop:GenericProperty>
            </Item>
          </Section>

        </Content>
      </Envelope>
    </data>

  </cmd>

</LDM_interface>
```

<LDM_interface> タグ

Logical Domains Manager に送信するすべてのコマンドは、<LDM_interface> タグで始まる必要があります。Logical Domains Manager に送信するドキュメントでは、ドキュメント内に含まれる <LDM_interface> タグは 1 つのみである必要があります。<LDM_interface> タグには、例5「1 つのオブジェクトで動作する 1 つのコマンドの形式」に示すバージョン属性が含まれている必要があります。

<cmd> タグ

ドキュメントでは、<LDM_interface> タグ内に 1 つ以上の <cmd> タグが含まれている必要があります。各 <cmd> セクションには、<action> タグを 1 つのみ含める必要があります。この <action> タグは、実行するコマンドを記述するために使用します。各 <cmd> タグに 1 つ以上の <data> タグを含めて、コマンドの処理対象のオブジェクトを記述する必要があります。

また、<cmd> タグには <options> タグも含めることができます。このタグは、一部のコマンドに関連付けられたオプションおよびフラグを指定するために使用されます。次のコマンドにはオプションが使用されます。

- `ldm remove-domain` コマンドには `-a` オプションを使用できます。
- `ldm bind-domain` コマンドには、`-f` オプションを使用できます。
- `ldm add-vdsdev` コマンドには、`-f` オプションを使用できます。
- `ldm cancel-operation` コマンドには、`migration` または `reconf` オプションを使用できます。
- `ldm add-spconfig` コマンドには、`-r` *autosave-name* オプションを使用できます。
- `ldm remove-spconfig` コマンドには、`-r` オプションを使用できます。
- `ldm list-spconfig` コマンドには、`-r` [*autosave-name*] オプションを使用できます。
- `ldm stop-domain` コマンドでは、次のタグを使用してコマンド引数を設定できます。
  - <force> は -f オプションを表します。
  - <halt> は -h オプションを表します。
  - <message> は -m オプションを表します。
  - <quick> は -q オプションを表します。
  - <reboot> は -r オプションを表します。
  - <timeout> は -t オプションを表します。

  このタグにはコンテンツ値を指定できません。ただし、-t および -m オプションには、null 以外の値 (<timeout>10</timeout> や <message>Shutting down now</message> など) を指定する必要があります。

次の XML の例は、リブートメッセージを含むリブートリクエストを `ldm stop-domain` コマンドに渡す方法を示しています。

```
<action>stop-domain</action>
<arguments>
    <reboot/>
    <message>my reboot message</message>
</arguments>
```

<data> タグ

各 <data> セクションには、指定したコマンドに関連するオブジェクトの記述を含めます。<data> セクションの形式は、Open Virtualization Format (OVF) ドラフト仕様の XML スキーマ部分に基づいています。このスキーマは、<References> タグ (Oracle VM Server for SPARC では未使用)、<Content> セクション、および <Section> セクションを含む <Envelope> セクションを定義します。

Oracle VM Server for SPARC の場合、<Content> セクションは、特定のドメインを指定および記述するために使用されます。<Content> ノードの id= 属性に指定するドメイン名で、ドメインが識別されます。<Content> セクション内には、特定のコマンドの必要に応じて、ドメインのリソースを記述するための <Section> セクションが 1 つ以上あります。

ドメイン名を指定するだけの場合は、<Section> タグを使用する必要はありません。逆に、コマンドでドメイン識別子が不要な場合は、そのコマンドで必要となるリソースを記述した <Section> セクションを、<Content> セクションの外側で、<Envelope> セクションの内側に配置する必要があります。

オブジェクト情報が推測可能な場合は、<data> セクションに <Envelope> タグを含める必要はありません。この状況は主に、ある処理に該当するすべてのオブジェクトのモニタリング要求、イベントの登録および登録解除の要求に当てはまります。

2 つの追加の OVF タイプによって、OVF 仕様のスキーマを使用して、すべてのタイプのオブジェクトを適切に定義できます。

- <gprop:GenericProperty> タグ
- <Binding> タグ

<gprop:GenericProperty> タグは、OVF 仕様に定義がないオブジェクトのプロパティーを扱います。プロパティー名はノードの key= 属性に定義され、プロパティーの値はノードの内容になります。<binding> タグは、ほかのリソースにバインドされたリソースを定義するために、`ldm list-bindings` コマンド出力で使用されます。

応答メッセージ

送信 XML 応答は、含まれているコマンドおよびオブジェクトに関して受信要求と厳密に一致した構造を持ちますが、そのほかに、指定されている各オブジェクトおよび各コマンド用の <Response> セクションと、要求に対する全体の <Response> セクションが追加されています。<Response> セクションはステータスおよびメッセージ情報を示します。次の例に、基本的な XML 要求への応答の構造を示します。

例 6 1 つのオブジェクトで動作する 1 つのコマンドに対する応答の形式

```
<LDM_interface version="1.3">
  <cmd>
    <action>Place command here</action>
    <data version="3.0">
      <Envelope>
        <References/>
        <!-- Note a <Section> section can be here instead of <Content>
        <Content xsi:type="ovf:VirtualSystem_Type" id="Domain name">
          <Section xsi:type="ovf:ResourceAllocationSection_type">
            <Item>
              <rasd:OtherResourceType>
                LDom Resource Type
              </rasd:OtherResourceType>
              <gprop:GenericProperty
              key="Property name">
                Property Value
            </gprop:GenericProperty>
            </Item>
          </Section>
          <!-- Note: More <Section>
        </Content>
      </Envelope>
      <response>
        <status>success or failure</status>
        <resp_msg>Reason for failure</resp_msg>
      </response>
    </data>

    <response>
      <status>success or failure</status>
      <resp_msg>Reason for failure</resp_msg>
    </response>
  </cmd>

