Oracle® VM Server for SPARC 3.2 管理指南

文件号码 E56425
2015 年 5 月

ORACLE

文档可访问性

有关 Oracle 对可访问性的承诺，请访问 Oracle Accessibility Program 网站 http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc。

获得 Oracle 支持

购买了支持服务的 Oracle 客户可通过 My Oracle Support 获得电子支持。有关信息，请访问 http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info；如果您听力受损，请访问 http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs。

目录

使用本文档

- **概述** – 提供了有关在受支持的服务器、刀片和服务器模块上运行的 Oracle VM Server for SPARC 3.2 软件的详细信息，包括概述和安全注意事项方面的信息，以及安装、配置、修改和执行常规任务的具体过程。请参见《Oracle VM Server for SPARC 3.2 安装指南 》中的“受支持的平台”。

注 - 本书中介绍的功能可与《Oracle VM Server for SPARC 3.2 安装指南 》中列出的所有受支持系统软件和硬件平台结合使用。但是，有些功能只适用于一部分受支持的系统软件和硬件平台。有关这些例外的信息，请参见《Oracle VM Server for SPARC 3.2 发行说明 》中的“本发行版新增功能”和 What's New in Oracle VM Server for SPARC Software (`http://www.oracle.com/technetwork/server-storage/vm/documentation/sparc-whatsnew-330281.html`)（Oracle VM Server for SPARC 软件中的新增功能）。

- **目标读者** – 管理 SPARC 服务器上的虚拟化功能的系统管理员。
- **必需的知识** – 这些服务器的系统管理员必须具有 UNIX 系统和 Oracle Solaris 操作系统 (Oracle Solaris operating system, Oracle Solaris OS) 的实际应用知识。

产品文档库

位于 `http://www.oracle.com/pls/topic/lookup?ctx=E56447` 的文档库中包含此产品的最新信息和已知问题。

反馈

可以在 `http://www.oracle.com/goto/docfeedback` 上提供有关本文档的反馈。

部分 I

Oracle VM Server for SPARC 3.2 软件

本部分介绍 Oracle VM Server for SPARC 3.2 软件，该软件可为 SPARC T 系列、SPARC M 系列和 Fujitsu M10 平台提供高效率企业级虚拟化功能。

♦ ♦ ♦ 第 1 章

1 Oracle VM Server for SPARC 软件概述

本章对 Oracle VM Server for SPARC 软件进行了概述。

Oracle VM Server for SPARC 可为 SPARC T 系列和 SPARC M 系列平台以及 Fujitsu M10 平台提供高效率企业级虚拟化功能。使用 Oracle VM Server for SPARC 软件，可以在单个系统上创建最多 128 个虚拟服务器（称为逻辑域）。这种配置使您能够利用由 SPARC T 系列和 SPARC M 系列平台、Fujitsu M10 平台和 Oracle Solaris OS 提供的海量线程处理能力。

本章包括以下主题：

关于 Oracle VM Server for SPARC 和 Oracle Solaris OS 版本

Oracle VM Server for SPARC 软件依赖于特定 Oracle Solaris OS 版本、所需的软件修补程序以及特定版本的系统固件。有关更多信息，请参见《Oracle VM Server for SPARC 3.2 安装指南 》中的“全限定 Oracle Solaris OS 版本”。

在来宾域上运行的 Oracle Solaris OS 版本与在 `primary` 域上运行的 Oracle Solaris OS 版本相互独立。因此，如果您在 `primary` 域中运行 Oracle Solaris 10 OS，则仍可以在来宾域中运行 Oracle Solaris 11 OS。同样，如果您在 `primary` 域中运行 Oracle Solaris 11 OS，那么仍然可以在来宾域中运行 Oracle Solaris 10 OS。

注 - 有些平台不再支持 `primary` 域中的 Oracle Solaris 10 OS。请查阅您的平台文档。您可以在其他域中继续运行 Oracle Solaris 10 OS。

在 `primary` 域上运行 Oracle Solaris 10 OS 或 Oracle Solaris 11 OS 的唯一区别是每个 OS 中的功能差异。

虚拟机管理程序和 Logical Domains

本节对支持逻辑域的 SPARC 虚拟机管理程序进行了概述。

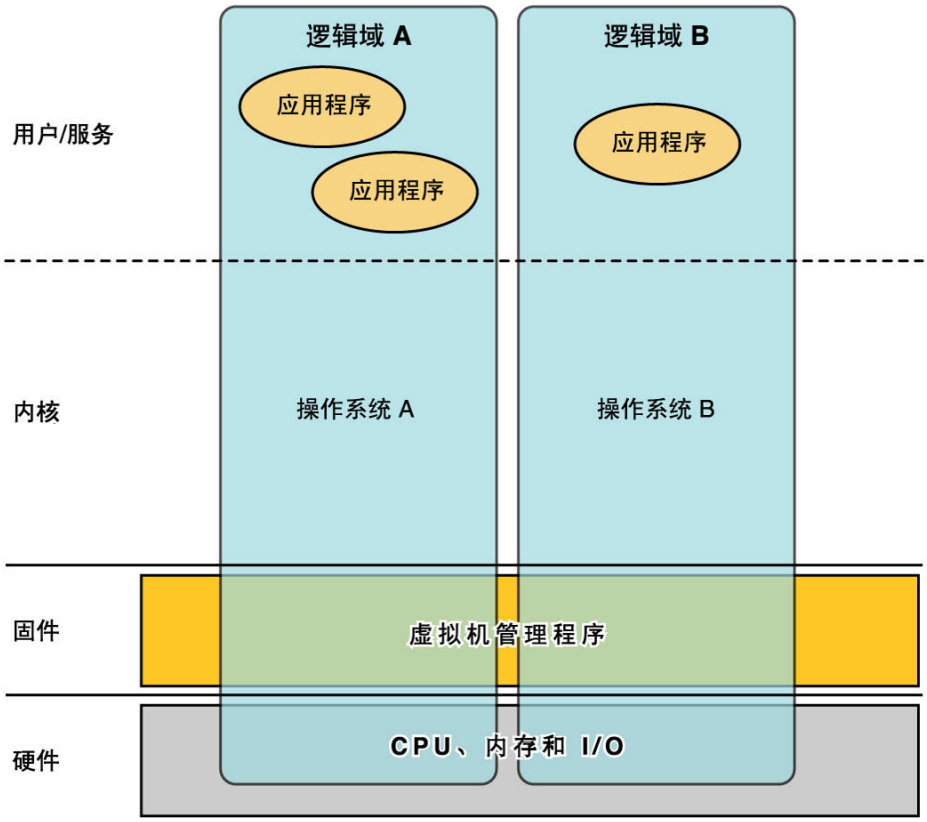
SPARC 虚拟机管理程序是一个小固件层，提供了可以向其中写入操作系统的稳定虚拟机体系结构。使用虚拟机管理程序的 SPARC 服务器提供相关的硬件功能，支持虚拟机管理程序对逻辑操作系统的活动进行控制。

逻辑域是由离散的逻辑分组资源组成的虚拟机。逻辑域在单个计算机系统内具有自己的操作系统和身份。可以单独创建、销毁、重新配置及重新引导每个逻辑域，而无需对服务器执行关开机循环。可以在不同的逻辑域中运行各种应用程序软件，并使其保持相互独立，以获得相应的性能和安全。

每个逻辑域只能发现由虚拟机管理程序设置为可用的那些服务器资源并与之交互。Logical Domains Manager 允许您指定虚拟机管理程序应通过控制域执行的操作。这样，虚拟机管理程序会执行服务器资源的划分，并向多个操作系统环境提供有限的子集。这种划分和置备操作是创建逻辑域的基本机制。下图显示了支持两个逻辑域的虚拟机管理程序。图中还显示了构成 Oracle VM Server for SPARC 功能的以下各层：

- 用户/服务（应用程序）
- 内核（操作系统）
- 固件（虚拟机管理程序）
- 硬件，包括 CPU、内存和 I/O

图 1-1 支持两个域的虚拟机管理程序



特定的 SPARC 虚拟机管理程序所支持的逻辑域的数量和功能是与服务器相关的特性。虚拟机管理程序可以向给定的逻辑域分配服务器的全部 CPU、内存和 I/O 资源的子集。通过该功能可以同时支持多个操作系统，每个操作系统位于自己的逻辑域内。资源可以在不同逻辑域之间以任意粒度重新布置。例如，可以按 CPU 线程为粒度将 CPU 分配给逻辑域。

每个逻辑域都可以作为完全独立的机器进行管理，并拥有自己的资源，例如：

- 内核、修补程序和调节参数
- 用户帐户和管理员
- 磁盘
- 网络接口、MAC 地址和 IP 地址

可以互相独立地停止、启动和重新引导每个逻辑域，而无需对服务器执行关开机循环。

虚拟机管理程序软件负责保持各个逻辑域之间的相互独立。虚拟机管理程序软件还提供逻辑域通道 (logical domain channel, LDC)，使逻辑域可以相互通信。使用 LDC，域可以相互提供服务，如联网或磁盘服务。

服务处理器 (service processor, SP)（也称为系统控制器 (system controller, SC)）监视和运行物理机，但不管理逻辑域。Logical Domains Manager 管理逻辑域。

除了使用 `ldm` 命令管理 Oracle VM Server for SPARC 软件以外，现在还可以使用 Oracle VM Manager。

Oracle VM Manager 是基于 Web 的用户界面，可用于管理 Oracle VM 环境。此用户界面的较早版本仅管理 Oracle VM Server x86 软件；但从 Oracle VM Manager 3.2 和 Oracle VM Server for SPARC 3.0 开始，还可管理 Oracle VM Server for SPARC 软件。有关 Oracle VM Manager 的更多信息，请参见 Oracle VM Documentation (`http://www.oracle.com/technetwork/documentation/vm-096300.html`)（Oracle VM 文档）。

Logical Domains Manager

Logical Domains Manager 用于创建和管理逻辑域，以及将逻辑域映射到物理资源。一台服务器上只能运行一个 Logical Domains Manager。

域的角色

所有逻辑域都是相同的，可以基于为其指定的角色将各个逻辑域区分开。逻辑域可执行以下角色：

- **控制域。**Logical Domains Manager 在此域中运行，使您能够创建和管理其他逻辑域，并将虚拟资源分配给其他域。每台服务器只能有一个控制域。控制域是安装 Oracle VM Server for SPARC 软件时创建的第一个域。控制域名为 `primary`。
- **服务域。**服务域向其他域提供虚拟设备服务，如虚拟交换机、虚拟控制台集中器以及虚拟磁盘服务器。您可以有多个服务域，并且可将任何域配置为服务域。
- **I/O 域。**I/O 域可以直接访问物理 I/O 设备，例如 PCI EXPRESS (PCIe) 控制器中的网卡。I/O 域可以拥有以下内容：
  - PCIe 根联合体。
  - PCIe 插槽或板载 PCIe 设备（通过使用直接 I/O (direct I/O, DIO) 功能）。请参见“通过分配 PCIe 端点设备创建 I/O 域” [67]。
  - PCIe SR-IOV 虚拟功能。请参见第 8 章 使用 PCIe SR-IOV 虚拟功能创建 I/O 域。

  当 I/O 域也用作服务域时，I/O 域能够以虚拟设备形式与其他域共享物理 I/O 设备。
- **根域。**根域有一个分配给它的 PCIe 根联合体。此域拥有 PCIe 结构并提供所有与结构相关的服务，如结构错误处理。根域也是 I/O 域，因为它拥有对物理 I/O 设备的直接访问权限。

您可以拥有的根域的数量取决于您的平台体系结构。例如，如果使用的是 Oracle Sun SPARC Enterprise T5440 服务器，最多可以有四个根域。

默认根域为 `primary` 域。可以使用非 `primary` 域作为根域。

- **来宾域。**来宾域是非 I/O 域，它使用一个或多个服务域提供的虚拟设备服务。来宾域没有任何物理 I/O 设备，只有虚拟 I/O 设备，如虚拟磁盘和虚拟网络接口。

可以在尚未配置 Oracle VM Server for SPARC 的现有系统上安装 Logical Domains Manager。在这种情况下，OS 的当前实例会成为控制域。而且，系统上仅配置一个域（即控制域）。在配置控制域之后，可以通过添加域并将这些应用程序从控制域移至新域来平衡其他域之间的应用程序负载，从而最大程度地提高整个系统的使用效率。

命令行界面

Logical Domains Manager 使用命令行界面 (command-line interface, CLI) 来创建和配置逻辑域。该 CLI 是具有多个子命令的单个命令 `ldm`。请参见 `ldm`(1M) 手册页。

Logical Domains Manager 守护进程 `ldmd` 必须正在运行才能使用 Logical Domains Manager CLI。

虚拟输入/输出

在 Oracle VM Server for SPARC 环境中，最多可以在一个系统上置备 128 个域（在 Fujitsu M10 服务器上最多可置备 256 个）。某些服务器（尤其是单处理器和部分双处理器系统）只拥有有限数量的 I/O 总线和物理 I/O 插槽。因此，可能无法向这些系统上的所有域提供对物理磁盘和网络设备的独占访问。可以将 PCIe 总线或端点设备分配给域，以为其提供对物理设备的访问。请注意，此解决方案不足以向所有域提供独占的设备访问。通过实施虚拟化 I/O 模型，可以解决对可直接访问的物理 I/O 设备数量的此限制。请参见第 5 章 配置 I/O 域。

没有物理 I/O 访问的所有逻辑域都配置有与服务域进行通信的虚拟 I/O 设备。服务域运行虚拟设备服务以提供对物理设备或其功能的访问。在此客户机-服务器模型中，虚拟 I/O 设备通过称为逻辑域通道 (logical domain channel, LDC) 的域际信道相互通信或与对应服务通信。虚拟化 I/O 功能包含对虚拟网络、存储和控制台的支持。

虚拟网络

Oracle VM Server for SPARC 使用虚拟网络设备和虚拟网络交换机设备来实现虚拟网络。虚拟网络 (`vnet`) 设备可模仿以太网设备并通过使用点对点通道与系统中的其他 `vnet` 设备进行通信。虚拟交换机 (`vsw`) 设备主要充当所有虚拟网络的传入和传出包的多路复用器。`vsw` 设备可直接与服务域上的物理网络适配器进行通信，并代表虚拟网络发送和

接收包。vsw 设备还充当简单的第二层交换机，在系统内与其连接的 vnet 设备之间交换包。

虚拟存储

虚拟存储基础结构采用客户机-服务器模型，以使逻辑域能够访问未直接分配给它们的块级存储。该模型使用以下组件：

- 虚拟磁盘客户机 (vdc)，用于导出块设备接口
- 虚拟磁盘服务 (vds)，用于以虚拟磁盘客户机的名义处理磁盘请求，并将这些请求提交给驻留在服务域上的后端存储

虽然虚拟磁盘显示为客户机域上的常规磁盘，但是大多数磁盘操作都会转发给虚拟磁盘服务并在服务域上进行处理。

虚拟控制台

在 Oracle VM Server for SPARC 环境中，来自 primary 域的控制台 I/O 会定向至服务处理器。其他所有域的控制台 I/O 会重定向至运行虚拟控制台集中器 (vcc) 的服务域。运行 vcc 的域通常为 primary 域。虚拟控制台集中器服务可以充当所有域的控制台通信流量的集中器，并可与虚拟网络终端服务器守护进程 (vntsd) 进行连接，以便通过 UNIX 套接字访问每个控制台。

资源配置

运行 Oracle VM Server for SPARC 软件的系统可以配置诸如虚拟 CPU、虚拟 I/O 设备、加密单元和内存等资源。某些资源可以在正在运行的域上动态地进行配置，而其他一些资源则必须在停止的域上进行配置。如果无法在控制域上动态地配置资源，必须首先启动延迟重新配置。延迟重新配置会将配置活动推迟到控制域进行重新引导后。有关更多信息，请参见“资源重新配置” [251]。

持久性配置

可以使用 ldm 命令将逻辑域的当前配置存储在服务处理器上。可以添加配置、指定要使用的配置、删除配置，以及列出配置。有关详细信息，请参见 ldm(1M) 手册页。您还可以指定要从 SP 引导的配置（如“将 Oracle VM Server for SPARC 与服务处理器结合使用” [306]中所述）。

有关管理配置的信息，请参见“管理域配置” [287]。

Oracle VM Server for SPARC 物理机到虚拟机转换工具

Oracle VM Server for SPARC 物理机到虚拟机 (Physical-to-Virtual, P2V) 转换工具可以自动将现有物理系统转换为在芯片多线程 (chip multithreading, CMT) 系统上的逻辑域中运行 Oracle Solaris 10 OS 的虚拟系统。您可以从运行 Oracle Solaris 10 OS 或 Oracle Solaris 11 OS 的控制域运行 `ldmp2v` 命令，以将以下其中一个源系统转换为逻辑域：

- 运行版本至少为 Solaris 8、Solaris 9 或 Oracle Solaris 10 OS 的任何基于 `sun4u` SPARC 的系统
- 运行 Oracle Solaris 10 OS 但不运行 Oracle VM Server for SPARC 软件的任何 `sun4v` 系统

注 - 您不能使用 P2V 工具将 Oracle Solaris 11 物理系统转换为虚拟系统。

有关该工具及其安装的信息，请参见第 17 章 Oracle VM Server for SPARC 物理机到虚拟机转换工具。有关 `ldmp2v` 命令的信息，请参见 `ldmp2v`(1M) 手册页。

Oracle VM Server for SPARC Configuration Assistant

Oracle VM Server for SPARC Configuration Assistant 将引导您完成通过设置基本属性配置逻辑域的过程。可将其用于配置已安装 Oracle VM Server for SPARC 软件但尚未对其进行配置的任何系统。

收集配置数据之后，Configuration Assistant 将创建适合于作为逻辑域引导的配置。还可以使用 Configuration Assistant 选定的默认值创建可用的系统配置。

注 - `ldmconfig` 命令仅在 Oracle Solaris 10 系统上受支持。

Configuration Assistant 是基于终端的工具。

有关更多信息，请参见第 18 章 Oracle VM Server for SPARC Configuration Assistant (Oracle Solaris 10) 和 `ldmconfig`(1M) 手册页。

Oracle VM Server for SPARC 管理信息库

Oracle VM Server for SPARC 管理信息库 (Management Information Base, MIB) 使第三方系统管理应用程序能够使用简单网络管理协议 (Simple Network Management Protocol, SNMP) 对域执行远程监视并启动和停止逻辑域（简称域）。有关更多信息，请参见第 20 章 使用 Oracle VM Server for SPARC 管理信息库软件。

Oracle VM Server for SPARC 故障排除

可以通过以下出版物获取有关特定的 Oracle VM Server for SPARC 软件问题的信息：

- 《Oracle VM Server for SPARC 3.2 发行说明 》中的“已知问题”
- Information Center: Overview of Oracle VM Server for SPARC (LDoms) (Doc ID 1589473.2) (https://mosemp.us.oracle.com/epmos/faces/DocumentDisplay?_afrLoop=227880986952919&id=1589473.2&_afrWindowMode=0&_adf.ctrl-state=wu098o5r6_96)（信息中心：Oracle VM Server for SPARC (LDoms) 概述（文档号 1589473.2））

♦ ♦ ♦ 第 2 章

Oracle VM Server for SPARC 安全

本章介绍可以在 Oracle VM Server for SPARC 系统上启用的一些安全功能。

本章包括以下主题：

注 - 本书中所显示的示例均由超级用户执行。但是，您可以改用配置文件使用户获得更精细的权限来执行管理任务。

通过使用权限委托管理逻辑域

Logical Domains Manager 软件包可向本地权限配置中添加两个预定义权限配置文件。这些权限配置文件可将管理特权委托给非特权用户：

- `LDoms Management` 配置文件允许用户使用所有 `ldm` 子命令。
- `LDoms Review` 配置文件允许用户使用所有与列表有关的 `ldm` 子命令。

这些权限配置文件可以直接分配给用户或之后会分配给用户的角色。如果将其中一种配置文件直接分配给用户，您必须使用 `pfexec` 命令或配置文件 shell（例如 `pfbash` 或 `pfksh`），以成功使用 `ldm` 命令管理您的域。根据您的权限配置确定是使用角色还是权限配置文件。请参见《System Administration Guide: Security Services》或《Oracle Solaris 11.1 Administration: Security Services》中的第 III 部分, “Roles, Rights Profiles, and Privileges”。

用户、授权、权限配置文件和角色可以按如下方式进行配置：

- 使用文件在系统本地配置
- 在命名服务（如 LDAP）中集中配置

安装 Logical Domains Manager 会在本地文件中添加必需的权限配置文件。要在命名服务中配置配置文件和角色，请参见《System Administration Guide: Naming and

Directory Services (DNS, NIS, and LDAP) 》。有关 Logical Domains Manager 软件包所提供的授权和执行属性的概述，请参见“Logical Domains Manager 配置文件内容” [26]。本章中的所有示例都假定权限配置使用本地文件。

使用权限配置文件和角色

注意 - 使用 `usermod` 和 `rolemod` 命令添加授权、权限配置文件或角色时要谨慎。

- 对于 Oracle Solaris 10 OS，`usermod` 或 `rolemod` 命令会取代任何现有值。

  要添加值而不是取代值，请指定现有值和新值的逗号分隔列表。
- 在 Oracle Solaris 11 OS 中，对于添加的每个授权，使用加号 (+) 来添加值。

  例如，`usermod -A +`*auth username* 命令可将 *auth* 授权授予给 *username* 用户，与 `rolemod` 命令类似。

管理用户权限配置文件

以下过程说明如何在系统上使用本地文件管理用户权限配置文件。要在命名服务中管理用户配置文件，请参见《System Administration Guide: Naming and Directory Services (DNS, NIS, and LDAP) 》。

▼ 如何将权限配置文件分配给用户

直接分配有 `LDoms Management` 配置文件的用户必须调用配置文件 shell 才能运行具有安全属性的 `ldm` 命令。有关更多信息，请参见《System Administration Guide: Security Services 》或《Oracle Solaris 11.1 Administration: Security Services 》中的第 III 部分, “Roles, Rights Profiles, and Privileges”。

1. 成为管理员。

   - 对于 Oracle Solaris 10，请参见《System Administration Guide: Security Services 》中的“Configuring RBAC (Task Map)”。
   - 对于 Oracle Solaris 11.2，请参见《Securing Users and Processes in Oracle Solaris 11.2 》中的第 1 章 “About Using Rights to Control Users and Processes”。

2. 为本地用户帐户分配管理配置文件。

   您可以为用户帐户分配 `LDoms Review` 配置文件或 `LDoms Management` 配置文件。

   ```
   # usermod -P "profile-name" username
   ```

以下命令可以将 LDoms Management 配置文件分配给用户 sam：

```
# usermod -P "LDoms Management" sam
```

给用户分配角色

以下过程说明如何使用本地文件创建角色并将其分配给用户。要在命名服务中管理角色，请参见《System Administration Guide: Naming and Directory Services (DNS, NIS, and LDAP)》。

使用此过程的优势是，只有分配有特定角色的用户才能承担该角色。承担角色时，如果已经为该角色分配了一个密码，则需要输入该密码。这两个安全层可以防止未获得某一角色的用户承担该角色（即使该用户知道密码）。

▼ 如何创建角色并将该角色分配给用户

1. 成为管理员。
   - 对于 Oracle Solaris 10，请参见《System Administration Guide: Security Services》中的“Configuring RBAC (Task Map)”。
   - 对于 Oracle Solaris 11.2，请参见《Securing Users and Processes in Oracle Solaris 11.2》中的第 1 章 “About Using Rights to Control Users and Processes”。
2. 创建角色。

   ```
   # roleadd -P "profile-name" role-name
   ```
3. 为该角色指定密码。
   系统将提示您指定并确认新密码。

   ```
   # passwd role-name
   ```
4. 将该角色分配给用户。

   ```
   # useradd -R role-name username
   ```
5. 为用户分配密码。
   系统将提示您指定并确认新密码。

   ```
   # passwd username
   ```
6. 必要时成为该用户并提供密码。

   ```
   # su username
   ```

7. 验证该用户是否能够访问为其分配的角色。

```
$ id
uid=nn(username) gid=nn(group-name)
$ roles
role-name
```

8. 必要时承担该角色并提供密码。

```
$ su role-name
```

9. 验证该用户是否能够承担此角色。

```
$ id
uid=nn(role-name) gid=nn(group-name)
```

例 2-1 创建角色并将该角色分配给用户

此示例说明如何创建 `ldm_read` 角色、将该角色分配给 `user_1` 用户、成为 `user_1` 用户并承担 `ldm_read` 角色。

```
# roleadd -P "LDoms Review" ldm_read
# passwd ldm_read
New Password:
Re-enter new Password:
passwd: password successfully changed for ldm_read
# useradd -R ldm_read user_1
# passwd user_1
New Password:
Re-enter new Password:
passwd: password successfully changed for user_1
# su user_1
Password:
$ id
uid=95555(user_1) gid=10(staff)
$ roles
ldm_read
$ su ldm_read
Password:
$ id
uid=99667(ldm_read) gid=14(sysadmin)
```

Logical Domains Manager 配置文件内容

Logical Domains Manager 软件包可向本地权限配置文件说明数据库中添加以下权限配置文件：

```
LDoms Power Mgmt Observability:::View LDoms Power Consumption:auths=solaris.ldoms.ldmpower
LDoms Review:::Review LDoms configuration:profiles=LDoms Power Mgmt
 Observability;auths=solaris.ldoms.read
```

```
LDoms Management:::Manage LDoms domains:profiles=LDoms Power Mgmt
 Observability;auths=solaris.ldoms.*
```

Logical Domains Manager 软件包还可向本地执行配置文件数据库中添加以下与 LDoms Management 配置文件和 LDoms Power Mgmt Observability 配置文件关联的执行属性：

```
LDoms Management:suser:cmd:::/usr/sbin/ldm:privs=file_dac_read,file_dac_search
LDoms Power Mgmt Observability:suser:cmd:::/usr/sbin/ldmpower:privs=file_dac_search
```

下表列出了 ldm 子命令以及执行这些命令所需的相应用户授权。

表 2-1 ldm 子命令和用户授权

ldm 子命令[†]	用户授权
add-*	solaris.ldoms.write
bind-domain	solaris.ldoms.write
list	solaris.ldoms.read
list-*	solaris.ldoms.read
panic-domain	solaris.ldoms.write
remove-*	solaris.ldoms.write
set-*	solaris.ldoms.write
start-domain	solaris.ldoms.write
stop-domain	solaris.ldoms.write
unbind-domain	solaris.ldoms.write

[†]涉及您可以添加、列出、删除或设置的所有资源。

通过使用权限控制对域控制台的访问

默认情况下，任何用户都可以访问所有域控制台。要控制对域控制台的访问，请配置 vntsd 守护进程以执行授权检查。vntsd 守护进程提供一个名为 vntsd/authorization 的服务管理工具 (Service Management Facility, SMF) 属性。可以配置此属性以对域控制台或控制台组启用用户和角色的授权检查。要启用授权检查，请使用 svccfg 命令将此属性的值设置为 true。当此选项处于启用状态时，vntsd 将仅侦听并接受 localhost 上的连接。如果启用 vntsd/authorization 时 listen_addr 属性指定备用 IP 地址，vntsd 将忽略该备用 IP 地址并继续仅在 localhost 上侦听。

注意 - 请勿将 vntsd 服务配置为使用 localhost 以外的主机。

如果您指定的是 localhost 以外的主机，系统将不再限制您从控制域连接到来宾域控制台。如果使用 telnet 命令远程连接到来宾域，则登录凭据在网络中以明文形式传递。

默认情况下，用于访问所有来宾控制台的授权位于本地授权说明数据库中。

```
solaris.vntsd.consoles:::Access All LDoms Guest Consoles::
```

使用 `usermod` 命令可以在本地文件中为用户或角色分配必需的授权。此命令仅允许具有所需授权的用户或角色访问给定的域控制台或控制台组。要在命名服务中为用户或角色分配授权，请参见《System Administration Guide: Naming and Directory Services (DNS, NIS, and LDAP)》。

您可以控制对所有域控制台的访问，也可以控制对单一域控制台的访问。

- 要控制对所有域控制台的访问，请参见如何使用角色控制对所有域控制台的访问 [28]和如何使用权限配置文件控制对所有域控制台的访问 [29]。
- 要控制对单一域控制台的访问，请参见如何使用角色控制对单一控制台的访问 [31]和如何使用权限配置文件控制对单个控制台的访问 [33]。

▼ 如何使用角色控制对所有域控制台的访问

1. 通过启用控制台授权检查来限制对域控制台的访问。

```
primary# svccfg -s vntsd setprop vntsd/authorization = true
primary# svcadm refresh vntsd
primary# svcadm restart vntsd
```

2. 创建具有 **solaris.vntsd.consoles** 授权的角色，该授权允许访问所有域控制台。

```
primary# roleadd -A solaris.vntsd.consoles role-name
primary# passwd all_cons
```

3. 将新角色分配给用户。

```
primary# usermod -R role-name username
```

例 2-2 使用角色控制对所有域控制台的访问

首先，启用控制台授权检查以限制对域控制台的访问。

```
primary# svccfg -s vntsd setprop vntsd/authorization = true
primary# svcadm refresh vntsd
primary# svcadm restart vntsd
primary# ldm ls
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
primary          active     -n-cv-  UART    8     16G      0.2%  47m
ldg1             active     -n--v-  5000    2     1G       0.1%  17h 50m
ldg2             active     -t----  5001    4     2G        25%  11s
```

以下示例说明如何创建具有 `solaris.vntsd.consoles` 授权的 `all_cons` 角色，该授权允许访问所有域控制台。

```
primary# roleadd -A solaris.vntsd.consoles all_cons
primary# passwd all_cons
```

```
New Password:
Re-enter new Password:
passwd: password successfully changed for all_cons
```

此命令将 all_cons 角色分配给 sam 用户。

```
primary# usermod -R all_cons sam
```

用户 sam 承担 all_cons 角色，可以访问所有控制台。例如：

```
$ id
uid=700299(sam) gid=1(other)
$ su all_cons
Password:
$ telnet localhost 5000
Trying 0.0.0.0...
Connected to 0.
Escape character is '^]'.

Connecting to console "ldg1" in group "ldg1" ....
Press ~? for control options ..

$ telnet localhost 5001
Trying 0.0.0.0...
Connected to 0.
Escape character is '^]'.

Connecting to console "ldg2" in group "ldg2" ....
Press ~? for control options ..
```

本示例说明未经授权的用户 dana 尝试访问域控制台时所发生的情况：

```
$ id
uid=702048(dana) gid=1(other)
$ telnet localhost 5000
Trying 0.0.0.0...
Connected to 0.
Escape character is '^]'.
Connection to 0 closed by foreign host.
```

▼ 如何使用权限配置文件控制对所有域控制台的访问

1. 通过启用控制台授权检查来限制对域控制台的访问。

```
primary# svccfg -s vntsd setprop vntsd/authorization = true
primary# svcadm refresh vntsd
primary# svcadm restart vntsd
```

2. 创建具有 **solaris.vntsd.consoles** 授权的权限配置文件。

   - Oracle Solaris 10 OS：编辑 **/etc/security/prof_attr** 文件。

包括以下条目：

```
LDoms Consoles:::Access LDoms Consoles:auths=solaris.vntsd.consoles
```

- Oracle Solaris 11 OS：使用 **profiles** 命令创建一个新配置文件。

```
primary# profiles -p "LDoms Consoles" \
'set desc="Access LDoms Consoles"; set auths=solaris.vntsd.consoles'
```

3. 将权限配置文件分配给用户。

- Oracle Solaris 10 OS：将权限配置文件分配给用户。

```
primary# usermod -P "All,Basic Solaris User,LDoms Consoles" username
```

在添加 LDoms Consoles 配置文件期间指定任何先前存在的配置文件时要谨慎。之前的命令显示用户已经拥有 All 和 Basic Solaris User 配置文件。

- Oracle Solaris 11 OS：将权限配置文件分配给用户。

```
primary# usermod -P +"LDoms Consoles" username
```

4. 作为用户连接到域控制台。

```
$ telnet localhost 5000
```

例 2-3 使用权限配置文件控制对所有域控制台的访问

以下示例说明如何使用权限配置文件控制对所有域控制台的访问：

- Oracle Solaris 10：通过将以下条目添加到 /etc/security/prof_attr 文件创建具有 solaris.vntsd.consoles 授权的权限配置文件：

```
LDoms Consoles:::Access LDoms Consoles:auths=solaris.vntsd.consoles
```

将权限配置文件分配给 *username*。

```
primary# usermod -P "All,Basic Solaris User,LDoms Consoles" username
```

以下命令显示如何验证用户是否为 sam 以及 All、Basic Solaris User 和 LDoms Consoles 权限配置文件是否有效。telnet 命令显示如何访问 ldg1 域控制台。

```
$ id
uid=702048(sam) gid=1(other)
$ profiles
All
Basic Solaris User
LDoms Consoles
$ telnet localhost 5000
Trying 0.0.0.0...
```

```
Connected to 0.
Escape character is '^]'.

Connecting to console "ldg1" in group "ldg1" ....
Press ~? for control options ..
```

- Oracle Solaris 11：使用 `profiles` 命令在权限配置文件说明数据库中创建具有 `solaris.vntsd.consoles` 授权的权限配置文件。

```
primary# profiles -p "LDoms Consoles" \
'set desc="Access LDoms Consoles"; set auths=solaris.vntsd.consoles'
```

将权限配置文件分配给用户。

```
primary# usermod -P +"LDoms Consoles" sam
```

以下命令显示如何验证用户是否为 `sam` 以及 `All`、`Basic Solaris User` 和 `LDoms Consoles` 权限配置文件是否有效。`telnet` 命令显示如何访问 `ldg1` 域控制台。

```
$ id
uid=702048(sam) gid=1(other)
$ profiles
All
Basic Solaris User
LDoms Consoles
$ telnet localhost 5000
Trying 0.0.0.0...
Connected to 0.
Escape character is '^]'.

Connecting to console "ldg1" in group "ldg1" ....
Press ~? for control options ..
```

▼ 如何使用角色控制对单一控制台的访问

1. 通过启用控制台授权检查来限制对域控制台的访问。

```
primary# svccfg -s vntsd setprop vntsd/authorization = true
primary# svcadm refresh vntsd
primary# svcadm restart vntsd
```

2. 将单个域的授权添加到授权说明数据库中。

   授权名称从域名派生而来，并且形式为 `solaris.vntsd.console-`*domain*：

   `solaris.vntsd.console-`*domain*`:::Access` *domain* `Console::`

3. 创建具有新授权的角色，以便仅允许访问该域控制台。

```
primary# roleadd -A solaris.vntsd.console-domain role-name
primary# passwd role-name
New Password:
Re-enter new Password:
passwd: password successfully changed for role-name
```

4. 将 *role-name* 角色分配给用户。

```
primary# usermod -R role-name username
```

例 2-4 访问单一域控制台

本示例说明用户 `terry` 如何承担 `ldg1cons` 角色，以及如何访问 `ldg1` 域控制台。

首先，将单个域 (`ldg1`) 的授权添加到授权说明数据库中。

```
solaris.vntsd.console-ldg1:::Access ldg1 Console::
```

然后，创建具有新授权的角色，以便仅允许访问该域控制台。

```
primary# roleadd -A solaris.vntsd.console-ldg1 ldg1cons
primary# passwd ldg1cons
New Password:
Re-enter new Password:
passwd: password successfully changed for ldg1cons
```

将 `ldg1cons` 角色分配给用户 `terry`，其承担 `ldg1cons` 角色并访问域控制台。

```
primary# usermod -R ldg1cons terry
primary# su terry
Password:
$ id
uid=700300(terry) gid=1(other)
$ su ldg1cons
Password:
$ id
uid=700303(ldg1cons) gid=1(other)
$ telnet localhost 5000
Trying 0.0.0.0...
Escape character is '^]'.

Connecting to console "ldg1" in group "ldg1" ....
Press ~? for control options ..
```

以下示例显示用户 `terry` 无法访问 `ldg2` 域控制台：

```
$ telnet localhost 5001
Trying 0.0.0.0...
Connected to 0.
Escape character is '^]'.
Connection to 0 closed by foreign host.
```

▼ 如何使用权限配置文件控制对单个控制台的访问

1. 通过启用控制台授权检查来限制对域控制台的访问。

```
primary# svccfg -s vntsd setprop vntsd/authorization = true
primary# svcadm refresh vntsd
primary# svcadm restart vntsd
```

2. 将单个域的授权添加到授权说明数据库中。
以下示例条目可添加域控制台的授权：

```
solaris.vntsd.console-domain:::Access domain Console::
```

3. 创建具有访问特定域控制台的授权的权限配置文件。

   - Oracle Solaris 10 OS：编辑 **/etc/security/prof_attr** 文件。

     ```
     domain Console:::Access domain
     Console:auths=solaris.vntsd.console-domain
     ```

     此条目必须在一行上。

   - Oracle Solaris 11 OS：使用 **profiles** 命令创建一个新配置文件。

     ```
     primary# profiles -p "domain Console" \
     'set desc="Access domain Console";
     set auths=solaris.vntsd.console-domain'
     ```

4. 将权限配置文件分配给用户。
以下命令将配置文件分配给用户：

   - Oracle Solaris 10 OS：分配权限配置文件。

     ```
     primary# usermod -P "All,Basic Solaris User,domain Console" username
     ```

     请注意，All 和 Basic Solaris User 配置文件是必需的。

   - Oracle Solaris 11 OS：分配权限配置文件。

     ```
     primary# usermod -P +"domain Console" username
     ```

启用并使用审计

Logical Domains Manager 使用 Oracle Solaris OS 审计功能检查控制域上发生的操作和事件的历史记录。这些历史记录会保存在一个日志中，用于跟踪已执行的操作、执行时间、执行者以及受影响的内容。

您可以基于系统上运行的 Oracle Solaris OS 的版本，按如下方式启用和禁用审计功能：

- Oracle Solaris 10 OS：使用 `bsmconv` 和 `bsmunconv` 命令。请参见 `bsmconv`(1M) 和 `bsmunconv`(1M) 手册页以及《System Administration Guide: Security Services》中的第 VII 部分, “Auditing in Oracle Solaris”。
- Oracle Solaris 11 OS：使用 `audit` 命令。请参见 `audit`(1M) 手册页和《Oracle Solaris 11.1 Administration: Security Services》中的第 VII 部分, “Auditing in Oracle Solaris”。

▼ 如何启用审计

必须在您的系统上配置和启用 Oracle Solaris 审计功能。默认情况下，Oracle Solaris 11 审计功能处于启用状态，但您仍然需要执行某些配置步骤。

注 - 不会为虚拟化软件 (`vs`) 类审计先前已存在的进程。确保在一般用户登录系统之前执行此步骤。

---

1. 向 **`/etc/security/audit_event`** 和 **`/etc/security/audit_class`** 文件中添加定制内容。
这些定制内容在 Oracle Solaris 升级之后会保留下来，但是，在全新安装 Oracle Solaris 之后应当重新添加这些定制内容。

   a. 将以下条目添加到 **`audit_event`** 文件中（如果尚未添加）：

   ```
   40700:AUE_ldoms:ldoms administration:vs
   ```

   b. 将以下条目添加到 **`audit_class`** 文件中（如果尚未添加）：

   ```
   0x10000000:vs:virtualization_software
   ```

2. (Oracle Solaris 10) 将 **`vs`** 类添加到 **`/etc/security/audit_control`** 文件中。
以下示例 `/etc/security/audit_control` 片段说明您可以如何指定 `vs` 类：

   ```
   dir:/var/audit
   flags:lo,vs
   minfree:20
   naflags:lo,na
   ```

3. (Oracle Solaris 10) 启用审计功能。

   a. 运行 **`bsmconv`** 命令。

   ```
   # /etc/security/bsmconv
   ```

   b. 重新引导系统。

4. (Oracle Solaris 11) 预选 **vs** 审计类。

   a. 确定已经选择哪些审计类。

      确保已选择的所有审计类均属于更新的类集合。以下示例说明已经选择 `lo` 类：

      ```
      # auditconfig -getflags
      active user default audit flags = lo(0x1000,0x1000)
      configured user default audit flags = lo(0x1000,0x1000)
      ```

   b. 添加 **vs** 审计类。

      ```
      # auditconfig -setflags [class],vs
      ```

      class 为零个或更多审计类，用逗号分隔。您可以在 `/etc/security/audit_class` 文件中查看审计类列表。请确保您的 Oracle VM Server for SPARC 系统包含 `vs` 类。

      例如，以下命令选择 `lo` 和 `vs` 类：

      ```
      # auditconfig -setflags lo,vs
      ```

   c. （可选）无论您要以管理员还是配置者的身份审计进程，均请注销系统。

      如果不想注销，请参见《Oracle Solaris 11.1 Administration: Security Services》中的"How to Update the Preselection Mask of Logged In Users"。

5. 确认审计软件正在运行。

   ```
   # auditconfig -getcond
   ```

   如果审计软件正在运行，则输出中会出现 `audit condition = auditing`。

▼ 如何禁用审计

- 禁用审计功能。

  - Oracle Solaris 10 OS：

    a. 运行 **bsmunconv** 命令。

       ```
       # /etc/security/bsmunconv
       Are you sure you want to continue? [y/n] y
       This script is used to disable the Basic Security Module (BSM).
       Shall we continue the reversion to a non-BSM system now? [y/n] y
       bsmunconv: INFO: removing c2audit:audit_load from /etc/system.
       bsmunconv: INFO: stopping the cron daemon.