  <response>
    <status>success or failure</status>
    <resp_msg>Reason for failure</resp_msg>
  </response>
</LDM_interface>
```

全体の応答

この <response> セクションは、<LDM_interface> セクションの直下の子であり、要求全体の成功または失敗を示します。受信 XML ドキュメントが不正な形式でない

かぎり、<response> セクションには、<status> タグだけが含まれます。この応答ステータスが成功を示している場合、すべてのオブジェクトに対するすべてのコマンドが成功しています。この応答ステータスが失敗を示し、<resp_msg> タグがない場合は、元の要求内のコマンドのいずれかが失敗しています。<resp_msg> タグは、XML ドキュメント自体の問題を記述する場合にのみ使用されます。

コマンドの応答

<cmd> セクションの下にある <response> セクションは、特定のコマンドの成功または失敗についてユーザーに通知します。<status> タグは、このコマンドが成功したか失敗したかを示します。全体の応答の場合と同様に、コマンドが失敗した場合で、要求の <cmd> セクションの内容の形式が不正なときは、<response> セクションには <resp_msg> タグのみが含まれます。それ以外の場合の失敗ステータスは、コマンドが実行されたオブジェクトのいずれかが原因で失敗したことを示しています。

オブジェクトの応答

最後に、<cmd> セクション内の各 <data> セクションにも、<response> セクションがあります。このセクションでは、この特定のオブジェクトで実行されたコマンドが成功したか失敗したかを示します。応答のステータスが SUCCESS の場合、<response> セクション内に <resp_msg> タグはありません。ステータスが FAILURE の場合、そのオブジェクトでのコマンドの実行時に発生したエラーに応じて、<response> フィールドには 1 つ以上の <resp_msg> タグがあります。オブジェクトエラーは、コマンドの実行時に検出された問題、または不正な形式または不明なオブジェクトが原因で発生する可能性があります。

<response> セクションのほかに、<data> セクションにその他の情報が含まれていることがあります。この情報は、受信 <data> フィールドと同じ形式で、失敗の原因となったオブジェクトを記述しています。25 ページの「<data> タグ」を参照してください。この追加情報は、次の場合に特に有用です。

- コマンドの実行が、特定の <data> セクションに対して失敗したが、別の <data> セクションに対しては成功した場合
- 空の <data> セクションがコマンドに渡されて、一部のドメインでは実行に失敗したが、ほかのドメインでは成功した場合

イベントメッセージ

ポーリングの代わりに、特定の状態変化が発生した場合にイベント通知を受信するように登録できます。個々に、または一括して登録できるイベントのタイプは 3 つあります。詳細は、29 ページの「イベントタイプ」 を参照してください。

登録および登録解除

イベントを登録するには、<LDM_interface> メッセージを使用します。24 ページの「<LDM_interface> タグ」を参照してください。<action> タグには登録または登録解除するイベントのタイプを記述し、<data> セクションは空白のままにしておきます。

例 7 イベントの登録要求メッセージの例

```
<LDM_interface version="1.3">
  <cmd>
    <action>reg-domain-events</action>
    <data version="3.0"/>
  </cmd>
</LDM_interface>
```

Logical Domains Manager は、登録または登録解除が成功したかどうかを示す <LDM_interface> 応答メッセージで応答します。

例 8 イベントの登録応答メッセージの例

```
<LDM_interface version="1.3">
  <cmd>
    <action>reg-domain-events</action>
    <data version="3.0"/>
      <response>
        <status>success</status>
      </response>
    </data>
    <response>
      <status>success</status>
    </response>
  </cmd>
  <response>
    <status>success</status>
  </response>
</LDM_interface>
```

各タイプのイベントの処理文字列は、イベントサブセクションにリストされます。

<LDM_event> メッセージ

イベントメッセージの形式は受信 <LDM_interface> メッセージと同じですが、このメッセージの開始タグは <LDM_event> になる点が異なります。メッセージの <action> タグは、イベントをトリガーするために実行されたアクションです。メッセージの <data> セクションにはイベントに関連付けられたオブジェクトが記述されます。詳細は、発生したイベントのタイプによって異なります。

例 9 <LDM_event> 通知の例

```
<LDM_event version='1.1'>
  <cmd>
    <action>Event command here</action>
    <data version='3.0'>
      <Envelope
        <References/>
        <Content xsi:type='ovf:VirtualSystem_Type' ovf:id='ldg1'/>
          <Section xsi:type="ovf:ResourceAllocationSection_type">
            <Item>
              <rasd:OtherResourceType>LDom Resource Type</rasd:OtherResourceType>
              <gprop:GenericProperty
              key="Property name">Property Value</gprop:GenericProperty>
            </Item>
          </Section>
      </Envelope>
    </data>
  </cmd>
</LDM_event>
```

イベントタイプ

次のイベントタイプを登録できます。

- ドメインイベント
- ハードウェアイベント
- 進捗イベント
- リソースイベント

すべてのイベントが ldm サブコマンドに対応しています。

ドメインイベント

ドメインイベントは、ドメインに直接実行できるアクションを記述します。次のドメインイベントを、<LDM_event> メッセージの <action> タグに指定することができます。

- add-domain
- bind-domain
- domain-reset
- migrate-domain
- panic-domain
- remove-domain
- start-domain
- stop-domain
- unbind-domain

これらのイベントでは、常に、OVF <data> セクションに、イベントが発生したドメインを記述する <Content> タグのみが含まれます。ドメインイベントを登録するには、<action> タグを reg-domain-events に設定した <LDM_interface> メッセージを送信します。これらのイベントを登録解除するには、<action> タグを unreg-domain-events に設定した <LDM_interface> メッセージを送信します。

ハードウェアイベント

ハードウェアイベントは、物理的なシステムハードウェアの変更に関係しています。Oracle VM Server for SPARC ソフトウェアの場合、ハードウェア変更は、サービスプロセッサ (SP) 構成の追加、削除、または設定を行う場合の SP への変更だけです。現在、このタイプのイベントは次の 3 つだけです。

- add-spconfig
- set-spconfig
- remove-spconfig

ハードウェアイベントでは、常に、OVF <data> セクションにイベントが発生している SP 構成を記述する <Section> タグのみが含まれます。これらのイベントを登録するには、<action> タグを reg-hardware-events に設定した <LDM_interface> メッセージを送信します。これらのイベントを登録解除するには、<action> タグを unreg-hardware-events に設定した <LDM_interface> メッセージを送信します。

進捗イベント

進捗イベントは、ドメインの移行など、長時間にわたって実行されるコマンドに対して発行されます。このイベントは、コマンド実行期間中のそれまでの進捗量を報告します。この時点では、migration-process イベントのみが報告されます。

進捗イベントでは、常に、OVF <data> セクションにイベントの影響を受ける SP 構成が記述された <Section> タグのみが含まれます。これらのイベントを登録するには、<action> タグを reg-hardware-events に設定した <LDM_interface> メッセージを送信します。これらのイベントを登録解除するには、<action> タグを unreg-hardware-events に設定した <LDM_interface> メッセージを送信します。

進捗イベントの <data> セクションは、影響を受けるドメインを記述する <content> セクションによって構成されています。この <content> セクションでは、ldom_info <Section> タグを使用して進捗を更新します。次の汎用プロパティーが ldom_info セクションに表示されます。

- --progress – コマンドの進捗の割合
- --status – コマンドのステータス。ongoing、failed、または done のいずれか
- --source – 進捗を報告しているマシン

リソースイベント

任意のドメインでリソースを追加、削除、または変更すると、リソースイベントが発生します。これらの一部のイベントの<data> セクションには、OVF <data> セクションにサービス名が示されている <Section> タグがある、<Content> タグが含まれています。

次のイベントを、<LDM_event> メッセージの <action> タグに指定することができます。

- add-vdiskserverdevice
- remove-vdiskserverdevice
- set-vdiskserverdevice
- remove-vdiskserver
- set-vconscon
- remove-vconscon
- set-vswitch
- remove-vswitch

次のリソースイベントでは、常に、OVF <data> セクションに、イベントの発生したドメインが記述された <Content> タグのみが含まれます。

- add-vcpu

- add-crypto
- add-memory
- add-io
- add-variable
- add-vconscon
- add-vdisk
- add-vdiskserver
- add-vnet
- add-vsan
- add-vswitch
- remove-crypto
- remove-io
- remove-memory
- remove-variable
- remove-vcpu
- remove-vdisk
- remove-vnet
- set-crypto
- set-memory
- set-variable
- set-vconsole
- set-vcpu
- set-vdisk
- set-vnet

リソースイベントを登録するには、<action> タグを reg-resource-events に設定した <LDM_interface> メッセージを送信します。これらのイベントの登録を解除するには、<action> タグを unreg-resource-events に設定した <LDM_interface> メッセージが必要です。

すべてのイベント

各イベントを個別に登録しないで、3 つのタイプすべてのイベントを待機するように登録することもできます。3 タイプすべてのイベントを同時に登録するには、<action> タグを reg-all-events に設定した <LDM_interface> メッセージを送信します。これらのイベントの登録を解除するには、<action> タグを unreg-all-events に設定した <LDM_interface> メッセージが必要です。

Logical Domains Manager のアクション

<action> タグで指定されたコマンドは、--events コマンドを除き、ldm コマンド行インタフェースのコマンドに対応しています。ldm サブコマンドの詳細については、ldm(1M)マニュアルページを参照してください。

注記 - XML インタフェースは、Logical Domains Manager CLI でサポートされている動詞またはコマンドの別名はサポートしていません。

<action> タグでサポートされている文字列は、次のとおりです。

- add-domain
- add-io
- add-mau
- add-memory
- add-spconfig
- add-variable
- add-vconscon
- add-vcpu
- add-vdisk
- add-vdiskserver
- add-vdiskserverdevice
- add-vhba
- add-vnet
- add-vsan
- add-vswitch
- bind-domain
- cancel-operation
- list-bindings
- list-constraints
- list-dependencies
- list-devices
- list-domain
- list-hba
- list-rsrc-group
- list-services
- list-spconfig

- list-variable
- list-vmapi
- migrate-domain
- reg-all-events
- reg-domain-events
- reg-hardware-events
- reg-resource-events
- remove-domain
- remove-io
- remove-mau
- remove-memory
- remove-reconf
- remove-spconfig
- remove-variable
- remove-vconscon
- remove-vcpu
- remove-vdisk
- remove-vdiskserver
- remove-vdiskserverdevice
- remove-vhba
- remove-vmapi
- remove-vnet
- remove-vsan
- remove-vswitch
- rescan-vhba
- set-domain
- set-mau
- set-memory
- set-spconfig
- set-variable
- set-vconscon
- set-vconsole
- set-vcpu
- set-vhba
- set-vmapi
- set-vnet
- set-vswitch

- start-domain
- stop-domain
- unbind-domain
- unreg-all-events
- unreg-domain-events
- unreg-hardware-events
- unreg-resource-events

Logical Domains Manager のリソースとプロパティー

このセクションでは、Logical Domains Manager リソースの例と、それらの各リソースに定義できるプロパティーについて説明します。XML の例では、リソースおよびプロパティーは**太字**で示されています。これらの例は、バインド出力ではなくリソースを示しています。制約出力は、ドメイン移行の出力を除いて、Logical Domains Manager のアクションの入力を作成する場合に使用できます。50 ページの「ドメインの移行」を参照してください。各リソースは、<Section> の OVF セクションで定義され、<rasd:OtherResourceType> タグによって指定されます。

ドメインの情報 (ldom_info) リソース

次の例に、ldom_info リソースのオプションのプロパティーを示します。

例 10 ldom_info の XML 出力の例

次の例に、uuid、hostid、Address など、複数の ldom_info プロパティーに指定された値を示します。

```
<Envelope>
  <References/>
  <Content xsi:type="ovf:VirtualSystem_Type" id="primary">
    <Section xsi:type="ovf:ResourceAllocationSection_type">
      <Item>
        <rasd:OtherResourceType>ldom_info</rasd:OtherResourceType>
        <uuid>c2c3d93b-a3f9-60f6-a45e-f35d55c05fb6</uuid>
        <rasd:Address>00:03:ba:d8:ba:f6</rasd:Address>
        <gprop:GenericProperty key="hostid">83d8baf6</gprop:GenericProperty>
        <gprop:GenericProperty key="master">plum</gprop:GenericProperty>
        <gprop:GenericProperty key="failure-policy">reset</gprop:GenericProperty>
        <gprop:GenericProperty key="extended-mapin-space">on</gprop:GenericProperty>
        <gprop:GenericProperty key="progress">45%</gprop:GenericProperty>
        <gprop:GenericProperty key="status">ongoing</gprop:GenericProperty>
        <gprop:GenericProperty key="source">system1</gprop:GenericProperty>
```

```
        <gprop:GenericProperty key="rc-add-policy"></gprop:GenericProperty>
        <gprop:GenericProperty key="perf-counters">global</gprop:GenericProperty>
      </Item>
    </Section>
  </Content>
</Envelope>
```

ldom_info リソースは、<Content> セクション内に必ず含まれます。ldom_info リソース内の次のプロパティーは、オプションです。

- <uuid> タグ。ドメインの UUID を指定します。
- <rasd:Address> タグ。ドメインに割り当てる MAC アドレスを指定します。
- <gprop:GenericProperty key="extended-mapin-space"> タグ。これは、ドメインの拡張されたマップイン領域を有効 (on) にするか、または無効 (off) にするかを指定します。デフォルト値は off です。
- <gprop:GenericProperty key="failure-policy"> タグ。マスタードメインに障害が発生した場合のスレーブドメインの動作を指定します。デフォルト値は ignore です。次に、有効なプロパティー値を示します。
  - ignore は、マスタードメインの障害を無視します。スレーブドメインは影響を受けません。
  - panic は、マスタードメインに障害が発生した場合、すべてのスレーブドメインにパニックを発生させます。
  - reset は、マスタードメインに障害が発生した場合、すべてのスレーブドメインをリセットします。
  - stop は、マスタードメインに障害が発生した場合、すべてのスレーブドメインを停止します。
- <gprop:GenericProperty key="hostid"> タグ。ドメインに割り当てるホスト ID を指定します。
- <gprop:GenericProperty key="master"> タグ。最大 4 つのマスタードメイン名をコンマで区切って指定します。
- <gprop:GenericProperty key="progress"> タグ。コマンドの進捗の割合を指定します。
- <gprop:GenericProperty key="source"> タグ。コマンドの進捗を報告するマシンを指定します。
- <gprop:GenericProperty key="status"> タグ。コマンドのステータス (done、failed、または ongoing) を指定します。
- <gprop:GenericProperty key="rc-add-policy"> タグは、特定のドメインに追加される可能性のあるルートコンプレックスで、直接 I/O および SR-IOV I/O 仮想化操作を有効または無効にするかどうかを指定します。有効な値は、iov および値なし (rc-add-policy=) です。
- アクセスするパフォーマンスレジスタセット (global、htstrand、strand) を指定する、<gprop:GenericProperty key="perf-counters"> タグ。

  プラットフォームにパフォーマンスアクセス機能がない場合、perf-counters プロパティー値は無視されます。

CPU (cpu) リソース

cpu リソースの割り当て単位プロパティーの <rasd:AllocationUnits> では、コアの数ではなく仮想 CPU の数を常に指定します。

cpu リソースは、<Content> セクション内に必ず含まれます。

例 11 ldm list-bindings コマンドによる cpu XML セクションの出力

次の例では、ldm list-bindings コマンドの使用による <cpu> セクションの XML 出力を示します。