       The Basic Security Module has been disabled.
       ```

```
Reboot this system now to come up without BSM.
```

b. 重新引导系统。

- Oracle Solaris 11 OS：

a. 运行 **audit -t** 命令。

```
# audit -t
```

b. 确认审计软件不再运行。

```
# auditconfig -getcond
audit condition = noaudit
```

▼ 如何查看审计记录

- 使用以下方法之一查看 **vs** 审计输出：
  - 使用 **auditreduce** 和 **praudit** 命令查看审计输出。

```
# auditreduce -c vs | praudit
# auditreduce -c vs -a 20060502000000 | praudit
```

  - 使用 **praudit -x** 命令以 XML 格式显示审计记录。

▼ 如何轮转审计日志

- 使用 **audit -n** 命令轮转审计日志。
  轮转审计日志会关闭当前的审计文件并打开当前审计目录中的一个新日志。

使用域控制台日志记录

在 Oracle VM Server for SPARC 环境中，来自 `primary` 域的控制台 I/O 会定向至服务处理器 (service processor, SP)。而来自所有其他域的控制台 I/O 则会重定向至运行虚拟控制台集中器 `vcc` 的服务域。如果服务域运行 Oracle Solaris 11 OS，则来宾域控制台输出可能会记录到一个文件中。

服务域支持逻辑域控制台日志记录。虽然服务域必须运行 Oracle Solaris 11 OS，但要记录日志的来宾域可以运行 Oracle Solaris 10 OS 或 Oracle Solaris 11 OS。

域控制台日志将保存到提供 vcc 服务的服务域 /var/log/vntsd/*domain*/console-log 上的一个文件中。可以使用 logadm 命令轮转控制台日志文件。请参见 logadm(1M) 和 logadm.conf(4) 手册页。

使用 Oracle VM Server for SPARC 软件，您可以有选择地为每个逻辑域启用和禁用控制台日志记录。默认情况下，控制台日志记录将处于启用状态。

▼ 如何启用或禁用控制台日志记录

必须分别为每个逻辑域启用或禁用控制台日志记录，即使这些域属于同一个控制台组也是如此。

1. **列出域的当前控制台设置。**

```
primary# ldm list -o console domain
```

2. **停止并解除绑定域。**

要修改控制台设置，域必须处于非活动和未绑定状态。

```
primary# ldm stop domain
primary# ldm unbind domain
```

3. **启用或禁用控制台日志记录。**

- **启用控制台日志记录。**

```
primary# ldm set-vcons log=on domain
```

- **禁用控制台日志记录。**

```
primary# ldm set-vcons log=off domain
```

域控制台日志记录的服务域要求

如果某个域连接到所运行的 OS 版本早于 Oracle Solaris 11.1 的服务域，则无法对该域进行日志记录。

注 - 即使为某个域启用了控制台日志记录，但如果服务域上不具有所需的支持，也不会对该域的虚拟控制台进行日志记录。

♦ ♦ ♦ 第 3 章

3 设置服务和控制域

本章介绍了设置默认服务和控制域所需的过程。

您也可以使用 Oracle VM Server for SPARC Configuration Assistant 配置逻辑域和服务。请参见第 18 章 Oracle VM Server for SPARC Configuration Assistant (Oracle Solaris 10)。

本章包括以下主题：

输出消息

如果无法在控制域上动态配置某个资源，则建议做法是首先启动延迟重新配置。延迟重新配置会将配置活动推迟到控制域进行重新引导后。

在 `primary` 域上启动延迟重新配置时，您将收到以下消息：

```
Initiating a delayed reconfiguration operation on the primary domain.
All configuration changes for other domains are disabled until the
primary domain reboots, at which time the new configuration for the
primary domain also takes effect.
```

在重新引导之前每次对 `primary` 域执行后续操作之后，将收到以下通知：

```
Notice: The primary domain is in the process of a delayed reconfiguration.
Any changes made to the primary domain will only take effect after it reboots.
```

创建默认服务

必须创建下列虚拟设备服务，以便将控制域作为服务域以及为其他域创建虚拟设备：

- `vcc` – 虚拟控制台集中器服务
- `vds` – 虚拟磁盘服务器
- `vsw` – 虚拟交换机服务

▼ 如何创建默认服务

1. **创建虚拟控制台集中器 (`vcc`) 服务，以供虚拟网络终端服务器守护进程 (`vntsd`) 使用，并将该服务器作为所有逻辑域控制台的集中器。**

   例如，以下命令会将端口范围从 5000 到 5100 的虚拟控制台集中器服务 (`primary-vcc0`) 添加到控制域 (`primary`)。

   ```
   primary# ldm add-vcc port-range=5000-5100 primary-vcc0 primary
   ```

2. **创建虚拟磁盘服务器 (`vds`)，以允许将虚拟磁盘导入到逻辑域中。**

   例如，以下命令会将虚拟磁盘服务器 (`primary-vds0`) 添加到控制域 (`primary`)。

   ```
   primary# ldm add-vds primary-vds0 primary
   ```

3. **创建虚拟交换机服务 (`vsw`)，以启用逻辑域中的虚拟网络 (`vnet`) 设备之间的联网。**

   如果每个逻辑域都必须通过虚拟交换机与外界进行通信，请将符合 GLDv3 的网络适配器分配给虚拟交换机。

   - 在 Oracle Solaris 10 中，将网络适配器上的虚拟交换机服务添加到控制域。

     ```
     primary# ldm add-vsw net-dev=network-device vsw-service primary
     ```

     例如，以下命令会将网络适配器 `nxge0` 上的虚拟交换机服务 (`primary-vsw0`) 添加到控制域 (`primary`)：

     ```
     primary# ldm add-vsw net-dev=nxge0 primary-vsw0 primary
     ```

   - 在 Oracle Solaris 11 中，在要用于来宾域联网的网络设备上添加虚拟交换机服务 (`primary-vsw0`)：

     ```
     primary# ldm add-vsw net-dev=network-device vsw-service primary
     ```

     例如，以下命令会将网络设备 `net0` 上的虚拟交换机服务 (`primary-vsw0`) 添加到控制域 (`primary`)：

     ```
     primary# ldm add-vsw net-dev=net0 primary-vsw0 primary
     ```

可以使用 `ldm list-netdev -b` 命令来确定可用于虚拟交换机的后端网络设备。请参见“虚拟交换机” [189]。

可以通过使用 `ldm set-vsw` 命令动态地更新 `net-dev` 属性值。

- 以下过程仅适用于 Oracle Solaris 10 OS，不应在 Oracle Solaris 11 系统上执行。

  此命令会自动将 MAC 地址分配给虚拟交换机。您可以指定将 MAC 地址作为 `ldm add-vsw` 命令的选项。但是，在这种情况下，您需要负责确保指定的 MAC 地址与现有的 MAC 地址不冲突。

  如果添加的虚拟交换机将取代底层物理适配器作为主网络接口，则必须为其分配物理适配器的 MAC 地址，以便动态主机配置协议 (Dynamic Host Configuration Protocol, DHCP) 服务器会为该域分配相同的 IP 地址。请参见“启用控制/服务域与其他域之间的联网（仅限 Oracle Solaris 10）” [44]。

```
primary# ldm add-vsw mac-addr=2:04:4f:fb:9f:0d net-dev=nxge0 primary-vsw0 primary
```

4. **检验是否已使用 `list-services` 子命令创建服务。**

   输出结果应该与以下内容类似：

```
primary# ldm list-services primary
VDS
    NAME             VOLUME         OPTIONS          DEVICE
    primary-vds0

VCC
    NAME             PORT-RANGE
    primary-vcc0     5000-5100

VSW
    NAME             MAC               NET-DEV   DEVICE     MODE
    primary-vsw0     02:04:4f:fb:9f:0d nxge0     switch@0   prog,promisc
```

控制域的初始配置

最初，所有系统资源都分配给控制域。要允许创建其他逻辑域，您必须释放其中一些资源。

配置控制域

▼ 如何配置控制域

以下过程举例说明了为控制域设置的资源。这些数字仅为示例，并且所使用的值可能不适用于您的控制域。

有关域大小设置建议，请参见 *Oracle VM Server for SPARC Best Practices* (http://www.oracle.com/technetwork/server-storage/vm/ovmsparc-best-practices-2334546.pdf)（Oracle VM Server for SPARC 最佳实践）。

1. **为控制域分配虚拟 CPU。**

   服务域（包括控制域）需要 CPU 和内存资源来执行针对来宾域的虚拟磁盘和虚拟网络 I/O 操作。要分配的 CPU 和内存资源的数量取决于来宾域的工作负荷。

   例如，以下命令将两个 CPU 核心（16 个虚拟 CPU 线程）分配给控制域 `primary`。其余的虚拟 CPU 线程可用于来宾域。

   ```
   primary# ldm set-core 2 primary
   ```

   可以根据应用程序要求动态地更改实际的 CPU 分配。使用 `ldm list` 命令可确定控制域的 CPU 利用率。如果控制域的 CPU 利用率很高，可使用 `ldm add-core` 和 `ldm set-core` 命令向服务域增加 CPU 资源。

2. **确定控制域中是否需要加密设备。**

   请注意，只有 UltraSPARC T2、UltraSPARC T2 Plus 和 SPARC T3 平台才具有加密设备 (MAU)。较新的平台（如 SPARC T4 系统和 Fujitsu M10 服务器）已经提供了加密加速，因此无需为这些平台分配加密加速器。

   如果使用的某个处理器较旧，则为控制域中的每个 CPU 整体核心分配一个加密单元。

   以下示例向控制域 `primary` 分配了两个加密资源：

   ```
   primary# ldm set-crypto 2 primary
   ```

3. **在控制域上启动延迟重新配置。**

   ```
   primary# ldm start-reconf primary
   ```

4. **为控制域分配内存。**

   例如，以下命令向控制域 `primary` 分配 16 GB 内存。此设置会留下其余内存供来宾域使用。

   ```
   primary# ldm set-memory 16G primary
   ```

5. **将域配置保存到服务处理器 (service processor, SP)。**

   例如，以下命令会添加名为 `initial` 的配置。

   ```
   primary# ldm add-config initial
   ```

6. **检验该配置在下次重新引导时是否可以使用。**

   ```
   primary# ldm list-config
   factory-default
   initial [current]
   ```

此 ldm list-config 命令表明执行关开机循环后将使用 initial 配置集。

7. 重新引导控制域以使重新配置更改生效。

减少控制域的初始 factory-default 配置中的 CPU 和内存资源

可以使用 CPU DR 来减少初始 factory-default 配置中控制域的核心数。但是，必须使用延迟重新配置（而不是内存 DR）来减少控制域的内存。

当采用 factory-default 配置时，控制域拥有主机系统的所有内存。之所以不太适合使用内存 DR 功能来实现此目的，是因为并不保证活动域能够添加（或者更典型的情况下能够放弃）所有请求的内存。当然，该域中运行的 OS 会尽最大可能来完成请求。另外，内存删除操作可能要运行很长时间。涉及到大量内存操作时这些问题会扩大化，正如初始减少控制域内存的情况。

注 - 在 ZFS 文件系统上安装了 Oracle Solaris OS 时，它会根据现有的物理内存数量自动确定大小并创建交换区域和转储区域作为 ZFS 根池中的 ZFS 卷。如果更改域的内存分配，则可能会改变这些域的建议大小。在减少控制域内存后，分配可能会大于所需量。有关交换空间建议，请参见《Managing File Systems in Oracle Solaris 11.2 》中的“Planning for Swap Space”。在释放磁盘空间之前，可以有选择地更改交换空间和转储空间。请参见《Managing ZFS File Systems in Oracle Solaris 11.2 》中的“Managing Your ZFS Swap and Dump Devices”。

▼ 如何减少控制域的初始 factory-default 配置中的 CPU 和内存资源

此过程展示如何减少控制域的初始 factory-default 配置中的 CPU 和内存资源。首先使用 CPU DR 减少核心数，然后在启动延迟重新配置后减少内存数。

示例值针对的是小控制域的 CPU 和内存大小，该控制域有足够的资源来运行 ldmd 守护进程以及执行迁移。但是，如果要将控制域用于其他目的，则可根据需要向控制域分配更多核心和内存。

1. 引导 **factory-default** 配置。

2. 配置控制域。

   请参见如何配置控制域 [41]。

重新引导以使用域

必须重新引导控制域才能使配置更改生效，并释放资源供其他逻辑域使用。

▼ 如何重新引导

- 关闭并重新引导控制域。

```
primary# shutdown -y -g0 -i6
```

注 - 重新引导或开关机循环都能够实例化新配置。实际上，只有开关机循环才能引导保存到服务处理器 (service processor, SP) 的配置，该配置将反映在 `list-config` 输出中。

启用控制/服务域与其他域之间的联网（仅限 Oracle Solaris 10）

注意 - 本节仅适用于 Oracle Solaris 10 系统。不要在 Oracle Solaris 11 系统上配置 `vsw` 接口。

默认情况下，会禁用系统中控制域与其他域之间的联网。要启用联网功能，应将虚拟交换机设备配置为网络设备。虚拟交换机可以取代底层物理设备（在此示例中为 `nxge0`）作为主接口，或者配置为域中的附加网络接口。

只要在同一虚拟 LAN 或虚拟网络中配置相应的网络后端设备，来宾域即可与控制域或服务域自动进行通信。

▼ 如何将虚拟交换机配置为主接口

注 - 请从控制域的控制台执行以下过程，因为此过程可能会暂时中断到域的网络连接。

如有必要，可以配置虚拟交换机和物理网络设备。这种情况下，需要像步骤 2 中那样创建虚拟交换机，但不需要删除物理设备（跳过步骤 3）。然后，必须使用静态 IP 地址或动态 IP 地址配置虚拟交换机。可以从 DHCP 服务器获取动态 IP 地址。有关其他信息和这种情况的示例，请参见“针对 NAT 和路由配置虚拟交换机和服务域” [204]。

1. 显示所有接口的寻址信息。

```
primary# ifconfig -a
```

2. 配置虚拟交换机网络接口。

```
primary# ifconfig vsw0 plumb
```

3. 删除设备中分配给虚拟交换机 (**net-dev**) 的物理接口。

```
primary# ifconfig nxge0 down unplumb
```

4. 要将物理网络设备 (**nxge0**) 的属性迁移到虚拟交换机设备 (**vsw0**)，请执行以下操作之一：

   - 如果使用静态 IP 地址配置网络，请对虚拟交换机重用 **nxge0** 的 IP 地址和网络掩码。

     ```
     primary# ifconfig vsw0 IP-of-nxge0 netmask netmask-of-nxge0 broadcast + up
     ```

   - 如果使用 DHCP 配置网络，请为虚拟交换机启用 DHCP。

     ```
     primary# ifconfig vsw0 dhcp start
     ```

5. 进行必需的配置文件修改，使其成为永久性更改。

```
primary# mv /etc/hostname.nxge0 /etc/hostname.vsw0
primary# mv /etc/dhcp.nxge0 /etc/dhcp.vsw0
```

启用虚拟网络终端服务器守护进程

必须启用虚拟网络终端服务器守护进程 (vntsd)，以提供对每个逻辑域的虚拟控制台的访问。有关如何使用此守护进程的信息，请参阅 vntsd(1M) 手册页。

▼ 如何启用虚拟网络终端服务器守护进程

注 - 在启用 vntsd 之前，请确保已在控制域上创建了默认服务 vconscon (vcc)。有关更多信息，请参见“创建默认服务” [40]。

1. 启用虚拟网络终端服务器守护进程 **vntsd**。

```
primary# svcadm enable vntsd
```

2. 验证是否已启用 **vntsd** 守护进程。

```
primary# svcs vntsd
```

```
STATE          STIME    FMRI
online         Oct_08   svc:/ldoms/vntsd:default
```

♦ ♦ ♦ 第 4 章

设置来宾域

本章介绍了设置来宾域所必需的过程。

您也可以使用 Oracle VM Server for SPARC Configuration Assistant 配置逻辑域和服务。请参见第 18 章 Oracle VM Server for SPARC Configuration Assistant (Oracle Solaris 10)。

本章包括以下主题：

创建和启动来宾域

来宾域必须运行既与 sun4v 平台兼容又与虚拟机管理程序提供的虚拟设备兼容的操作系统。目前，此要求意味着必须至少运行 Oracle Solaris 10 11/06 OS。通过运行 Oracle Solaris 10 1/13 OS，您可获得所有的 Oracle VM Server for SPARC 3.2 功能。有关可能需要的任何特定修补程序的信息，请参见《Oracle VM Server for SPARC 3.2 安装指南》。一旦创建了默认服务并重新分配了控制域的资源，您就可以创建和启动来宾域。

注 - 分配的 CPU 多于 1024 个的来宾域无法运行 Oracle Solaris 10 OS。此外，您无法使用 CPU DR 将 CPU 数目缩减到 1024 以下来运行 Oracle Solaris 10 OS。

要解决此问题，请对来宾域解除绑定，删除 CPU，直到 CPU 数目不多于 1024，然后重新绑定来宾域。然后，您可以在此来宾域上运行 Oracle Solaris 10 OS。

▼ 如何创建和启动来宾域

1. **创建逻辑域。**

   以下命令将创建一个名为 ldg1 的来宾域。

```
primary# ldm add-domain ldg1
```

2. 为来宾域添加 CPU。
   执行以下操作之一：

   - 添加虚拟 CPU。
     以下命令将向来宾域 `ldg1` 添加八个虚拟 CPU。

     ```
     primary# ldm add-vcpu 8 ldg1
     ```

   - 添加完整核心。
     以下命令将向来宾域 `ldg1` 添加两个完整核心。

     ```
     primary# ldm add-core 2 ldg1
     ```

3. 为来宾域添加内存。
   以下命令将向来宾域 `ldg1` 添加 2 GB 内存。

   ```
   primary# ldm add-memory 2G ldg1
   ```

4. 为来宾域添加虚拟网络设备。
   以下命令将向来宾域 `ldg1` 添加具有下列特定信息的虚拟网络设备。

   ```
   primary# ldm add-vnet vnet1 primary-vsw0 ldg1
   ```

   其中：

   - `vnet1` 是逻辑域的唯一接口名称，它被分配到此虚拟网络设备实例，供在后续 `set-vnet` 或 `remove-vnet` 子命令上引用。
   - `primary-vsw0` 是要连接到的现有网络服务（虚拟交换机）的名称。

   注 - 步骤 5 和 6 是简化后的说明，主要说明如何将虚拟磁盘服务器设备 `vdsdev` 添加到主域以及如何将虚拟磁盘 `vdisk` 添加到来宾域。要了解如何将 ZFS 卷和文件系统用作虚拟磁盘，请参见如何将 ZFS 卷作为具有单个分片的磁盘导出 [164] 和“将 ZFS 用于虚拟磁盘” [175]。

   ---

5. 指定要由虚拟磁盘服务器导出并用作来宾域的虚拟磁盘的设备。
   可以将物理磁盘、磁盘分片、卷或文件导出为块设备。以下示例对物理磁盘和文件进行了说明。

   - 物理磁盘示例。此示例会添加具有以下特定信息的物理磁盘：

     ```
     primary# ldm add-vdsdev /dev/dsk/c2t1d0s2 vol1@primary-vds0
     ```

     其中：

     - `/dev/dsk/c2t1d0s2` 是实际物理设备的路径名称。添加设备时，路径名必须与设备名称成对出现。

- vol1 是必须为添加到虚拟磁盘服务器的设备指定的唯一名称。该卷名称对于此虚拟磁盘服务器实例必须是唯一的，因为该名称会由此虚拟磁盘服务器导出到客户机以便进行添加。添加设备时，卷名称必须与实际设备的路径名成对出现。
- primary-vds0 是要将此设备添加到的虚拟磁盘服务器的名称。

- **文件示例。**此示例会将文件导出为块设备。

```
primary# ldm add-vdsdev backend vol1@primary-vds0
```

其中：

- *backend* 是导出为块设备的实际文件的路径名称。添加设备时，后端必须与设备名称成对出现。
- vol1 是必须为添加到虚拟磁盘服务器的设备指定的唯一名称。该卷名称对于此虚拟磁盘服务器实例必须是唯一的，因为该名称会由此虚拟磁盘服务器导出到客户机以便进行添加。添加设备时，卷名称必须与实际设备的路径名成对出现。
- primary-vds0 是要将此设备添加到的虚拟磁盘服务器的名称。

6. **为来宾域添加虚拟磁盘。**

以下示例会为来宾域 ldg1 添加虚拟磁盘。

```
primary# ldm add-vdisk vdisk1 vol1@primary-vds0 ldg1
```

其中：

- vdisk1 是虚拟磁盘的名称。
- vol1 是要连接到的现有卷的名称。
- primary-vds0 是要连接到的现有虚拟磁盘服务器的名称。

注 - 虚拟磁盘是与各种类型的物理设备、卷或文件相关联的通用块设备。虚拟磁盘与 SCSI 磁盘不同义，因此磁盘标签中不包含目标 ID。逻辑域中的虚拟磁盘的格式如下：c*N*d*N*s*N*，其中 c*N* 是虚拟控制器，d*N* 是虚拟磁盘号，s*N* 是磁盘分片。

---

7. **为来宾域设置 auto-boot? 和 boot-device 变量。**

以下示例会将来宾域 ldg1 的 auto-boot? 设置为 true。

```
primary# ldm set-var auto-boot\?=true ldg1
```

第一个示例命令会将来宾域 ldg1 的 boot-device? 设置为 vdisk1。

```
primary# ldm set-var boot-device=vdisk1 ldg1
```

8. **将资源绑定到来宾域 ldg1，然后列出该域以检验它是否已绑定。**

```
primary# ldm bind-domain ldg1
primary# ldm list-domain ldg1
NAME          STATE    FLAGS  CONS   VCPU MEMORY   UTIL  UPTIME
ldg1          bound    -----  5000   8    2G
```

9. 要查找来宾域的控制台端口，可以查看上述 **list-domain** 子命令的输出。

   在标题 CONS 下可以看见逻辑域来宾 1 (ldg1) 已将其控制台输出绑定到端口 5000。

10. 通过登录到控制域并直接连接到本地主机的控制台端口，可从其他终端连接到来宾域的控制台。

```
$ ssh hostname.domain-name
$ telnet localhost 5000
```

11. 启动来宾域 **ldg1**。

```
primary# ldm start-domain ldg1
```

在来宾域上安装 Oracle Solaris OS

本节说明了在来宾域上安装 Oracle Solaris OS 的一些不同方法。

注意 - 在 Oracle Solaris OS 安装期间，不要与虚拟控制台断开连接。

对于 Oracle Solaris 11 域，请使用 DefaultFixed 网络配置配置文件 (network configuration profile, NCP)。您可以在安装期间或之后启用此配置文件。

在 Oracle Solaris 11 安装期间，选择“手动”网络配置。Oracle Solaris 11 安装之后，请确保使用 netadm list 命令启用 DefaultFixed NCP。请参见《Connecting Systems Using Fixed Network Configuration in Oracle Solaris 11.1 》和《Connecting Systems Using Reactive Network Configuration in Oracle Solaris 11.1 》。

内存大小要求

在创建域时，Oracle VM Server for SPARC 软件不施加内存大小限制。内存大小要求是客操作系统的特性。如果所提供的内存量小于建议的大小，某些 Oracle VM Server for SPARC 功能可能不起作用。有关 Oracle Solaris 10 OS 的建议内存大小及其最小内存要求，请参见《Oracle Solaris 10 1/13 Installation Guide: Planning for Installation and Upgrade 》中的“System Requirements and Recommendations”。有关 Oracle Solaris 11 OS 的建议内存大小及最小内存要求，请参见《Oracle Solaris 11 Release Notes 》、《Oracle Solaris 11.1 Release Notes 》和《Oracle Solaris 11.2 Release Notes 》。

OpenBoot PROM 对于域有最小大小限制。目前，该限制为 12 MB。对于小于 12 MB 的域，Logical Domains Manager 会自动将该域扩展到 12 MB。Fujitsu M10 服务器的最小大小限制是 256 MB。有关内存大小要求的信息，请参阅系统固件的发行说明。

内存动态重新配置 (Dynamic Reconfiguration, DR) 功能可对给定操作中所涉及内存的地址和大小强制执行 256 MB 对齐。请参见“内存对齐” [268]。

▼ 如何通过 DVD 在来宾域上安装 Oracle Solaris OS

1. 在 DVD 驱动器中插入 Oracle Solaris 10 OS 或 Oracle Solaris 11 OS DVD。

2. 停止 **primary** 域上的卷管理守护进程 **vold**(1M)。

```
primary# svcadm disable volfs
```

3. 停止并解除绑定来宾域 (**ldg1**)。

```
primary# ldm stop ldg1
primary# ldm unbind ldg1
```

4. 将 DVD 与 DVD-ROM 介质添加为辅助卷和虚拟磁盘。

   以下示例会将 `c0t0d0s2` 作为驻留 Oracle Solaris 介质的 DVD 驱动器，将 `dvd_vol@primary-vds0` 作为辅助卷，将 `vdisk_cd_media` 作为虚拟磁盘。

```
primary# ldm add-vdsdev options=ro /dev/dsk/c0t0d0s2 dvd_vol@primary-vds0
primary# ldm add-vdisk vdisk_cd_media dvd_vol@primary-vds0 ldg1
```

5. 验证是否已将 DVD 作为辅助卷和虚拟磁盘添加。

```
primary# ldm list-bindings
NAME             STATE    FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
primary          active   -n-cv   SP      8     8G       0.2%  22h 45m
...
VDS
   NAME             VOLUME         OPTIONS          DEVICE
   primary-vds0     vol1                            /dev/dsk/c2t1d0s2
   dvd_vol                                          /dev/dsk/c0t0d0s2
....
------------------------------------------------------------------------------
NAME             STATE    FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
ldg1             inactive -----           60    6G
...
DISK
   NAME             VOLUME                      TOUT DEVICE  SERVER
   vdisk1           vol1@primary-vds0
   vdisk_cd_media   dvd_vol@primary-vds0
....
```

6. 绑定并启动来宾域 (**ldg1**)。

```
primary# ldm bind ldg1
primary# ldm start ldg1
LDom ldg1 started
primary# telnet localhost 5000
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.

Connecting to console "ldg1" in group "ldg1" ....
Press ~? for control options ..
```

7. 在客户机 OpenBoot PROM 中显示设备别名。

   在此示例中，可看到 `vdisk_cd_media` （即 Oracle Solaris DVD）的设备别名和 `vdisk1` （即可安装 Oracle Solaris OS 的虚拟磁盘）的设备别名。

```
ok devalias
vdisk_cd_media  /virtual-devices@100/channel-devices@200/disk@1
vdisk1          /virtual-devices@100/channel-devices@200/disk@0
vnet1           /virtual-devices@100/channel-devices@200/network@0
virtual-console /virtual-devices/console@1
name            aliases
```

8. 在来宾域的控制台上，将从分片 **f** 上的 **vdisk_cd_media** (**disk@1**) 进行引导。

```
ok boot vdisk_cd_media:f
Boot device: /virtual-devices@100/channel-devices@200/disk@1:f  File and args: -s
SunOS Release 5.10 Version Generic_139555-08 64-bit
Copyright (c), 1983-2010, Oracle and/or its affiliates.  All rights reserved.
```

9. 继续 Oracle Solaris OS 安装。

▼ 如何通过 Oracle Solaris ISO 文件在来宾域上安装 Oracle Solaris OS

1. 停止并解除绑定来宾域 (**ldg1**)。

```
primary# ldm stop ldg1
primary# ldm unbind ldg1
```

2. 将 Oracle Solaris ISO 文件添加为辅助卷和虚拟磁盘。

   以下示例会将 `solarisdvd.iso` 作为 Oracle Solaris ISO 文件，将 `iso_vol@primary-vds0` 作为辅助卷，将 `vdisk_iso` 作为虚拟磁盘。

```
primary# ldm add-vdsdev /export/solarisdvd.iso  iso_vol@primary-vds0
primary# ldm add-vdisk vdisk_iso iso_vol@primary-vds0 ldg1
```

3. 验证是否已将 Oracle Solaris ISO 文件添加为辅助卷和虚拟磁盘。

```
primary# ldm list-bindings
NAME             STATE    FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
primary          active   -n-cv   SP      8     8G       0.2%  22h 45m
...
VDS
   NAME             VOLUME         OPTIONS          DEVICE
   primary-vds0     vol1                            /dev/dsk/c2t1d0s2
   iso_vol                                          /export/solarisdvd.iso
....
--------------------------------------------------------------------------------
NAME             STATE    FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
ldg1             inactive -----           60    6G
...
DISK
   NAME       VOLUME               TOUT ID DEVICE  SERVER  MPGROUP
   vdisk1     vol1@primary-vds0
   vdisk_iso  iso_vol@primary-vds0
....
```

4. 绑定并启动来宾域 (**ldg1**)。

```
primary# ldm bind ldg1
primary# ldm start ldg1
LDom ldg1 started
primary# telnet localhost 5000
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.

Connecting to console "ldg1" in group "ldg1" ....
Press ~? for control options ..
```

5. 在客户机 OpenBoot PROM 中显示设备别名。
   在此示例中，可看到 `vdisk_iso`（即 Oracle Solaris ISO 映像）的设备别名和 `vdisk_install`（即磁盘空间）的设备别名。

```
ok devalias
vdisk_iso       /virtual-devices@100/channel-devices@200/disk@1
vdisk1          /virtual-devices@100/channel-devices@200/disk@0
vnet1           /virtual-devices@100/channel-devices@200/network@0
virtual-console /virtual-devices/console@1
name            aliases
```

6. 在来宾域的控制台上，将从分片 **f** 上的 **vdisk_iso** (**disk@1**) 进行引导。

```
ok boot vdisk_iso:f
Boot device: /virtual-devices@100/channel-devices@200/disk@1:f  File and args: -s
SunOS Release 5.10 Version Generic_139555-08 64-bit
Copyright (c) 1983-2010, Oracle and/or its affiliates.  All rights reserved.
```

7. 继续 Oracle Solaris OS 安装。

▼ 如何在 Oracle Solaris 10 来宾域上使用 Oracle Solaris JumpStart 功能

注 - Oracle Solaris JumpStart 功能仅可用于 Oracle Solaris 10 OS。请参见《Oracle Solaris 10 1/13 Installation Guide: JumpStart Installations》。

要执行 Oracle Solaris 11 OS 的自动安装，您可以使用自动化安装程序 (Automated Installer, AI) 功能。请参见《Transitioning From Oracle Solaris 10 to Oracle Solaris 11.2》。

- **修改 JumpStart 配置文件以反映来宾域的不同磁盘设备名称格式。**

  逻辑域中的虚拟磁盘设备名称不同于物理磁盘设备名称。因为虚拟磁盘设备名称不包含目标 ID (t*N*)。相对于常用 c*N*t*N*d*N*s*N*，虚拟磁盘设备名称改用 c*N*d*N*s*N* 格式。其中 c*N* 是虚拟控制器，d*N* 是虚拟磁盘号，而 s*N* 是分片号码。

  注 - 虚拟磁盘可显示为完整磁盘或具有单个分片的磁盘。通过使用指定多个分区的常规 JumpStart 配置文件，可在完整磁盘上安装 Oracle Solaris OS。具有单个分片的磁盘仅有一个分区 `s0`，使用整个磁盘。要在单个磁盘上安装 Oracle Solaris OS，必须使用拥有单个分区 (/) 的配置文件，此分区使用整个磁盘。无法定义任何其他分区，例如交换区。有关完整磁盘和具有单个分片的磁盘的更多信息，请参见“虚拟磁盘外观” [157]。

  - **用于安装 UFS 根文件系统的 JumpStart 配置文件。**

    请参见《Oracle Solaris 10 1/13 Installation Guide: JumpStart Installations》。

    普通 UFS 配置文件

    ```
    filesys c1t1d0s0 free /
    filesys c1t1d0s1 2048 swap
    filesys c1t1d0s5 120 /spare1
    filesys c1t1d0s6 120 /spare2
    ```

    用于在完整磁盘上安装域的实际 UFS 配置文件

    ```
    filesys c0d0s0 free /
    filesys c0d0s1 2048 swap
    filesys c0d0s5 120 /spare1
    filesys c0d0s6 120 /spare2
    ```

    用于在具有单个分片的磁盘上安装域的实际 UFS 配置文件

    ```
    filesys c0d0s0 free /
    ```

  - **用于安装 ZFS 根文件系统的 JumpStart 配置文件。**

    请参见《Oracle Solaris 10 1/13 Installation Guide: JumpStart Installations》中的第 9 章 “Installing a ZFS Root Pool With JumpStart”。

普通 ZFS 配置文件

```
pool rpool auto 2G 2G c1t1d0s0
```

用于安装域的实际 ZFS 配置文件

```
pool rpool auto 2G 2G c0d0s0
```

♦ ♦ ♦ 第 5 章

5 配置 I/O 域

本章介绍 I/O 域以及如何在 Oracle VM Server for SPARC 环境中配置 I/O 域。

本章包括以下主题：

I/O 域概述

I/O 域对物理 I/O 设备具有直接所有权和直接访问权。可以通过将 PCI EXPRESS (PCIe) 总线、PCIe 端点设备或 PCIe SR-IOV 虚拟功能分配给某个域来创建 I/O 域。使用 `ldm add-io` 命令可将总线、设备或虚拟功能分配给域。

由于以下原因，可能需要配置 I/O 域：

- I/O 域可以直接访问物理 I/O 设备，这样可避免与虚拟 I/O 有关的性能开销。因此，I/O 域上的 I/O 性能更接近于裸机系统上的 I/O 性能。
- I/O 域可托管要供来宾域使用的虚拟 I/O 服务。

有关配置 I/O 域的信息，请参见以下章节中的信息：

- 第 6 章 通过分配 PCIe 总线创建根域
- 第 7 章 使用直接 I/O 创建 I/O 域
- 第 8 章 使用 PCIe SR-IOV 虚拟功能创建 I/O 域
- 第 9 章 使用非 `primary` 根域

注 - 不能迁移具有 PCIe 总线、PCIe 端点设备或 SR-IOV 虚拟功能的域。有关其他迁移限制的信息，请参见第 12 章 迁移域。

创建 I/O 域的一般准则

I/O 域可以直接访问一个或多个 I/O 设备，例如 PCIe 总线、网络接口单元 (network interface unit, NIU)、PCIe 端点设备和 PCIe 单一根 I/O 虚拟化 (single root I/O virtualization, SR-IOV) 虚拟功能。

此类型的对 I/O 设备的直接访问意味着更多的 I/O 带宽可用于：

- 向 I/O 域中的应用程序提供服务
- 向来宾域提供虚拟 I/O 服务

以下基本准则可帮助您有效地利用 I/O 带宽：

- 以 CPU 核心粒度分配 CPU 资源。根据 I/O 域中 I/O 设备的类型和 I/O 设备的数量分配一个或多个 CPU 核心。

  例如，1-Gbps 的以太网设备与 10-Gbps 的以太网设备相比，可能只需较少的 CPU 核心即能使用全部的带宽。
- 遵守内存要求。内存要求取决于分配给域的 I/O 设备类型。建议每个 I/O 设备至少 4 GB。分配的 I/O 设备越多，必须分配的内存也越多。
- 当您使用 PCIe SR-IOV 功能时，对于每个 SR-IOV 虚拟功能，请遵循与其他 I/O 设备相同的准则。因此，请分配一个或多个 CPU 核心和内存（以 GB 为单位）以充分利用可从虚拟功能中获得的带宽。

请注意，创建大量的虚拟功能并将其分配给 CPU 和内存资源不足的域，不可能产生最优的配置。

SPARC 系统（最高包括 SPARC T5 和 SPARC M6 平台）提供中断数量有限，因此 Oracle Solaris 会限制每个设备可以使用的中断数量。默认限制应与典型系统配置的需求匹配，但可能需要为某些系统配置调整此值。有关详细信息，请参见“调整中断限制” [318]。

♦ ♦ ♦ 第 6 章

通过分配 PCIe 总线创建根域

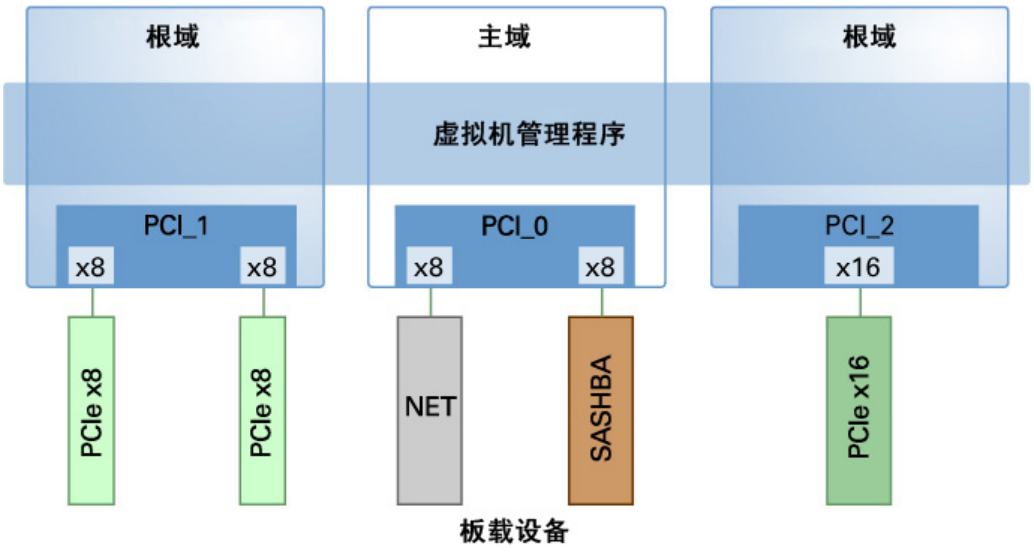
本章介绍如何通过分配 PCIe 总线来创建根域。

通过分配 PCIe 总线创建根域

可以使用 Oracle VM Server for SPARC 软件将完整的 PCIe 总线（也称为根联合体）分配到域。完整 PCIe 总线由 PCIe 总线本身及其所有 PCI 交换机和设备组成。服务器上的 PCIe 总线使用名称标识，例如 `pci@400` (`pci_0`)。配置有完整 PCIe 总线的 I/O 域也称为根域。

下图显示了具有三个根联合体（`pci_0`、`pci_1` 和 `pci_2`）的系统。

图 6-1 将 PCIe 总线分配到根域



使用 PCIe 总线可以创建的最大根域数取决于服务器上可用的 PCIe 总线数。使用 `ldm list-io` 确定系统上的可用 PCIe 总线数。

将 PCIe 总线分配到根域时，该总线上的所有设备都归该根域所有。可以将该总线上的任何 PCIe 端点设备分配到其他域。

在 Oracle VM Server for SPARC 环境中初次配置服务器或使用 `factory-default` 配置时，`primary` 域可访问所有物理设备资源。因此，`primary` 域是系统上配置的唯一根域，全部 PCIe 总线都归其所有。

静态 PCIe 总线分配

根域的静态 PCIe 总线分配方法要求您在分配或删除 PCIe 总线时在根域上启动延迟重新配置。打算对未拥有 PCIe 总线的域使用此方法时，必须先停止该域，然后才能分配 PCIe 总线。在根域上完成配置步骤后，必须重新引导此根域。如果系统中未安装 Oracle VM Server for SPARC 3.2 固件，或者相应域中安装的 OS 版本不支持动态 PCIe 总线分配，则必须使用静态方法。

根域停止或者处于延迟重新配置中时，在重新引导根域之前，可以先运行 `ldm add-io` 和 `ldm remove-io` 命令中的一个或多个。要最大限度地缩短域的停机时间，请在分配或删除 PCIe 总线之前提前进行规划。

- 对于根域，`primary` 和非 `primary` 都使用延迟重新配置。添加或删除了 PCIe 总线后，重新引导域以使更改生效。

  ```
  primary# ldm start-reconf root-domain
  Add or remove the PCIe bus by using the ldm add-io or ldm remove-io command
  primary# ldm stop -r domain-name
  ```

  请注意，仅当域已经拥有 PCIe 总线时才能使用延迟重新配置。
- 对于非根域，停止域，然后添加或删除 PCie 总线。

  ```
  primary# ldm stop domain-name
  Add or remove the PCIe bus by using the ldm add-io or ldm remove-io command
  primary# ldm start domain-name
  ```

动态 PCIe 总线分配

使用动态 PCIe 总线分配功能，可以在根域中动态分配或删除 PCIe 总线。

系统运行必需的固件和软件时启用动态 PCIe 总线分配功能。请参见“动态 PCIe 总线分配要求” [61]。如果系统不运行必需的固件和软件，`ldm add-io` 和 `ldm remove-io` 命令将正常失败。

启用后，可以运行 `ldm add-io` 和 `ldm remove-io` 命令而不需要停止根域或将根域置于延迟重新配置中。

动态 PCIe 总线分配要求

要使用动态 PCIe 总线分配功能，必须满足 Oracle VM Server for SPARC 3.2 发行版的软件和固件要求。

SPARC M5、SPARC M6 和 Fujitsu M10 平台支持此功能。

▼ 如何通过分配 PCIe 总线创建根域

此示例过程说明如何使用初始配置来创建新根域，在该配置中，`primary` 域拥有多个总线。默认情况下，系统上的所有总线都归 `primary` 域所有。此示例适用于 SPARC T4-2 服务器。在其他服务器上也可以使用此过程。虽然面向其他服务器的说明可能与这些说明稍有不同，但是您可以通过此处的示例了解基本原则。

确保不将托管引导磁盘和主网络接口的 PCIe 总线从 `primary` 域中删除。

注意 - 在支持的服务器上，所有的内部磁盘都可以连接到一个 PCIe 总线。如果域从内部磁盘进行引导，请不要将该总线从域中删除。

确保不删除具有由域使用的设备（例如网络端口）的总线。如果错误地删除了总线，则域可能将无法访问所需的设备并变为不可用。要删除具有由域使用的设备的总线，请重新配置该域，以使用其他总线的设备。例如，可能需要重新配置该域，以使用其他板载网络端口或其他 PCIe 插槽中的 PCIe 卡。

在某些 SPARC 服务器上，可以删除包含图形控制器和其他设备的 PCIe 总线。但是，不能向任何其他域添加此类 PCIe 总线。此类 PCIe 总线只能添加到 `primary` 域。

在此示例中，`primary` 域仅使用 ZFS 池 (`rpool`) 和网络接口 (`igb0`)。如果 `primary` 域使用多个设备，请对每个设备重复步骤 2 到步骤 4，以确保没有设备位于将要删除的总线上。

可以通过使用设备路径 (`pci@`*nnn*) 或 pseudonym (`pci_`*n*) 在域中添加或删除总线。`ldm list-bindings primary` 或 `ldm list -l -o physio primary` 命令会显示以下内容：

- `pci@400` 对应于 `pci_0`
- `pci@500` 对应于 `pci_1`
- `pci@600` 对应于 `pci_2`
- `pci@700` 对应于 `pci_3`

1. **检验 `primary` 域是否拥有多个 PCIe 总线。**

```
primary# ldm list-io
NAME                                      TYPE   BUS      DOMAIN   STATUS
----                                      ----   ---      ------   ------
pci_0                                     BUS    pci_0    primary
pci_1                                     BUS    pci_1    primary
pci_2                                     BUS    pci_2    primary
pci_3                                     BUS    pci_3    primary
/SYS/MB/PCIE1                             PCIE   pci_0    primary  EMP
/SYS/MB/SASHBA0                           PCIE   pci_0    primary  OCC
/SYS/MB/NET0                              PCIE   pci_0    primary  OCC
/SYS/MB/PCIE5                             PCIE   pci_1    primary  EMP
/SYS/MB/PCIE6                             PCIE   pci_1    primary  EMP
/SYS/MB/PCIE7                             PCIE   pci_1    primary  EMP
/SYS/MB/PCIE2                             PCIE   pci_2    primary  EMP
/SYS/MB/PCIE3                             PCIE   pci_2    primary  EMP
/SYS/MB/PCIE4                             PCIE   pci_2    primary  EMP
/SYS/MB/PCIE8                             PCIE   pci_3    primary  EMP
/SYS/MB/SASHBA1                           PCIE   pci_3    primary  OCC
/SYS/MB/NET2                              PCIE   pci_3    primary  OCC
/SYS/MB/NET0/IOVNET.PF0                   PF     pci_0    primary
/SYS/MB/NET0/IOVNET.PF1                   PF     pci_0    primary
/SYS/MB/NET2/IOVNET.PF0                   PF     pci_3    primary
/SYS/MB/NET2/IOVNET.PF1                   PF     pci_3    primary
```

2. 确定必须保留的引导磁盘的设备路径。

   - 对于 UFS 文件系统，请运行 **df /** 命令，以确定引导磁盘的设备路径。

     ```
     primary# df /
     /                  (/dev/dsk/c0t5000CCA03C138904d0s0):22755742 blocks  2225374 files
     ```

   - 对于 ZFS 文件系统，首先运行 **df /** 命令以确定池名称。然后，运行 **zpool status** 命令以确定引导磁盘的设备路径。

     ```
     primary# zpool status rpool
       pool: rpool
      state: ONLINE
       scan: none requested
     config:

            NAME                       STATE     READ WRITE CKSUM
            rpool                      ONLINE       0     0     0
              c0t5000CCA03C138904d0s0  ONLINE       0     0     0
     ```

3. 获取有关系统引导磁盘的信息。

   - 对于使用 Solaris I/O 多路径管理的磁盘，使用 **mpathadm** 命令确定连接有引导磁盘的 PCIe 总线。

     从 SPARC T3 服务器开始，内部磁盘通过 Solaris I/O 多路径来管理。

a. 查找磁盘连接到的启动器端口。

```
primary# mpathadm show lu /dev/rdsk/c0t5000CCA03C138904d0s0
Logical Unit:  /dev/rdsk/c0t5000CCA03C138904d0s2
        mpath-support:  libmpscsi_vhci.so
        Vendor:  HITACHI
        Product:  H106030SDSUN300G
        Revision:  A2B0
        Name Type:  unknown type
        Name:  5000cca03c138904
        Asymmetric:  no
        Current Load Balance:  round-robin
        Logical Unit Group ID:  NA
        Auto Failback:  on
        Auto Probing:  NA

        Paths:
                Initiator Port Name:  w50800200014100c8
                Target Port Name:  w5000cca03c138905
                Override Path:  NA
                Path State:  OK
                Disabled:  no

        Target Ports:
                Name:  w5000cca03c138905
                Relative ID:  0
```

b. 确定启动器端口所在的 PCIe 总线。

```
primary# mpathadm show initiator-port w50800200014100c8
Initiator Port:  w50800200014100c8
        Transport Type:  unknown
        OS Device File:  /devices/pci@400/pci@2/pci@0/pci@e/scsi@0/iport@1
```

- 对于未使用 Solaris I/O 多路径管理的磁盘，使用 **`ls -l`** 命令确定块设备所链接的物理设备。

  请对 UltraSPARC T2 或 UltraSPARC T2 Plus 系统上未使用 Solaris I/O 多路径来管理的磁盘使用此命令。

  以下示例使用块设备 `c1t0d0s0`：

```
primary# ls -l /dev/dsk/c0t1d0s0
lrwxrwxrwx   1 root     root          49 Oct  1 10:39 /dev/dsk/c0t1d0s0 ->
../../devices/pci@400/pci@0/pci@1/scsi@0/sd@1,0:a
```

  在此示例中，用作 `primary` 域的引导磁盘的物理设备会连接到 `pci@400` 总线。

4. 确定由系统使用的网络接口。

   使用 `ifconfig` 命令确定“已激活的”主网络接口。激活的接口设置了流，以使 IP 协议可以使用此设备。

- Oracle Solaris 10：

```
primary# ifconfig -a
lo0: flags=2001000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
igb0: flags=1004843<UP,BROADCAST,RUNNING,MULTICAST,DHCP,IPv4> mtu 1500 index 3
        inet 10.129.241.135 netmask ffffff00 broadcast 10.129.241.255
        ether 0:10:e0:e:f1:78
```

- Oracle Solaris 11：

```
primary# ifconfig -a
lo0: flags=2001000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
net0: flags=1004843<UP,BROADCAST,RUNNING,MULTICAST,DHCP,IPv4> mtu 1500 index 3
        inet 10.129.241.135 netmask ffffff00 broadcast 10.129.241.255
        ether 0:10:e0:e:f1:78

primary# dladm show-phys net0
LINK              MEDIA               STATE      SPEED  DUPLEX    DEVICE
net0              Ethernet            up         1000   full      igb0
```

5. 确定网络接口连接到的物理设备。

   以下命令使用 `igb0` 网络接口：

```
primary# ls -l /dev/igb0
lrwxrwxrwx   1 root     root          46 Oct  1 10:39 /dev/igb0  ->
../devices/pci@500/pci@0/pci@c/network@0:igb0
```

   还执行 `ls -l /dev/usbecm` 命令。

   在此示例中，域 `primary` 使用的网络接口的物理设备位于总线 `pci@500` 下，该总线对应于前面列出的 `pci_1`。这样，由于其他两个总线 `pci_2` (`pci@600`) 和 `pci_3` (`pci@700`) 未被 `primary` 域使用，因此可以将它们安全地分配到其他域。

   如果 `primary` 域使用的网络接口位于要分配到其他域的总线上，请重新配置 `primary` 域，使其使用其他网络接口。

6. 将不包含引导磁盘或网络接口的总线从 **`primary`** 域删除。

   在此示例中，将从 `primary` 域中删除总线 `pci_2`。

   - 动态方法：

     确保 `pci_2` 总线中的设备未由 `primary` 域 OS 使用。如果它们被使用，此命令可能无法删除该总线。使用静态方法强行删除 `pci_2` 总线。

```
primary# ldm remove-io pci_2 primary
```

   - 静态方法:

删除总线之前，必须先启动延迟重新配置。

```
primary# ldm start-reconf primary
primary# ldm remove-io pci_2 primary
primary# shutdown -y -g0 -i6
```