```
<?xml version="1.0"?>
<LDM_interface
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ovf="./schemas/envelope"
xmlns:rasd="./schemas/CIM_ResourceAllocationSettingData"
xmlns:vssd="./schemas/CIM_VirtualSystemSettingData"
xmlns:gprop="./schemas/GenericProperty"
xmlns:bind="./schemas/Binding"
version="1.3"
xsi:noNamespaceSchemaLocation="./schemas/combined-v3.xsd">
  <cmd>
    <action>list-bindings</action>
    <data version="3.0">
      <Envelope>
        <References/>
        <Content xsi:type="ovf:VirtualSystem_Type" ovf:id="primary">
          <Section xsi:type="ovf:ResourceAllocationSection_Type">
            <Item>
              <rasd:OtherResourceType>ldom_info</rasd:OtherResourceType>
              <uuid>1e04cdbd-472a-e8b9-ba4c-d3eee86e7725</uuid>
              <rasd:Address>00:21:28:f5:11:6a</rasd:Address>
              <gprop:GenericProperty key="hostid">0x8486632a</gprop:GenericProperty>
              <failure-policy>fff</failure-policy>
              <wcore>0</wcore>
              <extended-mapin-space>0</extended-mapin-space>
              <cpu-arch>native</cpu-arch>
              <rc-add-policy/>
              <gprop:GenericProperty key="state">active</gprop:GenericProperty>
            </Item>
          </Section>
          <Section xsi:type="ovf:VirtualHardwareSection_Type">
            <Item>
              <rasd:OtherResourceType>cpu</rasd:OtherResourceType>
              <rasd:AllocationUnits>8</rasd:AllocationUnits>
              <bind:Binding>
                <Item>
                  <rasd:OtherResourceType>cpu</rasd:OtherResourceType>
                  <gprop:GenericProperty key="vid">0</gprop:GenericProperty>
                  <gprop:GenericProperty key="pid">0</gprop:GenericProperty>
                  <gprop:GenericProperty key="cid">0</gprop:GenericProperty>
                  <gprop:GenericProperty key="strand_percent">100</gprop:GenericProperty>
                  <gprop:GenericProperty key="util_percent">1.1%</gprop:GenericProperty>
                  <gprop:GenericProperty key="normalized_utilization">0.1%</gprop:
GenericProperty>
                </Item>
```

```
          </Section>
        </Content>
      </Envelope>
    </data>
  </cmd>
</LDM_interface>
```

例 12 `ldm list-domain` コマンドによる `cpu` XML セクションの出力

次の例では、`ldm list-domain` コマンドの使用による `<cpu>` セクションの XML 出力を示します。

```
<?xml version="1.0"?>
<LDM_interface
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ovf="./schemas/envelope"
xmlns:rasd="./schemas/CIM_ResourceAllocationSettingData"
xmlns:vssd="./schemas/CIM_VirtualSystemSettingData"
xmlns:gprop="./schemas/GenericProperty"
xmlns:bind="./schemas/Binding"
version="1.3"
xsi:noNamespaceSchemaLocation="./schemas/combined-v3.xsd">
  <cmd>
    <action>list-domain</action>
    <data version="3.0">
      <Envelope>
        <References/>
        <Content xsi:type="ovf:VirtualSystem_Type" ovf:id="primary">
          <Section xsi:type="ovf:ResourceAllocationSection_Type">
            <Item>
              <rasd:OtherResourceType>ldom_info</rasd:OtherResourceType>
              <gprop:GenericProperty key="state">active</gprop:GenericProperty>
              <gprop:GenericProperty key="flags">-n-cv-</gprop:GenericProperty>
              <gprop:GenericProperty key="utilization">0.7%</gprop:GenericProperty>
              <gprop:GenericProperty key="uptime">3h</gprop:GenericProperty>
              <gprop:GenericProperty key="normalized_utilization">0.1%</gprop:
GenericProperty>
            </Item>
          </Section>
        </Content>
      </Envelope>
    </data>
  </cmd>
</LDM_interface>
```

MAU (`mau`) リソース

`mau` リソースは、`<Content>` セクション内に必ず含まれます。プロパティーは `<rasd:AllocationUnits>` タグのみで、MAU またはその他の暗号化装置の数を指定します。

注記 - `mau` リソースは、サポートされているサーバーでサポートする暗号化装置です。現在、モジュラー演算ユニット (Modular Arithmetic Unit、MAU) と Control Word Queue (CWQ) の 2 つの暗号化装置がサポートされています。

例 13 mau の XML の例

```
<Envelope>
  <References/>
  <Content xsi:type="ovf:VirtualSystem_Type" id="ldg1">
    <Section xsi:type="ovf:VirtualHardwareSection_Type">
      <Item>
        <rasd:OtherResourceType>mau</rasd:OtherResourceType>
        <rasd:AllocationUnits>1</rasd:AllocationUnits>
      </Item>
    </Section>
  </Content>
</Envelope>
```

メモリー (memory) リソース

メモリーリソースは、<Content> セクション内に必ず含まれます。プロパティーは <rasd:AllocationUnits> タグのみで、メモリーの量を指定します。

例 14 memory の XML の例

```
<Envelope>
  <References/>
  <Content xsi:type="ovf:VirtualSystem_Type" id="ldg1">
    <Section xsi:type="ovf:VirtualHardwareSection_Type">
      <Item>
        <rasd:OtherResourceType>memory</rasd:OtherResourceType>
        <rasd:AllocationUnits>4G</rasd:AllocationUnits>
      </Item>
    </Section>
  </Content>
</Envelope>
```

仮想 SAN (vsan) リソース

仮想 SAN (vsan) リソースは、<Content> セクション内に存在する場合があります。これは、次のキーを持つ <gprop:GenericProperty> タグを使用する必要があります。

- service_name – 仮想 SAN の名前
- vsan_iport – 仮想 SAN のイニシエータポート

vsan_volume セクション内に 1 つの vol_name プロパティーを含む項目が存在する必要があります。この vol_name プロパティーは、* に設定されている必要があります。

例 15 vsan の XML の例

```
<Envelope>
    <References/>
    <Section xsi:type="ovf:VirtualHardwareSection_Type">
        <Item>
            <rasd:OtherResourceType>vsan</rasd:OtherResourceType>
            <gprop:GenericProperty key="service_name">vs1</gprop:GenericProperty>
            <gprop:GenericProperty key="vsan_iport">
            /pci@400/pci@1/pci@0/pci@8/SUNW,emlxs@0/fp@0,0</gprop:GenericProperty>
            <bind:Binding>
                <Item>
                    <rasd:OtherResourceType>vsan_volume</rasd:OtherResourceType>
                    <gprop:GenericProperty key="vol_name">*</gprop:GenericProperty>
                </Item>
            </bind:Binding>
        </Item>
    </Section>
</Envelope>
```

仮想ディスクサーバー (vds) リソース

仮想ディスクサーバー (vds) リソースは、ドメイン記述の一部として <Content> セクションに含まれることも、単独で <Envelope> セクションに記述されることもあります。プロパティーは <gprop:GenericProperty> タグのみです。このタグには、"service_name" というキーがあり、記述される vds リソースの名前が含まれています。

例 16 vds の XML の例