不能将 primary 域用于引导磁盘和网络设备的总线分配到其他域。可将其他任何总线分配到其他域。在此示例中，primary 域未使用 pci@600，因此可将其重新分配到其他域。

7. 向域添加总线。

   在此示例中，向 ldg1 域添加 pci_2 总线。

   - 动态方法：

     ```
     primary# ldm add-io pci_2 ldg1
     ```

   - 静态方法:

     删除总线之前，必须先停止域。

     ```
     primary# ldm stop-domain ldg1
     primary# ldm add-io pci_2 ldg1
     primary# ldm start-domain ldg1
     ```

8. 将此配置保存到服务处理器。

   在此示例中，配置为 io-domain。

   ```
   primary# ldm add-config io-domain
   ```

   此配置 io-domain 还设置为重新引导后要使用的下一个配置。

9. 确认仍然为 **primary** 域分配了相应的总线，并为 **ldg1** 域分配了相应的总线。

```
primary# ldm list-io
NAME                                      TYPE   BUS      DOMAIN   STATUS
----                                      ----   ---      ------   ------
pci_0                                     BUS    pci_0    primary
pci_1                                     BUS    pci_1    primary
pci_2                                     BUS    pci_2    ldg1
pci_3                                     BUS    pci_3    primary
/SYS/MB/PCIE1                             PCIE   pci_0    primary  EMP
/SYS/MB/SASHBA0                           PCIE   pci_0    primary  OCC
/SYS/MB/NET0                              PCIE   pci_0    primary  OCC
/SYS/MB/PCIE5                             PCIE   pci_1    primary  EMP
/SYS/MB/PCIE6                             PCIE   pci_1    primary  EMP
/SYS/MB/PCIE7                             PCIE   pci_1    primary  EMP
/SYS/MB/PCIE2                             PCIE   pci_2    ldg1     EMP
/SYS/MB/PCIE3                             PCIE   pci_2    ldg1     EMP
/SYS/MB/PCIE4                             PCIE   pci_2    ldg1     EMP
/SYS/MB/PCIE8                             PCIE   pci_3    primary  EMP
/SYS/MB/SASHBA1                           PCIE   pci_3    primary  OCC
```

```
/SYS/MB/NET2                            PCIE   pci_3    primary  OCC
/SYS/MB/NET0/IOVNET.PF0                 PF     pci_0    primary
/SYS/MB/NET0/IOVNET.PF1                 PF     pci_0    primary
/SYS/MB/NET2/IOVNET.PF0                 PF     pci_3    primary
/SYS/MB/NET2/IOVNET.PF1                 PF     pci_3    primary
```

此输出确认已将 PCIe 总线 `pci_0`、`pci_1` 和 `pci_3` 及其设备分配到 `primary` 域。它还确认已将 PCIe 总线 `pci_2` 及其设备分配到 `ldg1` 域。

♦ ♦ ♦ 第 7 章

使用直接 I/O 创建 I/O 域

本章包括以下直接 I/O 主题：

通过分配 PCIe 端点设备创建 I/O 域

您可以将单个 PCIe 端点（或可分配的直接 I/O）设备分配到域。这种 PCIe 端点设备的使用增加了将设备分配到 I/O 域的粒度。这种功能是通过直接 I/O (Direct I/O, DIO) 功能提供的。

通过 DIO 功能，您可以在系统中创建比 PCIe 总线数更多的 I/O 域。可能的 I/O 域数当前仅受 PCIe 端点设备数限制。

PCIe 端点设备可以是下列任意一个：

- 插槽中的 PCIe 卡
- 由平台标识的板载 PCIe 设备

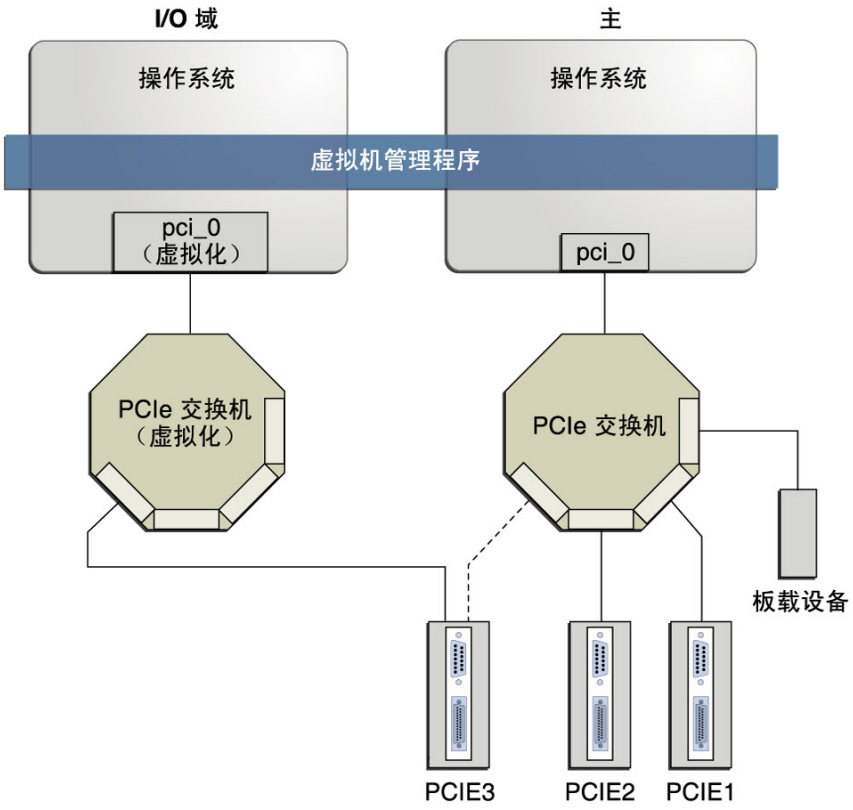
注 - 由于根域之间不能具有依赖关系，因此，拥有 PCIe 总线的根域不能将其 PCIe 端点设备或 SR-IOV 虚拟功能分配给另一个根域。不过，可以将 PCIe 总线中的 PCIe 端点设备或虚拟功能分配给拥有该总线的根域。

下图显示了将 PCIe 端点设备 `PCIE3` 分配到 I/O 域。I/O 域中的总线 `pci_0` 和交换机都是虚拟的。不可再在 `primary` 域中访问`PCIE3` 端点设备。

在 I/O 域中，`pci_0` 块和交换机分别为虚拟根联合体和虚拟 PCIe 交换机。该块和交换机类似于 `primary` 域中的 `pci_0` 块和交换机。在 `primary` 域中，插槽 `PCIE3` 中的设备是原始设备的“影子”格式，并标识为 `SUNW,assigned`。

注意 - 在使用 `ldm remove-io` 命令从 `primary` 域中删除某个 PCIe 端点设备之后，就无法通过 Oracle Solaris 热插拔操作“热删除”该设备。有关更换或删除 PCIe 端点设备的信息，请参见“更改 PCIe 硬件” [73]。

图 7-1 将 PCIe 端点设备分配到 I/O 域



使用 `ldm list-io` 命令列出 PCIe 端点设备。

即使 DIO 功能允许将插槽中的任意 PCIe 卡分配到 I/O 域，也仅支持某些 PCIe 卡。请参见“直接 I/O 硬件和软件要求” [69]。

注意 - 不支持具有桥的 PCIe 卡。也不支持 PCIe 函数级分配。将不受支持的 PCIe 卡分配到 I/O 域可能会导致不可预测的行为。

以下各项介绍了有关 DIO 功能的重要详细信息：

- 仅当满足所有软件要求时才会启用此功能。请参见“直接 I/O 硬件和软件要求” [69]。
- 使用 DIO 功能，只可将连接到已分配至根域的 PCIe 总线的 PCIe 端点分配到其他域。
- 只有当根域正在运行时，使用 DIO 的 I/O 域才能访问 PCIe 端点设备。
- 重新引导根域会对具有 PCIe 端点设备的 I/O 域产生影响。请参见“重新引导配置了 PCIe 端点的根域” [71]。根域还执行以下任务：
  - 初始化和管理 PCIe 总线。
  - 处理所有由分配到 I/O 域的 PCIe 端点设备触发的总线错误。请注意，只有 `primary` 域才会收到所有与 PCIe 总线相关的错误。

直接 I/O 硬件和软件要求

要成功使用直接 I/O (Direct I/O, DIO) 功能将直接 I/O 设备指定给域，必须运行适当的软件并使用支持的 PCIe 卡。

- **硬件要求。**只有某些 PCIe 卡才能用作 I/O 域上的直接 I/O 端点设备。在 Oracle VM Server for SPARC 环境中仍可以使用其他卡，但是这些卡不能用于 DIO 功能，而是用于指定了整个根联合体的服务域和 I/O 域。

  请参阅您平台的硬件文档来确认可以在您平台上使用的卡。有关受支持的 PCIe 卡的最新列表，请参见 `https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&doctype=REFERENCE&id=1325454.1`。
- **软件要求。**要使用 DIO 功能，以下域必须运行支持的 OS：
  - **根域。**至少 Oracle Solaris 10 9/10 OS 加修补程序 ID 145868-01 或 Oracle Solaris 11 OS。

    建议的做法是所有域至少运行 Oracle Solaris 10 1/13 OS 加http://www.oracle.com/pls/topic/lookup?ctx=E56447&id=LDSIGreqdrecommendedsolarisos 中的必需修补程序或运行 Oracle Solaris 11.1.10.5.0 OS。
  - **I/O 域。**平台支持的任何 Oracle Solaris OS。

注 - 在根域中，支持某个平台支持的所有 PCIe 卡。有关支持的 PCIe 卡列表，请参见平台的文档。但是，只能将支持直接 I/O 的 PCIe 卡分配到 I/O 域。

要通过使用直接 I/O 功能添加或删除 PCIe 端点设备，必须首先对 PCIe 总线本身启用 I/O 虚拟化。

可以使用 `ldm set-io` 或 `ldm add-io` 命令将 `iov` 属性设置为 `on`。还可以使用 `ldm add-domain` 或 `ldm set-domain` 命令将 `rc-add-policy` 属性设置为 `iov`。请参见 `ldm`(1M) 手册页。

重新引导根域会影响直接 I/O，因此，请仔细规划直接 I/O 配置更改，以最大限度地增加对根域的直接 I/O 相关更改，最大限度地减少根域重新引导。

当前直接 I/O 功能限制

有关如何解决限制的信息，请参见“规划 PCIe 端点设备配置” [70]。

只有当任何非根域已停止或处于非活动状态时，才允许向该域分配 PCIe 端点设备或从该域中删除该端点设备。

注 - Fujitsu M10 服务器 支持动态重新配置 PCIe 端点设备。可以在不重新引导根域或停止 I/O 域的情况下分配或删除 PCIe 端点设备。

有关此功能的最新信息，请参见《*Fujitsu M10/SPARC M10 Systems System Operation and Administration Guide*》中的相关型号，该文档位于 `http://www.fujitsu.com/global/services/computing/server/sparc/downloads/manual/`。

SPARC 系统（最高包括 SPARC T5 和 SPARC M6 平台）提供中断数量有限，因此 Oracle Solaris 会限制每个设备可以使用的中断数量。默认限制应与典型系统配置的需求匹配，但可能需要为某些系统配置调整此值。有关详细信息，请参见“调整中断限制” [318]。

规划 PCIe 端点设备配置

分配或删除 PCIe 端点设备时提前仔细规划以避免根域宕机。重新引导根域不仅会影响根域本身提供的服务，还会影响已分配有 PCIe 端点设备的 I/O 域。即使对每个 I/O 域的更改不会影响其他域，提前规划有助于最大限度地减小对由该域所提供的服务的影响。

在延迟重新配置期间，可以继续添加或删除更多设备，然后只需重新引导一次根域，便可使所有更改生效。

有关示例，请参见如何通过分配 PCIe 端点设备创建 I/O 域 [75]。

要规划和执行 DIO 设备配置，必须执行以下常规步骤：

1. 了解和记录系统硬件配置。

   具体地说，记录系统中有关 PCIe 卡的部件号及其他详细信息的信息。

   使用 `ldm list-io -l` 和 `prtdiag -v` 命令获取信息并保存起来供将来参考。
2. 确定 `primary` 域中所需的 PCIe 端点设备。

   例如，确定提供对以下内容的访问权限的 PCIe 端点设备：

- 引导磁盘设备
- 网络设备
- `primary` 域提供作为服务的其他设备

3. 删除可能会在 I/O 域中使用的所有 PCIe 端点设备。

   由于重新引导会对 I/O 域产生影响，因此，此步骤有助于您避免对根域执行后续重新引导操作。

   使用 `ldm remove-io` 命令删除 PCIe 端点设备。使用 pseudonyms 而非设备路径将设备指定到 `remove-io` 和 `add-io` 子命令。

注 - 在延迟重新配置期间删除所需的所有设备后，只需要重新引导一次根域，便可使所有更改生效。

---

4. 将此配置保存到服务处理器 (service processor, SP)。

   使用 `ldm add-config` 命令。

5. 重新引导根域以释放在步骤 3 中删除的 PCIe 端点设备。
6. 确认不再将删除的 PCIe 端点设备分配到根域。

   使用 `ldm list-io -l` 命令检验删除的设备在输出中是否显示为 `SUNW,assigned-device`。

7. 将可用的 PCIe 端点设备分配到来宾域，以提供对物理设备的直接访问权。

   分配完成后，将无法再通过域迁移功能将来宾域迁移到其他物理系统。

8. 将 PCIe 端点设备添加到来宾域或将其从来宾域删除。

   使用 `ldm add-io` 命令。

   通过减少重新引导操作并避免由 I/O 域提供的服务宕机来最大限度地减少对该域的更改。

9. （可选）更改 PCIe 硬件。

   请参见“更改 PCIe 硬件” [73]。

重新引导配置了 PCIe 端点的根域

PCIe 总线属于根域，该域负责初始化和管理该总线。根域必须处于活动状态，并在运行支持 DIO 或 SR-IOV 功能的 Oracle Solaris OS 版本。关闭、停止或重新引导根域将中断对 PCIe 总线的访问。PCIe 总线不可用时，该总线上的 PCIe 设备会受到影响，可能会变为不可用。

当正在运行具有 PCIe 端点设备的 I/O 域时，如果重新引导根域，则这些 I/O 域将出现不可预测的行为。例如，具有 PCIe 端点设备的 I/O 域可能会在重新引导过程中或在重新引导后出现紧急情况。重新引导根域后，您可能需要手动停止并启动每个域。

注 - 当关联的根域没有运行时，I/O 域无法启动。

要解决这些问题，请执行以下步骤之一：

- 在关闭根域之前，请手动关闭系统上已分配有 PCIe 端点设备的任何域。

  此步骤可确保您在关闭、停止或重新引导根域之前完全关闭这些域。

  要查找已将 PCIe 端点设备分配到其中的所有域，请运行 `ldm list-io` 命令。通过此命令，可以列出系统上已分配到域的 PCIe 端点设备。有关此命令输出的详细说明，请参见 `ldm`(1M) 手册页。

  对于找到的每个域，请通过运行 `ldm stop` 命令停止它。
- 配置根域和已分配有 PCIe 端点设备的域之间的域依赖关系。

  这种依赖关系可确保具有 PCIe 端点设备的域会在根域因某种原因而重新引导时自动重新启动。

  请注意，此依赖关系会强行重设那些域，并且那些域不能完全关闭。但是，此依赖关系不会对手动关闭的任何域产生影响。

  ```
  primary# ldm set-domain failure-policy=reset primary
  primary# ldm set-domain master=primary domain-name
  ```

例 7-1 为使用非 `primary` 根域和 I/O 域的配置环境配置故障策略依赖关系

以下示例介绍如何在使用非 `primary` 根域和 I/O 域的配置环境下配置故障策略依赖关系。

在此示例中，`ldg1` 是一个非 `primary` 根域。`ldg2` 是一个 I/O 域，该 I/O 域中的 PCIe SR-IOV 虚拟功能或 PCIe 端点设备是从 `ldg1` 域所拥有的根联合体分配的。

```
primary# ldm set-domain failure-policy=stop ldg1
primary# ldm set-domain master=ldg1 ldg2
```

这种依赖关系可确保该 I/O 域会在 `ldg1` 域重新引导时停止。

- 如果非 `primary` 根域重新引导，则此依赖关系可确保该 I/O 域停止。请在非 `primary` 根域引导后启动该 I/O 域。

  ```
  primary# ldm start ldg2
  ```
- 如果 `primary` 域重新引导，则此策略设置会同时停止非 `primary` 根域和所依赖的 I/O 域。`primary` 域引导后，必须先启动非 `primary` 根域。在该域引导后，请启动 I/O 域。

  ```
  primary# ldm start ldg1
  ```

  请等待 `ldg1` 域处于活动状态，然后再启动 I/O 域。

  ```
  primary# ldm start ldg2
  ```

更改 PCIe 硬件

以下步骤有助于避免错误地配置 PCIe 端点分配。有关安装和删除特定硬件的平台特定信息，请参见平台的文档。

- 如果要将 PCIe 卡安装到空插槽内，无需执行任何操作。此 PCIe 卡自动归拥有 PCIe 总线的域所有。

  要将新的 PCIe 卡分配到 I/O 域，请使用 `ldm remove-io` 命令先从根域中将该卡删除。然后，使用 `ldm add-io` 命令将卡分配到 I/O 域。
- 如果要将 PCIe 卡从系统中删除并将其分配到根域，则无需执行任何操作。
- 要删除已分配到某个 I/O 域的 PCIe 卡，请先将设备从 I/O 域删除。然后，将该设备添加到根域，然后再将其从系统中删除。
- 要替换已分配到某个 I/O 域的 PCIe 卡，请检验 DIO 功能是否支持新卡。

  如果支持，则会将新卡自动分配到当前 I/O 域而无需执行任何操作。

  如果不支持，请先使用 `ldm remove-io` 命令将该 PCIe 卡从 I/O 域删除。接下来，使用 `ldm add-io` 命令将 PCIe 卡重新分配到根域。之后，使用其他 PCIe 卡物理替换已分配到根域的 PCIe 卡。通过这些步骤，您可以避免不受 DIO 功能支持的配置。

在移除 PCIe 卡时最大限度减少来宾域中断

在移除或更换运行 Oracle VM Server for SPARC 软件的系统中的 PCIe 卡时，依赖于此硬件的域将不可用。要最大限度减少此类来宾域中断，必须让系统做好使用热插拔功能的准备以便将卡以物理方式移除。

▼ 如何在移除 PCIe 卡时最大限度减少来宾域中断

使用此过程，可以避免没有分配直接 I/O 或 SR-IOV 设备且配置了多个路径的来宾域中断。请注意，此过程要求重新引导 `primary` 域两次。

注 - 此过程在 PCIe 卡位于非 `primary` 根域拥有的根联合体上时不适用。有关替代方法，请参见How to Replace PCIe Direct I/O Cards Assigned to an Oracle VM Server for SPARC Guest Domain (Doc ID 1684273.1) (`https://support.oracle.com/epmos/faces/DocumentDisplay?_afrLoop=226878266536565&id=1684273.1&_adf.ctrl-state=bo9fbmr1n_49`)（如何更换分配到 Oracle VM Server for SPARC 来宾域的 PCIe 直接 I/O 卡（文档号 1684273.1））。

---

1. 停止分配有 PCIe 插槽的来宾域。

```
primary# ldm stop domain-name
```

2. 从该来宾域中删除 PCIe 插槽。

```
primary# ldm remove-io PCIe-slot domain-name
```

3. 停止分配有 PCIe 插槽和 SR-IOV 虚拟功能的来宾域。

```
primary# ldm stop domain-name
```

注 - 不需要停止分配有 PCIe 总线的来宾域，因为他们可能向来宾域提供了网络和磁带设备的替代路径。

---

4. 在 **primary** 域上启动延迟重新配置以便向其分配此插槽。

```
primary# ldm start-reconf primary
```

5. 将 PCIe 插槽添加到 **primary** 域。

```
primary# ldm add-io PCIe-slot domain-name
```

6. 重新引导 **primary** 域。

```
primary# shutdown -i6 -g0 -y
```

7. 使用热插拔命令更换 PCIe 卡。

   有关 Oracle Solaris OS 热插拔功能的信息，请参见《Managing Devices in Oracle Solaris 11.2》中的第 2 章 “Dynamically Configuring Devices”。

8. 在更换了卡之后，如果必须将同一 PCIe 插槽重新分配给来宾域，请执行以下步骤：

   a. 在 **primary** 域上启动延迟重新配置。

   ```
   primary# ldm start-reconf primary
   ```

   b. 从 **primary** 域中删除 PCIe 插槽。

   ```
   primary# ldm remove-io PCIe-slot domain-name
   ```

   c. 重新引导 **primary** 域以使 PCIe 插槽删除生效。

   ```
   primary# shutdown -i6 -g0 -y
   ```

   d. 将 PCIe 插槽重新分配给来宾域。

   ```
   primary# ldm add-io PCIe-slot domain-name
   ```

   e. 启动要向其分配 PCIe 插槽和 SR-IOV 虚拟功能的来宾域。

   ```
   primary# ldm start domain-name
   ```

通过分配 PCIe 端点设备创建 I/O 域

▼ 如何通过分配 PCIe 端点设备创建 I/O 域

提前规划所有 DIO 部署，从而最大限度地缩短停机时间。

注意 - 如果您将 SPARC T3-1 或 SPARC T4-1 系统上的 `/SYS/MB/SASHBA1` 槽分配给 DIO 域，那么 `primary` 域将失去对于板载 DVD 设备的访问权限。

SPARC T3-1 和 SPARC T4-1 系统包括两个用于板载存储的 DIO 槽，它们由 `/SYS/MB/SASHBA0` 和 `/SYS/MB/SASHBA1` 路径表示。除了承载多显示端板载磁盘之外，`/SYS/MB/SASHBA1` 插槽还会承载板载 DVD 设备。所以，如果您将 `/SYS/MB/SASHBA1` 分配给 DIO 域，那么 `primary` 域将失去对于板载 DVD 设备的访问权限。

SPARC T3-2 和 SPARC T4-2 系统具有承载所有板载磁盘以及板载 DVD 设备的单一 `SASHBA` 槽。因此，如果您将 `SASHBA` 分配给 DIO 域，则板载磁盘和板载 DVD 设备将借给 DIO 域，而无法由 `primary` 域使用。

有关通过添加 PCIe 端点设备来创建 I/O 域的示例，请参见“规划 PCIe 端点设备配置” [70]。

注 - 在本发行版中，请使用 `DefaultFixed` NCP 在 Oracle Solaris 11 系统上配置数据链路和网络接口。

Oracle Solaris 11 OS 包括以下 NCP：

- `DefaultFixed` – 允许您使用 `dladm` 或 `ipadm` 命令管理网络
- `Automatic` – 允许您使用 `netcfg` 或 `netadm` 命令管理网络

使用 `netadm list` 命令确保已启用 `DefaultFixed` NCP。请参见《Oracle Solaris Administration: Network Interfaces and Network Virtualization》中的第 7 章 “Using Datalink and Interface Configuration Commands on Profiles”。

1. **标识和归档系统上当前安装的设备。**

   `ldm list-io -l` 命令的输出显示了当前配置 I/O 设备的方式。可使用 `prtdiag -v` 命令获取更多详细信息。

注 - 将设备分配到 I/O 域之后，只可在 I/O 域中确定设备的标识。

```
primary# ldm list-io -l
NAME                                      TYPE   BUS      DOMAIN   STATUS
----                                      ----   ---      ------   ------
niu_0                                     NIU    niu_0    primary
```

```
[niu@480]
niu_1                                   NIU    niu_1    primary
[niu@580]
pci_0                                   BUS    pci_0    primary
[pci@400]
pci_1                                   BUS    pci_1    primary
[pci@500]
/SYS/MB/PCIE0                           PCIE   pci_0    primary  OCC
[pci@400/pci@2/pci@0/pci@8]
    SUNW,emlxs@0/fp/disk
    SUNW,emlxs@0/fp/tape
    SUNW,emlxs@0/fp@0,0
    SUNW,emlxs@0,1/fp/disk
    SUNW,emlxs@0,1/fp/tape
    SUNW,emlxs@0,1/fp@0,0
/SYS/MB/PCIE2                           PCIE   pci_0    primary  OCC
[pci@400/pci@2/pci@0/pci@4]
    pci/scsi/disk
    pci/scsi/tape
    pci/scsi/disk
    pci/scsi/tape
/SYS/MB/PCIE4                           PCIE   pci_0    primary  OCC
[pci@400/pci@2/pci@0/pci@0]
    ethernet@0
    ethernet@0,1
    SUNW,qlc@0,2/fp/disk
    SUNW,qlc@0,2/fp@0,0
    SUNW,qlc@0,3/fp/disk
    SUNW,qlc@0,3/fp@0,0
/SYS/MB/PCIE6                           PCIE   pci_0    primary  EMP
[pci@400/pci@1/pci@0/pci@8]
/SYS/MB/PCIE8                           PCIE   pci_0    primary  EMP
[pci@400/pci@1/pci@0/pci@c]
/SYS/MB/SASHBA                          PCIE   pci_0    primary  OCC
[pci@400/pci@2/pci@0/pci@e]
    scsi@0/iport@1
    scsi@0/iport@2
    scsi@0/iport@4
    scsi@0/iport@8
    scsi@0/iport@80/cdrom@p7,0
    scsi@0/iport@v0
/SYS/MB/NET0                            PCIE   pci_0    primary  OCC
[pci@400/pci@1/pci@0/pci@4]
    network@0
    network@0,1
/SYS/MB/PCIE1                           PCIE   pci_1    primary  OCC
[pci@500/pci@2/pci@0/pci@a]
    SUNW,qlc@0/fp/disk
    SUNW,qlc@0/fp@0,0
    SUNW,qlc@0,1/fp/disk
    SUNW,qlc@0,1/fp@0,0
/SYS/MB/PCIE3                           PCIE   pci_1    primary  OCC
[pci@500/pci@2/pci@0/pci@6]
    network@0
```

```
    network@0,1
    network@0,2
    network@0,3
/SYS/MB/PCIE5                              PCIE   pci_1    primary  OCC
[pci@500/pci@2/pci@0/pci@0]
    network@0
    network@0,1
/SYS/MB/PCIE7                              PCIE   pci_1    primary  EMP
[pci@500/pci@1/pci@0/pci@6]
/SYS/MB/PCIE9                              PCIE   pci_1    primary  EMP
[pci@500/pci@1/pci@0/pci@0]
/SYS/MB/NET2                               PCIE   pci_1    primary  OCC
[pci@500/pci@1/pci@0/pci@5]
    network@0
    network@0,1
    ethernet@0,80
/SYS/MB/NET0/IOVNET.PF0                    PF     pci_0    primary
[pci@400/pci@1/pci@0/pci@4/network@0]
    maxvfs = 7
/SYS/MB/NET0/IOVNET.PF1                    PF     pci_0    primary
[pci@400/pci@1/pci@0/pci@4/network@0,1]
    maxvfs = 7
/SYS/MB/PCIE5/IOVNET.PF0                   PF     pci_1    primary
[pci@500/pci@2/pci@0/pci@0/network@0]
    maxvfs = 63
/SYS/MB/PCIE5/IOVNET.PF1                   PF     pci_1    primary
[pci@500/pci@2/pci@0/pci@0/network@0,1]
    maxvfs = 63
/SYS/MB/NET2/IOVNET.PF0                    PF     pci_1    primary
[pci@500/pci@1/pci@0/pci@5/network@0]
    maxvfs = 7
/SYS/MB/NET2/IOVNET.PF1                    PF     pci_1    primary
[pci@500/pci@1/pci@0/pci@5/network@0,1]
    maxvfs = 7
```

2. **确定必须保留的引导磁盘的设备路径。**
   请参见如何通过分配 PCIe 总线创建根域 [61] 中的步骤 2。

3. **确定块设备连接到的物理设备。**
   请参见如何通过分配 PCIe 总线创建根域 [61] 中的步骤 3。

4. **确定由系统使用的网络接口。**
   请参见如何通过分配 PCIe 总线创建根域 [61] 中的步骤 4。

5. **确定网络接口连接到的物理设备。**
   以下命令使用 `igb0` 网络接口：

```
primary# ls -l /dev/igb0
lrwxrwxrwx   1 root     root          46 Jul 30 17:29 /dev/igb0 ->
../devices/pci@500/pci@0/pci@8/network@0:igb0
```

在此示例中，primary 域使用的网络接口的物理设备已连接到 PCIe 端点设备 (pci@500/pci@0/pci@8)，该端点设备对应于步骤 1 中列出的 MB/NET0。因此，您不希望将此设备从 primary 域删除。由于所有其他 PCIe 设备都未被 primary 域使用，因此可以将它们安全地分配到其他域。

如果 primary 域使用的网络接口位于您要分配到其他域的总线上，则需要重新配置 primary 域以使用其他网络接口。

6. 删除可能会在 I/O 域中使用的 PCIe 端点设备。

   在此示例中，您可以删除 PCIE2、PCIE3、PCIE4 和 PCIE5 端点设备，因为它们没有被 primary 域使用。

   a. 删除 PCIe 端点设备。

注意 - 不要删除 primary 域所使用或所需要的设备。不要删除具有域使用的设备（如网络端口）的总线。

如果不小心删除了错误设备，请使用 ldm cancel-reconf primary 命令取消 primary 域上的延迟重新配置。

可以一次删除多个设备以避免多次重新引导。

```
primary# ldm start-reconf primary
primary# ldm set-io iov=on pci_1
All configuration changes for other domains are disabled until the primary
domain reboots, at which time the new configuration for the primary domain
will also take effect.
primary# ldm remove-io /SYS/MB/PCIE1 primary
------------------------------------------------------------------------------
Notice: The primary domain is in the process of a delayed reconfiguration.
Any changes made to the primary domain will only take effect after it reboots.
------------------------------------------------------------------------------
primary# ldm remove-io /SYS/MB/PCIE3 primary
------------------------------------------------------------------------------
Notice: The primary domain is in the process of a delayed reconfiguration.
Any changes made to the primary domain will only take effect after it reboots.
------------------------------------------------------------------------------
primary# ldm remove-io /SYS/MB/PCIE5 primary
------------------------------------------------------------------------------
Notice: The primary domain is in the process of a delayed reconfiguration.
Any changes made to the primary domain will only take effect after it reboots.
------------------------------------------------------------------------------
```

   b. 将新配置保存到服务处理器 (service processor, SP)。

   以下命令会将配置保存到名为 dio 的文件中：

```
primary# ldm add-config dio
```

c. 重新引导系统以反映 PCIe 端点设备的删除。

```
primary# shutdown -i6 -g0 -y
```

7. 登录 **primary** 域，检验是否不再将 PCIe 端点设备分配到域。

```
primary# ldm list-io
NAME                                    TYPE   BUS      DOMAIN   STATUS
----                                    ----   ---      ------   ------
niu_0                                   NIU    niu_0    primary
niu_1                                   NIU    niu_1    primary
pci_0                                   BUS    pci_0    primary
pci_1                                   BUS    pci_1    primary  IOV
/SYS/MB/PCIE0                           PCIE   pci_0    primary  OCC
/SYS/MB/PCIE2                           PCIE   pci_0    primary  OCC
/SYS/MB/PCIE4                           PCIE   pci_0    primary  OCC
/SYS/MB/PCIE6                           PCIE   pci_0    primary  EMP
/SYS/MB/PCIE8                           PCIE   pci_0    primary  EMP
/SYS/MB/SASHBA                          PCIE   pci_0    primary  OCC
/SYS/MB/NET0                            PCIE   pci_0    primary  OCC
/SYS/MB/PCIE1                           PCIE   pci_1             OCC
/SYS/MB/PCIE3                           PCIE   pci_1             OCC
/SYS/MB/PCIE5                           PCIE   pci_1             OCC
/SYS/MB/PCIE7                           PCIE   pci_1    primary  EMP
/SYS/MB/PCIE9                           PCIE   pci_1    primary  EMP
/SYS/MB/NET2                            PCIE   pci_1    primary  OCC
/SYS/MB/NET0/IOVNET.PF0                 PF     pci_0    primary
/SYS/MB/NET0/IOVNET.PF1                 PF     pci_0    primary
/SYS/MB/NET2/IOVNET.PF0                 PF     pci_1    primary
/SYS/MB/NET2/IOVNET.PF1                 PF     pci_1    primary
```

注 - 对于删除的 PCIe 端点设备，`ldm list-io -l` 输出可能会显示 `SUNW,assigned-device`。不可再从 `primary` 域获取实际信息，但是要将设备分配到其中的域包含此信息。

---

8. 将 PCIe 端点设备分配到域。

a. 将 **PCIE3** 设备添加到 **ldg1** 域。

```
primary# ldm add-io /SYS/MB/PCIE3 ldg1
```

b. 绑定并启动 **ldg1** 域。

```
primary# ldm bind ldg1
primary# ldm start ldg1
LDom ldg1 started
```

9. 登录 **ldg1** 域并检验设备是否可用。

验证网络设备是否可用，然后配置网络设备以便在域中使用。

- Oracle Solaris 10 OS：运行以下命令：

```
primary# dladm show-dev
nxge0           link: unknown   speed: 0     Mbps       duplex: unknown
nxge1           link: unknown   speed: 0     Mbps       duplex: unknown
nxge2           link: unknown   speed: 0     Mbps       duplex: unknown
nxge3           link: unknown   speed: 0     Mbps       duplex: unknown
```

- Oracle Solaris 11 OS：运行以下命令：

```
primary# dladm show-phys
LINK              MEDIA               STATE      SPEED  DUPLEX    DEVICE
net0              Ethernet            unknown    0      unknown   nxge0
net1              Ethernet            unknown    0      unknown   nxge1
net2              Ethernet            unknown    0      unknown   nxge2
net3              Ethernet            unknown    0      unknown   nxge3
```

♦ ♦ ♦ 第 8 章

使用 PCIe SR-IOV 虚拟功能创建 I/O 域

本章包括以下 PCIe SR-IOV 主题：

SR-IOV 概述

注 - 由于根域之间不能具有依赖关系，因此，拥有 PCIe 总线的根域不能将其 PCIe 端点设备或 SR-IOV 虚拟功能分配给另一个根域。不过，可以将 PCIe 总线中的 PCIe 端点设备或虚拟功能分配给拥有该总线的根域。

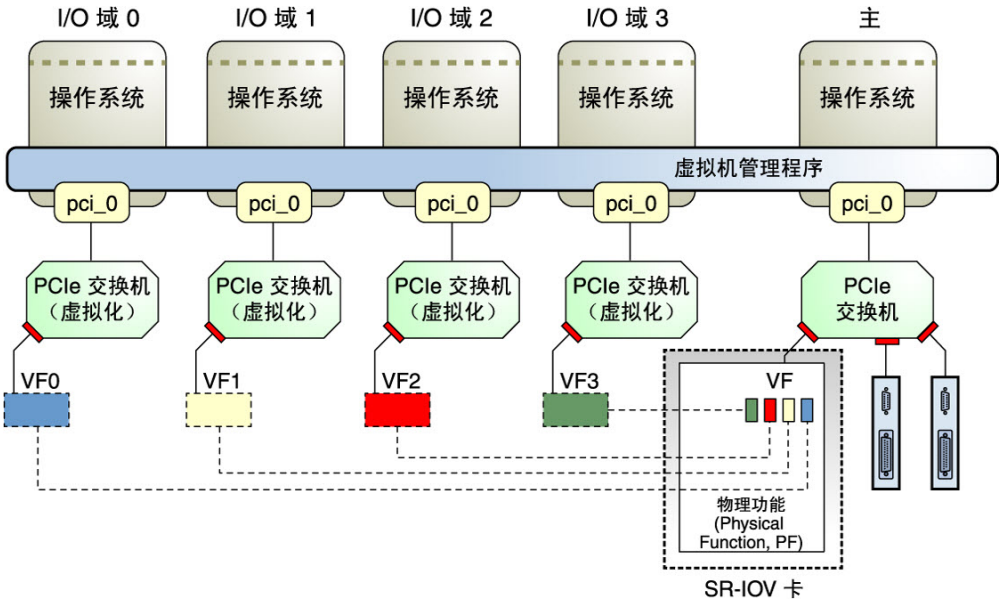
外设部件互连 Express (Peripheral Component Interconnect Express, PCIe) 单根 I/O 虚拟化 (single root I/O virtualization, SR-IOV) 的实现基于由 PCI-SIG 定义的 1.1 版标准。SR-IOV 标准允许在虚拟机之间高效共享 PCIe 设备，并在硬件中实现以获得能与本机性能媲美的 I/O 性能。SR-IOV 规范定义了新的标准，其中创建的新设备允许虚拟机直接连接到 I/O 设备。

单一 I/O 资源（称为物理功能）可以由多个虚拟机共享。共享的设备提供专用的资源，并且还使用共享的通用资源。这样，每个虚拟机都可访问唯一的资源。因此，启用了 SR-IOV 并且具有适当的硬件和 OS 支持的 PCIe 设备（例如以太网端口）可以显示为多个单独的物理设备，每个都具有其自己的 PCIe 配置空间。

有关 SR-IOV 的更多信息，请参见 PCI-SIG Web 站点 (http://www.pcisig.com/)。

下图显示了 I/O 域中虚拟功能和物理功能之间的关系。

图 8-1 在 I/O 域中使用虚拟功能和物理功能



SR-IOV 具有以下功能类型：

- **物理功能** – 支持 SR-IOV 规范定义的 SR-IOV 功能的 PCI 功能。物理功能包含 SR-IOV 功能结构并管理 SR-IOV 功能。物理功能是全面的 PCIe 功能，可以像其他任何 PCIe 设备一样发现、管理和处理。物理功能可用于配置和控制 PCIe 设备。
- **虚拟功能** – 与物理功能关联的 PCI 功能。虚拟功能是与物理功能和与该物理功能关联的虚拟功能共享一个或多个物理资源的轻量 PCIe 功能。与物理功能不同，虚拟功能仅可配置其自己的行为。

每个 SR-IOV 设备都可具有一个物理功能，并且每个物理功能都可最多具有 256 个与其关联的虚拟功能。此数字取决于特定的 SR-IOV 设备。虚拟功能由物理功能创建。

在物理功能中启用 SR-IOV 后，每个虚拟功能的 PCI 配置空间可以通过总线、设备和物理功能的功能编号访问。每个虚拟功能都具有一个 PCI 内存空间，用于映射其寄存器集。虚拟功能设备驱动程序对寄存器集进行操作以启用其功能，并且虚拟功能显示为实际 PCI 设备。创建后，您可以直接将虚拟功能分配给 I/O 域。此功能允许虚拟功能共享物理设备，以及在没有 CPU 和虚拟机管理程序软件开销的情况下执行 I/O。

在您的环境中使用 SR-IOV 功能可获得以下优势：

- **更高的性能和更短的延迟** – 从虚拟机环境直接访问硬件
- **降低的成本** – 节省资本支出和运营支出，这包括：
  - 节能
  - 减少了适配器数量
  - 简化了布线
  - 减少了交换机端口

Oracle VM Server for SPARC SR-IOV 的实现包括静态和动态配置方法。有关更多信息，请参见“静态 SR-IOV” [87]和“动态 SR-IOV” [88]。

通过 Oracle VM Server for SPARC SR-IOV 功能，可以执行以下操作：

- 在指定的物理功能上创建虚拟功能
- 在物理功能上销毁指定的虚拟功能
- 将虚拟功能分配到域
- 从域中删除虚拟功能

要在 SR-IOV 物理功能设备中创建和销毁虚拟功能，必须首先对该 PCIe 总线启用 I/O 虚拟化。可以使用 `ldm set-io` 或 `ldm add-io` 命令将 `iov` 属性设置为 `on`。还可以使用 `ldm add-domain` 或 `ldm set-domain` 命令将 `rc-add-policy` 属性设置为 `iov`。请参见 `ldm`(1M) 手册页。

注 - 在 Fujitsu M10 服务器上，默认情况下为 I/O 虚拟化启用 PCIe 总线。

向域分配 SR-IOV 虚拟功能会创建对提供 SR-IOV 物理功能服务的域的隐式依赖关系。可以使用 `ldm list-dependencies` 命令查看这些依赖关系或者查看依赖此 SR-IOV 物理功能的域。请参见“列出域 I/O 依赖关系” [320]。

SR-IOV 硬件和软件要求

SPARC T4、SPARC T5、SPARC M5 和 SPARC M6 平台支持动态和静态 PCIe SR-IOV 功能。仅当其他设备类型要求您使用静态方法时，Fujitsu M10 平台上才针对以太网设备支持动态功能。SPARC T3 平台仅支持静态 PCIe SR-IOV 功能。

注 - 将 InfiniBand SR-IOV 部署到 Oracle VM Server for SPARC 3.1 环境之前，请查看《Oracle VM Server for SPARC 3.2 发行说明》中的“InfiniBand SR-IOV 问题”中的信息。

- **硬件要求。**

  请参阅您平台的硬件文档来确认可以在您平台上使用的卡。有关受支持的 PCIe 卡的最新列表，请参见 `https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&doctype=REFERENCE&id=1325454.1`。

- 以太网 SR-IOV。要使用 SR-IOV 功能，可以使用板载 PCIe SR-IOV 设备以及 PCIe SR-IOV 插件卡。给定平台中的所有板载 SR-IOV 设备都受支持，除非平台文档中明确指出了例外情况。
- InfiniBand SR-IOV。SPARC T4、SPARC T5、SPARC M5 和 SPARC M6 平台以及 Fujitsu M10 平台支持 InfiniBand 设备。
- 光纤通道 SR-IOV。SPARC T4、SPARC T5、SPARC M5 和 SPARC M6 平台以及 Fujitsu M10 平台支持光纤通道设备。

  有关 Fujitsu M10 平台上支持的设备的最新列表，请参见 http://www.fujitsu.com/global/services/computing/server/sparc/downloads/manual/ 上适用于您的产品型号的产品说明中的《*Fujitsu M10/SPARC M10 Systems PCI Card Installation Guide*》

- 固件要求。
  - 以太网 SR-IOV。要使用动态 SR-IOV 功能，SPARC T4 系统必须至少运行 8.4.0.a 版的系统固件。SPARC T5、SPARC M5 和 SPARC M6 平台必须至少运行 9.1.0.a 版本的系统固件。Fujitsu M10 服务器必须至少运行 XCP2210 版本的系统固件。SPARC T3 平台仅支持静态 SR-IOV 功能。

    要使用 SR-IOV 功能，PCIe SR-IOV 设备必须至少运行设备固件版本 3.01。执行以下步骤来更新 Sun Dual 10-Gigabit Ethernet SFP+ PCIe 2.0 网络适配器的固件：

    1. 确定您是否需要升级设备上的 FCode 版本。

       从 `ok` 提示符执行以下命令：

       ```
       {0} ok cd path-to-device
       {0} ok .properties
       ```

       输出中的 `version` 值必须是以下项之一：

`LP`	Sun Dual 10GbE SFP+ PCIe 2.0 LP FCode 3.01 4/2/2012
`PEM`	Sun Dual 10GbE SFP+ PCIe 2.0 EM FCode 3.01 4/2/2012
`FEM`	Sun Dual 10GbE SFP+ PCIe 2.0 FEM FCode 3.01 4/2/2012

    2. 从 My Oracle Support (https://support.oracle.com/CSP/ui/flash.html#tab=PatchHomePage(page=PatchHomePage&id=h0wvdxy6())) 下载 ID 为 13932765 的修补程序。
    3. 安装该修补程序。

       该修补程序包内含一个介绍如何使用工具来执行升级的文档。
  - InfiniBand SR-IOV。要使用此功能，您的系统必须至少运行以下版本的系统固件：

- SPARC T4 – 8.4
- SPARC T5、SPARC M5 和 SPARC M6 – 9.1.0.*x*
- Fujitsu M10 服务器 – XCP2210

要支持将 Dual 40-Gigabit (4x) InfiniBand Host Channel Adapter M2 用作 InfiniBand SR-IOV 设备，卡或 express 模块必须至少运行 2.11.2010 版的固件。可以通过安装以下修补程序获得此版本的固件：

- 窄板型 (X4242A) – 修补程序 ID 16340059
- Express 模块 (X4243A) – 修补程序 ID 16340042

可使用 Oracle Solaris 11.1 `fwflash` 命令列出和更新 `primary` 域中的固件。要列出当前的固件版本，请使用 `fwflash -lc IB` 命令。要更新固件，请使用 `fwflash -f` *firmware-file* `-d` *device* 命令。请参见 `fwflash`(1M) 手册页。

要使用 InfiniBand SR-IOV，请确保 InfiniBand 交换机至少具有 2.1.2 版的固件。可以通过安装以下修补程序获得此版本的固件：

- Sun Datacenter InfiniBand Switch 36 (X2821A-Z) – 修补程序 ID 16221424
- Sun Network QDR InfiniBand GatewaySwitch (X2826A-Z) – 修补程序 ID 16221538

有关如何更新固件的信息，请参见您的 InfiniBand 交换机文档。

- 光纤通道 SR-IOV。要使用此功能，您的系统必须至少运行以下版本的系统固件：
  - SPARC T4 – 8.4.2.c
  - SPARC T5、SPARC M5 和 SPARC M6 – 9.1.2.d
  - Fujitsu M10 服务器 – XCP2210

  Sun Storage 16 Gb 光纤通道通用 HBA 上的固件 Emulex 必须至少为 1.1.60.1 修订版才能启用光纤通道 SR-IOV 功能。随固件提供了安装说明。