```
<Envelope>
  <References/>
  <Content xsi:type="ovf:VirtualSystem_Type" id="ldg1">
    <Section xsi:type="ovf:VirtualHardwareSection_Type">
      <Item>
        <rasd:OtherResourceType>vds</rasd:OtherResourceType>
        <gprop:GenericProperty
          key="service_name">vdstmp</gprop:GenericProperty>
      </Item>
    </Section>
  </Content>
</Envelope>
```

仮想ディスクサーバーボリューム (vds_volume) リソース

vds_volume リソースは、ドメイン記述の一部として <Content> セクションに含まれることも、単独で <Envelope> セクションに記述されることもあります。次のキーを持つ <gprop:GenericProperty> タグが必要です。

- vol_name – ボリュームの名前
- service_name – このボリュームをバインドする仮想ディスクサーバーの名前
- block_dev – このボリュームに関連付けるファイルまたはデバイスの名前

オプションで、vds_volume リソースに次のプロパティーも設定できます。

- vol_opts – {ro,slice,excl} のように、これらの項目の 1 つ以上がコンマで区切られて、1 つの文字列となっているもの
- mpgroup – マルチパス (フェイルオーバー) グループの名前

例 17 vds_volume の XML の例

```
<Envelope>
  <References/>
    <Section xsi:type="ovf:VirtualHardwareSection_Type">
      <Item>
        <rasd:OtherResourceType>vds_volume</rasd:OtherResourceType>
        <gprop:GenericProperty key="vol_name">vdsdev0</gprop:GenericProperty>
        <gprop:GenericProperty key="service_name">primary-vds0</gprop:GenericProperty>
        <gprop:GenericProperty key="block_dev">
          opt/SUNWldm/domain_disks/testdisk1</gprop:GenericProperty>
        <gprop:GenericProperty key="vol_opts">ro</gprop:GenericProperty>
        <gprop:GenericProperty key="mpgroup">mpgroup-name</gprop:GenericProperty>
      </Item>
    </Section>
</Envelope>
```

ディスク (disk) リソース

disk リソースは、<Content> セクション内に必ず含まれます。次のキーを持つ <gprop:GenericProperty> タグが必要です。

- vdisk_name – 仮想ディスクの名前
- service_name – この仮想ディスクをバインドする仮想ディスクサーバーの名前
- vol_name – この仮想ディスクを関連付ける仮想ディスクサービスデバイス

オプションで、disk リソースに timeout プロパティーも含めることができます。このプロパティーは、仮想ディスククライアント (vdc) と仮想ディスクサーバー (vds)

の間に接続を確立するためのタイムアウト値です (秒単位)。複数の仮想ディスク (vdisk) パスがある場合、vdc は、別の vds への接続を試みることができます。タイムアウトにより、すべての vds への接続が、指定した時間内に確立されます。

例 18 disk の XML の例

```
<Envelope>
  <References/>
  <Content xsi:type="ovf:VirtualSystem_Type" id="ldg1">
    <Section xsi:type="ovf:VirtualHardwareSection_Type">
      <Item>
        <rasd:OtherResourceType>disk</rasd:OtherResourceType>
        <gprop:GenericProperty key="vdisk_name">vdisk0</gprop:GenericProperty>
        <gprop:GenericProperty
          key="service_name">primary-vds0</gprop:GenericProperty>
        <gprop:GenericProperty key="vol_name">vdsdev0</gprop:GenericProperty>
        <gprop:GenericProperty key="timeout">60</gprop:GenericProperty>
      </Item>
    </Section>
  </Content>
</Envelope>
```

仮想スイッチ (vsw) リソース

vsw リソースは、ドメイン記述の一部として <Content> セクションに含まれることも、単独で <Envelope> セクションに記載されることもあります。それには、service_name キーがある <gprop:GenericProperty> タグが必要です。その名前が仮想スイッチに割り当てられます。

オプションで、vsw リソースに次のプロパティーも設定できます。

- <rasd:Address> – MAC アドレスを仮想スイッチに割り当てます。
- default-vlan-id – 仮想ネットワークデバイスまたは仮想スイッチをメンバーにする必要があるデフォルトの仮想ローカルエリアネットワーク (VLAN) を、タグ付きモードで指定します。最初の VLAN ID (*vid1*) は、この default-vlan-id に予約されています。
- dev_path – この仮想スイッチに関連付けるネットワークデバイスのパス
- id= – 新しい仮想スイッチデバイスの ID を指定します。デフォルトでは ID 値は自動的に生成されるため、OS で既存のデバイス名に一致させる必要がある場合に、このプロパティーを設定します。
- inter_vnet_link – inter-vnet 通信用の LDC チャネルを割り当てるかどうかを指定します。有効な値は、on、off、および auto です。デフォルト値は auto です。
- linkprop – 仮想デバイスが物理リンク状態の更新 (phys-state のデフォルト値) を取得するように指定します。値が空白の場合、仮想デバイスは物理リンクステータスの更新を取得しません。
- mode – Oracle Solaris Cluster のハートビートサポートの場合は sc。

- pvid – ポート仮想ローカルエリアネットワーク (VLAN) 識別子 (ID)。仮想ネットワークをメンバーにする必要のある VLAN をタグなしモードで指定します。
- mtu – 仮想スイッチと、その仮想スイッチにバインドされている仮想ネットワークデバイス、またはその両方に最大転送単位 (Maximum Transmission Unit、MTU) を指定します。有効な値の範囲は 1500 - 16000 です。無効な値を指定すると、ldm コマンドでエラーが発生します。
- vid – 仮想ローカルエリアネットワーク (VLAN) 識別子 (ID)。仮想ネットワークおよび仮想スイッチをメンバーにする必要のある VLAN をタグ付きモードで指定します。

例 19 vsw の XML の例

```
<Envelope>
  <References/>
  <Content xsi:type="ovf:VirtualSystem_Type" id="ldg2">
    <Section xsi:type="ovf:VirtualHardwareSection_Type">
      <Item>
        <rasd:OtherResourceType>vsw</rasd:OtherResourceType>
        <rasd:Address>00:14:4f:fb:ec:00</rasd:Address>
        <gprop:GenericProperty key="service_name">test-vsw1</gprop:GenericProperty>
        <gprop:GenericProperty key="inter_vnet_link">auto</gprop:GenericProperty>
        <gprop:GenericProperty key="default-vlan-id">1</gprop:GenericProperty>
        <gprop:GenericProperty key="pvid">1</gprop:GenericProperty>
        <gprop:GenericProperty key="mtu">1500</gprop:GenericProperty>
        <gprop:GenericProperty key="dev_path">switch@0</gprop:GenericProperty>
        <gprop:GenericProperty key="id">0</gprop:GenericProperty>
      </Item>
    </Section>
  </Content>
</Envelope>
```

ネットワーク (network) リソース

network リソースは、<Content> セクション内に必ず含まれます。次のキーを持つ <gprop:GenericProperty> タグが必要です。

- linkprop – 仮想デバイスが物理リンク状態の更新 (phys-state のデフォルト値) を取得するように指定します。値が空白の場合、仮想デバイスは物理リンクステータスの更新を取得しません。
- vnet_name – 仮想ネットワーク (vnet) の名前
- service_name – この仮想ネットワークをバインドする仮想スイッチ (vswitch) の名前
- custom – 信頼されるホストから仮想ネットワークデバイスに割り当てることのできる VLAN および MAC アドレスの最大数のカスタム設定を有効または無効のどちらにするかを指定します。デフォルト値は disable です。
- custom/max-mac-addrs – 信頼されるホストから仮想ネットワークデバイスに割り当てることのできる MAC アドレスの最大数を指定します。デフォルト値は 4096 です。

- `custom/max-vlans` – 信頼されるホストから仮想ネットワークデバイスに割り当てることのできる VLAN の最大数を指定します。デフォルト値は 4096 です。

オプションで、`network` リソースに次のプロパティーも設定できます。

- `<rasd:Address>` – MAC アドレスを仮想スイッチに割り当てます。
- `pvid` – ポート仮想ローカルエリアネットワーク (VLAN) 識別子 (ID)。仮想ネットワークをメンバーにする必要のある VLAN をタグなしモードで指定します。
- `vid` – 仮想ローカルエリアネットワーク (VLAN) 識別子 (ID)。仮想ネットワークおよび仮想スイッチをメンバーにする必要のある VLAN をタグ付きモードで指定します。

例 20 `network` の XML の例