注意 - 只有计划使用光纤通道 SR-IOV 功能时才需要对光纤通道卡执行固件更新。

- 软件要求。不再支持在 SR-IOV 根域中使用 Oracle Solaris 10 OS。除非另有说明，否则 SR-IOV 根域必须至少运行 Oracle Solaris 11.1.10.5.0 OS。因此，请升级 SR-IOV 根域以便至少运行 Oracle Solaris 11.1.10.5.0 OS。另外，建议升级 I/O 域以便至少运行 Oracle Solaris 11.1.10.5.0 OS。
  - 以太网 SR-IOV。要使用 SR-IOV 功能，根域必须至少运行 Oracle Solaris 11.1.10.5.0 OS。I/O 域必须至少运行 Oracle Solaris 10 1/13 OS。
  - InfiniBand SR-IOV。以下域必须运行受支持的 Oracle Solaris OS：
    - `primary` 域或非 `primary` 根域必须至少运行 Oracle Solaris 11.1.10.6.0 OS。
    - I/O 域至少能运行 Oracle Solaris 11.1.10.6.0 OS 或 Oracle Solaris 10 1/13 OS。

- 在包含您计划从中配置虚拟功能的 InfiniBand SR-IOV 物理功能的任何根域上更新 `/etc/system` 文件。

  ```
  set ldc:ldc_maptable_entries = 0x20000
  ```

  有关正确创建或更新 `/etc/system` 属性值的信息，请参见“更新 /etc/system 文件中的属性值” [302]。

  在您要将虚拟功能添加到其中的 I/O 域上更新 `/etc/system` 文件。

  ```
  set rdsv3:rdsv3_fmr_pool_size = 16384
  ```

- **光纤通道 SR-IOV。**要使用 SR-IOV 功能，根域必须至少运行 Oracle Solaris 11.1.17.4.0 OS。I/O 域必须至少运行 Oracle Solaris 10 1/13 OS。

有关静态和动态 SR-IOV 软件要求的更多信息，请参见以下部分：

- “静态 SR-IOV 软件要求” [88]
- “动态 SR-IOV 软件要求” [88]

有关特定于类的 SR-IOV 硬件要求的更多信息，请参见以下部分：

- “以太网 SR-IOV 硬件要求” [91]
- “InfiniBand SR-IOV 硬件要求” [109]
- “光纤通道 SR-IOV 硬件要求” [122]

当前的 SR-IOV 功能限制

SR-IOV 功能具有以下限制：

- 如果任何关联的根域未在运行，则 I/O 域无法启动。
- 对分配了一个或多个虚拟功能的所有域禁用了迁移。
- 您只能销毁为物理功能创建的最后一个虚拟功能。所以，如果您创建了三个虚拟功能，那么可以销毁的首个虚拟功能必须是第三个。
- 如果通过直接 I/O (Direct I/O, DIO) 功能将 SR-IOV 卡分配到某个域，则不会为该卡启用 SR-IOV 功能。
- 最多可以从特定的 PCIe 总线将 PCIe 端点设备和 SR-IOV 虚拟功能分配给支持的 SPARC T 系列和 SPARC M 系列系统上的 15 个域。在 Fujitsu M10 服务器上，可以将 PCIe 端点设备和 SR-IOV 虚拟功能从特定的 PCIe 总线分配到最多 24 个域。系统在根域和 I/O 域之间分配 PCIe 资源，例如每个 PCIe 总线的中断向量。因此，您可以分配给特定 I/O 域的设备数量也受到了限制。确保您未将大量虚拟功能分配到同一个 I/O 域。有关与 SR-IOV 相关的问题的说明，请参见《Oracle VM Server for SPARC 3.2 发行说明》。
- PCIe 总线属于根域，该域负责初始化和管理该总线。根域必须处于活动状态，并在运行支持 SR-IOV 功能的 Oracle Solaris OS 版本。关闭、停止或重新引导根域将中

断对 PCIe 总线的访问。PCIe 总线不可用时，该总线上的 PCIe 设备会受到影响，可能会变为不可用。

当正在运行具有 PCIe SR-IOV 虚拟功能的 I/O 域时，如果重新引导根域，则这些 I/O 域将出现不可预测的行为。例如，具有 PCIe 端点设备的 I/O 域可能会在重新引导过程中或在重新引导后出现紧急情况。重新引导根域后，您可能需要手动停止并启动每个域。

如果 I/O 域具有弹性，即使作为 PCIe 总线所有者的根域变为不可用，该 I/O 域也可以继续操作。请参见“I/O 域弹性” [135]。

- SPARC 系统（最高包括 SPARC T5 和 SPARC M6 平台）提供中断数量有限，因此 Oracle Solaris 会限制每个设备可以使用的中断数量。默认限制应与典型系统配置的需求匹配，但可能需要为某些系统配置调整此值。有关详细信息，请参见“调整中断限制” [318]。

静态 SR-IOV

静态 SR-IOV 方法要求在执行 SR-IOV 操作时，根域处于延迟重新配置状态或 I/O 域已停止。在根域上完成配置步骤后，必须重新引导此根域。如果系统中未安装 Oracle VM Server for SPARC 3.1 固件，或者相应域中安装的 OS 版本不支持动态 SR-IOV，则必须使用此方法。

要创建或销毁某个 SR-IOV 虚拟功能，必须首先在根域上启动延迟重新配置。然后，可以运行一个或多个 `ldm create-vf` 和 `ldm destroy-vf` 命令来配置虚拟功能。最后，重新引导根域。以下命令显示如何在非 `primary` 根域上创建虚拟功能：

```
primary# ldm start-reconf root-domain-name
primary# ldm create-vf pf-name
primary# ldm stop-domain -r root-domain-name

primary# shutdown -i6 -g0 -y
```

要在来宾域中静态添加或删除虚拟功能，必须首先停止该来宾域。然后，执行 `ldm add-io` 和 `ldm remove-io` 命令来配置虚拟功能。完成更改后，启动该域。以下命令显示如何使用此方法来分配虚拟功能：

```
primary# ldm stop guest-domain
primary# ldm add-io vf-name guest-domain
primary# ldm start guest-domain
```

也可以在根域（而不是来宾域）中添加或删除虚拟功能。要在根域中添加或删除 SR-IOV 虚拟功能，请首先在该根域上启动延迟重新配置。然后，可以运行一个或多个 `ldm add-io` 和 `ldm remove-io` 命令。最后，重新引导根域。

要最大限度地缩短域的停机时间，请在配置虚拟功能之前提前进行规划。

注 - 只有静态 SR-IOV 才支持 InfiniBand SR-IOV 设备。

静态 SR-IOV 软件要求

Oracle VM Server for SPARC 3.0 软件和固件可支持静态 SR-IOV 功能。请参见《Oracle VM Server for SPARC 3.0 Release Notes》中的"PCIe SR-IOV Hardware and Software Requirements"。

可以使用 `ldm set-io` 或 `ldm add-io` 命令将 `iov` 属性设置为 `on`。还可以使用 `ldm add-domain` 或 `ldm set-domain` 命令将 `rc-add-policy` 属性设置为 `iov`。请参见 `ldm`(1M) 手册页。

重新引导根域会影响 SR-IOV 产生影响，因此，请仔细规划直接 I/O 配置更改，以最大限度地增加对根域的 SR-IOV 相关更改，最大限度地减少根域重新引导。

动态 SR-IOV

动态 SR-IOV 功能可消除以下静态 SR-IOV 要求：

- **根域**。在根域上启动延迟重新配置，创建或销毁虚拟功能，然后重新引导根域
- **I/O 域**。停止 I/O 域，添加或删除虚拟功能，然后启动 I/O 域

通过动态 SR-IOV，您可以动态创建或销毁虚拟功能，而无需在根域上启动延迟重新配置。也可以在 I/O 域中动态添加或删除虚拟功能，而无需停止该域。Logical Domains Manager 会与 Logical Domains 代理和 Oracle Solaris I/O 虚拟化框架进行通信，以动态方式使这些更改生效。

动态 SR-IOV 软件要求

有关所需的 PCIe SR-IOV 软件和固件版本的信息，请参见"SR-IOV 硬件和软件要求" [83]。

注 - 如果系统不满足动态 SR-IOV 软件和固件要求，则必须使用静态 SR-IOV 方法执行与 SR-IOV 相关的任务。请参见"静态 SR-IOV" [87]。

动态 SR-IOV 配置要求

要动态创建或销毁某个虚拟功能，请确保满足以下条件：

- 在开始配置虚拟功能之前，已对 PCIe 总线启用了 I/O 虚拟化。

- 在根域上和在 I/O 域上运行的 OS 至少是 Oracle Solaris 11.1.10.5.0 OS 或 Oracle Solaris 10 1/13 OS 以及《Oracle VM Server for SPARC 3.2 安装指南 》中的“全限定 Oracle Solaris OS 版本”中必需的修补程序。
- 物理功能设备未在 OS 中配置，也未采用多路径配置。例如，可以取消激活以太网 SR-IOV 设备或将其置于 IPMP 或聚合中，以便成功创建或销毁虚拟功能。

  创建或销毁虚拟功能的操作需要物理功能设备驱动程序在脱机和联机状态之间切换。多路径配置可以使设备驱动程序在这两个状态之间切换。
- 在从 I/O 域中删除某个虚拟功能之前，该虚拟功能未在使用或采用多路径配置。例如，可以取消激活以太网 SR-IOV 虚拟功能，或者使该虚拟功能不采用 IPMP 配置。

注 - 无法对以太网 SR-IOV 虚拟功能使用聚合，因为当前多路径实现不支持虚拟功能。

启用 I/O 虚拟化

必须先对 PCIe 总线启用 I/O 虚拟化，并且根域处于延迟重新配置状态，才能配置 SR-IOV 虚拟功能。重新引导域以使更改生效。

注 - 在 Fujitsu M10 服务器上，默认情况下为 I/O 虚拟化启用 PCIe 总线。

▼ 如何对 PCIe 总线启用 I/O 虚拟化

只需对每个根联合体执行此过程一次即可。根联合体必须正在相同 SP 配置中运行。

1. **在根域上启动延迟重新配置。**

   ```
   primary# ldm start-reconf root-domain-name
   ```

2. **对 PCIe 总线启用 I/O 虚拟化操作。**

   请只有在未对具有物理功能的总线启用 I/O 虚拟化时才执行此步骤。

   运行以下命令之一：

   - **如果指定的 PCIe 总线已分配到根域，请启用 I/O 虚拟化。**

     ```
     primary# ldm set-io iov=on bus
     ```

   - **在向根域添加 PCIe 总线时启用 I/O 虚拟化。**

     ```
     primary# ldm add-io iov=on bus
     ```

3. **重新引导根域。**
   运行以下命令之一：

   - **重新引导非 `primary` 根域。**

     ```
     primary# ldm stop-domain -r root-domain
     ```

   - **重新引导 `primary` 根域。**

     ```
     primary# shutdown -i6 -g0 -y
     ```

PCIe SR-IOV 虚拟功能的使用计划

提前进行规划，以确定希望如何在配置中使用虚拟功能。确定 SR-IOV 设备中的哪些虚拟功能可满足当前和未来的配置需求。

如果尚未启用 I/O 虚拟化（这需要使用静态方法），请将此步骤与创建虚拟功能的步骤合并。通过合并这些步骤，您只需重新引导根域一次即可。

即使可以使用动态 SR-IOV，建议做法仍为一次性创建所有虚拟功能，因为在将这些虚拟功能分配到 I/O 域之后，您可能无法动态创建它们。

如果使用静态 SR-IOV，提前规划有助于您避免多次重新引导根域，而每次重新引导都可能对 I/O 域产生负面影响。

有关 I/O 域的信息，请参见“创建 I/O 域的一般准则” [58]。

使用以下常规步骤计划和执行 SR-IOV 虚拟功能配置和分配：

1. 确定系统上可用的 PCIe SR-IOV 物理功能，以及哪些物理功能最适合您的需要。
   使用以下命令确定所需的信息：

`ldm list-io`	确定可用的 SR-IOV 物理功能设备。
`prtdiag -v`	确定哪些 PCIe SR-IOV 卡和板载设备可用。
`ldm list-io -l` *pf-name*	确定指定物理功能的其他信息，例如设备支持的虚拟功能最大数量。
`ldm list-io -d` *pf-name*	确定设备支持的特定于设备的属性。请参见“高级 SR-IOV 主题：以太网 SR-IOV” [102]。

2. 对 PCIe 总线启用 I/O 虚拟化操作。
   请参见如何对 PCIe 总线启用 I/O 虚拟化 [89]。

3. 在指定的 SR-IOV 物理功能上创建所需数量的虚拟功能。

   使用以下命令创建物理功能的虚拟功能：

   ```
   primary# ldm create-vf -n max pf-name
   ```

   有关更多信息，请参见如何创建以太网 SR-IOV 虚拟功能 [92]、如何创建 InfiniBand 虚拟功能 [109]和如何创建光纤通道 SR-IOV 虚拟功能 [125]。

4. 使用 `ldm add-config` 命令将配置保存到 SP。

   有关更多信息，请参见如何向 I/O 域添加以太网 SR-IOV 虚拟功能 [100]、如何将 InfiniBand 虚拟功能添加到 I/O 域 [113]和如何向 I/O 域添加光纤通道 SR-IOV 虚拟功能 [132]。

使用以太网 SR-IOV 虚拟功能

您可以同时使用静态和动态 SR-IOV 方法管理以太网 SR-IOV 设备。

以太网 SR-IOV 硬件要求

有关所需的 PCIe 以太网 SR-IOV 硬件的信息，请参见“SR-IOV 硬件和软件要求” [83]。

以太网 SR-IOV 限制

以太网 SR-IOV 功能在本发行版中具有以下限制：

- 您可以通过设置 `pvid` 或 `vid` 属性启用虚拟功能的 VLAN 配置。您不能同时设置这两个虚拟功能属性。
- 不能将 SR-IOV 虚拟功能用作虚拟交换机的后端设备。

规划以太网 SR-IOV 虚拟功能的使用

动态创建虚拟功能时，请确保物理功能使用多路径或未被激活。

如果无法使用多路径或必须激活物理功能，请使用静态方法创建虚拟功能。请参见“静态 SR-IOV” [87]。

特定于以太网设备和特定于网络的属性

使用 `ldm create-vf` 命令可设置特定于设备和特定于网络的虚拟功能属性。`unicast-slots` 属性是特定于设备的属性。`mac-addr`、`alt-mac-addrs`、`mtu`、`pvid` 和 `vid` 属性是特定于网络的属性。

请注意，只有在将虚拟功能分配到处于延迟重新配置状态的 `primary` 域时，才能更改特定于网络的属性 `mac-addr`、`alt-mac-addrs` 和 `mtu`。

如果虚拟功能按如下方式分配，则尝试更改这些属性将失败：

- 虚拟功能分配到活动的 I/O 域：属性更改请求将被拒绝，因为必须在所属域处于非活动或绑定状态时，才能进行更改。
- 虚拟功能分配到非 `primary` 域且延迟重新配置已生效：属性更改请求将失败，并显示错误消息。

`pvid` 和 `vid` 特定于网络的属性可以随意更改，没有任何限制。

创建以太网虚拟功能

本节介绍如何动态创建和销毁虚拟功能。如果无法使用动态方法来执行这些操作，请在创建或销毁虚拟功能之前在根域上启动延迟重新配置。

▼ 如何创建以太网 SR-IOV 虚拟功能

如果无法使用此动态方法，请改用静态方法。请参见“静态 SR-IOV” [87]。

1. **确定物理功能设备。**

   ```
   primary# ldm list-io
   ```

   请注意，物理功能的名称包括 PCIe SR-IOV 卡或板载设备的位置信息。

2. **如果具有物理功能的总线尚未启用 I/O 虚拟化，请启用它。**
   请只有在未对具有物理功能的总线启用 I/O 虚拟化时才执行此步骤。
   请参见如何对 PCIe 总线启用 I/O 虚拟化 [89]。

3. **基于以太网物理功能用动态或静态方法创建单个虚拟功能或多个虚拟功能。**
   在创建一个或多个虚拟功能后，您可以将其分配给来宾域。

   - **动态方法：**
     - **要基于一个物理功能同时创建多个虚拟功能，请使用以下命令：**

```
primary# ldm create-vf -n number | max pf-name
```

使用 `ldm create-vf -n max` 命令可一次创建该物理功能的所有虚拟功能。

注意 - 系统使用 Intel 10-G 以太网卡时，基于每个物理功能创建的虚拟功能请勿超过 31 个，以便最大程度地提高性能。

您可以使用路径名称或 pseudonym 名称指定虚拟功能。但是，建议做法是使用 pseudonym 名称。

- **要基于一个物理功能创建一个虚拟功能，请使用以下命令：**

```
ldm create-vf [mac-addr=num] [alt-mac-addrs=[auto|num1,[auto|num2,...]]]
  [pvid=pvid] [vid=vid1,vid2,...] [mtu=size] [name=value...] pf-name
```

注 - 如果没有为网络设备显式分配 MAC 地址，则会自动分配 MAC 地址。

使用此命令可为该物理功能创建一个虚拟功能。您还可以手动指定特定于以太网类的属性值。

注 - OS 探测 IOV 设备时，有时无法立即使用新创建的虚拟功能。使用 `ldm list-io` 命令确定父物理功能及其子虚拟功能在 "Status"（状态）列中是否具有 `INV` 值。如果它们具有此值，则等待直到 `ldm list-io` 输出不再在 "Status"（状态）列中显示 `INV` 值（大约 45 秒），然后再使用该物理功能或其任何子虚拟功能。如果此状态持续存在，则设备存在问题。

根域重新引导（包括 `primary` 的重新引导）后或者您使用 `ldm create-vf` 或 `ldm destroy-vf` 命令后，状态可能立即为 `INV`。

- **静态方法:**

  a. **启动延迟重新配置。**

  ```
  primary# ldm start-reconf root-domain-name
  ```

  b. **基于以太网物理功能创建单个虚拟功能或多个虚拟功能。**

  使用如前所示的相同命令动态创建虚拟功能。

  c. **重新引导根域。**

  - **要重新引导非 `primary` 根域，请使用以下命令：**

  ```
  primary# ldm stop-domain -r root-domain
  ```

- 要重新引导 **primary** 根域，请使用以下命令：

```
primary# shutdown -i6 -g0 -y
```

例 8-1 显示有关以太网物理功能的信息

此示例显示了有关 /SYS/MB/NET0/IOVNET.PF0 物理功能的信息：

- 此物理功能来自板载 NET0 网络设备。
- IOVNET 字符串表示物理功能是一个网络 SR-IOV 设备。

```
primary# ldm list-io
NAME                                    TYPE   BUS      DOMAIN   STATUS
----                                    ----   ---      ------   ------
niu_0                                   NIU    niu_0    primary
niu_1                                   NIU    niu_1    primary
pci_0                                   BUS    pci_0    primary
pci_1                                   BUS    pci_1    primary
/SYS/MB/PCIE0                           PCIE   pci_0    primary  OCC
/SYS/MB/PCIE2                           PCIE   pci_0    primary  OCC
/SYS/MB/PCIE4                           PCIE   pci_0    primary  OCC
/SYS/MB/PCIE6                           PCIE   pci_0    primary  EMP
/SYS/MB/PCIE8                           PCIE   pci_0    primary  EMP
/SYS/MB/SASHBA                          PCIE   pci_0    primary  OCC
/SYS/MB/NET0                            PCIE   pci_0    primary  OCC
/SYS/MB/PCIE1                           PCIE   pci_1    primary  OCC
/SYS/MB/PCIE3                           PCIE   pci_1    primary  OCC
/SYS/MB/PCIE5                           PCIE   pci_1    primary  OCC
/SYS/MB/PCIE7                           PCIE   pci_1    primary  EMP
/SYS/MB/PCIE9                           PCIE   pci_1    primary  EMP
/SYS/MB/NET2                            PCIE   pci_1    primary  OCC
/SYS/MB/NET0/IOVNET.PF0                 PF     pci_0    primary
/SYS/MB/NET0/IOVNET.PF1                 PF     pci_0    primary
/SYS/MB/PCIE5/IOVNET.PF0                PF     pci_1    primary
/SYS/MB/PCIE5/IOVNET.PF1                PF     pci_1    primary
/SYS/MB/NET2/IOVNET.PF0                 PF     pci_1    primary
/SYS/MB/NET2/IOVNET.PF1                 PF     pci_1    primary
```

以下命令显示有关指定物理功能的更多详细信息。maxvfs 值表示设备支持的虚拟功能最大数量。

```
primary# ldm list-io -l /SYS/MB/NET0/IOVNET.PF0
NAME                                    TYPE   BUS      DOMAIN   STATUS
----                                    ----   ---      ------   ------
/SYS/MB/NET0/IOVNET.PF0                 PF     pci_0    primary
[pci@400/pci@1/pci@0/pci@4/network@0]
    maxvfs = 7
```

例 8-2 动态创建以太网虚拟功能而不设置可选属性

此示例会动态创建一个虚拟功能，而不设置任何可选属性。在这种情况下，自动为网络级虚拟功能分配 MAC 地址。

确保已对 pci_0 PCIe 总线启用 I/O 虚拟化。请参见如何对 PCIe 总线启用 I/O 虚拟化 [89]。

现在，您可以使用 ldm create-vf 命令从 /SYS/MB/NET0/IOVNET.PF0 物理功能创建虚拟功能。

```
primary# ldm create-vf /SYS/MB/NET0/IOVNET.PF0
Created new vf: /SYS/MB/NET0/IOVNET.PF0.VF0
```

例 8-3 动态创建以太网虚拟功能并设置属性

此示例会动态创建虚拟功能，并将 mac-addr 属性设置为 00:14:2f:f9:14:c0，以及将 vid 属性设置为 VLAN ID 2 和 3。

```
primary# ldm create-vf mac-addr=00:14:2f:f9:14:c0 vid=2,3 /SYS/MB/NET0/IOVNET.PF0
```

例 8-4 动态创建具有两个备用 MAC 地址的以太网虚拟功能

此示例会动态创建一个具有两个备用 MAC 地址的虚拟功能。一个 MAC 地址是自动分配的，另一个则显式指定为 00:14:2f:f9:14:c2。

```
primary# ldm create-vf alt-mac-addrs=auto,00:14:2f:f9:14:c2 /SYS/MB/NET0/IOVNET.PF0
```

例 8-5 静态创建虚拟功能而不设置可选属性

此示例会静态创建一个虚拟功能，而不设置任何可选属性。在这种情况下，自动为网络级虚拟功能分配 MAC 地址。

首先，在 primary 域上启动延迟重新配置，然后，在 pci_0 PCIe 总线上启用 I/O 虚拟化。由于已将 pci_0 总线分配到 primary 根域，因此，请使用 ldm set-io 命令启用 I/O 虚拟化。

```
primary# ldm start-reconf primary
Initiating a delayed reconfiguration operation on the primary domain.
All configuration changes for other domains are disabled until the primary
domain reboots, at which time the new configuration for the primary domain
will also take effect.

primary# ldm set-io iov=on pci_0
```

现在，您可以使用 ldm create-vf 命令从 /SYS/MB/NET0/IOVNET.PF0 物理功能创建虚拟功能。

```
primary# ldm create-vf /SYS/MB/NET0/IOVNET.PF0

------------------------------------------------------------------------------
```

```
Notice: The primary domain is in the process of a delayed reconfiguration.
Any changes made to the primary domain will only take effect after it reboots.
--------------------------------------------------------------------------------

Created new vf: /SYS/MB/NET0/IOVNET.PF0.VF0
```

最后，重新引导 `primary` 根域以使更改生效。

```
primary# shutdown -i6 -g0 -y
```

例 8-6 创建多个 SR-IOV 以太网虚拟功能

以下命令展示了如何基于 `/SYS/MB/NET2/IOVNET.PF1` 物理功能创建四个虚拟功能：

```
primary# ldm create-vf -n 31 /SYS/MB/NET2/IOVNET.PF1
Created new vf: /SYS/MB/NET2/IOVNET.PF1.VF0
Created new vf: /SYS/MB/NET2/IOVNET.PF1.VF1
Created new vf: /SYS/MB/NET2/IOVNET.PF1.VF2
...
Created new vf: /SYS/MB/NET2/IOVNET.PF1.VF30
```

请注意，`ldm create-vf -n` 命令创建了多个设置有默认属性值（如果适用）的虚拟功能。之后，您可以使用 `ldm set-io` 命令指定非默认属性值。

销毁以太网虚拟功能

如果当前未将虚拟功能分配给域，则可以销毁该虚拟功能。虚拟功能只能按照与创建时相反的顺序进行销毁，因此，只能销毁已创建的最后一个虚拟功能。生成的配置由物理功能驱动程序验证。

▼ 如何销毁以太网 SR-IOV 虚拟功能

如果无法使用此动态方法，请改用静态方法。请参见“静态 SR-IOV” [87]。

1. **确定物理功能设备。**

   ```
   primary# ldm list-io
   ```

2. **以动态或静态方法销毁单个虚拟功能或多个虚拟功能。**

   - **动态方法：**

     - **要同时销毁基于一个物理功能的一些或所有虚拟功能，请使用以下命令：**

       ```
       primary# ldm destroy-vf -n number | max pf-name
       ```

       使用 `ldm destroy-vf -n max` 命令可一次销毁该物理功能的所有虚拟功能。

如果您指定 *number* 作为 -n 选项的参数，则会销毁最后 *number* 个虚拟功能。请使用此方法，因为执行此操作时只进行一次物理功能设备驱动程序状态转换。

您可以使用路径名称或 pseudonym 名称指定虚拟功能。但是，建议做法是使用 pseudonym 名称。

- 要销毁指定的虚拟功能，请使用以下命令：

```
primary# ldm destroy-vf vf-name
```

由于受影响的设备中以及 OS 中的延迟，受影响的物理功能以及任何其余子虚拟功能可能无法立即使用。使用 `ldm list-io` 命令确定父物理功能及其子虚拟功能在 "Status"（状态）列中是否具有 `INV` 值。如果它们具有此值，则等待直到 `ldm list-io` 输出不再在 "Status"（状态）列中显示 `INV` 值（大约 45 秒）。此时，您可以安全使用该物理功能或其任何子虚拟功能。如果此状态持续存在，则设备存在问题。

根域重新引导（包括 `primary` 的重新引导）后或者您使用 `ldm create-vf` 或 `ldm destroy-vf` 命令后，状态可能立即为 `INV`。

- 静态方法:

a. 启动延迟重新配置。

```
primary# ldm start-reconf root-domain-name
```

b. 销毁单个虚拟功能或多个虚拟功能。

- 要同时销毁基于指定物理功能的所有虚拟功能，请使用以下命令：

```
primary# ldm destroy-vf -n number | max pf-name
```

您可以使用路径名称或 pseudonym 名称指定虚拟功能。但是，建议做法是使用 pseudonym 名称。

- 要销毁指定的虚拟功能，请使用以下命令：

```
primary# ldm destroy-vf vf-name
```

c. 重新引导根域。

- 要重新引导非 **primary** 根域，请使用以下命令：

```
primary# ldm stop-domain -r root-domain
```

- 要重新引导 **primary** 根域，请使用以下命令：

```
primary# shutdown -i6 -g0 -y
```

例 8-7 销毁以太网虚拟功能

此示例展示了如何动态销毁 /SYS/MB/NET0/IOVNET.PF0.VF0 虚拟功能。

```
primary# ldm destroy-vf /SYS/MB/NET0/IOVNET.PF0.VF0
```

以下示例说明如何静态销毁 /SYS/MB/NET0/IOVNET.PF0.VF0 虚拟功能。

```
primary# ldm start-reconf primary
Initiating a delayed reconfiguration operation on the primary domain.
All configuration changes for other domains are disabled until the primary
domain reboots, at which time the new configuration for the primary domain
will also take effect.

primary# ldm destroy-vf /SYS/MB/NET0/IOVNET.PF0.VF0
primary# shutdown -i6 -g0 -y
```

例 8-8 销毁多个以太网 SR-IOV 虚拟功能

此示例展示了销毁基于 /SYS/MB/NET2/IOVNET.PF1 物理功能的所有虚拟功能的结果。ldm list-io 输出显示物理功能具有七个虚拟功能。ldm destroy-vf 命令销毁所有虚拟功能，并且 ldm list-io 最终输出显示未保留任何虚拟功能。

```
primary# ldm list-io
...
/SYS/MB/NET2/IOVNET.PF1                   PF     pci_1
/SYS/MB/NET2/IOVNET.PF1.VF0               VF     pci_1
/SYS/MB/NET2/IOVNET.PF1.VF1               VF     pci_1
/SYS/MB/NET2/IOVNET.PF1.VF2               VF     pci_1
/SYS/MB/NET2/IOVNET.PF1.VF3               VF     pci_1
/SYS/MB/NET2/IOVNET.PF1.VF4               VF     pci_1
/SYS/MB/NET2/IOVNET.PF1.VF5               VF     pci_1
/SYS/MB/NET2/IOVNET.PF1.VF6               VF     pci_1
primary# ldm destroy-vf -n max /SYS/MB/NET2/IOVNET.PF1
primary# ldm list-io
...
/SYS/MB/NET2/IOVNET.PF1                   PF     pci_1    ldg1
```

修改以太网 SR-IOV 虚拟功能

ldm set-io *vf-name* 命令通过更改属性值或设置新的属性来修改虚拟功能的当前配置。此命令既可以修改特定于网络的属性，也可以修改特定于设备的属性。有关特定于设备的属性的信息，请参见“高级 SR-IOV 主题：以太网 SR-IOV” [102]。

如果无法使用此动态方法，请改用静态方法。请参见“静态 SR-IOV” [87]。

您可以使用 ldm set-io 命令修改以下属性：

- `mac-addr`、`alt-mac-addrs` 和 `mtu`

  要更改这些虚拟功能属性，请停止虚拟功能所属的域，并使用 `ldm set-io` 命令更改属性值，然后启动该域。

- `pvid` 和 `vid`

  您可以在将虚拟功能分配给域的同时，动态更改这些属性。请注意，执行此操作可能会使活动虚拟功能的网络通信发生更改；设置 `pvid` 属性可启用透明 VLAN。设置 `vid` 属性以指定 VLAN ID 将允许与那些指定 VLAN 的 VLAN 通信流量。

- **特定于设备的属性**

  使用 `ldm list-io -d` *pf-name* 命令可查看特定于设备的有效属性列表。您可以为物理功能和虚拟功能修改这些属性。必须使用静态方法修改特定于设备的属性。请参见“静态 SR-IOV” [87]。有关特定于设备的属性的更多信息，请参见“高级 SR-IOV 主题：以太网 SR-IOV” [102]。

▼ 如何修改以太网 SR-IOV 虚拟功能属性

1. **确定物理功能设备。**

   ```
   primary# ldm list-io
   ```

   请注意，物理功能的名称包括 PCIe SR-IOV 卡或板载设备的位置信息。

2. **修改虚拟功能属性。**

   ```
   ldm set-io name=value [name=value...] vf-name
   ```

例 8-9 修改以太网虚拟功能属性

这些示例说明了如何使用 `ldm set-io` 命令设置以太网虚拟功能的属性。

- 以下示例会修改指定的虚拟功能 `/SYS/MB/NET0/IOVNET.PF0.VF0` 的属性，使其成为 VLAN ID `2`、`3` 和 `4` 的一部分。

  ```
  primary# ldm set-io vid=2,3,4 /SYS/MB/NET0/IOVNET.PF0.VF0
  ```

  请注意，此命令动态更改虚拟功能的 VLAN 关联。要使用这些 VLAN，I/O 域中的 VLAN 接口必须使用相应的 Oracle Solaris OS 网络命令进行配置。

- 以下示例为 `/SYS/MB/NET0/IOVNET.PF0.VF0` 虚拟功能将 `pvid` 属性设置为 `2`，这使得该虚拟功能透明地成为 VLAN `2` 的一部分。换句话说，该虚拟功能不会查看任何标记的 VLAN 通信流量。

  ```
  primary# ldm set-io pvid=2 /SYS/MB/NET0/IOVNET.PF0.VF0
  ```

- 以下示例将自动分配的三个备用 MAC 地址分配给虚拟功能。备用地址允许在虚拟功能之上创建 Oracle Solaris 11 虚拟网络接口卡 (virtual network interface card, VNIC)。请注意，要使用 VNIC，必须在域中运行 Oracle Solaris 11 OS。

注 - 在运行此命令之前，请停止拥有该虚拟功能的域。

```
primary# ldm set-io alt-mac-addrs=auto,auto,auto /SYS/MB/NET0/IOVNET.PF0.VF0
```

- 以下示例为指定的虚拟功能将特定于设备的 `unicast-slots` 属性设置为 `12`。要查找对于物理功能有效的特定于设备的属性，请使用 `ldm list-io -d` *pf-name* 命令。

```
primary# ldm set-io unicast-slots=12 /SYS/MB/NET0/IOVNET.PF0.VF0

All configuration changes for other domains are disabled until the primary
domain reboots, at which time the new configuration for the primary domain
will also take effect.
```

在 I/O 域中添加和删除以太网 SR-IOV 虚拟功能

▼ 如何向 I/O 域添加以太网 SR-IOV 虚拟功能

如果无法以动态方法删除虚拟功能，请使用静态方法。请参见“静态 SR-IOV” [87]。

1. **确定要添加到 I/O 域的虚拟功能。**

```
primary# ldm list-io
```

2. **以动态或静态方法添加虚拟功能。**

   - **要以动态方法添加虚拟功能，请使用以下命令：**

     ```
     primary# ldm add-io vf-name domain-name
     ```

     vf-name 是虚拟功能的 pseudonym 名称或路径名称。建议做法是使用 pseudonym 名称。*domain-name* 用于指定要将虚拟功能添加到的域的名称。

     域中虚拟功能的设备路径名称是 `list-io -l` 输出中显示的路径。

   - **要以静态方法添加虚拟功能，请使用以下命令：**

     a. **启动延迟重新配置，然后添加虚拟功能。**

     ```
     primary# ldm start-reconf root-domain-name
     primary# ldm add-io vf-name domain-name
     ```

     vf-name 是虚拟功能的 pseudonym 名称或路径名称。建议做法是使用 pseudonym 名称。*domain-name* 用于指定要将虚拟功能添加到的域的名称。指定的来宾必须处于非活动状态或绑定状态。

域中虚拟功能的设备路径名称是 `list-io -l` 输出中显示的路径。

b. 重新引导根域。

- 要重新引导非 **primary** 根域，请使用以下命令：

```
primary# ldm stop-domain -r root-domain
```

- 要重新引导 **primary** 根域，请使用以下命令：

```
primary# shutdown -i6 -g0 -y
```

例 8-10 添加以太网虚拟功能

此示例展示了如何以动态方法将 `/SYS/MB/NET0/IOVNET.PF0.VF0` 虚拟功能添加到 `ldg1` 域。

```
primary# ldm add-io /SYS/MB/NET0/IOVNET.PF0.VF0 ldg1
```

如果无法动态添加虚拟功能，请使用静态方法：

```
primary# ldm stop-domain ldg1
primary# ldm add-io /SYS/MB/NET0/IOVNET.PF0.VF0 ldg1
primary# ldm start-domain ldg1
```

▼ 如何从 I/O 域删除以太网 SR-IOV 虚拟功能

如果无法以动态方法删除虚拟功能，请使用静态方法。请参见“静态 SR-IOV” [87]。

注意 - 从域中删除虚拟功能之前，请确保它对于引导该域不至关重要。

1. 确定要从 I/O 域删除的虚拟功能。

```
primary# ldm list-io
```

2. 以动态或静态方法删除虚拟功能。

- 要以动态方法删除虚拟功能，请使用以下命令：

```
primary# ldm remove-io vf-name domain-name
```

vf-name 是虚拟功能的 pseudonym 名称或路径名称。建议做法是使用设备 pseudonym。*domain-name* 用于指定要从中删除虚拟功能的域的名称。

- 要以静态方法删除虚拟功能，请使用以下命令：

a. 停止 I/O 域。

```
primary# ldm stop-domain domain-name
```

b. 删除虚拟功能。

```
primary# ldm remove-io vf-name domain-name
```

vf-name 是虚拟功能的 pseudonym 名称或路径名称。建议做法是使用设备 pseudonym。*domain-name* 用于指定要从中删除虚拟功能的域的名称。指定的来宾必须处于非活动状态或绑定状态。

c. 启动 I/O 域。

```
primary# ldm start-domain domain-name
```

例 8-11 动态删除以太网虚拟功能

此示例展示了如何以动态方法从 `ldg1` 域中删除 `/SYS/MB/NET0/IOVNET.PF0.VF0` 虚拟功能。

```
primary# ldm remove-io /SYS/MB/NET0/IOVNET.PF0.VF0 ldg1
```

如果命令成功，则从 `ldg1` 域删除该虚拟功能。重新启动 `ldg1` 后，指定的虚拟功能不再显示在该域中。

如果无法动态删除虚拟功能，请使用静态方法：

```
primary# ldm stop-domain ldg1
primary# ldm remove-io /SYS/MB/NET0/IOVNET.PF0.VF0 ldg1
primary# ldm start-domain ldg1
```

高级 SR-IOV 主题：以太网 SR-IOV

本节介绍与使用 SR-IOV 虚拟功能相关的一些高级主题。

虚拟功能的高级网络配置

使用 SR-IOV 虚拟功能时，请注意以下问题：

- SR-IOV 虚拟功能只能使用 Logical Domains Manager 分配的 MAC 地址。如果您使用其他 Oracle Solaris OS 网络命令更改 I/O 域上的 MAC 地址，则命令可能会失败或无法正常运行。
- 此时，不支持 I/O 域中的 SR-IOV 网络虚拟功能的链路聚合。如果您尝试创建链路聚合，它可能无法按预期方式发挥作用。

- 您可以创建虚拟 I/O 服务并将它们分配给 I/O 域。可以在从中创建虚拟功能的同一物理功能上创建这些虚拟 I/O 服务。例如，您可以使用板载的 1-Gbps 网络设备（`net0` 或 `igb0`）作为虚拟交换机的网络后端设备，并从同一物理功能设备创建虚拟功能。

使用 SR-IOV 虚拟功能引导 I/O 域

SR-IOV 虚拟功能可提供与任何其他类型的 PCIe 设备类似的功能，例如，可将虚拟功能用作逻辑域引导设备。例如，网络虚拟功能可以用于通过网络引导以在 I/O 域中安装 Oracle Solaris OS。

注 - 在从虚拟功能设备引导 Oracle Solaris OS 时，请验证要装入的 Oracle Solaris OS 是否支持虚拟功能设备。如果支持，则可以按计划继续进行其余安装。

SR-IOV 特定于设备的属性

SR-IOV 物理功能设备驱动程序可以导出特定于设备的属性。这些属性可用于调节物理功能及其虚拟功能的资源分配。有关这些属性的信息，请参见物理功能驱动程序的手册页，例如 `igb`(7D) 和 `ixgbe`(7D) 手册页。

`ldm list-io -d` 命令显示指定物理功能设备驱动程序导出的特定于设备的属性。每个属性的信息包括其名称、简要说明、默认值、最大值和一个或多个以下标志：

`P`	适用于物理功能
`V`	适用于虚拟功能
`R`	只读或只用于提供信息的参数

```
primary# ldm list-io -d pf-name
```

使用 `ldm create-vf` 或 `ldm set-io` 命令可为物理功能或虚拟功能设置读写属性。请注意，要设置特定于设备的属性，必须使用静态方法。请参见“静态 SR-IOV” [87]。

以下示例显示板载 Intel 1-Gbps SR-IOV 设备导出的特定于设备的属性：

```
primary# ldm list-io -d /SYS/MB/NET0/IOVNET.PF0
Device-specific Parameters
--------------------------
max-config-vfs
    Flags = PR
    Default = 7
    Descr = Max number of configurable VFs
max-vf-mtu
```

```
    Flags = VR
    Default = 9216
    Descr = Max MTU supported for a VF
max-vlans
    Flags = VR
    Default = 32
    Descr = Max number of VLAN filters supported
pvid-exclusive
    Flags = VR
    Default = 1
    Descr = Exclusive configuration of pvid required
unicast-slots
    Flags = PV
    Default = 0 Min = 0 Max = 24
    Descr = Number of unicast mac-address slots
```

以下示例将 `unicast-slots` 属性设置为 8：

```
primary# ldm create-vf unicast-slots=8 /SYS/MB/NET0/IOVNET.PF0
```

在 SR-IOV 虚拟功能上创建 VNIC

支持在 SR-IOV 虚拟功能上创建 Oracle Solaris 11 VNIC。但是，支持的 VNIC 数量限制为分配给虚拟功能的备用 MAC 地址（`alt-mac-addrs` 属性）的数量。确保对虚拟功能使用 VNIC 时分配足够数量的备用 MAC 地址。可使用 `ldm create-vf` 或 `ldm set-io` 命令通过备用 MAC 地址设置 `alt-mac-addrs` 属性。

以下示例说明如何在 SR-IOV 虚拟功能上创建四个 VNIC。第一个命令将备用 MAC 地址分配给虚拟功能设备。此命令使用自动分配方法将四个备用 MAC 地址分配给 `/SYS/MB/NET0/IOVNET.PF0.VF0` 虚拟功能设备：

```
primary# ldm set-io alt-mac-addrs=auto,auto,auto,auto /SYS/MB/NET0/IOVNET.PF0.VF0
```

下一个命令将启动 `ldg1` I/O 域。在此示例中，由于 `auto-boot?` 属性设置为 `true`，因此，Oracle Solaris 11 OS 也会在该 I/O 域中引导。

```
primary# ldm start ldg1
```

以下命令可在来宾域中使用 Oracle Solaris 11 `dladm` 命令显示具有备用 MAC 地址的虚拟功能。此输出显示 `net30` 虚拟功能具有四个备用 MAC 地址。

```
guest# dladm show-phys -m
LINK              SLOT     ADDRESS            INUSE CLIENT
net0              primary  0:14:4f:fa:b4:d1   yes   net0
net25             primary  0:14:4f:fa:c9:eb   no    --
net30             primary  0:14:4f:fb:de:4c   no    --
                  1        0:14:4f:f9:e8:73   no    --
                  2        0:14:4f:f8:21:58   no    --
                  3        0:14:4f:fa:9d:92   no    --
```

```
                4       0:14:4f:f9:8f:1d   no     --
```

以下命令会创建四个 VNIC。请注意，尝试创建多于通过备用 MAC 地址所指定的 VNIC 将会失败。

```
guest# dladm create-vnic -l net30 vnic0
guest# dladm create-vnic -l net30 vnic1
guest# dladm create-vnic -l net30 vnic2
guest# dladm create-vnic -l net30 vnic3
guest# dladm show-link
LINK                CLASS     MTU    STATE    OVER
net0                phys      1500   up       --
net25               phys      1500   up       --
net30               phys      1500   up       --
vnic0               vnic      1500   up       net30
vnic1               vnic      1500   up       net30
vnic2               vnic      1500   up       net30
vnic3               vnic      1500   up       net30
```

使用 SR-IOV 虚拟功能创建 I/O 域

以下过程介绍了如何创建包括 PCIe SR-IOV 虚拟功能的 I/O 域。

▼ 如何通过向其分配 SR-IOV 虚拟功能来创建 I/O 域

提前进行计划，以便最大程度地减少根域的重新引导次数，从而最大限度地缩短停机时间。

开始之前

开始之前，请确保已为用于创建虚拟功能的物理功能的父项 PCIe 总线启用了 I/O 虚拟化。请参见如何对 PCIe 总线启用 I/O 虚拟化 [89]。

1. **确定要与使用 SR-IOV 功能的 I/O 域共享的 SR-IOV 物理功能。**

   ```
   primary# ldm list-io
   ```

2. **为物理功能创建一个或多个虚拟功能。**

   ```
   primary# ldm create-vf pf-name
   ```

   您可以为要创建的每个虚拟功能运行此命令。您还可以使用 -n 选项通过单个命令创建基于同一物理功能的多个虚拟功能。请参见例 8-6 “创建多个 SR-IOV 以太网虚拟功能” 和 ldm(1M) 手册页。

注 - 如果已从关联的物理功能创建了其他虚拟功能，并且其中任一虚拟功能已绑定到其他域，此命令将失败。

---

3. 查看根域上可用虚拟功能的列表。

```
primary# ldm list-io
```

4. 将步骤 2 中创建的虚拟功能分配到其目标 I/O 域。

```
primary# ldm add-io vf-name domain-name
```

注 - 如果目标 I/O 域中的 OS 不支持动态 SR-IOV，则必须使用静态方法。请参见“静态 SR-IOV” [87]。

---

5. 验证虚拟功能在 I/O 域上是否可用。

以下 Oracle Solaris 11 命令显示虚拟功能的可用性：

```
guest# dladm show-phys
```

例 8-12 通过分配 SR-IOV 虚拟功能来动态创建 I/O 域

以下动态示例说明如何为物理功能 /SYS/MB/NET0/IOVNET.PF0 创建虚拟功能 /SYS/MB/NET0/IOVNET.PF0.VF0，并将该虚拟功能分配给 ldg1 I/O 域。

本示例假定满足以下情况：

- primary 域上的 OS 支持动态 SR-IOV 操作
- pci_0 总线已分配到 primary 域，并且已针对 I/O 虚拟化操作进行了初始化
- /SYS/MB/NET0/IOVNET.PF0 物理功能属于 pci_0 总线
- /SYS/MB/NET0/IOVNET.PF0 物理功能未将任何现有虚拟功能分配到域
- ldg1 域处于活动状态并已引导，并且其 OS 支持动态 SR-IOV 操作

从 /SYS/MB/NET0/IOVNET.PF0 物理功能创建虚拟功能。

```
primary# ldm create-vf /SYS/MB/NET0/IOVNET.PF0
Created new vf: /SYS/MB/NET0/IOVNET.PF0.VF0
```

将 /SYS/MB/NET0/IOVNET.PF0.VF0 虚拟功能添加到 ldg1 域。

```
primary# ldm add-io /SYS/MB/NET0/IOVNET.PF0.VF0 ldg1
```

以下命令显示已将虚拟功能添加到 ldg1 域。

```
primary# ldm list-io
NAME                          TYPE  BUS      DOMAIN    STATUS
----                          ----  ---      ------    ------
niu_0                         NIU   niu_0    primary
niu_1                         NIU   niu_1    primary
pci_0                         BUS   pci_0    primary   IOV
pci_1                         BUS   pci_1    primary
/SYS/MB/PCIE0                 PCIE  pci_0    primary   OCC
```

```
/SYS/MB/PCIE2                 PCIE  pci_0    primary   OCC
/SYS/MB/PCIE4                 PCIE  pci_0    primary   OCC
/SYS/MB/PCIE6                 PCIE  pci_0    primary   EMP
/SYS/MB/PCIE8                 PCIE  pci_0    primary   EMP
/SYS/MB/SASHBA                PCIE  pci_0    primary   OCC
/SYS/MB/NET0                  PCIE  pci_0    primary   OCC
/SYS/MB/PCIE1                 PCIE  pci_1    primary   OCC
/SYS/MB/PCIE3                 PCIE  pci_1    primary   OCC
/SYS/MB/PCIE5                 PCIE  pci_1    primary   OCC
/SYS/MB/PCIE7                 PCIE  pci_1    primary   EMP
/SYS/MB/PCIE9                 PCIE  pci_1    primary   EMP
/SYS/MB/NET2                  PCIE  pci_1    primary   OCC
/SYS/MB/NET0/IOVNET.PF0       PF    pci_0    primary
/SYS/MB/NET0/IOVNET.PF1       PF    pci_0    primary
/SYS/MB/PCIE5/IOVNET.PF0      PF    pci_1    primary
/SYS/MB/PCIE5/IOVNET.PF1      PF    pci_1    primary
/SYS/MB/NET2/IOVNET.PF0       PF    pci_1    primary
/SYS/MB/NET2/IOVNET.PF1       PF    pci_1    primary
/SYS/MB/NET0/IOVNET.PF0.VF0   VF    pci_0    ldg1
```

例 8-13 通过分配 SR-IOV 虚拟功能来静态创建 I/O 域

以下静态示例说明如何为物理功能 `/SYS/MB/NET0/IOVNET.PF0` 创建虚拟功能 `/SYS/MB/NET0/IOVNET.PF0.VF0`，并将该虚拟功能分配给 `ldg1` I/O 域。

本示例假定满足以下情况：

- `primary` 域上的 OS 不支持动态 SR-IOV 操作
- `pci_0` 总线已分配到 `primary` 域，但尚未针对 I/O 虚拟化操作进行初始化
- `/SYS/MB/NET0/IOVNET.PF0` 物理功能属于 `pci_0` 总线
- `/SYS/MB/NET0/IOVNET.PF0` 物理功能未将任何现有虚拟功能分配到域
- `ldg1` 域处于活动状态并已引导，但其 OS 不支持动态 SR-IOV 操作
- `ldg1` 域已将 `auto-boot?` 属性设置为 `true`，因此，该域可以在启动后自动引导

首先，在 `primary` 域上启动延迟重新配置，启用 I/O 虚拟化，然后从 `/SYS/MB/NET0/IOVNET.PF0` 物理功能创建虚拟功能。

```
primary# ldm start-reconf primary
Initiating a delayed reconfiguration operation on the primary domain.
All configuration changes for other domains are disabled until the primary
domain reboots, at which time the new configuration for the primary domain
will also take effect.

primary# ldm set-io iov=on pci_0
primary# ldm create-vf /SYS/MB/NET0/IOVNET.PF0

------------------------------------------------------------------------------
Notice: The primary domain is in the process of a delayed reconfiguration.
Any changes made to the primary domain will only take effect after it reboots.
------------------------------------------------------------------------------
```

```
Created new vf: /SYS/MB/NET0/IOVNET.PF0.VF0
```

然后，关闭 primary 域。

```
primary# shutdown -i6 -g0 -y
```

停止 ldg1 域，添加虚拟功能，然后启动该域。

```
primary# ldm stop ldg1
primary# ldm add-io /SYS/MB/NET0/IOVNET.PF0.VF0 ldg1
primary# ldm start ldg1
```

以下命令显示已将虚拟功能添加到 ldg1 域。

```
primary# ldm list-io
NAME                           TYPE  BUS      DOMAIN    STATUS
----                           ----  ---      ------    ------
niu_0                          NIU   niu_0    primary
niu_1                          NIU   niu_1    primary
pci_0                          BUS   pci_0    primary   IOV
pci_1                          BUS   pci_1    primary
/SYS/MB/PCIE0                  PCIE  pci_0    primary   OCC
/SYS/MB/PCIE2                  PCIE  pci_0    primary   OCC
/SYS/MB/PCIE4                  PCIE  pci_0    primary   OCC
/SYS/MB/PCIE6                  PCIE  pci_0    primary   EMP
/SYS/MB/PCIE8                  PCIE  pci_0    primary   EMP
/SYS/MB/SASHBA                 PCIE  pci_0    primary   OCC
/SYS/MB/NET0                   PCIE  pci_0    primary   OCC
/SYS/MB/PCIE1                  PCIE  pci_1    primary   OCC
/SYS/MB/PCIE3                  PCIE  pci_1    primary   OCC
/SYS/MB/PCIE5                  PCIE  pci_1    primary   OCC
/SYS/MB/PCIE7                  PCIE  pci_1    primary   EMP
/SYS/MB/PCIE9                  PCIE  pci_1    primary   EMP
/SYS/MB/NET2                   PCIE  pci_1    primary   OCC
/SYS/MB/NET0/IOVNET.PF0        PF    pci_0    primary
/SYS/MB/NET0/IOVNET.PF1        PF    pci_0    primary
/SYS/MB/PCIE5/IOVNET.PF0       PF    pci_1    primary
/SYS/MB/PCIE5/IOVNET.PF1       PF    pci_1    primary
/SYS/MB/NET2/IOVNET.PF0        PF    pci_1    primary
/SYS/MB/NET2/IOVNET.PF1        PF    pci_1    primary
/SYS/MB/NET0/IOVNET.PF0.VF0    VF    pci_0    ldg1
```

使用 InfiniBand SR-IOV 虚拟功能

InfiniBand SR-IOV 设备仅支持静态 SR-IOV 功能。

要最大限度地缩短停机时间，请在根域处于延迟重新配置状态或来宾域已停止时作为一个组运行所有 SR-IOV 命令。以这种方式限制的 SR-IOV 命令包括 ldm create-vf、ldm destroy-vf、ldm add-io 和 ldm remove-io 命令。

通常，虚拟功能会分配到多个来宾域。重新引导根域会影响已分配有该根域的虚拟功能的所有来宾域。

由于未使用的 InfiniBand 虚拟功能几乎不会产生任何开销，因此，可以通过提前创建所需的虚拟功能（即使不会立即使用这些虚拟功能）来避免停机时间。

InfiniBand SR-IOV 硬件要求

有关所需的 PCIe InfiniBand SR-IOV 硬件的信息，请参见“SR-IOV 硬件和软件要求” [83]。

要获得 InfiniBand SR-IOV 支持，根域必须至少运行 Oracle Solaris 11.1.10.5.0 OS。I/O 域可以运行 Oracle Solaris 10 1/13 OS 加上修补程序 148888-04，或者至少运行 Oracle Solaris 11.1.10.5.0 OS。

创建和销毁 InfiniBand 虚拟功能

▼ 如何创建 InfiniBand 虚拟功能

此过程介绍如何创建 InfiniBand SR-IOV 虚拟功能。

1. **在根域上启动延迟重新配置。**

```
primary# ldm start-reconf root-domain-name
```

2. **通过设置 `iov=on` 启用 I/O 虚拟化。**

请只有在未对具有物理功能的总线启用 I/O 虚拟化时才执行此步骤。

```
primary# ldm set-io iov=on bus
```

3. **创建一个或多个与该根域中的物理功能关联的虚拟功能。**

```
primary# ldm create-vf pf-name
```

您可以为要创建的每个虚拟功能运行此命令。您还可以使用 -n 选项通过单个命令创建基于同一物理功能的多个虚拟功能。请参见例 8-6 “创建多个 SR-IOV 以太网虚拟功能” 和 `ldm`(1M) 手册页。

4. **重新引导根域。**

运行以下命令之一：

- **重新引导非 `primary` 根域。**

```
primary# ldm stop-domain -r root-domain
```

- 重新引导 **primary** 根域。

```
primary# shutdown -i6 -g0 -y
```

例 8-14 创建 InfiniBand 虚拟功能

以下示例显示有关物理功能 /SYS/MB/RISER1/PCIE4/IOVIB.PF0 物理功能的信息：

- 此物理功能位于 PCIE 插槽 4 中。
- IOVIB 字符串表示该物理功能是一个 InfiniBand SR-IOV 设备。

```
primary# ldm list-io
NAME                                    TYPE   BUS      DOMAIN   STATUS
----                                    ----   ---      ------   ------
pci_0                                   BUS    pci_0    primary
niu_0                                   NIU    niu_0    primary
/SYS/MB/RISER0/PCIE0                    PCIE   pci_0    primary  EMP
/SYS/MB/RISER1/PCIE1                    PCIE   pci_0    primary  EMP
/SYS/MB/RISER2/PCIE2                    PCIE   pci_0    primary  EMP
/SYS/MB/RISER0/PCIE3                    PCIE   pci_0    primary  OCC
/SYS/MB/RISER1/PCIE4                    PCIE   pci_0    primary  OCC
/SYS/MB/RISER2/PCIE5                    PCIE   pci_0    primary  EMP
/SYS/MB/SASHBA0                         PCIE   pci_0    primary  OCC
/SYS/MB/SASHBA1                         PCIE   pci_0    primary  OCC
/SYS/MB/NET0                            PCIE   pci_0    primary  OCC
/SYS/MB/NET2                            PCIE   pci_0    primary  OCC
/SYS/MB/RISER0/PCIE3/IOVIB.PF0          PF     pci_0    primary
/SYS/MB/RISER1/PCIE4/IOVIB.PF0          PF     pci_0    primary
/SYS/MB/NET0/IOVNET.PF0                 PF     pci_0    primary
/SYS/MB/NET0/IOVNET.PF1                 PF     pci_0    primary
/SYS/MB/NET2/IOVNET.PF0                 PF     pci_0    primary
/SYS/MB/NET2/IOVNET.PF1                 PF     pci_0    primary
```

以下命令显示有关指定物理功能的更多详细信息。maxvfs 值表示该设备所支持的最多虚拟功能数。

```
primary# ldm list-io -l /SYS/MB/RISER1/PCIE4/IOVIB.PF0
NAME                                    TYPE   BUS      DOMAIN   STATUS
----                                    ----   ---      ------   ------
/SYS/MB/RISER1/PCIE4/IOVIB.PF0          PF     pci_0    primary
[pci@400/pci@1/pci@0/pci@0/pciex15b3,673c@0]
    maxvfs = 64
```

以下示例说明如何创建静态虚拟功能。首先，在 primary 域上启动延迟重新配置，然后对 pci_0 PCIe 总线启用 I/O 虚拟化。由于已将 pci_0 总线分配到 primary 根域，因此，请使用 ldm set-io 命令启用 I/O 虚拟化。

```
primary# ldm start-reconf primary
Initiating a delayed reconfiguration operation on the primary domain.
```

```
All configuration changes for other domains are disabled until the primary
domain reboots, at which time the new configuration for the primary domain
will also take effect.

primary# ldm set-io iov=on pci_0
------------------------------------------------------------------------
Notice: The primary domain is in the process of a delayed reconfiguration.
Any changes made to the primary domain will only take effect after it reboots.
------------------------------------------------------------------------
```

现在，可以使用 `ldm create-vf` 命令从 `/SYS/MB/RISER1/PCIE4/IOVIB.PF0` 物理功能创建虚拟功能。

```
primary# ldm create-vf /SYS/MB/RISER1/PCIE4/IOVIB.PF0
------------------------------------------------------------------------
Notice: The primary domain is in the process of a delayed reconfiguration.
Any changes made to the primary domain will only take effect after it reboots.
------------------------------------------------------------------------
Created new vf: /SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF0
```

请注意，在同一个延迟重新配置过程中，可以创建多个虚拟功能。以下命令可以再创建一个虚拟功能：

```
primary# ldm create-vf /SYS/MB/RISER1/PCIE4/IOVIB.PF0
------------------------------------------------------------------------
Notice: The primary domain is in the process of a delayed reconfiguration.
Any changes made to the primary domain will only take effect after it reboots.
------------------------------------------------------------------------
Created new vf: /SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF1
```

最后，重新引导 `primary` 根域以使更改生效。

```
primary# shutdown -i6 -g0 -y
Shutdown started.

Changing to init state 6 - please wait
...
```

▼ 如何销毁 InfiniBand 虚拟功能

此过程介绍如何销毁 InfiniBand SR-IOV 虚拟功能。

如果当前未将虚拟功能分配给域，则可以销毁该虚拟功能。虚拟功能只能按照与创建时相反的顺序进行销毁，因此，只能销毁已创建的最后一个虚拟功能。生成的配置由物理功能驱动程序验证。

1. **在根域上启动延迟重新配置。**

   ```
   primary# ldm start-reconf root-domain-name
   ```

2. **销毁一个或多个与该根域中的物理功能关联的虚拟功能。**

```
primary# ldm destroy-vf vf-name
```

您可以为要销毁的每个虚拟功能运行此命令。您还可以使用 -n 选项通过单个命令销毁基于同一物理功能的多个虚拟功能。请参见例 8-8 “销毁多个以太网 SR-IOV 虚拟功能” 和 ldm(1M) 手册页。

3. 重新引导根域。

   运行以下命令之一：

   - 重新引导非 **primary** 根域。

     ```
     primary# ldm stop-domain -r root-domain
     ```

   - 重新引导 **primary** 根域。

     ```
     primary# shutdown -i6 -g0 -y
     ```

例 8-15 销毁 InfiniBand 虚拟功能

以下示例说明如何销毁静态 InfiniBand 虚拟功能 /SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF1。

ldm list-io 命令可显示有关总线、物理功能和虚拟功能的信息。

```
primary# ldm list-io
NAME                                    TYPE   BUS      DOMAIN STATUS
----                                    ----   ---      ------ ------
pci_0                                   BUS    pci_0    primary  IOV
...
/SYS/MB/RISER1/PCIE4/IOVIB.PF0          PF     pci_0    primary
...
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF0      VF     pci_0
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF1      VF     pci_0
```

可以使用 ldm list-io -l 命令获取有关物理功能和相关虚拟功能的更多详细信息。

```
primary# ldm list-io -l /SYS/MB/RISER1/PCIE4/IOVIB.PF0
NAME                                    TYPE   BUS      DOMAIN STATUS
----                                    ----   ---      ------ ------
/SYS/MB/RISER1/PCIE4/IOVIB.PF0          PF     pci_0    primary
[pci@400/pci@1/pci@0/pci@0/pciex15b3,673c@0]
    maxvfs = 64
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF0      VF     pci_0
[pci@400/pci@1/pci@0/pci@0/pciex15b3,673c@0,1]
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF1      VF     pci_0
[pci@400/pci@1/pci@0/pci@0/pciex15b3,673c@0,2]
```

只有当未将某一虚拟功能分配到域时，才能销毁该虚拟功能。ldm list-io -l 输出中的 DOMAIN 列可显示将虚拟功能分配到的任何域的名称。此外，虚拟功能必须按照

与创建时相反的顺序来销毁。因此，在此示例中，必须先销毁 /SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF1 虚拟功能，然后再销毁 /SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF0 虚拟功能。

在确定了正确的虚拟功能之后，可以将其销毁。首先，启动延迟重新配置。

```
primary# ldm start-reconf primary
Initiating a delayed reconfiguration operation on the primary domain.
All configuration changes for other domains are disabled until the primary
domain reboots, at which time the new configuration for the primary domain
will also take effect.

primary# ldm destroy-vf /SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF1
--------------------------------------------------------------------------
Notice: The primary domain is in the process of a delayed reconfiguration.
Any changes made to the primary domain will only take effect after it reboots.
--------------------------------------------------------------------------
```

可以在延迟重新配置状态下发出多个 ldm destroy-vf 命令。这样，您也可以销毁 /SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF0。

最后，重新引导 primary 根域以使更改生效。

```
primary# shutdown -i6 -g0 -y
Shutdown started.

Changing to init state 6 - please wait
...
```

在 I/O 域中添加和删除 InfiniBand 虚拟功能

▼ 如何将 InfiniBand 虚拟功能添加到 I/O 域

此过程介绍如何向 I/O 域添加 InfiniBand SR-IOV 虚拟功能。

1. 停止 I/O 域。

   ```
   primary# ldm stop-domain domain-name
   ```

2. 将一个或多个虚拟功能添加到 I/O 域。

   vf-name 是虚拟功能的 pseudonym 名称或路径名称。建议做法是使用 pseudonym 名称。*domain-name* 用于指定要将虚拟功能添加到的域的名称。指定的 I/O 域必须处于非活动或绑定状态。

   ```
   primary# ldm add-io vf-name domain-name
   ```

3. 启动 I/O 域。

   ```
   primary# ldm start-domain domain-name
   ```

例 8-16 添加 InfiniBand 虚拟功能

以下示例说明如何将 /SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF2 虚拟功能添加到 iodom1 I/O 域。

首先，确定要分配的虚拟功能。

```
primary# ldm list-io
NAME                                    TYPE   BUS      DOMAIN STATUS
----                                    ----   ---      ------ ------
pci_0                                   BUS    pci_0    primary  IOV
...
/SYS/MB/RISER1/PCIE4/IOVIB.PF0          PF     pci_0    primary
...
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF0      VF     pci_0
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF1      VF     pci_0
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF2      VF     pci_0
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF3      VF     pci_0
```

要将某个虚拟功能添加到 I/O 域，该虚拟功能必须尚未分配。DOMAIN 列可指示要将该虚拟功能分配到的域的名称。在此示例中，/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF2 未分配到域。

要将虚拟功能添加到某个域，该域必须处于非活动或绑定状态。

```
primary# ldm list-domain
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  NORM  UPTIME
primary          active     -n-cv-  UART    32    64G      0.2%  0.2%  56m
iodom1           active     -n----  5000    8     8G        33%   33%  25m
```

ldm list-domain 输出显示 iodom1 I/O 域处于活动状态，因此，必须将其停止。

```
primary# ldm stop iodom1
LDom iodom1 stopped
primary# ldm list-domain
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  NORM  UPTIME
primary          active     -n-cv-  UART    32    64G      0.0%  0.0%  57m
iodom1           bound      ------  5000    8     8G
```

现在，您可以将该虚拟功能添加到该 I/O 域。

```
primary# ldm add-io /SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF2 iodom1
primary# ldm list-io
...
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF2      VF     pci_0    iodom1
```

请注意，可以在 I/O 域停止期间添加多个虚拟功能。例如，可以将其他未分配的虚拟功能（如 /SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF3）添加到 iodom1 中。添加虚拟功能后，可以重新启动 I/O 域。

```
primary# ldm start iodom1
LDom iodom1 started
```

```
primary# ldm list-domain
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  NORM  UPTIME
primary          active     -n-cv-  UART    32    64G      1.0%  1.0%  1h 18m
iodom1           active     -n----  5000    8     8G        36%   36%  1m
```

▼ 如何从 I/O 域删除 InfiniBand 虚拟功能

此过程介绍如何从 I/O 域中删除 InfiniBand SR-IOV 虚拟功能。

1. 停止 I/O 域。

```
primary# ldm stop-domain domain-name
```

2. 从 I/O 域中删除一个或多个虚拟功能。

vf-name 是虚拟功能的 pseudonym 名称或路径名称。建议做法是使用设备 pseudonym。*domain-name* 用于指定要从中删除虚拟功能的域的名称。指定的 I/O 域必须处于非活动或绑定状态。

注 - 从 I/O 域删除虚拟功能之前，请确保该虚拟功能在引导该域时并不至关重要。

```
primary# ldm remove-io vf-name domain-name
```

3. 启动 I/O 域。

```
primary# ldm start-domain domain-name
```

例 8-17 删除 InfiniBand 虚拟功能

以下示例说明如何从 `iodom1` I/O 域删除 `/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF2` 虚拟功能。

首先，确定要删除的虚拟功能。

```
primary# ldm list-io
NAME                                  TYPE   BUS      DOMAIN STATUS
----                                  ----   ---      ------ ------
pci_0                                 BUS    pci_0    primary  IOV
...
/SYS/MB/RISER1/PCIE4/IOVIB.PF0        PF     pci_0    primary
...
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF0    VF     pci_0
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF1    VF     pci_0
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF2    VF     pci_0    iodom1
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF3    VF     pci_0    iodom1
```

DOMAIN 列显示虚拟功能所分配到的域的名称。`/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF2` 虚拟功能已分配到 `iodom1`。

要从某个 I/O 域中删除虚拟功能，该域必须处于非活动或绑定状态。使用 `ldm list-domain` 命令可确定域的状态。

```
primary# ldm list-domain
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  NORM  UPTIME
primary          active     -n-cv-  UART    32    64G      0.3%  0.3%  29m
iodom1           active     -n----  5000    8     8G        17%   17%  11m
```

在此示例中，`iodom1` 域处于活动状态，因此，必须将其停止。

```
primary# ldm stop iodom1
LDOM iodom1 stopped
primary# ldm list-domain
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  NORM  UPTIME
primary          active     -n-cv-  UART    32    64G      0.0%  0.0%  31m
iodom1           bound      ------  5000    8     8G
```

现在，您可以从 `iodom1` 中删除 `/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF2` 虚拟功能。

```
primary# ldm remove-io /SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF2 iodom1
primary# ldm list-io
NAME                                    TYPE   BUS      DOMAIN STATUS
----                                    ----   ---      ------ ------
...
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF2      VF     pci_0
...
```

请注意，该虚拟功能的 DOMAIN 列现在为空。

可以在 I/O 域停止期间删除多个虚拟功能。在此示例中，您也可以删除 `/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF3` 虚拟功能。删除虚拟功能后，可以重新启动 I/O 域。

```
primary# ldm start iodom1
LDom iodom1 started
primary# ldm list-domain
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  NORM  UPTIME
primary          active     -n-cv-  UART    32    64G      0.3%  0.3%  39m
iodom1           active     -n----  5000    8     8G       9.4%  9.4%  5s
```

在根域中添加和删除 InfiniBand 虚拟功能

▼ 如何将 InfiniBand 虚拟功能添加到根域

此过程介绍如何向根域添加 InfiniBand SR-IOV 虚拟功能。

1. **启动延迟重新配置。**

```
primary# ldm start-reconf root-domain
```

2. **将一个或多个虚拟功能添加到根域。**

vf-name 是虚拟功能的 pseudonym 名称或路径名称。建议做法是使用 pseudonym 名称。*root-domain-name* 用于指定要将虚拟功能添加到的根域的名称。

```
primary# ldm add-io vf-name root-domain-name
```

3. 重新引导根域。
运行以下命令之一：

   - 重新引导非 **primary** 根域。

     ```
     primary# ldm stop-domain -r root-domain-name
     ```

   - 重新引导 **primary** 根域。

     ```
     primary# shutdown -i6 -g0 -y
     ```

▼ 如何从根域删除 InfiniBand 虚拟功能

此过程介绍如何从根域中删除 InfiniBand SR-IOV 虚拟功能。

1. 启动延迟重新配置。

   ```
   primary# ldm start-reconf root-domain
   ```

2. 从根域删除一个或多个虚拟功能。
vf-name 是虚拟功能的 pseudonym 名称或路径名称。建议做法是使用 pseudonym 名称。*root-domain-name* 用于指定要将虚拟功能添加到的根域的名称。

   ```
   primary# ldm remove-io vf-name root-domain-name
   ```

3. 重新引导根域。
运行以下命令之一：

   - 重新引导非 **primary** 根域。

     ```
     primary# ldm stop-domain -r root-domain-name
     ```

   - 重新引导 **primary** 根域。

     ```
     primary# shutdown -i6 -g0 -y
     ```

高级 SR-IOV 主题：InfiniBand SR-IOV

本节介绍如何确定 InfiniBand 物理功能和虚拟功能，以及如何将 Logical Domains Manager 与 InfiniBand 物理功能和虚拟功能的 Oracle Solaris 视图关联起来。

列出 InfiniBand SR-IOV 虚拟功能

以下示例说明使用不同方法来显示有关 /SYS/MB/RISER1/PCIE4/IOVIB.PF0 物理功能的信息。如果某个物理功能名称包含 IOVIB 字符串，则表示这是一个 InfiniBand SR-IOV 设备。

```
primary# ldm list-io
NAME                                    TYPE   BUS      DOMAIN   STATUS
----                                    ----   ---      ------   ------
pci_0                                   BUS    pci_0    primary  IOV
niu_0                                   NIU    niu_0    primary
/SYS/MB/RISER0/PCIE0                    PCIE   pci_0    primary  EMP
/SYS/MB/RISER1/PCIE1                    PCIE   pci_0    primary  EMP
/SYS/MB/RISER2/PCIE2                    PCIE   pci_0    primary  EMP
/SYS/MB/RISER0/PCIE3                    PCIE   pci_0    primary  OCC
/SYS/MB/RISER1/PCIE4                    PCIE   pci_0    primary  OCC
/SYS/MB/RISER2/PCIE5                    PCIE   pci_0    primary  EMP
/SYS/MB/SASHBA0                         PCIE   pci_0    primary  OCC
/SYS/MB/SASHBA1                         PCIE   pci_0    primary  OCC
/SYS/MB/NET0                            PCIE   pci_0    primary  OCC
/SYS/MB/NET2                            PCIE   pci_0    primary  OCC
/SYS/MB/RISER0/PCIE3/IOVIB.PF0          PF     pci_0    primary
/SYS/MB/RISER1/PCIE4/IOVIB.PF0          PF     pci_0    primary
/SYS/MB/NET0/IOVNET.PF0                 PF     pci_0    primary
/SYS/MB/NET0/IOVNET.PF1                 PF     pci_0    primary
/SYS/MB/NET2/IOVNET.PF0                 PF     pci_0    primary
/SYS/MB/NET2/IOVNET.PF1                 PF     pci_0    primary
/SYS/MB/RISER0/PCIE3/IOVIB.PF0.VF0      VF     pci_0    primary
/SYS/MB/RISER0/PCIE3/IOVIB.PF0.VF1      VF     pci_0    primary
/SYS/MB/RISER0/PCIE3/IOVIB.PF0.VF2      VF     pci_0    iodom1
/SYS/MB/RISER0/PCIE3/IOVIB.PF0.VF3      VF     pci_0    iodom1
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF0      VF     pci_0    primary
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF1      VF     pci_0    primary
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF2      VF     pci_0    iodom1
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF3      VF     pci_0    iodom1
```

ldm list-io -l 命令可提供有关指定物理功能设备 /SYS/MB/RISER1/PCIE4/IOVIB.PF0 的更多详细信息。maxvfs 值显示该物理设备可支持的最多虚拟功能数为 64。对于与物理功能关联的每个虚拟功能，输出会显示以下信息：

- 功能名称
- 功能类型
- 总线名称
- 域名
- 功能的可选状态
- 设备路径

此 ldm list-io -l 输出显示 VF0 和 VF1 已分配到 primary 域，而 VF2 和 VF3 已分配到 iodom1 I/O 域。

```
primary# ldm list-io -l /SYS/MB/RISER1/PCIE4/IOVIB.PF0
NAME                                    TYPE   BUS      DOMAIN   STATUS
----                                    ----   ---      ------   ------
/SYS/MB/RISER1/PCIE4/IOVIB.PF0          PF     pci_0    primary
[pci@400/pci@1/pci@0/pci@0/pciex15b3,673c@0]
    maxvfs = 64
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF0      VF     pci_0    primary
[pci@400/pci@1/pci@0/pci@0/pciex15b3,673c@0,1]
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF1      VF     pci_0    primary
[pci@400/pci@1/pci@0/pci@0/pciex15b3,673c@0,2]
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF2      VF     pci_0    iodom1
[pci@400/pci@1/pci@0/pci@0/pciex15b3,673c@0,3]
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF3      VF     pci_0    iodom1
[pci@400/pci@1/pci@0/pci@0/pciex15b3,673c@0,4]
```

确定 InfiniBand SR-IOV 功能

本节介绍如何在 Oracle Solaris 11 和 Oracle Solaris 10 系统上确定 InfiniBand SR-IOV 设备。

使用 `ldm list-io -l` 命令可显示与每个物理功能和虚拟功能关联的 Oracle Solaris 设备路径名称。

```
primary# ldm list-io -l /SYS/MB/RISER1/PCIE4/IOVIB.PF0
NAME                                    TYPE   BUS      DOMAIN   STATUS
----                                    ----   ---      ------   ------
/SYS/MB/RISER1/PCIE4/IOVIB.PF0          PF     pci_0    primary
[pci@400/pci@1/pci@0/pci@0/pciex15b3,673c@0]
    maxvfs = 64
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF0      VF     pci_0    primary
[pci@400/pci@1/pci@0/pci@0/pciex15b3,673c@0,1]
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF1      VF     pci_0    primary
[pci@400/pci@1/pci@0/pci@0/pciex15b3,673c@0,2]
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF2      VF     pci_0    iodom1
[pci@400/pci@1/pci@0/pci@0/pciex15b3,673c@0,3]
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF3      VF     pci_0    iodom1
[pci@400/pci@1/pci@0/pci@0/pciex15b3,673c@0,4]
```

Oracle Solaris 11：

使用 Oracle Solaris 11 `dladm show-phys -L` 命令可将每个 IP over InfiniBand (IPoIB) 实例与其物理卡匹配。例如，以下命令显示哪些 IPoIB 实例使用插槽 `PCIE4` 中的卡，即，前面的 `ldm list-io -l` 示例中显示的卡。

```
primary# dladm show-phys -L | grep PCIE4
net5              ibp0         PCIE4/PORT1
net6              ibp1         PCIE4/PORT2
net19             ibp8         PCIE4/PORT1
net9              ibp9         PCIE4/PORT2
net18             ibp4         PCIE4/PORT1
```

```
net11             ibp5         PCIE4/PORT2
```

每个 InfiniBand 主机通道适配器 (host channel adapter, HCA) 设备都有一个全局唯一标识符 (globally unique ID, GUID)。每个端口也有 GUID（通常，一个 HCA 具有两个端口）。InfiniBand HCA GUID 用于唯一标识相应的适配器。端口 GUID 用于唯一标识每个 HCA 端口，并承担与网络设备的 MAC 地址类似的角色。这些 16 位十六进制数字 GUID 可供 InfiniBand 管理工具和诊断工具使用。

使用 Oracle Solaris 11 `dladm show-ib` 命令可获取有关 InfiniBand SR-IOV 设备的 GUID 信息。同一设备的物理功能和虚拟功能具有相关的 HCA GUID 值。HCA GUID 的第 11 位十六进制数字显示了物理功能与其虚拟功能之间的关系。请注意，在 HCAGUID 和 PORTGUID 列中，前导零均已禁止显示。

例如，物理功能 `PF0` 具有两个虚拟功能：`VF0` 和 `VF1`，这两个虚拟功能已分配到 `primary` 域。每个虚拟功能的第 11 位十六进制数字都是在相关物理功能的基础上递增 1。因此，如果 `PF0` 的 GUID 为 `8`，则 `VF0` 和 `VF1` 的 GUID 将分别为 `9` 和 `A`。

以下 `dladm show-ib` 命令输出显示 `net5` 和 `net6` 链路属于物理功能 `PF0`。`net19` 和 `net9` 链路属于同一设备的 `VF0`，而 `net18` 和 `net11` 链路则属于 `VF1`。

```
primary# dladm show-ib
LINK         HCAGUID         PORTGUID        PORT STATE  PKEYS
net6         2128**0**001A17F56  21280001A17F58  2    up     FFFF
net5         2128**0**001A17F56  21280001A17F57  1    up     FFFF
net19        2129**0**001A17F56  14050000000001  1    up     FFFF
net9         2129**0**001A17F56  14050000000008  2    up     FFFF
net18        212**A**0001A17F56  14050000000002  1    up     FFFF
net11        212**A**0001A17F56  14050000000009  2    up     FFFF
```

以下 Oracle Solaris 11 `dladm show-phys` 输出中的设备显示了链路和底层 InfiniBand 端口设备 (`ibp`*X*) 之间的关系。

```
primary# dladm show-phys
LINK              MEDIA                STATE      SPEED  DUPLEX    DEVICE
...
net6              Infiniband           up         32000  unknown   ibp1
net5              Infiniband           up         32000  unknown   ibp0
net19             Infiniband           up         32000  unknown   ibp8
net9              Infiniband           up         32000  unknown   ibp9
net18             Infiniband           up         32000  unknown   ibp4
net11             Infiniband           up         32000  unknown   ibp5
```

使用 `ls -l` 命令可显示实际 InfiniBand 端口（IB 端口）设备路径。IB 端口设备是 `ldm list-io -l` 输出中所示设备路径的子项。物理功能的单元地址包含一部分（如 `pciex15b3,673c@0`），而虚拟功能的单元地址则包含两部分（如 `pciex15b3,1002@0,2`）。此单元地址的第二部分比虚拟功能编号大 1。（在此示例中，第二个组成部分为 `2`，因此，该设备为虚拟功能 `1`。）以下输出显示 `/dev/ibp0` 为物理功能，而 `/dev/ibp5` 为虚拟功能。

```
primary# ls -l /dev/ibp0
```

```
lrwxrwxrwx   1 root     root          83 Apr 18 12:02 /dev/ibp0 ->
../devices/pci@400/pci@1/pci@0/pci@0/pciex15b3,673c@0/hermon@0/ibport@1,0,ipib:ibp0
primary# ls -l /dev/ibp5
lrwxrwxrwx   1 root     root          85 Apr 22 23:29 /dev/ibp5 ->
../devices/pci@400/pci@1/pci@0/pci@0/pciex15b3,1002@0,2/hermon@3/ibport@2,0,ipib:ibp5
```

可以使用 OpenFabrics `ibv_devices` 命令查看 OpenFabrics 设备名称和节点 (HCA) GUID。如果具有虚拟功能，则 Type 列可指示此功能为物理功能还是虚拟功能。

```
primary# ibv_devices
device                 node GUID            type
------              ----------------        ----
mlx4_4              0002c90300a38910         PF
mlx4_5              0021280001a17f56         PF
mlx4_0              0002cb0300a38910         VF
mlx4_1              0002ca0300a38910         VF
mlx4_2              00212a0001a17f56         VF
mlx4_3              0021290001a17f56         VF
```

Oracle Solaris 10：

在 Oracle Solaris 10 来宾 I/O 域中，使用 `dladm show-dev` 命令可显示每个 IPoIB 实例，其名称格式为 `ibd`*xx*。

```
# dladm show-dev
vnet0           link: up        speed: 0     Mbps       duplex: unknown
ibd0            link: up        speed: 32000 Mbps       duplex: unknown
ibd1            link: up        speed: 32000 Mbps       duplex: unknown
ibd2            link: up        speed: 32000 Mbps       duplex: unknown
ibd3            link: up        speed: 32000 Mbps       duplex: unknown
```

可以对 `/devices/` 目录中的 HCA 路径名称使用 `ls -l` 命令来提取 HCA 及其 HCA GUID。

```
# ls -l /devices/ib\:[0-9]*
crw-r--r--   1 root     sys       67,  0 Jun 12 16:27 /devices/ib:212B0001A17F56
crw-r--r--   1 root     sys       67,  0 Jun 12 16:27 /devices/ib:212C0001A17F56
```

`ibv_devices` 输出中的 GUID（请注意第 11 位十六进制数字，即，此示例中的 "B" 和 "C"）表示这些虚拟功能已分配到 Oracle Solaris 10 域。可以通过对 `/dev` IPoIB 路径名称使用 `ls -l` 命令来获取有关 IPoIB 实例的更多信息。

```
# ls -l /dev/ibd*
lrwxrwxrwx   1 root     other         29 May 23 16:26 /dev/ibd ->
../devices/pseudo/clone@0:ibd
lrwxrwxrwx   1 root     root          89 May 31 10:52 /dev/ibd0 ->
../devices/pci@400/pci@1/pci@0/pci@0/pciex15b3,1002@0,3/hermon@0/ibport@1,ffff,ipib:ibd0
lrwxrwxrwx   1 root     root          89 May 31 10:52 /dev/ibd1 ->
../devices/pci@400/pci@1/pci@0/pci@0/pciex15b3,1002@0,3/hermon@0/ibport@2,ffff,ipib:ibd1
lrwxrwxrwx   1 root     root          89 Jun 12 18:36 /dev/ibd2 ->
../devices/pci@400/pci@1/pci@0/pci@0/pciex15b3,1002@0,4/hermon@1/ibport@1,ffff,ipib:ibd2
lrwxrwxrwx   1 root     root          89 Jun 12 18:36 /dev/ibd3 ->
../devices/pci@400/pci@1/pci@0/pci@0/pciex15b3,1002@0,4/hermon@1/ibport@2,ffff,ipib:ibd3
```

每个路径均以 `ldm list-io -l` 输出中显示的设备路径开头。虚拟功能（如 `pciex15b3,1002@0,4`）的单元地址包含两部分，其中的第二部分比虚拟功能编号（在此示例中为 VF3）大 1。

`ibport` 设备的单元地址包含三部分，后跟冒号和 IPoIB 设备实例名称。此单元地址的第一部分为端口号。第二部分为分区键（P 键）十六进制值。请注意，InfiniBand P 键值类似于以太网的 VLAN。第三部分为字符串 `ipib`。

`ls -l /dev/ibd3` 命令输出显示 `ibd3` IPoIB 实例使用端口 `2` 和 P 键值 `ffff`。

使用光纤通道 SR-IOV 虚拟功能

从 Oracle VM Server for SPARC 3.1.1 发行版起，提供了对 SR-IOV 光纤通道的支持。SR-IOV 光纤通道主机总线适配器 (host bus adapter, HBA) 可以具有一个或多个端口，其中每一个都显示为 SR-IOV 物理功能。对于光纤通道物理功能，您可以通过其设备名称中的 `IOVFC` 字符串来识别它们。

每个光纤通道物理功能都具有唯一的端口和节点全球名称 (world-wide name, WWN) 值，这些值由卡的制造商提供。当基于光纤通道物理功能创建虚拟功能时，虚拟功能的行为类似于光纤通道 HBA 设备。每个虚拟功能都必须具有一个由 SAN 结构的端口 WWN 和节点 WWN 指定的唯一标识。您可以使用 Logical Domains Manager 自动或手动分配端口和节点 WWN。通过分配您自己的值，您可以完全控制任何虚拟功能的标识。

光纤通道 HBA 虚拟功能使用 N_Port ID 虚拟化 (N_Port ID Virtualization, NPIV) 方法登录到 SAN 结构。根据此 NPIV 的要求，您必须将光纤通道 HBA 端口连接到支持 NPIV 的光纤通道交换机。虚拟功能完全由 SR-IOV 卡的硬件或固件进行管理。除了这些例外之外，光纤通道虚拟功能的工作和行为方式与非 SR-IOV 光纤通道 HBA 设备相同。SR-IOV 虚拟功能具有与非 SR-IOV 设备相同的功能，因此每种配置都支持所有类型的 SAN 存储设备。

通过虚拟功能唯一的端口 WWN 和节点 WWN 值，SAN 管理员可以使用处理非 SR-IOV 光纤通道 HBA 端口的相同方式向虚拟功能分配存储。此管理包括区域划分、LUN 掩码以及服务质量 (quality of service, QoS)。您可以对存储进行配置，使其只供特定的逻辑域独占访问，对根域中的物理功能不可见。

您可以同时使用静态和动态 SR-IOV 方法管理光纤通道 SR-IOV 设备。

光纤通道 SR-IOV 硬件要求

有关所需的 PCIe 光纤通道 SR-IOV 硬件的信息，请参见“SR-IOV 硬件和软件要求” [83]。

- 控制域。
  - Oracle Solaris 11 OS。
    - QLogic 卡。至少 Oracle Solaris 11.2.0.42.0 OS
    - Emulex 卡。至少 Oracle Solaris 11.1.17.4.0 OS
  - Oracle Solaris 10 OS。仅在 SPARC T 系列平台和 Fujitsu M10 服务器上。
    - QLogic 卡。至少 Oracle Solaris 10 1/13 OS、Oracle Solaris 10 修补程序 ID 150400-11 加上设备驱动程序修补程序 ID 149175-06
    - Emulex 卡。至少 Oracle Solaris 10 1/13 OS、Oracle Solaris 10 修补程序 ID 150400-11 加上 Fujitsu M10 服务器的设备驱动程序修补程序 ID 149173-04。SPARC T 系列和 SPARC M 系列系统上的 SR-IOV 功能不支持。
- I/O 域。
  - Oracle Solaris 11 OS。
    - QLogic 卡。至少 Oracle Solaris 11.2.0.42.0 OS
    - Emulex 卡。至少 Oracle Solaris 11.1.17.4.0 OS
  - Oracle Solaris 10 OS。
    - QLogic 卡。至少 Oracle Solaris 10 1/13 OS、Oracle Solaris 10 修补程序 ID 150400-11 加上设备驱动程序修补程序 ID 149175-06
    - Emulex 卡。至少 Oracle Solaris 10 1/13 OS、Oracle Solaris 10 修补程序 ID 150400-11 加上设备驱动程序修补程序 ID 149173-04

光纤通道 SR-IOV 要求和限制

光纤通道 SR-IOV 功能具有以下建议和限制：

- SR-IOV 卡必须运行支持 SR-IOV 功能的最新版本固件。
- 光纤通道 PCIe 卡必须连接到支持 NPIV 且与 PCIe 卡兼容的光纤通道交换机。
- Logical Domains Manager 通过将所有系统的控制域连接到同一个 SAN 结构并使其属于同一多播域来正确地自动生成唯一的 `port-wwn` 和 `node-wwn` 属性值。

  如果无法配置此环境，则在创建虚拟功能时必须手动提供 `node-wwn` 和 `port-wwn` 值。此行为用于确保没有命名冲突。请参见“光纤通道虚拟功能的全球名称分配” [124]。

特定于光纤通道设备类的属性

可以使用 `ldm create-vf` 或 `ldm set-io` 命令设置以下光纤通道虚拟功能属性：

`bw-percent` 指定要分配给光纤通道虚拟功能的带宽所占的百分比。有效值为 0 到 100。分配给光纤通道物理功能的虚拟功能的总带宽值不能超过

100。默认值为 0，以便使该虚拟功能从共享同一物理功能的其他虚拟功能尚未占用的带宽中获得公平的份额。

`node-wwn` 指定光纤通道虚拟功能的节点全球名称 (world-wide name, WWN)。有效值为非零值。默认情况下，此值是自动分配的。如果您手动指定此值，则还必须指定 `port-wwn` 属性的值。有关更多信息，请参见“光纤通道虚拟功能的全球名称分配” [124]。

`port-wwn` 指定光纤通道虚拟功能的端口 WWN。有效值为非零值。默认情况下，此值是自动分配的。如果您手动指定此值，则还必须指定 `node-wwn` 属性的值。有关更多信息，请参见“光纤通道虚拟功能的全球名称分配” [124]。

当光纤通道虚拟功能在使用中时，无法修改 `node-wwn` 或 `port-wwn` 属性。不过，即使光纤通道虚拟功能在使用中，您也可以动态修改 `bw-percent` 属性值。

光纤通道虚拟功能的全球名称分配

Logical Domains Manager 同时支持自动和手动分配光纤通道虚拟功能的全球名称。

自动分配全球名称

Logical Domains Manager 从其 MAC 地址自动分配池中分配一个唯一的 MAC 地址并创建 IEEE 格式的 `node-wwn` 和 `port-wwn` 属性值。

```
port-wwn = 10:00:XX:XX:XX:XX:XX:XX
node-wwn = 20:00:XX:XX:XX:XX:XX:XX
```

XX:*XX*:*XX*:*XX*:*XX*:*XX* 是自动分配的 MAC 地址。

当连接到同一光纤通道结构的所有系统的控制域还通过以太网连接并且属于同一多播域时，此自动分配方法将生成唯一的 WWN。如果不能满足此要求，则必须手动分配唯一的 WWN，这在 SAN 上是必需的。

手动分配全球名称

您可以使用任何方法构造唯一的 WWN。本节介绍如何从 Logical Domains Manager MAC 地址手动分配池创建 WWN。必须确保您分配的 WWN 是唯一的。

Logical Domains Manager 具有一个包含 256,000 个 MAC 地址的池，这些地址可用于在 `00:14:4F:FC:00:00` - `00:14:4F:FF:FF:FF` 范围内进行手动分配。

以下示例显示了基于 `00:14:4F:FC:00:01` MAC 地址的 `port-wwn` 和 `node-wwn` 属性值：

```
port-wwn = 10:00:00:14:4F:FC:00:01
node-wwn = 20:00:00:14:4F:FC:00:01
```

`00:14:4F:FC:00:01` 是手动分配的 MAC 地址。有关 MAC 地址自动分配的信息，请参见“自动或手动分配 MAC 地址” [201]。

注 - 要确保 SAN 存储配置是可预测的，最好手动分配 WWN。

并非所有系统都通过以太网连接到同一个多播域时，必须使用手动分配 WWN 方法。还可以使用此方法确保在销毁和重新创建光纤通道虚拟功能时使用相同的 WWN。

创建光纤通道 SR-IOV 虚拟功能

本节介绍如何动态创建和销毁虚拟功能。如果无法使用动态方法来执行这些操作，请在创建或销毁虚拟功能之前在根域上启动延迟重新配置。

▼ 如何创建光纤通道 SR-IOV 虚拟功能

如果无法使用此动态方法，请改用静态方法。请参见“静态 SR-IOV” [87]。

1. **确定物理功能设备。**

   ```
   primary# ldm list-io
   ```

   请注意，物理功能的名称包括 PCIe SR-IOV 卡或板载设备的位置信息。

2. **如果具有物理功能的总线尚未启用 I/O 虚拟化，请启用它。**
   请只有在未对具有物理功能的总线启用 I/O 虚拟化时才执行此步骤。
   请参见如何对 PCIe 总线启用 I/O 虚拟化 [89]。