```
<Envelope>
  <References/>
  <Content xsi:type="ovf:VirtualSystem_Type" id="ldg1">
    <Section xsi:type="ovf:VirtualHardwareSection_Type">
      <Item>
        <rasd:OtherResourceType>network</rasd:OtherResourceType>
        <gprop:GenericProperty key="linkprop">phys-state</gprop:GenericProperty>
        <gprop:GenericProperty key="custom">enable</gprop:GenericProperty>
        <gprop:GenericProperty key="custom/max-mac-addrs">4096</gprop:GenericProperty>
        <gprop:GenericProperty key="custom/max-vlans">12</gprop:GenericProperty>
        <gprop:GenericProperty key="vnet_name">ldg1-vnet0</gprop:GenericProperty>
        <gprop:GenericProperty
          key="service_name">primary-vsw0</gprop:GenericProperty>
        <rasd:Address>00:14:4f:fc:00:01</rasd:Address>
      </Item>
    </Section>
  </Content>
</Envelope>
```

仮想コンソール端末集配信装置 (vcc) リソース

`vcc` リソースは、ドメイン記述の一部として `<Content>` セクションに含まれることも、単独で `<Envelope>` セクションに記述されることもあります。次のキーを持つ `<gprop:GenericProperty>` タグを使用できます。

- `service_name` – 仮想コンソール端末集配信装置サービスに割り当てる名前
- `min_port` – この `vcc` に関連付ける最小ポート番号
- `max_port` – この `vcc` に関連付ける最大ポート番号

例 21 `vcc` の XML の例

```
<Envelope>
  <References/>
  <Content xsi:type="ovf:VirtualSystem_Type" id="ldg1">
    <Section xsi:type="ovf:VirtualHardwareSection_Type">
```

```
      <Item>
        <rasd:OtherResourceType>vcc</rasd:OtherResourceType>
        <gprop:GenericProperty key="service_name">vcc1</gprop:GenericProperty>
        <gprop:GenericProperty key="min_port">6000</gprop:GenericProperty>
        <gprop:GenericProperty key="max_port">6100</gprop:GenericProperty>
      </Item>
    </Section>
  </Content>
</Envelope>
```

変数 (var) リソース

var リソースは、<Content> セクション内に必ず含まれます。次のキーを持つ <gprop:GenericProperty> タグを使用できます。

- name – 変数の名前
- value – 変数の値

例 22 var の XML の例

```
<Envelope>
  <References/>
  <Content xsi:type="ovf:VirtualSystem_Type" id="ldg1">
    <Section xsi:type="ovf:VirtualHardwareSection_Type">
      <Item>
        <rasd:OtherResourceType>var</rasd:OtherResourceType>
        <gprop:GenericProperty key="name">test_var</gprop:GenericProperty>
        <gprop:GenericProperty key="value">test1</gprop:GenericProperty>
      </Item>
    </Section>
  </Content>
</Envelope>
```

物理 I/O デバイス (physio_device) リソース

physio_device リソースは、<Content> セクション内に必ず含まれます。このリソースは、add-io、set-io、remove-io、create-vf、destroy-vf、および set-domain サブコマンドを使用して変更できます。

例 23 physio_device の XML の例

次の例は、仮想機能、物理機能、ルートコンプレックスでアクションを実行する方法を示します。

- 次の XML の例では、ldm add-io コマンドを使用して、/SYS/MB/NET0/IOVNET.PF0.VF0 仮想機能を ldg1 ドメインに追加する方法を示します。

```
<LDM_interface version="1.3">
  <cmd>
    <action>add-io</action>
    <data version="3.0">
      <Envelope>
        <References/>
        <Content xsi:type="ovf:VirtualSystem_Type" ovf:id="ldg1">
          <Section xsi:type="ovf:VirtualHardwareSection_Type">
            <Item>
              <rasd:OtherResourceType>physio_device</rasd:OtherResourceType>
              <gprop:GenericProperty key="name">
              /SYS/MB/NET0/IOVNET.PF0.VF0</gprop:GenericProperty>
            </Item>
          </Section>
        </Content>
      </Envelope>
    </data>
  </cmd>
</LDM_interface>
```

- 次の XML の例に、`ldm set-io` コマンドを使用して、`pci_1` ルートコンプレックスの `iov_bus_enable_iov` プロパティー値を `on` に設定する方法を示します。

```
<LDM_interface version="1.3">
  <cmd>
    <action>set-io</action>
    <data version="3.0">
      <Envelope>
        <References/>
          <Section xsi:type="ovf:VirtualHardwareSection_Type">
            <Item>
              <rasd:OtherResourceType>physio_device</rasd:OtherResourceType>
              <gprop:GenericProperty key="name">pci_1</gprop:GenericProperty>
              <gprop:GenericProperty key="iov_bus_enable_iov">
              on</gprop:GenericProperty>
            </Item>
        </Section>
      </Envelope>
    </data>
  </cmd>
</LDM_interface>
```

- 次の XML の例に、`ldm set-io` コマンドを使用して、`/SYS/MB/NET0/IOVNET.PF1` 物理機能の `unicast-slots` プロパティー値を `6` に設定する方法を示します。

```
<LDM_interface version="1.3">
  <cmd>
```

```
    <action>set-io</action>
    <data version="3.0">
      <Envelope>
        <References/>
          <Section xsi:type="ovf:VirtualHardwareSection_Type">
            <Item>
              <rasd:OtherResourceType>physio_device</rasd:OtherResourceType>
              <gprop:GenericProperty key="name">
              /SYS/MB/NET0/IOVNET.PF1</gprop:GenericProperty>
              <gprop:GenericProperty key="unicast-slots">6</gprop:GenericProperty>
            </Item>
          </Section>
      </Envelope>
    </data>
  </cmd>
</LDM_interface>
```

- 次の XML の例では、`ldm create-vf` コマンドを使用して、次のプロパティー値で `/SYS/MB/NET0/IOVNET.PF1.VF0` 仮想機能を作成する方法を示します。
  - `unicast-slots=6`
  - `pvid=3`
  - `mtu=1600`