3. **基于物理功能用动态或静态方法创建单个虚拟功能或多个虚拟功能。**
   在创建一个或多个虚拟功能后，您可以将其分配给来宾域。

   - **动态方法：**

     - **要基于一个物理功能同时创建多个虚拟功能，请使用以下命令：**

       ```
       primary# ldm create-vf -n number | max pf-name
       ```

       使用 `ldm create-vf -n max` 命令可一次创建该物理功能的所有虚拟功能。此命令自动为每个虚拟功能分配端口和节点 WWN 并将 `bw-percent` 属性设置为默认值 `0`。此值指定向所有虚拟功能分配公平份额带宽。

提示 - 同时创建物理功能的所有虚拟功能。如果要手动分配 WWN，请首先创建所有虚拟功能，然后使用 `ldm set-io` 命令手动为每个虚拟功能分配 WWN 值。此方法最大程度地减少了基于物理功能创建虚拟功能时的状态转换次数。

您可以使用路径名称或 pseudonym 名称指定虚拟功能。但是，建议做法是使用 pseudonym 名称。

- **要基于一个物理功能创建一个虚拟功能，请使用以下命令：**

  ```
  ldm create-vf [bw-percent=value] [port-wwn=value node-wwn=value] pf-name
  ```

  您还可以手动指定特定于光纤通道类的属性值。

注 - OS 探测 IOV 设备时，有时无法立即使用新创建的虚拟功能。使用 `ldm list-io` 命令确定父物理功能及其子虚拟功能在 "Status"（状态）列中是否具有 `INV` 值。如果它们具有此值，则等待直到 `ldm list-io` 输出不再在 "Status"（状态）列中显示 `INV` 值（大约 45 秒），然后再使用该物理功能或其任何子虚拟功能。如果此状态持续存在，则设备存在问题。

根域重新引导（包括 `primary` 的重新引导）后或者您使用 `ldm create-vf` 或 `ldm destroy-vf` 命令后，状态可能立即为 `INV`。

- **静态方法:**

  a. **启动延迟重新配置。**

     ```
     primary# ldm start-reconf root-domain-name
     ```

  b. **基于物理功能创建单个虚拟功能或多个虚拟功能。**
     使用如前所示的相同命令动态创建虚拟功能。

  c. **重新引导根域。**

     - **要重新引导非 `primary` 根域，请使用以下命令：**

       ```
       primary# ldm stop-domain -r root-domain
       ```

     - **要重新引导 `primary` 根域，请使用以下命令：**

       ```
       primary# shutdown -i6 -g0 -y
       ```

例 8-18 显示有关光纤通道物理功能的信息

此示例显示了有关 `/SYS/MB/PCIE7/IOVFC.PF0` 物理功能的信息：

- 此物理功能来自 PCIe 槽 `PCIE7` 中的一个板。
- `IOVFC` 字符串指示此物理功能是一个光纤通道 SR-IOV 设备。

```
primary# ldm list-io
NAME                                    TYPE   BUS      DOMAIN   STATUS
----                                    ----   ---      ------   ------
pci_0                                   BUS    pci_0    primary  IOV
pci_1                                   BUS    pci_1    rootdom1 IOV
```

```
niu_0                                   NIU    niu_0    primary
niu_1                                   NIU    niu_1    primary
/SYS/MB/PCIE0                           PCIE   pci_0    primary  OCC
/SYS/MB/PCIE2                           PCIE   pci_0    primary  OCC
/SYS/MB/PCIE4                           PCIE   pci_0    primary  OCC
/SYS/MB/PCIE6                           PCIE   pci_0    primary  EMP
/SYS/MB/PCIE8                           PCIE   pci_0    primary  EMP
/SYS/MB/SASHBA                          PCIE   pci_0    primary  OCC
/SYS/MB/NET0                            PCIE   pci_0    primary  OCC
/SYS/MB/PCIE1                           PCIE   pci_1    rootdom1 OCC
/SYS/MB/PCIE3                           PCIE   pci_1    rootdom1 OCC
/SYS/MB/PCIE5                           PCIE   pci_1    rootdom1 OCC
/SYS/MB/PCIE7                           PCIE   pci_1    rootdom1 OCC
/SYS/MB/PCIE9                           PCIE   pci_1    rootdom1 OCC
/SYS/MB/NET2                            PCIE   pci_1    rootdom1 OCC
/SYS/MB/NET0/IOVNET.PF0                 PF     pci_0    primary
/SYS/MB/NET0/IOVNET.PF1                 PF     pci_0    primary
/SYS/MB/PCIE5/IOVNET.PF0                PF     pci_1    rootdom1
/SYS/MB/PCIE5/IOVNET.PF1                PF     pci_1    rootdom1
/SYS/MB/PCIE7/IOVFC.PF0                 PF     pci_1    rootdom1
/SYS/MB/PCIE7/IOVFC.PF1                 PF     pci_1    rootdom1
/SYS/MB/NET2/IOVNET.PF0                 PF     pci_1    rootdom1
/SYS/MB/NET2/IOVNET.PF1                 PF     pci_1    rootdom1
```

以下命令显示有关指定物理功能的更多详细信息。`maxvfs` 值表示设备支持的虚拟功能最大数量。

```
primary# ldm list-io -l /SYS/MB/PCIE7/IOVFC.PF0
NAME                                    TYPE   BUS      DOMAIN   STATUS
----                                    ----   ---      ------   ------
/SYS/MB/PCIE7/IOVNET.PF0                PF     pci_0    rootdom1
[pci@400/pci@1/pci@0/pci@6/SUNW,fcdev@0]
    maxvfs = 8
```

例 8-19 动态创建光纤通道虚拟功能而不设置可选属性

此示例会动态创建一个虚拟功能，而不设置任何可选属性。在本例中，`ldm create-vf` 命令自动分配默认带宽百分比、端口全球名称 (world-wide name, WWN) 以及节点 WWN 值。

确保已对 `pci_1` PCIe 总线启用 I/O 虚拟化。请参见如何对 PCIe 总线启用 I/O 虚拟化 [89]。

您可以使用 `ldm create-vf` 命令创建基于 `/SYS/MB/PCIE7/IOVFC.PF0` 物理功能的所有虚拟功能。

```
primary# ldm create-vf -n max /SYS/MB/PCIE7/IOVFC.PF0
Created new vf: /SYS/MB/PCIE7/IOVFC.PF0.VF0
Created new vf: /SYS/MB/PCIE7/IOVFC.PF0.VF1
Created new vf: /SYS/MB/PCIE7/IOVFC.PF0.VF2
Created new vf: /SYS/MB/PCIE7/IOVFC.PF0.VF3
Created new vf: /SYS/MB/PCIE7/IOVFC.PF0.VF4
```

```
Created new vf: /SYS/MB/PCIE7/IOVFC.PF0.VF5
Created new vf: /SYS/MB/PCIE7/IOVFC.PF0.VF6
Created new vf: /SYS/MB/PCIE7/IOVFC.PF0.VF7
```

例 8-20 动态创建光纤通道虚拟功能并设置属性

此示例以动态方法创建一个虚拟功能，同时将 bw-percent 属性值设置为 25 并指定端口和节点 WWN。

```
primary# ldm create-vf port-wwn=10:00:00:14:4F:FC:00:01 \
node-wwn=20:00:00:14:4F:FC:00:01 bw-percent=25 /SYS/MB/PCIE7/IOVFC.PF0
```

例 8-21 以静态方法创建光纤通道虚拟功能而不设置可选属性

此示例会静态创建一个虚拟功能，而不设置任何可选属性。在本例中，ldm create-vf 命令自动分配默认带宽百分比、端口全球名称 (world-wide name, WWN) 以及节点 WWN 值。

首先在 rootdom1 域上启动延迟重新配置。然后，在 pci_1 PCIe 总线上启用 I/O 虚拟化。由于 pci_1 总线已分配到 rootdom1 根域，因此，请使用 ldm set-io 命令启用 I/O 虚拟化。

```
primary# ldm start-reconf rootdom1
Initiating a delayed reconfiguration operation on the rootdom1 domain.
All configuration changes for other domains are disabled until the rootdom1
domain reboots, at which time the new configuration for the rootdom1 domain
will also take effect.

primary# ldm set-io iov=on pci_1
```

请注意，您可以使用 ldm create-vf 命令创建基于 /SYS/MB/PCIE7/IOVFC.PF0 物理功能的所有虚拟功能。

```
primary# ldm create-vf -n max /SYS/MB/PCIE7/IOVFC.PF0

------------------------------------------------------------------------------
Notice: The rootdom1 domain is in the process of a delayed reconfiguration.
Any changes made to the rootdom1 domain will only take effect after it reboots.
------------------------------------------------------------------------------

Created new vf: /SYS/MB/PCIE7/IOVFC.PF0.VF0
Created new vf: /SYS/MB/PCIE7/IOVFC.PF0.VF1
Created new vf: /SYS/MB/PCIE7/IOVFC.PF0.VF2
Created new vf: /SYS/MB/PCIE7/IOVFC.PF0.VF3
Created new vf: /SYS/MB/PCIE7/IOVFC.PF0.VF4
Created new vf: /SYS/MB/PCIE7/IOVFC.PF0.VF5
Created new vf: /SYS/MB/PCIE7/IOVFC.PF0.VF6
Created new vf: /SYS/MB/PCIE7/IOVFC.PF0.VF7
```

最后，采用下列方法之一重新引导 rootdom1 根域以使更改生效：

- rootdom1 是一个非 primary 根域

```
primary# ldm stop-domain -r rootdom1
```

- rootdom1 是 primary 域

```
primary# shutdown -i6 -g0 -y
```

销毁光纤通道 SR-IOV 虚拟功能

如果当前未将虚拟功能分配给域，则可以销毁该虚拟功能。虚拟功能只能按照与创建时相反的顺序进行销毁，因此，只能销毁已创建的最后一个虚拟功能。生成的配置由物理功能驱动程序验证。

▼ 如何销毁光纤通道 SR-IOV 虚拟功能

如果无法使用此动态方法，请改用静态方法。请参见“静态 SR-IOV” [87]。

1. **确定物理功能设备。**

```
primary# ldm list-io
```

2. **以动态或静态方法销毁单个虚拟功能或多个虚拟功能。**

   - **动态方法：**

     - **要同时销毁基于一个物理功能的所有虚拟功能，请使用以下命令：**

       ```
       primary# ldm destroy-vf -n number | max pf-name
       ```

       您可以使用路径名称或 pseudonym 名称指定虚拟功能。但是，建议做法是使用 pseudonym 名称。

       使用 `ldm destroy-vf -n max` 命令可一次销毁该物理功能的所有虚拟功能。

       如果您指定 *number* 作为 `-n` 选项的参数，则会销毁最后 *number* 个虚拟功能。请使用此方法，因为执行此操作时只进行一次物理功能设备驱动程序状态转换。

     - **要销毁指定的虚拟功能，请使用以下命令：**

       ```
       primary# ldm destroy-vf vf-name
       ```

     由于受影响的设备中以及 OS 中的延迟，受影响的物理功能以及任何其余子虚拟功能可能无法立即使用。使用 `ldm list-io` 命令确定父物理功能及其子虚拟功能在 "Status"（状态）列中是否具有 `INV` 值。如果它们具有此值，则等待直到 `ldm list-io` 输出不再在 "Status"（状态）列中显示 `INV` 值（大约 45 秒）。此时，您可以安全使用该物理功能或其任何子虚拟功能。如果此状态持续存在，则设备存在问题。

根域重新引导（包括 primary 的重新引导）后或者您使用 ldm create-vf 或 ldm destroy-vf 命令后，状态可能立即为 INV。

- 静态方法:

  a. 启动延迟重新配置。

  ```
  primary# ldm start-reconf root-domain-name
  ```

  b. 销毁单个虚拟功能或多个虚拟功能。

  - 要同时销毁基于指定物理功能的所有虚拟功能，请使用以下命令：

    ```
    primary# ldm destroy-vf -n number | max pf-name
    ```

    您可以使用路径名称或 pseudonym 名称指定虚拟功能。但是，建议做法是使用 pseudonym 名称。

  - 要销毁指定的虚拟功能，请使用以下命令：

    ```
    primary# ldm destroy-vf vf-name
    ```

  c. 重新引导根域。

  - 要重新引导非 **primary** 根域，请使用以下命令：

    ```
    primary# ldm stop-domain -r root-domain
    ```

  - 要重新引导 **primary** 根域，请使用以下命令：

    ```
    primary# shutdown -i6 -g0 -y
    ```

例 8-22 动态销毁多个光纤通道 SR-IOV 虚拟功能

此示例展示了销毁基于 /SYS/MB/PCIE5/IOVFC.PF1 物理功能的所有虚拟功能的结果。ldm list-io 输出显示此物理功能具有八个虚拟功能。ldm destroy-vf -n max 命令销毁所有虚拟功能，并且 ldm list-io 最终输出显示未保留任何虚拟功能。

```
primary# ldm list-io
...
/SYS/MB/PCIE5/IOVFC.PF1                 PF     pci_1
/SYS/MB/PCIE5/IOVFC.PF1.VF0             VF     pci_1
/SYS/MB/PCIE5/IOVFC.PF1.VF1             VF     pci_1
/SYS/MB/PCIE5/IOVFC.PF1.VF2             VF     pci_1
/SYS/MB/PCIE5/IOVFC.PF1.VF3             VF     pci_1
/SYS/MB/PCIE5/IOVFC.PF1.VF4             VF     pci_1
/SYS/MB/PCIE5/IOVFC.PF1.VF5             VF     pci_1
/SYS/MB/PCIE5/IOVFC.PF1.VF6             VF     pci_1
```

```
/SYS/MB/PCIE5/IOVFC.PF1.VF7              VF     pci_1
primary# ldm destroy-vf -n max /SYS/MB/PCIE5/IOVFC.PF1
primary# ldm list-io
...
/SYS/MB/PCIE5/IOVFC.PF1                  PF     pci_1
```

例 8-23 销毁光纤通道虚拟功能

此示例展示了如何销毁基于 /SYS/MB/PCIE7/IOVFC.PF0 物理功能的所有虚拟功能。

```
primary# ldm start-reconf rootdom1
Initiating a delayed reconfiguration operation on the rootdom1 domain.
All configuration changes for other domains are disabled until the rootdom1
domain reboots, at which time the new configuration for the rootdom1 domain
will also take effect.

primary# ldm destroy-vf -n max /SYS/MB/PCIE7/IOVFC.PF0
primary# ldm stop-domain -r rootdom1
```

修改光纤通道 SR-IOV 虚拟功能

ldm set-io 命令通过更改属性值或设置新的属性来修改虚拟功能的当前配置。

如果无法使用此动态方法，请改用静态方法。请参见“静态 SR-IOV” [87]。

您可以使用 ldm set-io 命令修改 bw-percent、port-wwn 和 node-wwn 属性。

虚拟功能已分配给域后，只能动态更改 bw-percent 属性。

▼ 如何修改光纤通道 SR-IOV 虚拟功能属性

1. **确定物理功能设备。**

   ```
   primary# ldm list-io
   ```

   请注意，物理功能的名称包括 PCIe SR-IOV 卡或板载设备的位置信息。

2. **修改虚拟功能属性。**

   ldm set-io [bw-percent=*value*] [port-wwn=*value* node-wwn=*value*] *pf-name*

   与任何时候都能动态更改的 bw-percent 属性值不同，只有当虚拟功能未分配给域时，才能动态修改 port-wwn 和 node-wwn 属性值。

例 8-24 修改光纤通道 SR-IOV 虚拟功能属性

此示例通过指定带宽百分比以及端口和节点 WWN 值修改指定的虚拟功能 /SYS/MB/PCIE7/IOVFC.PF0.VF0 的属性。

```
primary# ldm set-io port-wwn=10:00:00:14:4f:fc:f4:7c \
node-wwn=20:00:00:14:4f:fc:f4:7c bw-percent=25 /SYS/MB/PCIE7/IOVFC.PF0.VF0
```

在 I/O 域中添加和删除光纤通道 SR-IOV 虚拟功能

▼ 如何向 I/O 域添加光纤通道 SR-IOV 虚拟功能

如果无法动态删除虚拟功能，请使用静态方法。请参见“静态 SR-IOV” [87]。

1. 确定要添加到 I/O 域的虚拟功能。

   ```
   primary# ldm list-io
   ```

2. 以动态或静态方法添加虚拟功能。

   - 要以动态方法添加虚拟功能，请使用以下命令：

     ```
     primary# ldm add-io vf-name domain-name
     ```

     vf-name 是虚拟功能的 pseudonym 名称或路径名称。建议做法是使用 pseudonym 名称。*domain-name* 用于指定要将虚拟功能添加到的域的名称。

     域中虚拟功能的设备路径名称是 `list-io -l` 输出中显示的路径。

   - 要以静态方法添加虚拟功能，请使用以下命令：

     a. 停止域，然后添加虚拟功能。

        ```
        primary# ldm stop-domain domain-name
        primary# ldm add-io vf-name domain-name
        ```

        vf-name 是虚拟功能的 pseudonym 名称或路径名称。建议做法是使用 pseudonym 名称。*domain-name* 用于指定要将虚拟功能添加到的域的名称。指定的来宾必须处于非活动状态或绑定状态。

        域中虚拟功能的设备路径名称是 `list-io -l` 输出中显示的路径。

     b. 重新启动域。

        ```
        primary# ldm start-domain domain-name
        ```

例 8-25 添加光纤通道虚拟功能

此示例展示了如何以动态方法将 `/SYS/MB/PCIE7/IOVFC.PF0.VF0` 虚拟功能添加到 `ldg2` 域。

```
primary# ldm add-io /SYS/MB/PCIE7/IOVFC.PF0.VF0 ldg2
```

如果无法动态添加虚拟功能，请使用静态方法：

```
primary# ldm stop-domain ldg2
primary# ldm add-io /SYS/MB/PCIE7/IOVFC.PF0.VF0 ldg2
primary# ldm start-domain ldg2
```

▼ 如何从 I/O 域中删除光纤通道 SR-IOV 虚拟功能

如果无法使用此动态方法，请改用静态方法。请参见“静态 SR-IOV” [87]。

注意 - 从域中删除虚拟功能之前，请确保它对于引导该域不至关重要。

1. 确定要从 I/O 域删除的虚拟功能。

   ```
   primary# ldm list-io
   ```

2. 以动态或静态方法删除虚拟功能。

   - 要以动态方法删除虚拟功能，请使用以下命令：

     ```
     primary# ldm remove-io vf-name domain-name
     ```

     vf-name 是虚拟功能的 pseudonym 名称或路径名称。建议做法是使用设备 pseudonym。*domain-name* 用于指定要从中删除虚拟功能的域的名称。

   - 要以静态方法删除虚拟功能，请使用以下命令：

     a. 停止 I/O 域。

        ```
        primary# ldm stop-domain domain-name
        ```

     b. 删除虚拟功能。

        ```
        primary# ldm remove-io vf-name domain-name
        ```

        vf-name 是虚拟功能的 pseudonym 名称或路径名称。建议做法是使用设备 pseudonym。*domain-name* 用于指定要从中删除虚拟功能的域的名称。指定的来宾必须处于非活动状态或绑定状态。

     c. 启动 I/O 域。

        ```
        primary# ldm start-domain domain-name
        ```

例 8-26 以动态方法删除光纤通道虚拟功能

此示例展示了如何以动态方法从 `ldg2` 域中删除 `/SYS/MB/PCIE7/IOVFC.PF0.VF0` 虚拟功能。

```
primary# ldm remove-io /SYS/MB/PCIE7/IOVFC.PF0.VF0 ldg2
```

如果命令成功，则会从 ldg2 域删除该虚拟功能。重新启动 ldg2 后，指定的虚拟功能不再显示在该域中。

如果无法动态删除虚拟功能，请使用静态方法：

```
primary# ldm stop-domain ldg2
primary# ldm remove-io /SYS/MB/PCIE7/IOVFC.PF0.VF0 ldg2
primary# ldm start-domain ldg2
```

高级 SR-IOV 主题：光纤通道 SR-IOV

本节介绍与使用光纤通道 SR-IOV 虚拟功能相关的一些高级主题。

在来宾域中访问光纤通道虚拟功能

ldg2 控制台日志显示了对分配的光纤通道虚拟功能设备执行的操作。使用 fcadm 命令可查看和访问光纤通道虚拟功能设备。

```
ldg2# fcadm hba-port
HBA Port WWN: 100000144ffb8a99
       Port Mode: Initiator
       Port ID: 13d02
       OS Device Name: /dev/cfg/c3
       Manufacturer: Emulex
       Model: 7101684
       Firmware Version: 7101684 1.1.60.1
       FCode/BIOS Version: Boot:1.1.60.1 Fcode:4.03a4
       Serial Number: 4925382+133400002R
       Driver Name: emlxs
       Driver Version: 2.90.15.0 (2014.01.22.14.50)
       Type: N-port
       State: online
       Supported Speeds: 4Gb 8Gb 16Gb
       Current Speed: 16Gb
       Node WWN: 200000144ffb8a99
       NPIV Not Supported
```

使用 format 命令可显示可见 LUN。

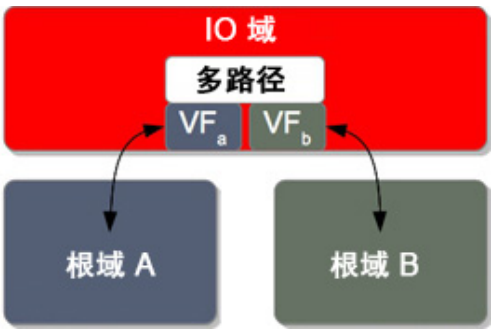
```
ldg2# format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
   0. c2d0 <Unknown-Unknown-0001-25.00GB>
      /virtual-devices@100/channel-devices@200/disk@0
```

```
   1. c3t21000024FF4C4BF8d0 <SUN-COMSTAR-1.0-10.00GB>
      /pci@340/pci@1/pci@0/pci@6/SUNW,emlxs@0,2/fp@0,0/ssd@w21000024ff4c4bf8,0
Specify disk (enter its number): ^D
ldg2#
```

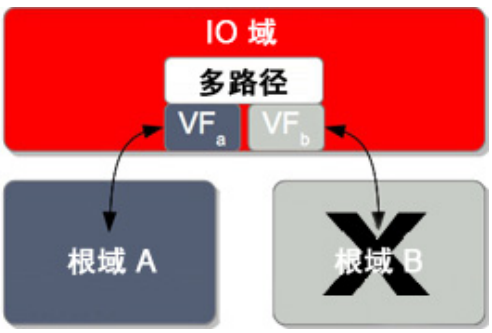
I/O 域弹性

通过 I/O 域弹性，即使 I/O 域一个关联的根域中断，I/O 域也能继续运行，从而改进了 I/O 域的可用性和性能。根域中断时，使用其服务的 I/O 域继续运行，其实现方法是使它的受影响设备故障转移到备用 I/O 路径。根域恢复服务时，弹性 I/O 域中的受影响设备也恢复服务并且故障转移功能将恢复。

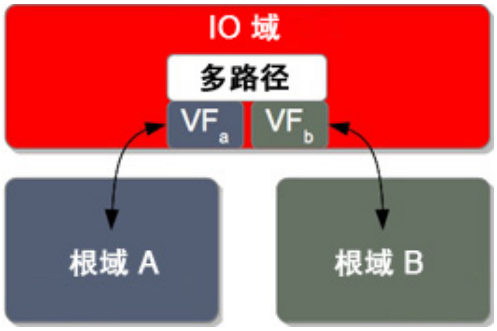
下图显示并说明了配置的一个根域发生故障时会进行哪些操作以及根域恢复服务时会进行哪些操作。



每个根域为 I/O 域提供一个虚拟功能。I/O 域对网络设备使用虚拟设备多路径功能（例如 IPMP），对光纤通道设备使用 MPxIO。



根域 B 由于紧急情况或重新引导而中断时，将在 I/O 域中暂停虚拟功能 B，然后多路径功能通过根域 A 路由所有 I/O。



根域 B 恢复服务时，虚拟功能 B 在 I/O 域中恢复操作。多路径组恢复为完全冗余。

在此配置中，虚拟功能可以是 vnet 设备，这表示 I/O 域可以配置有虚拟功能或 vnet 设备的任意组合。请注意，光纤通道设备只能是虚拟功能。

可以创建包含弹性和非弹性 I/O 域的配置。有关示例，请参见“示例 – 使用弹性和非弹性配置” [140]。

弹性 I/O 域要求

弹性 I/O 域必须满足以下要求：

- 至少运行 Oracle Solaris 11.2.8.0.0 (SRU 8) OS 并且其 primary 域至少运行 Oracle VM Server for SPARC 3.2 软件。
- 使用多路径功能为虚拟功能和虚拟设备创建故障转移配置。对于网络多路径，IPMP 组可以具有虚拟功能和 vnet 设备。
- 将 master 属性值设置为其 failure-policy 属性设置为 ignore 的根域的名称。任何其他故障策略设置（例如 stop、reset 或 panic）会取代 I/O 弹性并且 I/O 域将中断。
- 仅使用支持 I/O 域弹性的 SR-IOV 虚拟功能。请参见 https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&doctype=REFERENCE&id=1325454.1。

注 - 如果 I/O 域仅使用 I/O 弹性设备，但是未配置多路径，将影响应用程序，直到根域恢复服务并且受影响设备被恢复。

I/O 域弹性限制

- 如果将 SR-IOV 卡热插入根域，然后将其虚拟功能分配到 I/O 域，则根域发生故障时 I/O 域可能无法提供弹性。因此，仅应在根域关闭时添加 SR-IOV 卡。然后，在根域引导后分配虚拟功能。
- 如果您有弹性 I/O 域，然后以下列方式之一分配设备，则该 I/O 域不再具有弹性：

- 从不支持 I/O 弹性的卡添加虚拟功能
- 使用直接 I/O 功能直接分配设备

在这种情况下，将 failure-policy 从 `ignore` 设置为 `reset` 或 `stop`。

配置弹性 I/O 域

▼ 如何配置弹性 I/O 域

开始之前 仅使用支持 I/O 域弹性功能的 PCIe 卡。请参见 `https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&doctype=REFERENCE&id=1325454.1`。

确保 I/O 域、根域、服务域和 `primary` 域至少运行 Oracle Solaris 11.2.8.0.0 (SRU 8) OS 和 Logical Domains Manager 3.2 软件。

1. **在根域上，将 `failure-policy` 属性设置为 `ignore`。**

   ```
   primary# ldm set-domain failure-policy=ignore root-domain-name
   ```

   注 - 如果向 I/O 域添加不支持弹性的任何设备，该域将不再具有弹性。所以，将 `failure-policy` 属性值重置为 `stop`、`reset` 或 `panic`。

   有关域依赖关系的信息，请参见“配置域依赖关系” [306]。

2. **在 I/O 域上，将 `master` 属性设置为根域的名称。**

   ```
   primary# ldm set-domain master=root-domain-name  I/O-domain-name
   ```

3. **跨路径配置多路径功能。**

   - **以太网。使用 IPMP 跨路径配置多路径功能。**
     有关使用 IPMP 配置多路径功能的信息，请参见*Administering TCP/IP Networks, IPMI, and IP Tunnels in Oracle Solaris 11.2*。
   - **光纤通道。使用 MPxIO 跨路径配置多路径功能。**
     有关使用 MPxIO 配置多路径功能的信息，请参见*Managing SAN Devices and Multipathing Oracle Solaris 11.2*。

例 8-27 使用 IPMP 配置具有以太网 SR-IOV 功能的多路径功能

此示例显示如何使用 IPMP 为弹性 I/O 域配置网络虚拟功能设备。有关更多信息，请参见*Administering TCP/IP Networks, IPMI, and IP Tunnels in Oracle Solaris 11.2*。

1. 确定分配给不同根域的两个以太网 SR-IOV 物理功能。

在此示例中，root-1 和 root-2 根域具有以太网 SR-IOV 物理功能。

```
primary# ldm list-io | grep root-1 | grep PF
/SYS/PCI-EM8/IOVNET.PF0                   PF     pci_1    root-1
primary# ldm list-io | grep root-2 | grep PF
/SYS/RIO/NET2/IOVNET.PF0                  PF     pci_2    root-2
```

2. 在每个指定物理功能上创建两个以太网虚拟功能。

```
primary# ldm create-vf /SYS/MB/NET0/IOVNET.PF0
Created new vf: /SYS/PCI-EM8/IOVNET.PF0.VF0
primary# ldm create-vf /SYS/RIO/NET2/IOVNET.PF0
Created new vf: /SYS/RIO/NET2/IOVNET.PF0.VF0
```

3. 向 io-1 I/O 域分配以太网虚拟功能。

```
primary# ldm add-io /SYS/PCI-EM8/IOVNET.PF0.VF0 io-1
primary# ldm add-io /SYS/RIO/NET2/IOVNET.PF0.VF0 io-1
```

4. 将以太网虚拟功能配置到 I/O 域上的 IPMP 组中。

   a. 确定 I/O 域上新添加的网络设备 net1 和 net2。

```
i0-1# dladm show-phys
LINK        MEDIA        STATE    SPPED    DUPLEX    DEVICE
net0        Ethernet     up       0        unknown   vnet0
net1        Ethernet     up       1000     full      igbvf0
net2        Ethernet     up       1000     full      igbvf1
```

   b. 为新添加的网络设备创建 IP 接口。

```
io-1# ipadm create-ip net1
io-1# ipadm create-ip net2
```

   c. 为两个网络接口创建 ipmp0 IPMP 组。

```
io-1# ipadm create-ipmp -i net1 -i net2 ipmp0
```

   d. 向 IPMP 组分配 IP 地址。

   此示例配置 DHCP 选项。

```
io-1# ipadm create-addr -T dhcp ipmp0/v4
```

   e. 检查 IPMP 组接口的状态。

```
io-1# ipmpstat -g
```

例 8-28 使用 MPxIO 配置具有光纤通道 SR-IOV 功能的多路径功能

此示例显示如何使用 MPxIO 为弹性 I/O 域配置光纤通道虚拟功能设备。有关更多信息，请参见*Managing SAN Devices and Multipathing Oracle Solaris 11.2*。

1. 确定分配给不同根域的两个光纤通道 SR-IOV 物理功能。

在此示例中，root-1 和 root-2 根域具有光纤通道 SR-IOV 物理功能。

```
primary# ldm list-io | grep root-1 | grep PF
/SYS/PCI-EM4/IOVFC.PF0                    PF     pci_1    root-1
primary# ldm list-io | grep root-2 | grep PF
/SYS/PCI-EM15/IOVFC.PF0                   PF     pci_2    root-2
```

2. 在每个指定物理功能上创建两个虚拟功能。
   有关更多信息，请参见如何创建光纤通道 SR-IOV 虚拟功能 [125]。

```
primary# ldm create-vf port-wwn=10:00:00:14:4f:fc:60:00 \
node-wwn=20:00:00:14:4f:fc:60:00 /SYS/PCI-EM4/IOVFC.PF0
Created new vf: /SYS/PCI-EM4/IOVFC.PF0.VF0
primary# ldm create-vf port-wwn=10:00:00:14:4f:fc:70:00 \
node-wwn=20:00:00:14:4f:fc:70:00 /SYS/PCI-EM15/IOVFC.PF0
Created new vf: /SYS/PCI-EM15/IOVFC.PF0.VF0
```

3. 向 io-1 I/O 域添加新创建的虚拟功能。

```
primary# ldm add-io /SYS/PCI-EM4/IOVFC.PF0.VF0 io-1
primary# ldm add-io /SYS/PCI-EM15/IOVFC.PF0.VF0 io-1
```

4. 使用 prtconf -v 命令确定是否在 I/O 域上启用了 MPxIO。
   如果 fp 设备的输出包括以下设备属性设置，则启用了 MPxIO：

```
mpxio-disable="no"
```

   如果 mpxio-disable 属性设置为 yes，则在 /etc/driver/drv/fp.conf 文件中将该属性设置为 no，然后重新引导 I/O 域。

   如果 mpxio-disable 设备属性未显示在 prtconf -v 输出中，则将 mpxio-disable="no" 条目添加到 /etc/driver/drv/fp.conf 文件中，然后重新引导 I/O 域。

5. 检查 MPxIO 组的状态。

```
io-1# mpathadm show LU

Logical Unit:  /dev/rdsk/c0t600A0B80002A384600003D6B544EECD0d0s2
     mpath-support:  libmpscsi_vhci.so
     Vendor:  SUN
     Product:  CSM200_R
     Revision:  0660
     Name Type:  unknown type
     Name:  600a0b80002a384600003d6b544eecd0
     Asymmetric:  yes
     Current Load Balance:  round-robin
     Logical Unit Group ID:  NA
     Auto Failback:  on
     Auto Probing:  NA
```

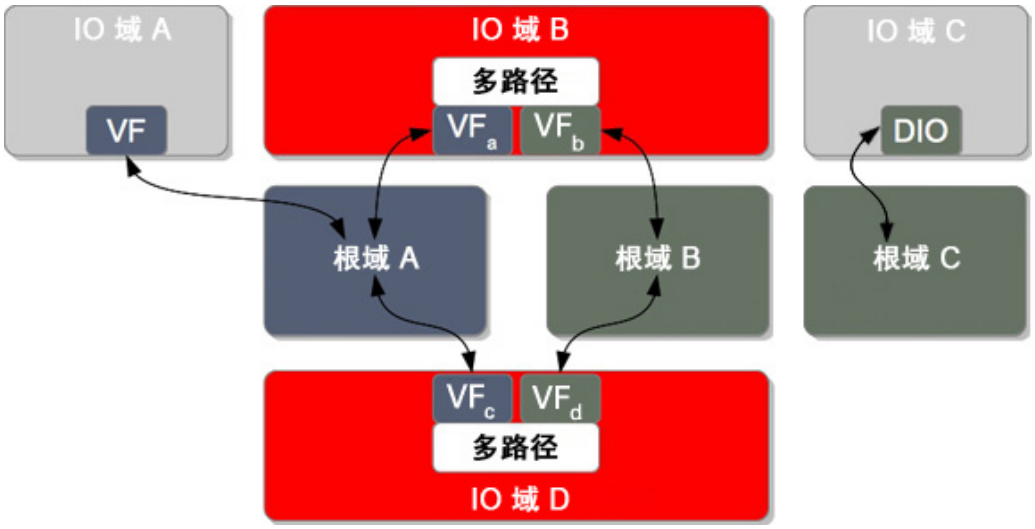
```
Paths:
        Initiator Port Name:  100000144ffc6000
        Target Port Name:  201700a0b82a3846
        Override Path:  NA
        Path State:  OK
        Disabled:  no

        Initiator Port Name:  100000144ffc7000
        Target Port Name:  201700a0b82a3846
        Override Path:  NA
        Path State:  OK
        Disabled:  no

Target Port Groups:
        ID:  1
        Explicit Failover:  yes
        Access State:  active
        Target Ports:
                Name:  201700a0b82a3846
                Relative ID:  0
```

示例 – 使用弹性和非弹性配置

可以使用同时具有弹性和非弹性域的配置。

下图显示 I/O 域 A 和 I/O 域 C 不具有弹性，因为两个都不使用多路径功能。I/O 域 A 具有虚拟功能，I/O 域 C 具有直接 I/O 设备。

图 8-2 具有弹性和非弹性 I/O 域的配置



I/O 域 B 和 I/O 域 D 具有弹性。I/O 域 A、B 和 D 依赖于根域 A。I/O 域 B 和 D 依赖于根域 B。I/O 域 C 依赖于根域 C。

如果根域 A 中断，I/O 域 A 也中断。但是，I/O 域 B 和 D 故障转移到备用路径并继续运行应用程序。如果根域 C 中断，I/O 域 C 以根域 C 的 `failure-policy` 属性值指定的方式发生故障。

重新引导配置了非弹性 I/O 域的根域

注 - 如果 I/O 域具有弹性，即使为其提供服务的根域中断，该 I/O 域也可以继续操作。有关配置弹性 I/O 域的信息，请参见“I/O 域弹性” [135]。

与 I/O 域中的 PCIe 插槽一样，“重新引导配置了 PCIe 端点的根域” [71]中所述的问题也适用于分配给 I/O 域的虚拟功能。

注 - 当关联的根域没有运行时，I/O 域无法启动。

♦ ♦ ♦ 第 9 章

使用非 primary 根域

本章包括以下非 primary 根域主题：

非 primary 根域概述

根域分配有 PCIe 根联合体。此域拥有 PCIe 结构并提供所有与结构相关的服务，如结构错误处理。根域也是 I/O 域，因为它拥有对物理 I/O 设备的直接访问权限。primary 域是默认根域。

可以在分配到任何根域的 PCIe 总线上执行直接 I/O 和 SR-IOV 操作。现在，用户可以对所有根域（包括非 primary 根域）执行以下操作：

- 显示 PCIe 插槽的状态
- 显示已有的 SR-IOV 物理功能
- 将 PCIe 插槽分配给 I/O 域或根域
- 将 PCIe 插槽从 I/O 域或根域删除
- 从物理功能创建虚拟功能
- 销毁虚拟功能
- 将虚拟功能分配到其他域
- 从其他域删除虚拟功能

Logical Domains Manager 可从非 primary 根域中运行的 Logical Domains 代理获取 PCIe 端点设备和 SR-IOV 物理功能设备。在首次发现根域后，如果该根域关闭，则会缓存此信息，直到该根域引导为止。

只有当根域处于活动状态时，才能执行直接 I/O 和 SR-IOV 操作。此时，Logical Domains Manager 将在已有的实际设备上运行。发生以下操作时，可能会刷新缓存的数据：

- Logical Domains 代理在指定的根域中重新启动
- 在指定的根域中执行了硬件更改，例如，热插拔操作

使用 `ldm list-io` 命令可查看 PCIe 端点设备状态。输出内容还会显示属于每个非 `primary` 根域的根联合体中的子设备和物理功能设备。

您可以对任何根域应用以下命令：

- `ldm add-io`
- `ldm remove-io`
- `ldm set-io`
- `ldm create-vf`
- `ldm destroy-vf`
- `ldm start-reconf`
- `ldm cancel-reconf`

延迟重新配置支持的范围已进行了扩展，可支持非 `primary` 根域。但是，它只能用于运行 `ldm add-io`、`ldm remove-io`、`ldm set-io`、`ldm create-vf` 和 `ldm destroy-vf` 命令。延迟重新配置可用于无法使用以下动态操作完成的任何操作：

- 执行直接 I/O 操作
- 在不符合动态 SR-IOV 配置要求的物理功能中创建和销毁虚拟功能。

注意 - 提前进行计划，以便最大程度地减少根域的重新引导次数，从而最大限度地缩短停机时间。

非 `primary` 根域要求

直接 I/O 和 SR-IOV 功能可以使用非 `primary` 根域。SPARC T4、SPARC T5、SPARC M5 和 SPARC M6 平台以及 Fujitsu M10 平台支持此功能。

- **硬件要求。**

  除了 https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&doctype=REFERENCE&id=1325454.1 中所述的适用于直接 I/O 和 SR-IOV 的 PCIe 卡之外，还可以使用其他 PCIe 卡，但不是用于 DIO 和 SR-IOV。要确定可以在您的平台上使用哪些卡，请参见您的平台硬件文档。
- **固件要求。**

  SPARC T4 平台必须至少运行 8.4.0.a 版的系统固件。

  SPARC T5、SPARC M5 和 SPARC M6 平台必须至少运行 9.1.0.*x* 版的系统固件。

Fujitsu M10 服务器必须至少运行 XCP2210 版本的系统固件。

- **软件要求。**

  所有域都必须至少运行 Oracle Solaris 11.1.10.5.0 OS，或者运行 Oracle Solaris 10 1/13 OS 加上 http://www.oracle.com/pls/topic/lookup?ctx=E56447&id=LDSIGreqdrecommendedsolarisos 或 Oracle Solaris 11.1.10.5.0 OS 中的必需修补程序。

非 primary 根域限制

非 primary 根域具有以下使用限制：

- 当关联的根域没有运行时，I/O 域无法启动。
- 延迟重新配置支持的范围已扩展，可支持非 primary 根域。在重新引导根域或取消延迟重新配置之前，只能运行以下命令：
  - `ldm add-io`
  - `ldm remove-io`
  - `ldm set-io`
  - `ldm create-vf`
  - `ldm destroy-vf`
- 要执行以下操作，根域必须处于活动状态并进行引导：
  - 创建和销毁 SR-IOV 虚拟功能
  - 添加和删除 PCIe 插槽
  - 添加和删除 SR-IOV 虚拟功能
- 在对 PCIe 插槽执行 `ldm add-io` 和 `ldm remove-io` 直接 I/O 操作时，必须在根域上启动延迟重新配置。
- 如果您的配置不符合动态 I/O 虚拟化要求，则必须对以下 SR-IOV 虚拟功能操作使用延迟重新配置：
  - `ldm create-vf`
  - `ldm destroy-vf`
  - `ldm add-io`
  - `ldm remove-io`
  - `ldm set-io`
- 重新引导某个根域将影响在属于该根域的 PCIe 总线中具有设备的任何 I/O 域。请参见“重新引导配置了 PCIe 端点的根域” [71]。
- 您不能将 SR-IOV 虚拟功能或 PCIe 插槽从一个根域分配到另一个根域。此限制可以防止发生循环依赖关系。

非 primary 根域示例

以下示例说明如何为 PCIe 总线启用 I/O 虚拟化、在非 primary 根域上管理直接 I/O 设备以及在非 primary 根域上管理 SR-IOV 虚拟功能。

为 PCIe 总线启用 I/O 虚拟化

以下示例说明如何使用 ldm add-io 和 ldm set-io 命令启用 I/O 虚拟化。

以下 SPARC T4-2 I/O 配置显示总线 pci_1 已从 primary 域中删除。

```
primary# ldm list-io
NAME                                      TYPE   BUS      DOMAIN   STATUS
----                                      ----   ---      ------   ------
pci_0                                     BUS    pci_0    primary  IOV
pci_1                                     BUS    pci_1
niu_0                                     NIU    niu_0    primary
niu_1                                     NIU    niu_1    primary
/SYS/MB/PCIE0                             PCIE   pci_0    primary  OCC
/SYS/MB/PCIE2                             PCIE   pci_0    primary  OCC
/SYS/MB/PCIE4                             PCIE   pci_0    primary  OCC
/SYS/MB/PCIE6                             PCIE   pci_0    primary  EMP
/SYS/MB/PCIE8                             PCIE   pci_0    primary  EMP
/SYS/MB/SASHBA                            PCIE   pci_0    primary  OCC
/SYS/MB/NET0                              PCIE   pci_0    primary  OCC
/SYS/MB/PCIE1                             PCIE   pci_1             UNK
/SYS/MB/PCIE3                             PCIE   pci_1             UNK
/SYS/MB/PCIE5                             PCIE   pci_1             UNK
/SYS/MB/PCIE7                             PCIE   pci_1             UNK
/SYS/MB/PCIE9                             PCIE   pci_1             UNK
/SYS/MB/NET2                              PCIE   pci_1             UNK
/SYS/MB/NET0/IOVNET.PF0                   PF     pci_0    primary
/SYS/MB/NET0/IOVNET.PF1                   PF     pci_0    primary
```

以下列表显示来宾域处于绑定状态：

```
primary# ldm list
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  NORM  UPTIME
primary          active     -n-cv-  UART    8     8G       0.6%  0.6%  8m
rootdom1         bound      ------  5000    8     4G
ldg2             bound      ------  5001    8     4G
ldg3             bound      ------  5002    8     4G
```

以下 ldm add-io 命令将 pci_1 总线添加到 rootdom1 域，并为该总线启用 I/O 虚拟化。ldm start 命令启动 rootdom1 域。

```
primary# ldm add-io iov=on pci_1 rootdom1
primary# ldm start rootdom1
LDom rootdom1 started
```

如果已将指定的 PCIe 总线分配给某个根域，请使用 ldm set-io 命令启用 I/O 虚拟化。

```
primary# ldm start-reconf rootdom1
primary# ldm set-io iov=on pci_1
primary# ldm stop-domain -r rootdom1
```

此根域必须正在运行其 OS，您才能配置 I/O 设备。如果未将来宾域设置为自动引导，请连接到 rootdom1 来宾域的控制台，然后引导 rootdom1 根域的 OS。

```
primary# telnet localhost 5000
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Connecting to console "rootdom1" in group "rootdom1" ....
Press ~? for control options ..
ok> boot
...
primary#
```

以下命令表明 pci_1 PCIe 总线及其子项现在属于 rootdom1 根域。

```
primary# ldm list-io
NAME                                    TYPE   BUS      DOMAIN   STATUS
----                                    ----   ---      ------   ------
pci_0                                   BUS    pci_0    primary  IOV
pci_1                                   BUS    pci_1    rootdom1     IOV
niu_0                                   NIU    niu_0    primary
niu_1                                   NIU    niu_1    primary
/SYS/MB/PCIE0                           PCIE   pci_0    primary  OCC
/SYS/MB/PCIE2                           PCIE   pci_0    primary  OCC
/SYS/MB/PCIE4                           PCIE   pci_0    primary  OCC
/SYS/MB/PCIE6                           PCIE   pci_0    primary  EMP
/SYS/MB/PCIE8                           PCIE   pci_0    primary  EMP
/SYS/MB/SASHBA                          PCIE   pci_0    primary  OCC
/SYS/MB/NET0                            PCIE   pci_0    primary  OCC
/SYS/MB/PCIE1                           PCIE   pci_1    rootdom1     OCC
/SYS/MB/PCIE3                           PCIE   pci_1    rootdom1     OCC
/SYS/MB/PCIE5                           PCIE   pci_1    rootdom1     OCC
/SYS/MB/PCIE7                           PCIE   pci_1    rootdom1     OCC
/SYS/MB/PCIE9                           PCIE   pci_1    rootdom1     EMP
/SYS/MB/NET2                            PCIE   pci_1    rootdom1     OCC
/SYS/MB/NET0/IOVNET.PF0                 PF     pci_0    primary
/SYS/MB/NET0/IOVNET.PF1                 PF     pci_0    primary
/SYS/MB/PCIE5/IOVNET.PF0                PF     pci_1    rootdom1
/SYS/MB/PCIE5/IOVNET.PF1                PF     pci_1    rootdom1
/SYS/MB/NET2/IOVNET.PF0                 PF     pci_1    rootdom1
/SYS/MB/NET2/IOVNET.PF1                 PF     pci_1    rootdom1
```

在非 primary 根域上管理直接 I/O 设备

以下示例说明如何在非 primary 根域上管理直接 I/O 设备。

以下命令将生成一个错误，因为它试图从根域中删除仍处于活动状态的插槽：

```
primary# ldm remove-io /SYS/MB/PCIE7 ldg1
Dynamic I/O operations on PCIe slots are not supported.
Use start-reconf command to trigger delayed reconfiguration and make I/O
changes statically.
```

以下命令显示删除插槽的正确方法，即，首先在根域上启动延迟重新配置。

```
primary# ldm start-reconf ldg1
Initiating a delayed reconfiguration operation on the ldg1 domain.
All configuration changes for other domains are disabled until the ldg1
domain reboots, at which time the new configuration for the ldg1 domain
will also take effect.
primary# ldm remove-io /SYS/MB/PCIE7 ldg1
------------------------------------------------------------------------------
Notice: The ldg1 domain is in the process of a delayed reconfiguration.
Any changes made to the ldg1 domain will only take effect after it reboots.
------------------------------------------------------------------------------
primary# ldm stop-domain -r ldg1
```

以下 ldm list-io 命令可验证 /SYS/MB/PCIE7 插槽是否不再位于根域中。

```
primary# ldm list-io
NAME                                    TYPE   BUS      DOMAIN   STATUS
----                                    ----   ---      ------   ------
pci_0                                   BUS    pci_0    primary  IOV
pci_1                                   BUS    pci_1    ldg1     IOV
niu_0                                   NIU    niu_0    primary
niu_1                                   NIU    niu_1    primary
/SYS/MB/PCIE0                           PCIE   pci_0    primary  OCC
/SYS/MB/PCIE2                           PCIE   pci_0    primary  OCC
/SYS/MB/PCIE4                           PCIE   pci_0    primary  OCC
/SYS/MB/PCIE6                           PCIE   pci_0    primary  EMP
/SYS/MB/PCIE8                           PCIE   pci_0    primary  EMP
/SYS/MB/SASHBA                          PCIE   pci_0    primary  OCC
/SYS/MB/NET0                            PCIE   pci_0    primary  OCC
/SYS/MB/PCIE1                           PCIE   pci_1    ldg1     OCC
/SYS/MB/PCIE3                           PCIE   pci_1    ldg1     OCC
/SYS/MB/PCIE5                           PCIE   pci_1    ldg1     OCC
/SYS/MB/PCIE7                           PCIE   pci_1             OCC
/SYS/MB/PCIE9                           PCIE   pci_1    ldg1     EMP
/SYS/MB/NET2                            PCIE   pci_1    ldg1     OCC
/SYS/MB/NET0/IOVNET.PF0                 PF     pci_0    primary
/SYS/MB/NET0/IOVNET.PF1                 PF     pci_0    primary
/SYS/MB/PCIE5/IOVNET.PF0                PF     pci_1    ldg1
/SYS/MB/PCIE5/IOVNET.PF1                PF     pci_1    ldg1
/SYS/MB/NET2/IOVNET.PF0                 PF     pci_1    ldg1
/SYS/MB/NET2/IOVNET.PF1                 PF     pci_1    ldg1
```

以下命令可将 /SYS/MB/PCIE7 插槽分配给 ldg2 域。ldm start 命令可启动 ldg2 域。

```
primary# ldm add-io /SYS/MB/PCIE7 ldg2
primary# ldm start ldg2
```

```
LDom ldg2 started
```

在非 primary 根域上管理 SR-IOV 虚拟功能

以下命令可利用属于非 primary 根域的两个物理功能创建两个虚拟功能。

```
primary# ldm create-vf /SYS/MB/PCIE5/IOVNET.PF0
Created new vf: /SYS/MB/PCIE5/IOVNET.PF0.VF0
primary# ldm create-vf /SYS/MB/PCIE5/IOVNET.PF0
Created new vf: /SYS/MB/PCIE5/IOVNET.PF0.VF1
primary# ldm create-vf /SYS/MB/NET2/IOVNET.PF1
Created new vf: /SYS/MB/NET2/IOVNET.PF1.VF0
primary# ldm create-vf /SYS/MB/NET2/IOVNET.PF1
Created new vf: /SYS/MB/NET2/IOVNET.PF1.VF1
```

您还可以在以下两个命令中使用 -n 选项来创建两个虚拟功能：

```
primary# ldm create-vf -n 2 /SYS/MB/PCIE5/IOVNET.PF0
Created new vf: /SYS/MB/PCIE5/IOVNET.PF0.VF0
Created new vf: /SYS/MB/PCIE5/IOVNET.PF0.VF1
primary# ldm create-vf -n 2 /SYS/MB/NET2/IOVNET.PF1
Created new vf: /SYS/MB/NET2/IOVNET.PF1.VF0
Created new vf: /SYS/MB/NET2/IOVNET.PF1.VF1
```

如果无法在给定物理功能上动态创建虚拟功能，请启动延迟重新配置以静态创建虚拟功能。

```
primary# ldm start-reconf ldg1
primary# ldm create-vf /SYS/MB/PCIE5/IOVNET.PF0
Created new vf: /SYS/MB/PCIE5/IOVNET.PF0.VF0
primary# ldm create-vf /SYS/MB/PCIE5/IOVNET.PF0
Created new vf: /SYS/MB/PCIE5/IOVNET.PF0.VF1
primary# ldm create-vf /SYS/MB/NET2/IOVNET.PF1
Created new vf: /SYS/MB/NET2/IOVNET.PF1.VF0
primary# ldm create-vf /SYS/MB/NET2/IOVNET.PF1
Created new vf: /SYS/MB/NET2/IOVNET.PF1.VF1
```

然后重新引导根域 ldg1，以使更改生效。

```
primary# ldm stop-domain -r ldg1
```

以下命令可显示新的虚拟功能。

```
primary# ldm list-io
NAME                                    TYPE   BUS      DOMAIN   STATUS
----                                    ----   ---      ------   ------
pci_0                                   BUS    pci_0    primary  IOV
pci_1                                   BUS    pci_1    ldg1     IOV
niu_0                                   NIU    niu_0    primary
niu_1                                   NIU    niu_1    primary
```

```
/SYS/MB/PCIE0                          PCIE   pci_0    primary  OCC
/SYS/MB/PCIE2                          PCIE   pci_0    primary  OCC
/SYS/MB/PCIE4                          PCIE   pci_0    primary  OCC
/SYS/MB/PCIE6                          PCIE   pci_0    primary  EMP
/SYS/MB/PCIE8                          PCIE   pci_0    primary  EMP
/SYS/MB/SASHBA                         PCIE   pci_0    primary  OCC
/SYS/MB/NET0                           PCIE   pci_0    primary  OCC
/SYS/MB/PCIE1                          PCIE   pci_1    ldg1     OCC
/SYS/MB/PCIE3                          PCIE   pci_1    ldg1     OCC
/SYS/MB/PCIE5                          PCIE   pci_1    ldg1     OCC
/SYS/MB/PCIE7                          PCIE   pci_1    ldg2     OCC
/SYS/MB/PCIE9                          PCIE   pci_1    ldg1     EMP
/SYS/MB/NET2                           PCIE   pci_1    ldg1     OCC
/SYS/MB/NET0/IOVNET.PF0                PF     pci_0    primary
/SYS/MB/NET0/IOVNET.PF1                PF     pci_0    primary
/SYS/MB/PCIE5/IOVNET.PF0               PF     pci_1    ldg1
/SYS/MB/PCIE5/IOVNET.PF1               PF     pci_1    ldg1
/SYS/MB/NET2/IOVNET.PF0                PF     pci_1    ldg1
/SYS/MB/NET2/IOVNET.PF1                PF     pci_1    ldg1
/SYS/MB/PCIE5/IOVNET.PF0.VF0           VF     pci_1
/SYS/MB/PCIE5/IOVNET.PF0.VF1           VF     pci_1
/SYS/MB/NET2/IOVNET.PF1.VF0            VF     pci_1
/SYS/MB/NET2/IOVNET.PF1.VF1            VF     pci_1
```

以下命令可将 /SYS/MB/PCIE5/IOVNET.PF0.VF1 虚拟功能动态添加到非 primary 根域 ldg1 中：

```
primary# ldm add-io /SYS/MB/PCIE5/IOVNET.PF0.VF1 ldg1
```

以下命令会将 /SYS/MB/NET2/IOVNET.PF1.VF0 虚拟功能动态添加到 ldg2 域：

```
primary# ldm add-io /SYS/MB/NET2/IOVNET.PF1.VF0 ldg2
```

以下命令会将 /SYS/MB/NET2/IOVNET.PF1.VF1 虚拟功能添加到已绑定的 ldg3 域：

```
primary# ldm add-io /SYS/MB/NET2/IOVNET.PF1.VF1 ldg3
primary# ldm start ldg3
LDom ldg3 started
```

连接到 ldg3 域的控制台，然后引导其 OS。

以下输出显示所有分配都是按预期进行的。其中有一个虚拟功能未分配，因此，可以将其动态分配到 ldg1、ldg2 或 ldg3 域。

```
# ldm list-io
NAME                                   TYPE   BUS      DOMAIN   STATUS
----                                   ----   ---      ------   ------
pci_0                                  BUS    pci_0    primary  IOV
pci_1                                  BUS    pci_1    ldg1     IOV
niu_0                                  NIU    niu_0    primary
niu_1                                  NIU    niu_1    primary
/SYS/MB/PCIE0                          PCIE   pci_0    primary  OCC
/SYS/MB/PCIE2                          PCIE   pci_0    primary  OCC
/SYS/MB/PCIE4                          PCIE   pci_0    primary  OCC
```

```
/SYS/MB/PCIE6                          PCIE   pci_0    primary  EMP
/SYS/MB/PCIE8                          PCIE   pci_0    primary  EMP
/SYS/MB/SASHBA                         PCIE   pci_0    primary  OCC
/SYS/MB/NET0                           PCIE   pci_0    primary  OCC
/SYS/MB/PCIE1                          PCIE   pci_1    ldg1     OCC
/SYS/MB/PCIE3                          PCIE   pci_1    ldg1     OCC
/SYS/MB/PCIE5                          PCIE   pci_1    ldg1     OCC
/SYS/MB/PCIE7                          PCIE   pci_1    ldg2     OCC
/SYS/MB/PCIE9                          PCIE   pci_1    ldg1     EMP
/SYS/MB/NET2                           PCIE   pci_1    ldg1     OCC
/SYS/MB/NET0/IOVNET.PF0                PF     pci_0    primary
/SYS/MB/NET0/IOVNET.PF1                PF     pci_0    primary
/SYS/MB/PCIE5/IOVNET.PF0               PF     pci_1    ldg1
/SYS/MB/PCIE5/IOVNET.PF1               PF     pci_1    ldg1
/SYS/MB/NET2/IOVNET.PF0                PF     pci_1    ldg1
/SYS/MB/NET2/IOVNET.PF1                PF     pci_1    ldg1
/SYS/MB/PCIE5/IOVNET.PF0.VF0           VF     pci_1
/SYS/MB/PCIE5/IOVNET.PF0.VF1           VF     pci_1    ldg1
/SYS/MB/NET2/IOVNET.PF1.VF0            VF     pci_1    ldg2
/SYS/MB/NET2/IOVNET.PF1.VF1            VF     pci_1    ldg3
```

♦ ♦ ♦ 第 10 章

10 使用虚拟磁盘

本章介绍如何将虚拟磁盘与 Oracle VM Server for SPARC 软件结合使用。
本章包括以下主题：

虚拟磁盘简介

虚拟磁盘包含两个组件：显示在来宾域中的虚拟磁盘本身，以及用于存储数据和发送虚拟 I/O 的虚拟磁盘后端。虚拟磁盘后端由虚拟磁盘服务器 (`vds`) 驱动程序从服务域导出。`vds` 驱动程序通过虚拟机管理程序，借助于逻辑域通道 (logical domain channel, LDC) 与来宾域中的虚拟磁盘客户机 (`vdc`) 驱动程序进行通信。最终，虚拟磁盘在来宾域中显示为 `/dev/[r]dsk/cXdYsZ` 设备。

注 - 可以通过使用 `/dev/dsk` 或 `/dev/rdsk` 作为磁盘路径名的一部分引用磁盘。任一引用都会产生相同结果。

注意 - 不要使用 d0 设备表示整个磁盘。仅当磁盘具有 EFI 标签（不是 VTOC 标签）时此设备才表示整个磁盘。使用 d0 设备会导致虚拟磁盘成为单分片磁盘，这可能会使您在写入磁盘的开头时损坏磁盘标签。

改用 s2 分片虚拟化整个磁盘。s2 分片与标签无关。

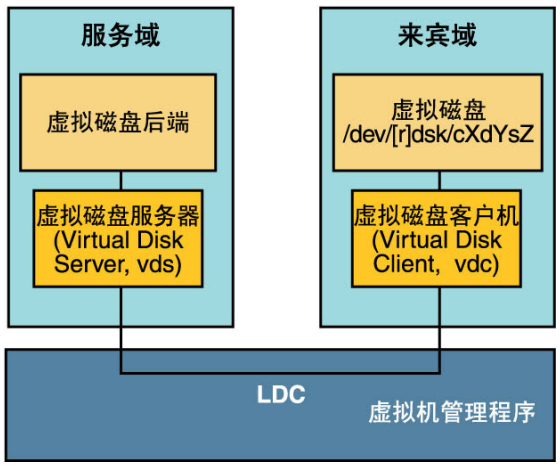
虚拟磁盘后端可以是物理设备，也可以是逻辑设备。物理设备可包括下列项：

- 物理磁盘或磁盘逻辑单元号 (logical unit number, LUN)
- 物理磁盘分片

逻辑设备可以是下列任意项：

- 位于本地文件系统（如 ZFS 或 UFS）或通过 NFS 提供的远程文件系统中的文件
- 卷管理器（如 ZFS、VxVM 或 Solaris Volume Manager）中的逻辑卷
- 可从服务域访问的任意磁盘伪设备

图 10-1 虚拟磁盘与 Oracle VM Server for SPARC



虚拟磁盘标识符和设备名称

当您使用 ldm add-vdisk 命令向域添加虚拟磁盘时，您可以通过设置 id 属性来指定其设备编号。

```
ldm add-vdisk [id=disk-id] disk-name volume-name@service-name domain-name
```

域的每个虚拟磁盘都有一个唯一的设备编号，该设备编号是在绑定域时指定的。如果使用显式设备编号（通过设置 `id` 属性获得）添加虚拟磁盘，则会使用指定的设备编号。否则，系统将自动指定最低的可用设备编号。在这种情况下，指定的设备编号将取决于向域添加虚拟磁盘的方式。绑定域后，最终指定给虚拟磁盘的设备编号会显示在 `ldm list-bindings` 命令的输出中。

当具有虚拟磁盘的域运行 Oracle Solaris OS 时，每个虚拟磁盘在该域中都将显示为 `c0d`*n* 磁盘设备，其中 *n* 是虚拟磁盘的设备编号。

在以下示例中，`ldg1` 域具有两个虚拟磁盘：`rootdisk` 和 `pdisk`。`rootdisk` 的设备编号为 `0` (`disk@0`)，在域中显示为 `c0d0` 磁盘设备。`pdisk` 的设备编号为 `1` (`disk@1`)，在域中显示为 `c0d1` 磁盘设备。

```
primary# ldm list-bindings ldg1
...
DISK
    NAME             VOLUME                         TOUT DEVICE  SERVER         MPGROUP
    rootdisk         dsk_nevada@primary-vds0             disk@0  primary
    pdisk            c3t40d1@primary-vds0                disk@1  primary
...
```

注意 - 如果设备编号未显式指定给虚拟磁盘，则在域解除绑定并稍后重新绑定时，可更改其设备编号。在这种情况下，由该域中运行的 OS 指定的设备名称还可以更改和解除系统的现有配置。例如，从域的配置中删除虚拟磁盘时，可能会出现这种情况。

管理虚拟磁盘

本节将介绍如何向来宾域添加虚拟磁盘、如何更改虚拟磁盘和超时选项，以及如何从来宾域中删除虚拟磁盘。有关虚拟磁盘选项的说明，请参见“虚拟磁盘后端选项” [158]。有关虚拟磁盘超时的说明，请参见“虚拟磁盘超时” [173]。

通过相同或不同的虚拟磁盘服务器可多次导出虚拟磁盘后端。每个导出的虚拟磁盘后端实例随后可指定给相同或不同的来宾域。

如果要多次导出虚拟磁盘后端，则不应使用独占 (`excl`) 选项将其导出。指定 `excl` 选项仅允许将后端导出一次。使用 `ro` 选项，可以将后端作为只读设备安全地导出多次。

向域分配虚拟设备会创建对提供虚拟磁盘服务的域的隐式依赖关系。可以使用 `ldm list-dependencies` 命令查看这些依赖关系或者查看依赖虚拟磁盘服务的域。请参见“列出域 I/O 依赖关系” [320]。

▼ 如何添加虚拟磁盘

1. **从服务域导出虚拟磁盘后端。**

   ```
   ldm add-vdsdev [-fq] [options={ro,slice,excl}] [mpgroup=mpgroup] \
   backend volume-name@service-name
   ```

2. **将后端指定给来宾域。**

   ```
   ldm add-vdisk [timeout=seconds] [id=disk-id] disk-name volume-name@service-name domain-name
   ```

   您可以通过设置 `id` 属性来指定新虚拟磁盘设备的定制 ID。默认情况下，ID 值会自动生成，所以，如果您需要匹配 OS 中的现有设备名称，请设置此属性。请参见“虚拟磁盘标识符和设备名称” [154]。

   注 - 绑定来宾域 (*domain-name*) 时，后端实际上是从服务域中导出并指定给来宾域的。

▼ 如何多次导出虚拟磁盘后端

注意 - 多次导出虚拟磁盘后端时，在来宾域上运行且使用该虚拟磁盘的应用程序将负责协调和同步并发写入访问，以确保数据一致性。

以下示例介绍如何通过同一个虚拟磁盘服务将同一个虚拟磁盘添加到两个不同的来宾域。

1. **从服务域导出虚拟磁盘后端两次。**

   ```
   ldm add-vdsdev [options={ro,slice}] backend volume1@service-name
   # ldm add-vdsdev -f [options={ro,slice}] backend volume2@service-name
   ```

   请注意，第二个 `ldm add-vdsdev` 命令将使用 `-f` 选项强制执行后端的第二次导出。将同一后端路径用于这两个命令且虚拟磁盘服务器位于同一服务域时，使用此选项。

2. **将导出的后端分配给每个来宾域。**

   对于 `ldom1` 和 `ldom2` 而言，*disk-name* 可以不同。

   ```
   ldm add-vdisk [timeout=seconds] disk-name volume1@service-name ldom1
   # ldm add-vdisk [timeout=seconds] disk-name volume2@service-name ldom2
   ```

▼ 如何更改虚拟磁盘选项

有关虚拟磁盘选项的更多信息，请参见“虚拟磁盘后端选项” [158]。

- **从服务域导出后端之后，可以更改虚拟磁盘选项。**

```
primary# ldm set-vdsdev options=[{ro,slice,excl}] volume-name@service-name
```

▼ 如何更改超时选项

有关虚拟磁盘选项的更多信息，请参见“虚拟磁盘后端选项” [158]。

- **将虚拟磁盘分配给来宾域之后，可以更改虚拟磁盘的超时。**

```
primary# ldm set-vdisk timeout=seconds disk-name domain-name
```

▼ 如何删除虚拟磁盘

1. **从来宾域删除虚拟磁盘。**

```
primary# ldm rm-vdisk disk-name domain-name
```

2. **停止从服务域导出相应的后端。**

```
primary# ldm rm-vdsdev volume-name@service-name
```

虚拟磁盘外观

将后端导出为虚拟磁盘之后，它可以在来宾域中显示为完整磁盘或具有单个分片的磁盘。它的显示方式取决于后端的类型以及导出后端时所使用的选项。

注意 - 单分片磁盘没有设备 ID。如果需要设备 ID，使用整个物理磁盘后端。

完整磁盘

将后端作为完整磁盘导出到某个域之后，该后端将在该域中显示为一个具有八个分片（s0 至 s7）的常规磁盘。可使用 format(1M) 命令来查看此类磁盘。使用 fmthard 或 format 命令可更改磁盘的分区表。

在 OS 安装软件中也可以显示完整磁盘，并且能够将其选作可安装 OS 的磁盘。

除了只能作为具有单个分片的磁盘导出的物理磁盘分片之外，所有后端都可作为完整磁盘导出。

单分片磁盘

将后端作为具有单个分片的磁盘导出到某个域之后，该后端将在该域中显示为一个具有八个分片（s0 至 s7）的常规磁盘。但是，只有第一个分片 (s0) 可用。可通过 format(1M) 命令来查看此类磁盘，但无法更改磁盘的分区表。

在 OS 安装软件中也可以显示具有单个分片的磁盘，并且能够将其选作可安装 OS 的磁盘。在这种情况下，如果使用 UNIX 文件系统 (UNIX File System, UFS) 安装 OS，则必须只定义根分区 (/)，并且该分区必须占用所有磁盘空间。

除只能作为完整磁盘导出的物理磁盘之外，所有后端都可作为具有单个分片的磁盘导出。

注 - 在 Oracle Solaris 10 10/08 OS 发行版之前，具有单个分片的磁盘显示为具有一个分区 (s0) 的磁盘。无法使用 format 命令来查看此类磁盘。在 OS 安装软件中也不会显示此类磁盘，并且也不能将其选作为可安装 OS 的磁盘设备。

虚拟磁盘后端选项

导出虚拟磁盘后端时可指定不同的选项。这些选项在 ldm add-vdsdev 命令的 options= 参数中以逗号分隔列表的形式来表示。有效选项包括：ro、slice 和 excl。

只读 (ro) 选项

只读 (ro) 选项指定将后端导出为只读设备。在这种情况下，只能对分配给来宾域的虚拟磁盘进行读取访问操作，对该虚拟磁盘进行的任何写入操作都将失败。

独占 (excl) 选项

独占 (excl) 选项指定在将服务域中的后端作为虚拟磁盘导出到另一个域时，该后端只能通过虚拟磁盘服务器以独占方式打开。以独占方式打开后端时，服务域中的其他应用程序将无法访问该后端。此限制可以防止服务域中运行的应用程序无意中使用正在由来宾域使用的后端。

注 - 某些驱动程序不支持 excl 选项，并且不允许以独占方式打开某些虚拟磁盘后端。已知 excl 选项适用于物理磁盘和分片，但该选项不适用于文件。该选项可能适用于伪设备（例如磁盘卷）。如果后端的驱动程序不支持独占打开，则会忽略后端的 excl 选项，且不能以独占方式打开后端。

因为 excl 选项可防止服务域中运行的应用程序访问导出到来宾域的后端，因此，请勿在以下情况下设置 excl 选项：

- 当来宾域正在运行时，如果您希望能够使用命令（如 format 或 luxadm）管理物理磁盘，则不要使用 excl 选项导出这些磁盘。
- 当您导出 Solaris Volume Manager（如 RAID 或镜像卷）时，请勿设置 excl 选项。否则，在 RAID 或镜像卷的组件出现故障时，这会阻止 Solaris Volume Manager 启动某个恢复操作。有关更多信息，请参见“通过 Solaris Volume Manager 使用虚拟磁盘” [179]。
- 如果 Veritas 卷管理器 (Veritas Volume Manager, VxVM) 已安装在服务域中，并且针对物理磁盘启用了 Veritas 动态多路径 (Veritas Dynamic Multipathing, VxDMP)，则在导出物理磁盘时不得使用 excl 选项（非默认选项）。否则，导出将失败，因为虚拟磁盘服务器 (virtual disk server, vds) 无法打开物理磁盘设备。有关更多信息，请参见“在安装了 VxVM 的情况下使用虚拟磁盘” [180]。
- 如果要从同一个虚拟磁盘服务多次导出同一个虚拟磁盘后端，请参见如何多次导出虚拟磁盘后端 [156]以了解更多信息。

默认情况下，后端以非独占方式打开。这样，将后端导出到其他域时，后端仍可以由服务域中运行的应用程序使用。

分片 (slice) 选项

通常，后端既可以导出为完整磁盘，也可以导出为具有单个分片的磁盘，具体取决于后端的类型。如果指定了 slice 选项，则会强制将后端导出为具有单个分片的磁盘。

当您要导出后端的原始内容时，此选项非常有用。例如，如果您有一个存有数据的 ZFS 或 Solaris Volume Manager 卷，且希望来宾域访问这些数据，则应使用 slice 选项导出 ZFS 或 Solaris Volume Manager 卷。

有关此选项的更多信息，请参见“虚拟磁盘后端” [160]。

虚拟磁盘后端

虚拟磁盘后端是存储虚拟磁盘数据的位置。后端可以是磁盘、磁盘分片、文件或卷（如 ZFS、Solaris Volume Manager 或 VxVM）。后端在来宾域中既可以显示为完整磁盘，也可以显示为具有单个分片的磁盘，具体取决于在从服务域导出后端时是否设置了 `slice` 选项。默认情况下，虚拟磁盘后端以非独占方式导出为可读写的完整磁盘。

物理磁盘或磁盘 LUN

物理磁盘或磁盘 LUN 始终作为完整磁盘导出。在这种情况下，虚拟磁盘驱动程序（`vds` 和 `vdc`）从虚拟磁盘转发 I/O，并充当到物理磁盘或磁盘 LUN 的传递通道。

通过在不设置 `slice` 选项的情况下导出与该磁盘分片 2 (`s2`) 相对应的设备，可从服务域导出物理磁盘或磁盘 LUN。如果使用 `slice` 选项导出某个磁盘的分片 2，则将只导出该分片，而非整个磁盘。

▼ 如何将物理磁盘作为虚拟磁盘导出

注意 - 配置虚拟磁盘时，确保每个虚拟磁盘引用不同的物理（后端）资源，如物理磁盘、磁盘分片、文件或卷。

某些磁盘（如光纤通道和 SAS）本身具有“双端口”，即同一个磁盘可以被两个不同的路径引用。确保分配给不同域的路径不会引用同一个物理磁盘。

1. **将物理磁盘作为虚拟磁盘导出。**
   例如，要将物理磁盘 `c1t48d0` 作为虚拟磁盘导出，则必须导出该磁盘的分片 2 (`c1t48d0s2`)。

   ```
   primary# ldm add-vdsdev /dev/dsk/c1t48d0s2 c1t48d0@primary-vds0
   ```

2. **将磁盘指定给来宾域。**
   例如，将磁盘 (`pdisk`) 指定给来宾域 `ldg1`。

   ```
   primary# ldm add-vdisk pdisk c1t48d0@primary-vds0 ldg1
   ```

3. **来宾域启动并运行 Oracle Solaris OS 之后，检验该磁盘是否可供访问且是否为完整磁盘。**
   完整磁盘是一个具有八 (8) 个分片的常规磁盘。
   例如，要检查的磁盘为 `c0d1`。

```
ldg1# ls -1 /dev/dsk/c0d1s*
/dev/dsk/c0d1s0
/dev/dsk/c0d1s1
/dev/dsk/c0d1s2
/dev/dsk/c0d1s3
/dev/dsk/c0d1s4
/dev/dsk/c0d1s5
/dev/dsk/c0d1s6
/dev/dsk/c0d1s7
```

物理磁盘分片

物理磁盘分片始终作为具有单个分片的磁盘导出。在这种情况下，虚拟磁盘驱动程序（vds 和 vdc）从虚拟磁盘转发 I/O，并充当到物理磁盘分片的传递通道。

通过导出相应的分片设备可从服务域导出物理磁盘分片。如果该设备与分片 2 不同，则无论是否指定了 slice 选项，该设备都会自动导出为具有单个分片的磁盘。如果该设备为磁盘的分片 2，则必须设置 slice 选项，以便仅将分片 2 作为单分片磁盘导出。否则会将整个磁盘作为完整磁盘导出。

▼ 如何将物理磁盘分片作为虚拟磁盘导出

1. **将物理磁盘的某个分片作为虚拟磁盘导出。**
   例如，要将物理磁盘 c1t57d0 的分片 0 作为虚拟磁盘导出，必须按照如下方式导出与分片 (c1t57d0s0) 相对应的设备。

   ```
   primary# ldm add-vdsdev /dev/dsk/c1t57d0s0 c1t57d0s0@primary-vds0
   ```

   您无需指定 slice 选项，因为分片始终作为具有单个分片的磁盘导出。

2. **将磁盘指定给来宾域。**
   例如，将磁盘 (pslice) 指定给来宾域 ldg1。

   ```
   primary# ldm add-vdisk pslice c1t57d0s0@primary-vds0 ldg1
   ```

3. **来宾域启动并运行 Oracle Solaris OS 之后，可列出磁盘（例如，c0d13），并且您可以查看该磁盘是否可供访问。**

   ```
   ldg1# ls -1 /dev/dsk/c0d13s*
   /dev/dsk/c0d13s0
   /dev/dsk/c0d13s1
   /dev/dsk/c0d13s2
   /dev/dsk/c0d13s3
   /dev/dsk/c0d13s4
   ```

```
/dev/dsk/c0d13s5
/dev/dsk/c0d13s6
/dev/dsk/c0d13s7
```

尽管有八个设备，但是，由于该磁盘是一个具有单个分片的磁盘，因此只能使用第一个分片 (`s0`)。

▼ 如何导出分片 2

- **要导出分片 2（例如，磁盘 `c1t57d0s2`），必须指定 `slice` 选项。否则会导出整个磁盘。**

```
primary# ldm add-vdsdev options=slice /dev/dsk/c1t57d0s2 c1t57d0s2@primary-vds0
```

文件和卷导出

文件或卷（例如，来自 ZFS 或 Solaris Volume Manager）既可以作为完整磁盘导出，也可以作为具有单个分片的磁盘导出，具体取决于是否设置了 `slice` 选项。

文件或卷作为完整磁盘导出

如果未设置 `slice` 选项，文件或卷会作为完整磁盘导出。在这种情况下，虚拟磁盘驱动程序（`vds` 和 `vdc`）从虚拟磁盘转发 I/O 并管理虚拟磁盘的分区。文件或卷最终会变成一个磁盘映像，其中含有虚拟磁盘所有分片中的数据以及用于管理分区和磁盘结构的元数据。

将空白文件或卷作为完整磁盘导出时，它将在来宾域中显示为未格式化的磁盘（即，无分区的磁盘）。然后，您需要在来宾域中运行 `format` 命令，以便定义可用的分区并写入有效的磁盘标签。如果虚拟磁盘未格式化，对该磁盘进行的所有 I/O 操作都将失败。

注 - 必须在来宾域中运行 `format` 命令来创建分区。

▼ 如何将文件作为完整磁盘导出

1. **在服务域中，创建一个文件（例如，`fdisk0`），用作虚拟磁盘。**

```
service# mkfile 100m /ldoms/domain/test/fdisk0
```

该文件的大小定义虚拟磁盘的大小。此示例会创建一个 100 MB 的空白文件，以获取 100 MB 的虚拟磁盘。

2. 在控制域中，将该文件导出为虚拟磁盘。

```
primary# ldm add-vdsdev /ldoms/domain/test/fdisk0 fdisk0@primary-vds0
```

在此示例中，未设置 slice 选项，所以，该文件将作为完整磁盘导出。

3. 在控制域中，将磁盘指定给来宾域。

例如，将磁盘 (fdisk) 指定给来宾域 ldg1。

```
primary# ldm add-vdisk fdisk fdisk0@primary-vds0 ldg1
```

4. 来宾域启动并运行 Oracle Solaris OS 之后，检验该磁盘是否可供访问且是否为完整磁盘。

完整磁盘是一个具有八个分片的常规磁盘。

以下示例介绍如何列出磁盘 c0d5，并检验该磁盘是否可供访问且是否为完整磁盘。

```
ldg1# ls -1 /dev/dsk/c0d5s*
/dev/dsk/c0d5s0
/dev/dsk/c0d5s1
/dev/dsk/c0d5s2
/dev/dsk/c0d5s3
/dev/dsk/c0d5s4
/dev/dsk/c0d5s5
/dev/dsk/c0d5s6
/dev/dsk/c0d5s7
```

▼ 如何将 ZFS 卷作为完整磁盘导出

1. 创建 ZFS 卷，用作完整磁盘。

以下示例说明如何创建要用作完整磁盘的 ZFS 卷 zdisk0。

```
service# zfs create -V 100m ldoms/domain/test/zdisk0
```

卷的大小定义虚拟磁盘的大小。此示例创建了一个 100 MB 的卷，以获取 100 MB 的虚拟磁盘。

2. 在控制域中，将相应的设备导出到该 ZFS 卷。

```
primary# ldm add-vdsdev /dev/zvol/dsk/ldoms/domain/test/zdisk0 \
zdisk0@primary-vds0
```

在此示例中，未设置 slice 选项，因此，该文件将作为完整磁盘导出。

3. 在控制域中，将卷指定给来宾域。

以下示例说明如何将卷 zdisk0 分配给来宾域 ldg1：

```
primary# ldm add-vdisk zdisk0 zdisk0@primary-vds0 ldg1
```

4. **来宾域启动并运行 Oracle Solaris OS 之后，检验该磁盘是否可供访问且是否为完整磁盘。**
   完整磁盘是一个具有八个分片的常规磁盘。
   以下示例说明如何列出磁盘 `c0d9`，并检验该磁盘是否可供访问且是否为完整磁盘：

```
ldg1# ls -1 /dev/dsk/c0d9s*
/dev/dsk/c0d9s0
/dev/dsk/c0d9s1
/dev/dsk/c0d9s2
/dev/dsk/c0d9s3
/dev/dsk/c0d9s4
/dev/dsk/c0d9s5
/dev/dsk/c0d9s6
/dev/dsk/c0d9s7
```

文件或卷作为具有单个分片的磁盘导出

如果已设置 `slice` 选项，则文件或卷会作为具有单个分片的磁盘导出。在这种情况下，虚拟磁盘仅具有一个分区 (`s0`)，该分区直接映射到文件或卷后端。文件或卷仅包含写入到虚拟磁盘的数据，而不包含额外数据（如分区信息或磁盘结构）。

将文件或卷作为具有单个分片的磁盘导出时，系统会模拟一个假磁盘分区，这使得文件或卷看起来像磁盘分片。因为对磁盘分区进行了模拟，所以，您无需为该磁盘创建分区。

▼ 如何将 ZFS 卷作为具有单个分片的磁盘导出

1. **创建 ZFS 卷，用作具有单个分片的磁盘。**
   以下示例介绍如何创建要用作具有单个分片的磁盘的 ZFS 卷 `zdisk0`。

```
service# zfs create -V 100m ldoms/domain/test/zdisk0
```

   卷的大小定义虚拟磁盘的大小。此示例会创建一个 100 MB 的卷，以获取 100 MB 的虚拟磁盘。

2. **在控制域中，将相应的设备导出到该 ZFS 卷，并设置 `slice` 选项，以便将该卷作为具有单个分片的磁盘导出。**

```
primary# ldm add-vsdev options=slice /dev/zvol/dsk/ldoms/domain/test/zdisk0 \
zdisk0@primary-vds0
```

3. **在控制域中，将卷指定给来宾域。**
   下面介绍如何将卷 `zdisk0` 指定给来宾域 `ldg1`。

```
primary# ldm add-vdisk zdisk0 zdisk0@primary-vds0 ldg1
```

4. 来宾域启动并运行 Oracle Solaris OS 之后，可列出磁盘（例如，**c0d9**），并且您可以查看该磁盘是否可供访问且是否为具有单个分片的磁盘 (**s0**)。

```
ldg1# ls -1 /dev/dsk/c0d9s*
/dev/dsk/c0d9s0
/dev/dsk/c0d9s1
/dev/dsk/c0d9s2
/dev/dsk/c0d9s3
/dev/dsk/c0d9s4
/dev/dsk/c0d9s5
/dev/dsk/c0d9s6
/dev/dsk/c0d9s7
```

导出卷以及向后兼容性

如果已配置为将卷作为虚拟磁盘导出，则卷将作为完整磁盘导出，而不是作为单分片磁盘导出。要保留旧行为并将卷作为具有单个分片的磁盘导出，您需要执行以下操作之一：

- 在 Oracle VM Server for SPARC 3.2 软件中使用 `ldm set-vdsdev` 命令，并针对要作为具有单个分片的磁盘导出的所有卷设置 `slice` 选项。请参见 `ldm`(1M) 手册页。
- 将以下行添加到服务域上的 `/etc/system` 文件中。

  ```
  set vds:vd_volume_force_slice = 1
  ```

  有关正确创建或更新 `/etc/system` 属性值的信息，请参见“更新 `/etc/system` 文件中的属性值” [302]。

注 - 设置此可调参数会将所有卷强制作为具有单个分片的磁盘导出，且无法将任何卷作为完整磁盘导出。

不同类型的后端的导出方式汇总

后端	没有分片选项	设置了分片选项
磁盘（磁盘分片 2）	完整磁盘[†]	单分片磁盘[‡]
磁盘分片（不是分片 2）	单分片磁盘[*]	单分片磁盘
文件	完整磁盘	单分片磁盘
卷（包括 ZFS、Solaris Volume Manager 或 Vx VM）	完整磁盘	单分片磁盘

[†]导出整个磁盘。

[‡]仅导出分片 2

*分片始终作为具有单个分片的磁盘导出。

将文件和磁盘分片作为虚拟磁盘导出的准则

本节包括将文件和磁盘分片作为虚拟磁盘导出的准则。

使用回送文件 (`lofi`) 驱动程序

如果使用回送文件 (`lofi`) 驱动程序将文件作为虚拟磁盘导出，则会添加一个额外的驱动程序层，并影响虚拟磁盘的性能。您可以改为直接将文件作为完整磁盘或具有单个分片的磁盘导出。请参见“文件和卷导出” [162]。

直接或间接导出磁盘分片

要将分片直接或间接（例如，通过 Solaris Volume Manager 卷）作为虚拟磁盘导出，应通过使用 `prtvtoc` 命令确保该分片不是始于物理磁盘的第一个块（块 0）。

如果直接或间接导出了始于物理磁盘的第一个块的磁盘分片，可能会覆盖物理磁盘的分区表，并导致该磁盘的所有分区不可访问。

配置虚拟磁盘多路径

通过虚拟磁盘多路径可以在来宾域中配置可通过多个路径对其后端存储进行访问的虚拟磁盘。这些路径指向不同的能够对同一后端存储（如磁盘 LUN）提供访问权限的服务域。即使在其中一个服务域关闭的情况下，此功能也能够使来宾域中的虚拟磁盘处于可访问状态。例如，您可以设置一个虚拟磁盘多路径配置，用于访问网络文件系统 (network file system, NFS) 服务器上的文件。或者，您可以使用此配置访问已连接到多个服务域的共享存储中的 LUN。所以，当来宾域访问虚拟磁盘时，虚拟磁盘驱动程序将通过其中一个服务域访问后端存储。如果虚拟磁盘驱动程序无法连接到服务域，则该虚拟磁盘会尝试通过其他服务域访问后端存储。

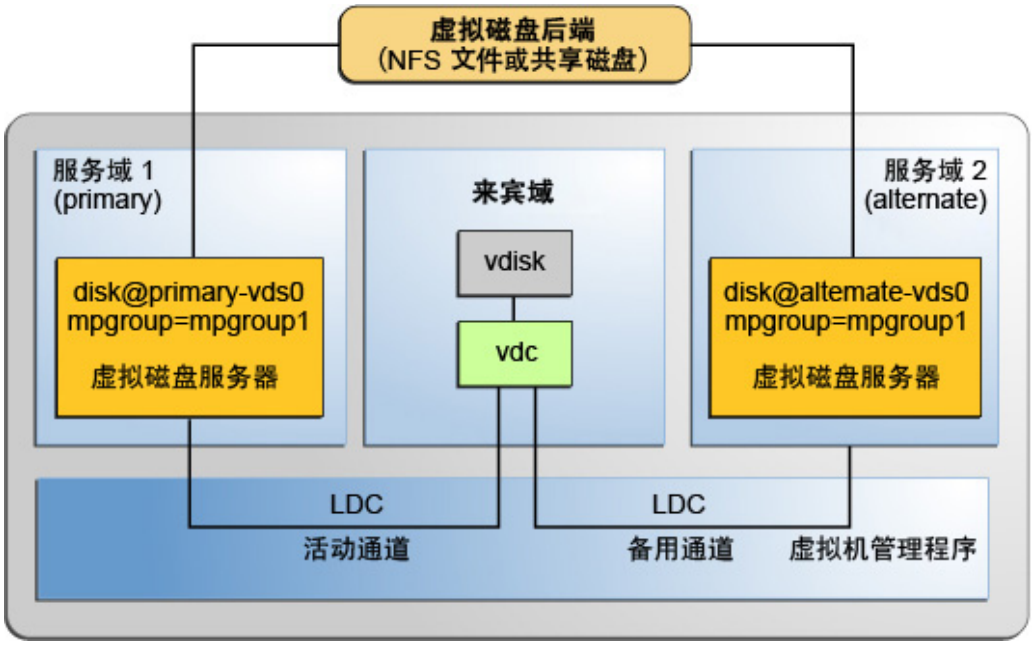
注 - 不能将 mpgroup 和 SCSI 预留空间一起使用。

请注意，虚拟磁盘多路径功能可以检测到服务域无法访问后端存储的时间。在此类实例中，虚拟磁盘驱动程序将尝试通过另一个路径访问后端存储。

要启用虚拟磁盘多路径功能，必须从每个服务域导出虚拟磁盘后端，并将虚拟磁盘添加到同一多路径组 (`mpgroup`) 中。`mpgroup` 由一个名称标识，并在导出虚拟磁盘后端之后进行配置。

下图显示了一个虚拟磁盘多路径配置，在如何配置虚拟磁盘多路径 [168]过程中将以此配置为例进行说明。在此示例中，名为 `mpgroup1` 的多路径组用于创建虚拟磁盘，通过以下两个服务域可对其后端进行访问：`primary` 和 `alternate`。

图 10-2 配置虚拟磁盘多路径



虚拟磁盘多路径和虚拟磁盘超时

使用虚拟磁盘多路径时，如果使用当前活动的路径无法访问后端，用于访问后端的路径会自动发生更改。此路径的更改与虚拟磁盘 `timeout` 属性的值无关。

虚拟磁盘 `timeout` 属性指定了一个时间量，如果在该时间量内没有服务域可用于处理 I/O，I/O 将会失败。此超时适用于所有虚拟磁盘，即使是使用虚拟磁盘多路径的磁盘也是如此。

因此，在配置了虚拟磁盘多路径的情况下设置虚拟磁盘超时可能会使多路径无法正常工作，尤其是超时值较小时更是如此。请避免为属于多路径组的虚拟磁盘设置虚拟磁盘超时。

有关更多信息，请参见“虚拟磁盘超时” [173]。

▼ 如何配置虚拟磁盘多路径

请参见图 10-2 “配置虚拟磁盘多路径”。

1. **从 `primary` 服务域导出虚拟磁盘后端。**

   ```
   primary# ldm add-vdsdev mpgroup=mpgroup1 backend-path1 volume@primary-vds0
   ```

   backend-path1 是指向 `primary` 域中虚拟磁盘后端的路径。

2. **从 `alternate` 服务域导出同一个虚拟磁盘后端。**

   ```
   primary# ldm add-vdsdev mpgroup=mpgroup1 backend-path2 volume@alternate-vds0
   ```

   backend-path2 是指向 `alternate` 域中虚拟磁盘后端的路径。

   注 - *backend-path1* 和 *backend-path2* 是指向两个不同的域（`primary` 和 `alternate`）中同一虚拟磁盘后端的路径。这些路径可以相同或不同，具体取决于 `primary` 和 `alternate` 域的配置。用户可以选择 *volume* 名。对于两个命令，卷名可以相同，也可以不同。

   ---

3. **将虚拟磁盘导出到来宾域。**

   ```
   primary# ldm add-vdisk disk-name volume@primary-vds0 domain-name
   ```

   注 - 尽管通过不同的服务域将虚拟磁盘后端多次导出，但只能为来宾域指定一个虚拟磁盘，并通过任意服务域将该磁盘与虚拟磁盘后端关联。

例 10-1 使用 Mpgroup 将 LUN 添加到主域和备用域的虚拟磁盘服务

下面显示了如何创建 LUN 并使用相同的 mpgroup 将该 LUN 添加到主域和备用域的虚拟磁盘服务：

要确定在访问 LUN 时首先使用的域，请在将磁盘添加到域时指定关联的路径。

- 创建虚拟磁盘设备：

  ```
  primary# ldm add-vdsdev mpgroup=ha lun1@primary-vds0
  primary# ldm add-vdsdev mpgroup=ha lun1@alternate-vds0
  ```