```
<LDM_interface version="1.3">
  <cmd>
    <action>create-vf</action>
    <data version="3.0">
      <Envelope>
        <References/>
          <Section xsi:type="ovf:VirtualHardwareSection_Type">
            <Item>
              <rasd:OtherResourceType>vf_device</rasd:OtherResourceType>
              <gprop:GenericProperty key="iov_pf_name">
              /SYS/MB/NET0/IOVNET.PF1</gprop:GenericProperty>
              <gprop:GenericProperty key="unicast-slots">6</gprop:GenericProperty>
              <gprop:GenericProperty key="pvid">3</gprop:GenericProperty>
              <gprop:GenericProperty key="mtu">1600</gprop:GenericProperty>
            </Item>
          </Section>
      </Envelope>
    </data>
  </cmd>
</LDM_interface>
```

- 次の部分的な XML の例は、`ldm create-vf` コマンドを使用して、`/SYS/MB/NET0/IOVNET.PF1` 物理機能で `iov_pf_repeat_count_str` 値 (3) によって指定された数

の仮想機能を作成する方法を示します。iov_pf_repeat_count_str プロパティーで複数の仮想機能を作成するときにほかのプロパティー値を指定することはできません。

```
<LDM_interface version="1.3">
  <cmd>
    <action>create-vf</action>
    <data version="3.0">
      <Envelope>
        <References/>
          <Section xsi:type="ovf:VirtualHardwareSection_Type">
            <Item>
              <rasd:OtherResourceType>vf_device</rasd:OtherResourceType>
              <gprop:GenericProperty key="iov_pf_name">
              /SYS/MB/NET0/IOVNET.PF1</gprop:GenericProperty>
              <gprop:GenericProperty key="iov_pf_repeat_count_str">
              3</gprop:GenericProperty>
            </Item>
          </Section>
      </Envelope>
    </data>
  </cmd>
</LDM_interface>
```

SP 構成 (spconfig) リソース

サービスプロセッサ (SP) 構成 (spconfig) リソースは、必ず単独で <Envelope> セクションに記述されます。次のキーを持つ <gprop:GenericProperty> タグを使用できます。

- spconfig_name – SP に格納されている構成の名前。
- spconfig_status – 特定の SP 構成の現在のステータス。このプロパティーは、ldm list-spconfig コマンドの出力で使用されます。

例 24 spconfig の XML の例

```
<Envelope>
    <Section xsi:type="ovf:ResourceAllocationSection_type">
      <Item>
        <rasd:OtherResourceType>spconfig</rasd:OtherResourceType>
        <gprop:GenericProperty
          key="spconfig_name">primary</gprop:GenericProperty>
        <gprop:GenericProperty
          key="spconfig_status">current</gprop:GenericProperty>
          key="spconfig_aux_status">degraded</gprop:GenericProperty>
      </Item>
```

```
    </Section>
</Envelope>
```

DRM ポリシー構成 (policy) リソース

DRM ポリシー (policy) リソースは、<Envelope> セクションに表示され、次のキーが付いた<gprop:GenericProperty> タグを持つことができます。

- policy_name – DRM ポリシーの名前
- policy_enable – DRM ポリシーが有効または無効のどちらであるかを指定します。
- policy_priority – DRM ポリシーの優先順位
- policy_vcpu_min – 1 つのドメインに対する仮想 CPU リソースの最小数
- policy_vcpu_max – 1 つのドメインに対する仮想 CPU リソースの最大数
- policy_util_lower – ポリシー分析がトリガーされるタイミングでの使用率の下限レベル
- policy_util_upper – ポリシー分析がトリガーされるタイミングでの使用率の上限レベル
- policy_tod_begin – DRM ポリシーの実効開始時間
- policy_tod_end – DRM ポリシーの実効停止時間
- policy_sample_rate – サンプリングレート。秒単位のサイクル時間
- policy_elastic_margin – CPU 使用率の上限および下限範囲のバッファ量
- policy_attack – いずれか 1 つのリソース制御サイクル中に追加されるリソースの最大量
- policy_decay – いずれか 1 つのリソース制御サイクル中に削除されるリソースの最大量

例 25 policy の XML の例

```
<Envelope>
   <Section xsi:type="ovf:VirtualHardwareSection_Type">
     <Item>
       <rasd:OtherResourceType>policy</rasd:OtherResourceType>
       <gprop:GenericProperty key="policy_name">test-policy</gprop:GenericProperty>
       <gprop:GenericProperty key="policy_enable">on</gprop:GenericProperty>
       <gprop:GenericProperty key="policy_priority">1</gprop:GenericProperty>
       <gprop:GenericProperty key="policy_vcpu_min">12</gprop:GenericProperty>
       <gprop:GenericProperty key="policy_vcpu_max">13</gprop:GenericProperty>
       <gprop:GenericProperty key="policy_util_lower">8</gprop:GenericProperty>
       <gprop:GenericProperty key="policy_util_upper">9</gprop:GenericProperty>
       <gprop:GenericProperty key="policy_tod_begin">07:08:09</gprop:GenericProperty>
       <gprop:GenericProperty key="policy_tod_end">09:08:07</gprop:GenericProperty>
       <gprop:GenericProperty key="policy_sample_rate">1</gprop:GenericProperty>
       <gprop:GenericProperty key="policy_elastic_margin">8</gprop:GenericProperty>
       <gprop:GenericProperty key="policy_attack">8</gprop:GenericProperty>
       <gprop:GenericProperty key="policy_decay">9</gprop:GenericProperty>
```

```
    </Item>
   </Section>
</Envelope>
```

コンソール (console) リソース

console リソースは、<Content> セクション内に必ず含まれます。次のキーを持つ <gprop:GenericProperty> タグを使用できます。

- port – この仮想コンソール (console) の変更先のポート
- service_name – この console をバインドする仮想コンソール端末集配信装置 (vcc) サービス
- group – この console をバインドするグループの名前
- enable-log – このコンソールの仮想コンソールロギングを有効または無効にします

例 26 console の XML の例

```
<Envelope>
  <References/>
  <Content xsi:type="ovf:VirtualSystem_Type" id="ldg1">
    <Section xsi:type="ovf:VirtualHardwareSection_Type">
      <Item>
        <rasd:OtherResourceType>console</rasd:OtherResourceType>
        <gprop:GenericProperty key="port">6000</gprop:GenericProperty>
        <gprop:GenericProperty key="service_name">vcc2</gprop:GenericProperty>
        <gprop:GenericProperty key="group">group-name</gprop:GenericProperty>
        <gprop:GenericProperty key="enable-log">on</gprop:GenericProperty>
      </Item>
    </Section>
  </Content>
</Envelope>
```

ドメインの移行

次の例は、ldm migrate-domain コマンドの <data> セクションの内容を示しています。

- 1 番めの <Content> ノード (<ldom_info> セクションなし) は、移行元のソースドメインです。
- 2 番めの <Content> ノード (<ldom_info> セクションあり) は、移行先のターゲットドメインです。ソースドメインとターゲットドメインの名前は同じにすることができます。
- ターゲットドメインの <ldom_info> セクションには、移行先のマシンおよびこのマシンへの移行に必要な詳細情報が記述されます。

- `target-host` は、移行先のターゲットマシンです。
- `user-name` は、ターゲットマシンのログインユーザー名で、SASL 64 ビットでエンコードされている必要があります。
- `password` は、ターゲットマシンへのログインに使用するパスワードで、SASL 64 ビットでエンコードされている必要があります。

注記 - Logical Domains Manager では、`sasl_decode64()` を使用してターゲットのユーザー名およびパスワードを復号化し、`sasl_encode64()` を使用してこれらの値をエンコードします。SASL 64 符号化は、`base64` 符号化に相当します。

例 27 `migrate-domain` の `<data>` セクションの例

```
<Envelope>
  <References/>
  <Content xsi:type="ovf:VirtualSystem_Type" ovf:id="ldg1"/>
  <Content xsi:type="ovf:VirtualSystem_Type" ovf:id="ldg1"/>
    <Section xsi:type="ovf:ResourceAllocationSection_Type">
      <Item>
        <rasd:OtherResourceType>ldom_info</rasd:OtherResourceType>
        <gprop:GenericProperty key="target">target-host</gprop:GenericProperty>
        <gprop:GenericProperty key="username">user-name</gprop:GenericProperty>
        <gprop:GenericProperty key="password">password</gprop:GenericProperty>
      <Item>
    </Section>
  </Content>
</Envelope>
```

XML スキーマ

Logical Domains Manager で使用される XML スキーマは、`/opt/SUNWldm/bin/schemas` ディレクトリにあります。ファイル名は、次のとおりです。

- `cim-common.xsd` – `cim-common.xsd` スキーマ
- `cim-rasd.xsd` – `cim-rasd.xsd` スキーマ
- `cim-vssd.xsd` – `cim-vssd.xsd` スキーマ
- `cli-list-constraint-v3.xsd` – `cli-list-constraint-v3.xsd` スキーマ
- `combined-v3.xsd` – `LDM_interface` XML スキーマ
- `event-v3.xsd` – `LDM_Event` XML スキーマ
- `ldmd-binding.xsd` – `Binding_Type` XML スキーマ
- `ldmd-property.xsd` – `GenericProperty` XML スキーマ
- `ovf-core.xsd` – `ovf-core.xsd` スキーマ
- `ovf-envelope.xsd` – `ovf-envelope.xsd` スキーマ
- `ovf-section.xsd` – `ovf-section.xsd` スキーマ

- `ovf-strings.xsd` – `ovf-strings.xsd` スキーマ
- `ovfenv-core.xsd` – `ovfenv-core.xsd` スキーマ
- `ovfenv-section.xsd` – `ovfenv-section.xsd` スキーマ

♦ ♦ ♦ 第 4 章

Logical Domains Manager の検出

この章では、サブネット上のシステムで実行されている Logical Domains Manager の検出について説明します。

Logical Domains Manager を実行しているシステムの検出

マルチキャストメッセージを使用すると、サブネット上で Logical Domains Manager を検出できます。`ldmd` デーモンは、ネットワーク上で特定のマルチキャストパケットを待機できます。そのマルチキャストメッセージが特定のタイプの場合、`ldmd` は呼び出し元に対して応答します。これにより、Oracle VM Server for SPARC を実行しているシステム上で `ldmd` を検出できます。

マルチキャスト通信

この検出メカニズムは、`ldmd` デーモンによって使用されるものと同じマルチキャストネットワークを使用して、MAC アドレスを自動的に割り当てるときに衝突を検出します。マルチキャストソケットを構成するには、次の情報を指定する必要があります。

```
#define MAC_MULTI_PORT  64535
#define MAC_MULTI_GROUP  "239.129.9.27"
```

デフォルトでは、マシンが接続されているサブネット上ではマルチキャストパケットのみを送信できます。この動作をは、`ldmd` デーモンに `ldmd/hops` SMF プロパティーを設定することによって変更できます。

メッセージ形式

検出メッセージは、他のメッセージと混同しないように明白にマークされている必要があります。次のマルチキャストメッセージ形式により、検出待機プロセスで検出メッセージを識別できます。

```
#include <netdb.h> /* Used for MAXHOSTNAMELEN definition */
#define MAC_MULTI_MAGIC_NO 92792004
#define MAC_MULTI_VERSION 1

enum {
 SEND_MSG = 0,
 RESPONSE_MSG,
 LDMD_DISC_SEND,
 LDMD_DISC_RESP,
};

typedef struct {
 uint32_t version_no;
 uint32_t magic_no;
 uint32_t msg_type;
 uint32_t resv;
 union {
  mac_lookup_t  Mac_lookup;
  ldmd_discovery_t Ldmd_discovery;
 } payload;
#define lookup  payload.Mac_lookup
#define discovery payload.Ldmd_discovery
} multicast_msg_t;

#define LDMD_VERSION_LEN 32

typedef struct {
 uint64_t mac_addr;
 char  source_ip[INET_ADDRSTRLEN];
} mac_lookup_t;

typedef struct {
 char  ldmd_version[LDMD_VERSION_LEN];
 char  hostname[MAXHOSTNAMELEN];
 struct in_addr ip_address;
 int  port_no;
} ldmd_discovery_t;
```

▼ サブネット上で実行している Logical Domains Manager を検出する方法

1. **マルチキャストソケットを開きます。**
   53 ページの「マルチキャスト通信」 に示すポートおよびグループの情報を使用していることを確認してください。

2. **ソケット経由で `multicast_msg_t` メッセージを送信します。**
   メッセージには次の内容を含めるようにしてください。

   - `version_no` の有効な値 (`MAC_MULTI_VERSION` によって定義されている `1`)
   - `magic_no` の有効な値 (`MAC_MULTI_MAGIC_NO` によって定義されている `92792004`)
   - `LDMD_DISC_SEND` の `msg_type`

3. **マルチキャストソケットで Logical Domains Manager からの応答を待機します。**

応答は、次の情報が含まれる `multicast_msg_t` メッセージである必要があります。

- `version_no` の有効な値
- `magic_no` の有効な値
- `LDMD_DISC_RESP` に設定された `msg_type`
- 次の情報が含まれる、`ldmd_discovery_t` 構造で構成されたペイロード
  - `ldmd_version` – システム上で実行されている Logical Domains Manager のバージョン
  - `hostname` – システムのホスト名
  - `ip_address` – システムの IP アドレス
  - `port_no` – Logical Domains Manager によって通信に使用されているポート番号で、XMPP ポート `6482` にする

Logical Domains Manager からの応答を待機する場合、自動割り当て MAC 衝突検出パケットが破棄されていることを確認してください。

♦ ♦ ♦ 第 5 章

5 仮想ドメイン情報コマンドおよび API の使用

この章では、仮想ドメイン情報コマンドおよび API について説明します。

仮想ドメイン情報コマンドの使用

`virtinfo` コマンドを使用して、実行中の仮想ドメインに関する情報を収集することができます。

コマンドまたは API を使用して、仮想ドメインについて収集できる情報の一覧を、次に示します。

- ドメインの種類 (実装、制御、ゲスト、I/O、サービス、ルート)
- 仮想ドメインマネージャーにより決定されるドメイン名
- ドメインのユニバーサル固有識別子 (UUID)
- ドメインの制御ドメインのネットワークノード名
- ドメインが実行されているシャーシのシリアル番号

`virtinfo` コマンドについては、`virtinfo`(1M) のマニュアルページを参照してください。

仮想ドメイン情報 API の使用

また、仮想ドメイン情報 API を使用して、仮想ドメインに関する情報を収集するプログラムを作成することもできます。`libv12n`(3LIB) と `v12n`(3EXT) のマニュアルページを参照してください。

索引