- 要首先使用来自 `primary-vds0` 的 LUN，请执行以下命令：

  ```
  primary# ldm add-vdisk disk1 lun1@primary-vds0 gd0
  ```

- 要首先使用来自 `alternate-vds0` 的 LUN，请执行以下命令：

```
primary# ldm add-vdisk disk1 lun1@alternate-vds0 gd0
```

配置虚拟磁盘多路径后所产生的结果

为虚拟磁盘配置了多路径并启动来宾域之后，虚拟磁盘将通过与其关联的服务域之一访问其后端。如果此服务域不可用，虚拟磁盘会尝试通过属于同一多路径组的其他服务域访问其后端。

注意 - 定义多路径组 (`mpgroup`) 时，应确保属于同一个 `mpgroup` 的虚拟磁盘后端实际上是同一个虚拟磁盘后端。如果将不同的后端添加到同一个 `mpgroup` 中，则可能会出现意外行为并有可能丢失或损坏存储在这些后端上的数据。

动态路径选择

从 Oracle VM Server for SPARC 3.2 软件开始，可以动态选择要用于至少运行 Oracle Solaris 11.2.1.0.0 (SRU 1) OS 的来宾域上虚拟磁盘的路径。

通过使用 `ldm set-vdisk` 命令将 `volume` 属性设置为表单 *volume-name*@*service-name* 中的值来更改 mpgroup 磁盘中的第一个路径时，会发生动态路径选择。支持动态路径选择的活动域仅能切换到选定路径。如果更新的驱动程序未在运行，则 Oracle Solaris OS 重新装入磁盘实例或者下次域重新引导时将选择此路径。

使用动态路径选择功能，可以在磁盘处于使用中时动态执行以下步骤：

- 指定连接磁盘时来宾域首先尝试的磁盘路径
- 将当前活动的路径更改为对已经连接的多路径磁盘指示的路径

现在将 `ldm add-vdisk` 命令与 mpgroup 磁盘一起使用可以将 *volume-name*@*service-name* 指示的路径指定为用来访问磁盘的选定路径。

选定的磁盘路径列在向来宾域提供的路径集中的第一个，而不考虑创建关联的 mpgroup 时该路径的排名。

可以在绑定、非活动和活动域上使用 `ldm set-vdisk` 命令。在活动域上使用时，此命令允许您仅选择 mpgroup 磁盘的指定路径。

`ldm list-bindings` 命令显示以下信息：

- 每个 mpgroup 路径的 `STATE` 列指示以下值之一：
  - `active` – mpgroup 的当前活动路径
  - `standby` – 路径当前未使用
  - `unknown` – 域不支持动态路径选择，设备未连接或者发生错误阻止检索路径状态
- 磁盘路径按用于选择活动路径的顺序列出

- 与磁盘关联的卷是 mpgroup 的选定路径，将会首先列出。

  以下示例显示选定路径为 `vol-ldg2@opath-ldg2`，当前使用的活动路径经过 `ldg1` 域。如果无法使用选定路径并转而使用第二个可能的路径，可能会看到这种情况。即使选定路径变为联机，也会继续使用非选定路径。要使第一个路径再次变为活动状态，请重新发出 `ldm set-vdisk` 命令将 `volume` 属性设置为所需的路径的名称。

```
DISK

NAME            VOLUME                   TOUT ID DEVICE SERVER  MPGROUP
disk            disk-ldg4@primary-vds0       0  disk@0 primary
tdiskgroup      vol-ldg2@opath-ldg2          1  disk@1 ldg2    testdiskgroup
    PORT MPGROUP VOLUME        MPGROUP SERVER   STATE
    2    vol-ldg2@opath-ldg2   ldg2            standby
    0    vol-ldg1@opath-vds    ldg1            active
    1    vol-prim@primary-vds0 primary         standby
```

如果在未至少运行 Oracle Solaris 11.2.1.0.0 (SRU 1) OS 的绑定域的 mpgroup 磁盘上使用 `ldm set-vdisk` 命令，该操作将更改路径优先级顺序，下次磁盘连接或重新引导过程中或者如果 OBP 需要访问新路径，可以首先使用新路径。

CD、DVD 和 ISO 映像

您可以按照与导出任意常规磁盘相同的方式导出光盘 (compact disc, CD) 或数字通用光盘 (digital versatile disc, DVD)。要将 CD 或 DVD 导出到来宾域，应将 CD 或 DVD 设备的分片 2 作为完整磁盘导出（即，不使用 `slice` 选项）。

注 - 不能导出 CD 或 DVD 驱动器本身。只能导出 CD 或 DVD 驱动器内的 CD 或 DVD。因此，CD 或 DVD 必须存在于该驱动器内部，才可以将其导出。此外，要能够导出 CD 或 DVD，该 CD 或 DVD 在服务域中不能处于使用状态。具体地说，卷管理文件系统 `volfs` 服务不得使用该 CD 或 DVD。有关如何使 `volfs` 停止使用设备的说明，请参见如何将 CD 或 DVD 从服务域导出到来宾域 [171]。

如果将 CD 或 DVD 的国际标准化组织 (International Organization for Standardization, ISO) 映像存储在文件或卷中，并将该文件或卷作为完整磁盘导出，则它将在来宾域中显示为一个 CD 或 DVD。

导出 CD、DVD 或 ISO 映像后，它将在来宾域中自动显示为只读设备。但是，您不能在来宾域中执行任何 CD 控制操作；即，无法从来宾域启动、停止或弹出 CD。如果导出的 CD、DVD 或 ISO 映像可引导，则可以在相应的虚拟磁盘中对来宾域进行引导。

例如，如果导出 Oracle Solaris OS 安装 DVD，则可以在与该 DVD 相对应的虚拟磁盘上引导来宾域，并从该 DVD 安装该来宾域。为此，当来宾域进入 `ok` 提示符时，请使用以下命令。

```
ok boot /virtual-devices@100/channel-devices@200/disk@n:f
```

其中 *n* 是表示已导出 DVD 的虚拟磁盘的索引。

注 - 如果导出 Oracle Solaris OS 安装 DVD，并在与该 DVD 相对应的虚拟磁盘上引导来宾域以对其进行安装，则在安装期间不能更改该 DVD。因此，您可能需要跳过请求其他 CD/DVD 的任何安装步骤，或者需要提供一个备用路径来访问所请求的这一介质。

▼ 如何将 CD 或 DVD 从服务域导出到来宾域

1. 在服务域中，检查卷管理守护进程 **vold** 是否正在运行且处于联机状态。

```
service# svcs volfs
STATE          STIME    FMRI
online         12:28:12 svc:/system/filesystem/volfs:default
```

2. 如果卷管理守护进程正在运行且处于联机状态（如步骤 1 的示例所示），请执行以下操作：

   a. 在 **/etc/vold.conf** 文件中，注释掉以下列文字开头的行：

   ```
   use cdrom drive....
   ```

   请参见 `vold.conf`(4) 手册页。

   b. 将 CD 或 DVD 插入 CD 或 DVD 驱动器。

   c. 在服务域中，重新启动卷管理文件系统服务。

   ```
   service# svcadm refresh volfs
   service# svcadm restart volfs
   ```

3. 在服务域中，查找 CD-ROM 设备的磁盘路径。

```
service# cdrw -l
Looking for CD devices...
   Node                  Connected Device                 Device type
----------------------+--------------------------------+-----------------
/dev/rdsk/c1t0d0s2    | MATSHITA CD-RW  CW-8124   DZ13 | CD Reader/Writer
```

4. 将 CD 或 DVD 磁盘设备作为完整磁盘导出。

```
primary# ldm add-vdsdev /dev/dsk/c1t0d0s2 cdrom@primary-vds0
```

5. 将导出的 CD 或 DVD 指定给来宾域。

   以下命令显示如何将导出的 CD 或 DVD 分配给域 `ldg1`：

```
primary# ldm add-vdisk cdrom cdrom@primary-vds0 ldg1
```

多次导出 CD 或 DVD

可以多次导出某个 CD 或 DVD 映像并将其分配给不同的来宾域。有关更多信息，请参见如何多次导出虚拟磁盘后端 [156]。

▼ 如何从控制域导出 ISO 映像以安装来宾域

开始之前 并假定 primary 域和来宾域均已配置。

例如，以下 ldm list 显示 primary 和 ldom1 域均已进行配置：

```
primary# ldm list
NAME             STATE    FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
primary          active   -n-cv   SP      8     8G       0.3%  15m
ldom1            active   -t---   5000    4     1G        25%  8m
```

1. **添加虚拟磁盘服务器设备以导出 ISO 映像。**

   在本示例中，ISO 映像是 /export/images/sol-10-u8-ga-sparc-dvd.iso。

   ```
   primary# ldm add-vdsdev /export/images/sol-10-u8-ga-sparc-dvd.iso dvd-iso@primary-vds0
   ```

2. **停止来宾域。**

   在本示例中，逻辑域是 ldom1。

   ```
   primary# ldm stop-domain ldom1
   LDom ldom1 stopped
   ```

3. **将 ISO 映像的虚拟磁盘添加到逻辑域。**

   在本示例中，逻辑域是 ldom1。

   ```
   primary# ldm add-vdisk s10-dvd dvd-iso@primary-vds0 ldom1
   ```

4. **重新启动来宾域。**

   在本示例中，逻辑域是 ldom1。

   ```
   primary# ldm start-domain ldom1
   LDom ldom1 started
   # ldm list
   NAME             STATE    FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
   primary          active   -n-cv   SP      8     8G       0.4%  25m
   ldom1            active   -t---   5000    4     1G       0.0%  0s
   ```

   在本示例中，ldm list 命令显示 ldom1 域已刚刚启动。

5. 连接到来宾域。

```
primary# telnet localhost 5000
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.

Connecting to console "ldom1" in group "ldom1" ....
Press ~? for control options ..
```

6. 验证 ISO 映像是否以虚拟磁盘形式存在。

```
{0} ok show-disks
a) /virtual-devices@100/channel-devices@200/disk@1
b) /virtual-devices@100/channel-devices@200/disk@0
q) NO SELECTION
Enter Selection, q to quit: q
```

在本示例中，新添加的设备是 `/virtual-devices@100/channel-devices@200/disk@1`。

7. 引导要从 ISO 映像安装的来宾域。

在本示例中，从 `/virtual-devices@100/channel-devices@200/disk@1` 磁盘的 `f` 分片引导。

```
{0} ok boot /virtual-devices@100/channel-devices@200/disk@1:f
```

虚拟磁盘超时

默认情况下，如果提供对虚拟磁盘后端访问的服务域已关闭，则从来宾域到相应虚拟磁盘的所有 I/O 都将被阻止。当服务域可以正常运行并能够向虚拟磁盘后端提供 I/O 请求服务时，I/O 可自动恢复。

但是，在某些情况下，文件系统或应用程序可能不希望 I/O 操作发生阻塞，而是希望在服务域关闭时间过长时，该操作会失败并报告错误。现在，您可以为每个虚拟磁盘设置一个连接超时期限，然后，使用它在来宾域上的虚拟磁盘客户机和服务域上的虚拟磁盘服务器之间建立连接。当达到超时期限时，只要服务域关闭且未在虚拟磁盘客户机和服务器之间重新建立连接，所有暂挂的 I/O 和所有新的 I/O 都将失败。

使用以下方法之一设置超时：

- 使用 `ldm add-vdisk` 命令。

  `ldm add-vdisk timeout=`*seconds disk-name volume-name@service-name domain-name*

- 使用 `ldm set-vdisk` 命令。

  `ldm set-vdisk timeout=`*seconds disk-name domain-name*

- 将以下行添加到来宾域上的 `/etc/system` 文件中。

```
set vdc:vdc_timeout=seconds
```

有关正确创建或更新 `/etc/system` 属性值的信息，请参见“更新 `/etc/system` 文件中的属性值” [302]。

注 - 如果已设置了此可调参数，它将覆盖使用 `ldm` CLI 进行的任何超时设置。此外，此可调参数为来宾域中的所有虚拟磁盘设置超时。

请以秒为单位指定超时。如果将超时设置为 `0`，则会禁用超时，并在服务域关闭时阻止 I/O（这是默认设置和行为）。

虚拟磁盘和 SCSI

如果将物理 SCSI 磁盘或 LUN 作为完整磁盘导出，则相应的虚拟磁盘会支持用户 SCSI 命令接口 `uscsi` 和多主机磁盘控制操作 `mhd`。其他虚拟磁盘（如将文件或卷作为后端的虚拟磁盘）不支持这些接口。

注 - 不能将 mpgroup 和 SCSI 预留空间一起使用。

因此，使用 SCSI 命令（例如，Solaris Volume Manager `metaset` 或 Oracle Solaris Cluster `shared devices`）的应用程序或产品功能在来宾域中只能与将物理 SCSI 磁盘作为后端的虚拟磁盘一起使用。

注 - 因为服务域对用作虚拟磁盘后端的物理 SCSI 磁盘或 LUN 进行管理，所以服务域可有效地执行 SCSI 操作。特别是，该服务域设置了 SCSI 预留空间。因此，在服务域和来宾域中运行的应用程序不应向同一物理 SCSI 磁盘发出 SCSI 命令。这样做会导致出现意外的磁盘状态。

虚拟磁盘和 `format` 命令

`format` 命令可识别域中存在的所有虚拟磁盘。但是，对于作为具有单个分片的磁盘导出的虚拟磁盘，`format` 命令无法更改此类虚拟磁盘的分区表。诸如 `label` 之类的命令都将失败，除非尝试写入与已与虚拟磁盘关联的磁盘标签类似的磁盘标签。

其后端为 SCSI 磁盘的虚拟磁盘支持所有 `format`(1M) 子命令。其后端不是 SCSI 磁盘的虚拟磁盘不支持某些 `format`(1M) 子命令（如 `repair` 和 `defect`）。在这种情况

下，format(1M) 的行为类似于集成驱动器电子 (Integrated Drive Electronics, IDE) 磁盘的行为。

将 ZFS 用于虚拟磁盘

本节介绍如何使用 Zettabyte 文件系统 (Zettabyte File System, ZFS) 存储已导出到来宾域的虚拟磁盘后端。ZFS 可提供一种简捷、有效的解决方案来创建和管理虚拟磁盘后端。使用 ZFS 可以执行以下操作：

- 将磁盘映像存储在 ZFS 卷或 ZFS 文件中
- 使用快照备份磁盘映像
- 使用克隆复制磁盘映像和置备更多的域

有关使用 ZFS 的更多信息，请参阅《Oracle Solaris ZFS Administration Guide 》。

在下面的介绍和示例中，主域还是存储磁盘映像的服务域。

在服务域中配置 ZFS 池

要存储磁盘映像，首先应在服务域中创建一个 ZFS 存储池。例如，以下命令在 primary 域中创建了一个包含磁盘 c1t50d0 的 ZFS 存储池 ldmpool。

```
primary# zpool create ldmpool c1t50d0
```

使用 ZFS 存储磁盘映像

以下命令为来宾域 ldg1 创建了一个磁盘映像。已为此来宾域创建了 ZFS 文件系统，从而此来宾域的所有磁盘映像都将存储在该文件系统上。

```
primary# zfs create ldmpool/ldg1
```

磁盘映像可以存储在 ZFS 卷或 ZFS 文件中。使用 zfs create -V 命令可快速创建 ZFS 卷，无论其大小为何。另一方面，必须使用 mkfile 命令创建 ZFS 文件。此命令可能需要一些时间才能完成，尤其是要创建的文件非常大时（通常在创建磁盘映像时）。

ZFS 卷和 ZFS 文件都可以利用 ZFS 的功能（例如，快照和克隆功能），但 ZFS 卷是一个伪设备，而 ZFS 文件是一个常规文件。

如果将磁盘映像用作安装有 OS 的虚拟磁盘，则该磁盘映像的大小必须足以满足 OS 安装的要求。此大小取决于 OS 的版本以及所执行的安装类型。如果要安装 Oracle

Solaris OS，则可以使用 20 GB 的磁盘大小，以满足任意版本的 Oracle Solaris OS 的任何安装类型的需要。

使用 ZFS 存储磁盘映像的示例

以下示例说明如何使用 ZFS 卷或 ZFS 文件存储磁盘映像。导出 ZFS 卷或文件的语法相同，但指向后端的路径不同。

启动来宾域后，ZFS 卷或文件将显示为可安装 Oracle Solaris OS 的虚拟磁盘。

例 10-2 使用 ZFS 卷存储磁盘映像

首先，在 ZFS 卷上创建 20 GB 映像。

```
primary# zfs create -V 20gb ldmpool/ldg1/disk0
```

然后，将 ZFS 卷作为虚拟磁盘导出。

```
primary# ldm add-vdsdev /dev/zvol/dsk/ldmpool/ldg1/disk0 ldg1_disk0@primary-vds0
```

将 ZFS 卷分配到 `ldg1` 来宾域。

```
primary# ldm add-vdisk disk0 ldg1_disk0@primary-vds0 ldg1
```

例 10-3 使用 ZFS 文件存储磁盘映像

首先，在 ZFS 卷上创建 20 GB 磁盘映像，然后创建 ZFS 文件。

```
primary# zfs create ldmpool/ldg1/disk0
primary# mkfile 20g /ldmpool/ldg1/disk0/file
```

然后，将 ZFS 文件作为虚拟磁盘导出。

```
primary# ldm add-vdsdev /ldmpool/ldg1/disk0/file ldg1_dis0@primary-vds0
```

将 ZFS 文件分配到 `ldg1` 来宾域。

```
primary# ldm add-vdisk disk0 ldg1_disk0@primary-vds0 ldg1
```

创建磁盘映像的快照

如果磁盘映像存储在 ZFS 卷或 ZFS 文件上，则可以使用 ZFS `snapshot` 命令创建该磁盘映像的快照。

创建磁盘映像的快照之前，请确保该磁盘在来宾域中当前未处于使用状态中，以确保当前存储在磁盘映像上的数据保持一致性。您可以使用以下方法之一确保磁盘在来宾域中处于未使用状态：

- 停止并解除绑定来宾域。这是一种最安全的解决方案，也是在创建用作来宾域引导磁盘的磁盘映像快照时唯一可用的解决方案。
- 卸载来宾域中正在使用的任何磁盘分片（您希望为这些分片创建快照），确保来宾域中未使用任何分片。

在本示例中，由于 ZFS 布局，不管磁盘映像是存储在 ZFS 卷上还是 ZFS 文件上，创建磁盘映像的快照时所使用的命令都相同。

例 10-4 创建磁盘映像的快照

此示例将创建为 `ldg1` 域创建的磁盘映像的快照。

```
primary# zfs snapshot ldmpool/ldg1/disk0@version_1
```

使用克隆置备新域

创建完磁盘映像的快照后，可以使用 ZFS `clone` 命令复制该磁盘映像。然后，可以将克隆的映像分配到其他域。通过克隆引导磁盘映像可以为新的来宾域快速创建引导磁盘，而无需执行整个 Oracle Solaris OS 安装过程。

例如，如果已创建的 `disk0` 是域 `ldg1` 的引导磁盘，请执行以下操作克隆该磁盘，以为域 `ldg2` 创建引导磁盘。

```
primary# zfs create ldmpool/ldg2
primary# zfs clone ldmpool/ldg1/disk0@version_1 ldmpool/ldg2/disk0
```

随后，`ldompool/ldg2/disk0` 可作为虚拟磁盘导出并指定给新的 `ldg2` 域。域 `ldg2` 可从该虚拟磁盘直接引导，而无需执行 OS 整个安装过程。

克隆引导磁盘映像

克隆引导磁盘映像后，新的映像与原始引导磁盘完全相同，它包含在克隆映像之前存储在引导磁盘上的所有信息（如主机名、IP 地址、已挂载的文件系统表或任意系统配置或可调参数）。

因为原始引导磁盘映像上已挂载文件系统表与已克隆的磁盘映像上的已挂载文件系统表相同，所以，将已克隆的磁盘映像指定给新域的的顺序必须与原始域上的相同。例如，

如果引导磁盘映像已指定为原始域的第一个磁盘，则已克隆的磁盘映像也必须指定为新域的第一个磁盘。否则，无法引导新域。

如果原始域是使用静态 IP 地址配置的，则使用克隆映像的新域必须以相同的 IP 地址开始。在这种情况下，您可以使用 Oracle Solaris 11 `sysconfig unconfigure` 命令或 Oracle Solaris 10 `sys-unconfig` 命令更改新域的网络配置。要避免此问题，您还可以为未配置的系统的磁盘映像创建快照。

如果原始域是使用动态主机配置协议 (Dynamic Host Configuration Protocol, DHCP) 配置的，则使用克隆映像的新域也应使用 DHCP。在这种情况下，您无需更改新域的网络配置，因为该域在其引导时可自动接收 IP 地址及其网络配置。

注 - 域的主机 ID 不会存储在引导磁盘上，而是在创建域时由 Logical Domains Manager 分配。因此，在克隆磁盘映像时，新域不会保留原始域的主机 ID。

▼ 如何创建未配置系统的磁盘映像的快照

1. **绑定和启动原始域。**
2. **取消配置系统。**

   - Oracle Solaris 11 OS：运行 **`sysconfig unconfigure`** 命令。
   - Oracle Solaris 10 OS：运行 **`sys-unconfig`** 命令。

   该操作完成后，域将停止。
3. **停止和解除绑定域；不要重新引导该域。**
4. **创建域引导磁盘映像的快照。**

   例如：

   ```
   primary# zfs snapshot ldmpool/ldg1/disk0@unconfigured
   ```

   此时，已经为未配置的系统的引导磁盘映像创建了快照。
5. **克隆此映像，以创建首次引导时要求对系统进行配置的新域。**

在 Oracle VM Server for SPARC 环境中使用卷管理器

本节介绍在 Oracle VM Server for SPARC 环境中使用卷管理器的信息。

通过卷管理器使用虚拟磁盘

可以将任何 ZFS、Solaris Volume Manager 或 Veritas 卷管理器 (Veritas Volume Manager, VxVM) 卷作为虚拟磁盘从服务域导出到来宾域。卷可以作为具有单个分片的磁盘导出（如果使用 `ldm add-vdsdev` 命令指定了 `slice` 选项）或作为完整磁盘导出。

注 - 本节的其余部分将以 Solaris Volume Manager 卷为例进行说明。但是，所讨论的内容也适用于 ZFS 卷和 VxVM 卷。

以下示例介绍如何将卷作为具有单个分片的磁盘导出。

来宾域中的虚拟磁盘（例如，`/dev/dsk/c0d2s0`）直接映射到相关联的卷（例如，`/dev/md/dsk/d0`），来宾域的虚拟磁盘上存储的数据直接存储到相关联的卷中，而没有额外的元数据。因此，存储在来宾域的虚拟磁盘上的数据也可以通过相关联的卷从服务域直接访问。

示例

- 如果将 Solaris Volume Manager 卷 `d0` 从 `primary` 域导出到 `domain1`，则 `domain1` 的配置需要一些额外的步骤。

```
primary# metainit d0 3 1 c2t70d0s6 1 c2t80d0s6 1 c2t90d0s6
primary# ldm add-vdsdev options=slice /dev/md/dsk/d0 vol3@primary-vds0
primary# ldm add-vdisk vdisk3 vol3@primary-vds0 domain1
```

- 例如，绑定并启动 `domain1` 之后，导出的卷会显示为 `/dev/dsk/c0d2s0`，并且您可以使用它。

```
domain1# newfs /dev/rdsk/c0d2s0
domain1# mount /dev/dsk/c0d2s0 /mnt
domain1# echo test-domain1 > /mnt/file
```

- 停止并解除绑定 `domain1` 后，可通过 Solaris Volume Manager 卷 `d0` 从主域直接访问 `domain1` 的虚拟磁盘上存储的数据。

```
primary# mount /dev/md/dsk/d0 /mnt
primary# cat /mnt/file
test-domain1
```

通过 Solaris Volume Manager 使用虚拟磁盘

当 RAID 或镜像 Solaris Volume Manager 卷由另一个域用作虚拟磁盘时，则在导出它时不得设置独占 (`excl`) 选项。否则，如果 Solaris Volume Manager 卷的某个组件出现故障，就无法启动使用 `metareplace` 命令或热备份恢复 Solaris Volume Manager 卷的过程。`metastat` 命令将卷视为正在进行重新同步，但并未进行重新同步。

例如，/dev/md/dsk/d0 是使用 excl 选项作为虚拟磁盘导出到另一个域的 RAID Solaris Volume Manager 卷，并且 d0 配置有一些热备份设备。如果 d0 的组件出现故障，则 Solaris Volume Manager 会将出现故障的组件替换为热备份，并重新同步 Solaris Volume Manager 卷。但是，重新同步并不会启动。卷会被报告为正在进行重新同步，但并没有进行重新同步。

```
primary# metastat d0
d0: RAID
    State: Resyncing
    Hot spare pool: hsp000
    Interlace: 32 blocks
    Size: 20097600 blocks (9.6 GB)
Original device:
    Size: 20100992 blocks (9.6 GB)
Device                                      Start Block  Dbase   State Reloc
c2t2d0s1                                           330  No       Okay  Yes
c4t12d0s1                                          330  No       Okay  Yes
/dev/dsk/c10t600C0FF0000000000015153295A4B100d0s1  330  No  Resyncing  Yes
```

这种情况下，必须停止并解除绑定将 Solaris Volume Manager 卷用作虚拟磁盘的域，才能完成重新同步。然后，可使用 metasync 命令重新同步 Solaris Volume Manager 卷。

```
# metasync d0
```

在安装了 VxVM 的情况下使用虚拟磁盘

如果在系统上安装了 VxVM，并在要作为虚拟磁盘导出的物理磁盘或分区上启用了 Veritas 动态多路径 (Dynamic Multipathing, DMP)，则在导出该磁盘或分区时不能设置（非默认）excl 选项。否则，在绑定使用此类磁盘的域时，则会在 /var/adm/messages 中收到错误。

```
vd_setup_vd():  ldi_open_by_name(/dev/dsk/c4t12d0s2) = errno 16
vds_add_vd():  Failed to add vdisk ID 0
```

您可以通过查看 vxdisk list 输出中的多路径信息，检查是否已启用 Veritas DMP。例如：

```
# vxdisk list Disk_3
Device:    Disk_3
devicetag: Disk_3
type:      auto
info:      format=none
flags:     online ready private autoconfig invalid
pubpaths:  block=/dev/vx/dmp/Disk_3s2 char=/dev/vx/rdmp/Disk_3s2
guid:      -
udid:      SEAGATE%5FST336753LSUN36G%5FDISKS%5F30323339483031444304E0000
site:      -
Multipathing information:
numpaths:  1
c4t12d0s2  state=enabled
```

或者，如果您针对要在设置了 excl 选项的情况下作为虚拟磁盘导出的磁盘或分片启用了 Veritas DMP，则可以使用 vxdmpadm 命令禁用 DMP。例如：

```
# vxdmpadm -f disable path=/dev/dsk/c4t12d0s2
```

将卷管理器用于虚拟磁盘

本节介绍如何将卷管理器用于虚拟磁盘。

将 ZFS 用于虚拟磁盘

任何虚拟磁盘都可以与 ZFS 结合使用。在任何域中都可以导入 ZFS 存储池 (zpool)，该域可以看到属于该 zpool 的所有存储设备，而不管该域将所有这些设备视为虚拟设备还是实际设备。

将 Solaris Volume Manager 用于虚拟磁盘

在 Solaris Volume Manager 本地磁盘集中可以使用任何虚拟磁盘。例如，虚拟磁盘可用于存储本地磁盘集的 Solaris Volume Manager 元设备状态数据库 metadb，或者用于在本地磁盘集中创建 Solaris Volume Manager 卷。

后端为 SCSI 磁盘的任何虚拟磁盘都可以在 Solaris Volume Manager 共享磁盘集 metaset 中使用。后端不是 SCSI 磁盘的虚拟磁盘不能添加到 Solaris Volume Manager 共享磁盘集。如果尝试将后端不是 SCSI 磁盘的虚拟磁盘添加到 Solaris Volume Manager 共享磁盘集，则会失败并显示类似下面内容的错误。

```
# metaset -s test -a c2d2
metaset: domain1: test: failed to reserve any drives
```

将 VxVM 用于虚拟磁盘

有关来宾域中的 VxVM 支持的信息，请参见 Symantec 的 VxVM 文档。

♦ ♦ ♦ 第 11 章

11 使用虚拟网络

本章介绍如何通过 Oracle VM Server for SPARC 软件使用虚拟网络，其中涵盖下列主题：

Oracle Solaris OS 联网在 Oracle Solaris 10 OS 与 Oracle Solaris 11 OS 之间有很大差别。有关要考虑的问题的信息，请参见“Oracle Solaris 10 联网概述” [184]、“Oracle Solaris 11 联网概述” [186]和“Oracle Solaris 11 中联网特定功能的差别” [230]。

虚拟网络简介

虚拟网络允许域在不使用任何外部物理网络的情况下进行相互通信。虚拟网络也可以允许域使用同一物理网络接口访问物理网络并与远程系统通信。配置虚拟交换机可创建虚拟网络，您可以将虚拟网络设备连接到此虚拟交换机。

Oracle Solaris 联网在 Oracle Solaris 10 OS 与 Oracle Solaris 11 OS 之间有很大差别。以下两节提供有关每个 OS 的联网的概述。

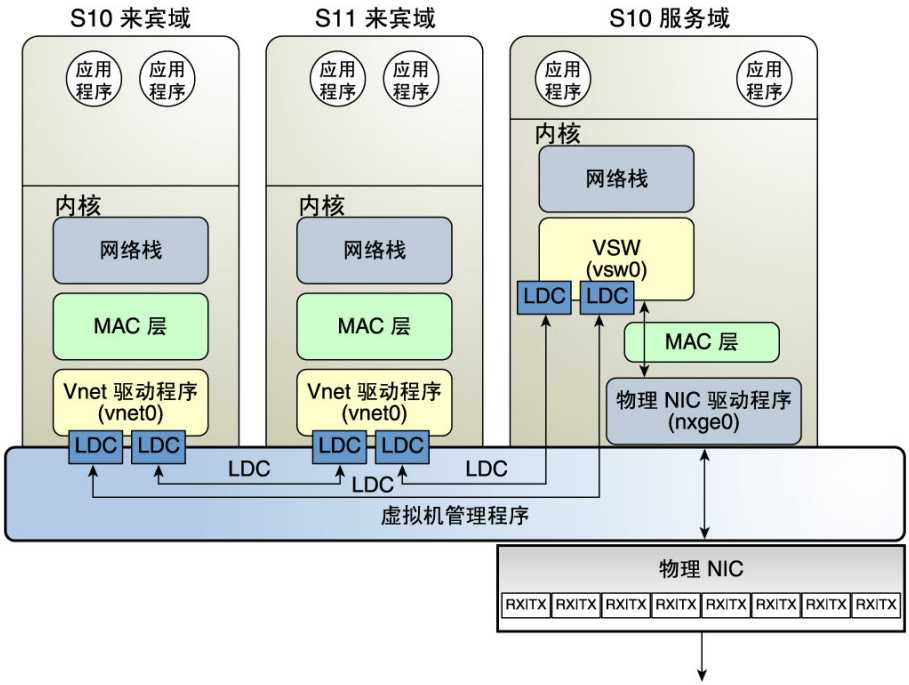
注 - Oracle Solaris 10 联网与域或系统的联网相同。Oracle Solaris 11 联网也一样。有关 Oracle Solaris OS 联网的更多信息，请参见 Oracle Solaris 10 Documentation（Oracle Solaris 10 文档）和 Oracle Solaris 11.1 Documentation（Oracle Solaris 11.1 文档）。

“Oracle Solaris 11 联网概述” [186]中介绍了 Oracle Solaris 10 与 Oracle Solaris 11 联网之间的功能差异。

Oracle Solaris 10 联网概述

下图显示运行 Oracle Solaris 11 OS 的来宾域与 Oracle Solaris 10 服务域完全兼容。唯一的差别是在 Oracle Solaris 11 OS 中添加或增强了相关功能。

图 11-1 Oracle Solaris 10 OS 的 Oracle VM Server for SPARC 网络概述



上图显示了仅适用于 Oracle Solaris 10 OS 的接口名称，例如，`nxge0`、`vsw0` 和 `vnet0`。另外，还要注意以下几点：

- 服务域中的虚拟交换机与来宾域相连接，以便来宾域可相互通信。
- 虚拟交换机也与物理网络接口 `nxge0` 相连接，以便来宾域可以与物理网络进行通信。
- 虚拟交换机网络接口 `vsw0` 会在服务域中创建，通过该接口，可以使两个来宾域与该服务域进行通信。
- 可以使用 Oracle Solaris 10 `ifconfig` 命令配置服务域中的虚拟交换机网络接口 `vsw0`。
- 可以使用 `ifconfig` 命令，将 Oracle Solaris 10 来宾域中的虚拟网络设备 `vnet0` 配置为网络接口。
- Oracle Solaris 11 来宾域中的虚拟网络设备 `vnet0` 可能显示通用链路名称，例如 `net0`。可以使用 `ipadm` 命令将其配置为网络接口。

虚拟交换机的行为与常规物理网络交换机相同，可在不同的系统（例如它连接到的来宾域、服务域和物理网络）之间交换网络包。`vsw` 驱动程序提供的网络设备功能允许您将虚拟交换机配置为网络接口。

Oracle Solaris 11 联网概述

Oracle Solaris 11 OS 引入了许多新的联网功能，在 Oracle Solaris 11.1 Documentation（Oracle Solaris 11.1 文档）中的 Oracle Solaris 11 联网文档中有对这些功能的介绍。

在使用 Oracle VM Server for SPARC 软件时，请务必了解以下 Oracle Solaris 11 联网功能：

- 所有网络配置都通过 `ipadm` 和 `dladm` 命令执行。
- “默认情况下的虚名”功能会为所有物理网络适配器生成通用链路名称，例如 `net0`。此功能还会为虚拟交换机 (`vsw`*n*) 和虚拟网络设备 (`vnet`*n*) 生成通用名称，这些名称显示为操作系统的物理网络适配器。要确定与物理网络设备相关联的通用链路名称，请使用 `dladm show-phys` 命令。

  在 Oracle Solaris 11 中，默认情况下，物理网络设备名称使用通用“虚”名。将使用通用名称（例如 `net0`）来代替 Oracle Solaris 10 中使用的设备驱动程序名称（例如 `nxge0`）。

  以下命令通过指定通用名称（`net0`）来代替驱动程序名称（例如 `nxge0`），为 `primary` 域创建虚拟交换机：

  ```
  primary# ldm add-vsw net-dev=net0 primary-vsw0 primary
  ```
- Oracle Solaris 11 OS 使用虚拟网络接口卡 (virtual network interface card, VNIC) 创建内部虚拟网络。

  VNIC 是物理网络设备的虚拟实例，可从物理网络设备创建并分配给一个区域。
- 在配置 Oracle VM Server for SPARC 软件时，请使用 Oracle Solaris 11 `DefaultFixed` 网络配置文件 (network configuration profile, NCP)。

  对于 Oracle Solaris 11 域，请使用 `DefaultFixed` NCP。您可以在安装期间或之后启用此配置文件。在 Oracle Solaris 11 安装期间，选择“手动”网络配置。
- 不要将主网络接口替换为虚拟交换机 (`vsw`) 接口。控制域可使用现有的主网络接口与将虚拟网络设备连接至相同虚拟交换机的来宾域进行通信。
- 不要对虚拟交换机使用物理网络适配器的 MAC 地址，因为对虚拟交换机使用物理适配器的 MAC 地址会与主网络接口发生冲突。

注 - 在本发行版中，请使用 `DefaultFixed` NCP 在 Oracle Solaris 11 系统上配置数据链路和网络接口。

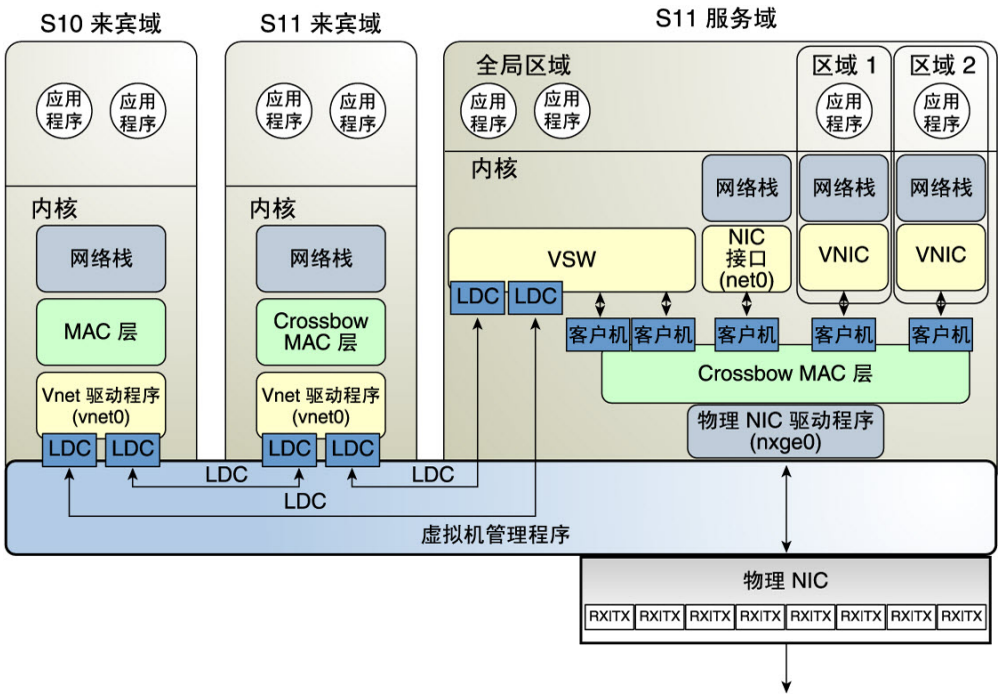
Oracle Solaris 11 OS 包括以下 NCP：

- `DefaultFixed` – 允许您使用 `dladm` 或 `ipadm` 命令管理网络
- `Automatic` – 允许您使用 `netcfg` 或 `netadm` 命令管理网络

使用 `netadm list` 命令确保已启用 `DefaultFixed` NCP。请参见《Oracle Solaris Administration: Network Interfaces and Network Virtualization》中的第 7 章 “Using Datalink and Interface Configuration Commands on Profiles”。

下图显示运行 Oracle Solaris 10 OS 的来宾域与 Oracle Solaris 11 服务域完全兼容。唯一的差别是在 Oracle Solaris 11 OS 中添加或增强了相关功能。

图 11-2 Oracle Solaris 11 OS 的 Oracle VM Server for SPARC 网络概述



该图显示网络设备名称（例如，`nxge0` 和 `vnet0`）可在 Oracle Solaris 11 域中表示为通用链路名称（例如，`net`*n*）。另外，还要注意以下几点：

- 服务域中的虚拟交换机与来宾域相连接，以便来宾域可相互通信。
- 虚拟交换机还会与物理网络设备 `nxge0` 相连接，以使来宾域能够与物理网络进行通信。

  虚拟交换机还允许来宾域与服务域网络接口 `net0` 以及与 `nxge0` 位于相同物理网络设备上的 VNIC 进行通信。这包括来宾域与 Oracle Solaris 11 服务域之间的通信。不要将虚拟交换机自身（`vsw`*n* 设备）配置为网络设备，因为不再支持此功能。
- 可以使用 `ifconfig` 命令，将 Oracle Solaris 10 来宾域中的虚拟网络设备 `vnet0` 配置为网络接口。
- Oracle Solaris 11 来宾域中的虚拟网络设备 `vnet0` 可能显示通用链路名称，例如 `net0`。可以使用 `ipadm` 命令将其配置为网络接口。

虚拟交换机的行为与常规物理网络交换机相似，可在它所连接的不同系统之间交换网络包。系统可以是来宾域、服务域或物理网络。

最大程度地提高虚拟网络性能

按照本节中所述配置平台和域，来宾网络和外部网络以及来宾域到来宾域的通信可以实现较高的传输速率。虚拟网络堆栈引入了对大型段负载转移 (large segment offload, LSO) 的支持，这可以产生高 TCP 性能且不需要使用巨型 (jumbo) 帧。

硬件和软件要求

要最大程度地提高域的网络性能，需要满足以下要求：

- **硬件要求。**只有 SPARC T4、SPARC T5、SPARC M5 和 SPARC M6 系统能实现这些性能改进。
- **系统固件要求。**这些 SPARC 系统必须运行最新的系统固件。请参见《Oracle VM Server for SPARC 3.2 安装指南 》中的“全限定系统固件版本”。
- **Oracle Solaris OS 要求。**确保服务域和来宾域运行以下 Oracle Solaris OS 版本：
  - **服务域。**至少 Oracle Solaris 11.1.9.0.0 OS 或具有 150031-03 修补程序的 Oracle Solaris 10 OS。
  - **来宾域。**至少 Oracle Solaris 11.1.9.0.0 OS 或具有 150031-03 修补程序的 Oracle Solaris 10 OS。
- **CPU 和内存要求。**确保为服务域和来宾域分配了足够的 CPU 和内存资源。
  - **服务域。**因为服务域充当来宾域的数据代理，所以至少向服务域分配 2 个 CPU 核心和 4 GB 内存。
  - **来宾域。**将每个来宾域配置为至少能够驱动 10-Gbps 的性能。至少为每个来宾域分配 2 个 CPU 核心和 4 GB 内存。

对域进行配置以最大程度地提高虚拟网络的性能

在早期版本的 Oracle VM Server for SPARC 和 Oracle Solaris OS 中，您可以通过配置巨型帧改进网络性能。此配置不再是必需的（除非由于其他原因而需要此配置），为服务域和来宾域使用标准 MTU 值 1500 是最好的选择。

要改进网络性能，请将服务域和来宾域的 `extended-mapin-space` 属性设置为 `on`，这是 Oracle VM Server for SPARC 3.1 软件和受支持的系统固件的默认设置。

```
primary# ldm set-domain extended-mapin-space=on domain-name
```

要检查 `extended-mapin-space` 属性值，请运行以下命令：

```
primary# ldm ls -l domain-name |grep extended-mapin
extended-mapin-space=on
```

注 - 更改 extended-mapin-space 属性值会在 primary 域上触发延迟配置。这种情况要求重新引导 primary 域。在更改此属性值之前，您还必须停止来宾域。

虚拟交换机

虚拟交换机 (vsw) 是在服务域中运行的组件，由虚拟交换机驱动程序管理。可以将虚拟交换机连接到某些来宾域，以在这些域之间进行网络通信。此外，如果虚拟交换机还与物理网络接口相关联，则可以通过该物理网络接口在来宾域和物理网络之间进行网络通信。在 Oracle Solaris 10 服务域中运行时，虚拟交换机还具有网络接口 vsw*n*，该接口允许服务域与连接到该虚拟交换机的其他域通信。可以像使用任何常规网络接口一样使用虚拟交换机，并使用 Oracle Solaris 10 ifconfig 命令对其进行配置。

向域分配虚拟网络设备会创建对提供虚拟交换机的域的隐式依赖关系。可以使用 ldm list-dependencies 命令查看这些依赖关系或者查看依赖此虚拟交换机的域。请参见“列出域 I/O 依赖关系” [320]。

在 Oracle Solaris 11 服务域中，虚拟交换机无法用作常规网络接口。如果虚拟交换机连接到物理网络接口，可以使用此物理接口与服务域进行通信。如果在没有物理接口的情况下进行配置，可以通过将 etherstub 用作与 VNIC 连接的网络设备 (net-dev) 来启用与服务域的通信。

要确定将哪个网络设备用作虚拟交换机的后端设备，请在 dladm show-phys 输出中搜索物理网络设备或使用 ldm list-netdev 命令列出逻辑域的网络设备。

注 - 将虚拟交换机添加到 Oracle Solaris 10 服务域时，并未创建其网络接口。因此，默认情况下，服务域无法与连接到其虚拟交换机的来宾域通信。要在来宾域和服务域之间实现网络通信，必须在服务域中创建并配置关联的虚拟交换机的网络接口。有关说明，请参见“启用控制/服务域与其他域之间的联网（仅限 Oracle Solaris 10）” [44]。

此情况仅会在 Oracle Solaris 10 OS 中发生，不会在 Oracle Solaris 11 OS 中发生。

可以使用 ldm add-vsw、ldm set-vsw 和 ldm rm-vsw 命令分别向域中添加虚拟交换机、为虚拟交换机设置选项以及删除虚拟交换机。请参见 ldm(1M) 手册页。

在具有 VLAN 标记的 NIC 或聚合实例上创建虚拟交换机时，必须在 ldm add-vsw 或 ldm set-vsw 命令中指定 NIC (nxge0)、聚合 (aggr3) 或虚名 (net0) 作为 net-dev 属性的值。

注 - 从 Oracle Solaris 11.2.1.0.0 (SRU 1) OS 开始，可以使用 ldm set-vsw 命令动态更新 net-dev 属性值。在以前的 Oracle Solaris OS 发行版中，使用 ldm set-vsw 命令更新 primary 域中的 net-dev 属性值会导致 primary 域进入延迟重新配置。

不能在 InfiniBand IP-over-InfiniBand (IPoIB) 网络设备之上添加虚拟交换机。尽管 `ldm add-vsw` 和 `ldm add-vnet` 命令似乎已执行成功，但由于这些设备通过 InfiniBand 传输层来传输 IP 包，因此不会传输任何数据。虚拟交换机仅支持以太网作为传输层。

以下示例说明如何在物理网络适配器上创建虚拟交换机：

- Oracle Solaris 10 OS：以下命令会在名为 `nxge0` 的物理网络适配器上创建虚拟交换机：

  ```
  primary# ldm add-vsw net-dev=nxge0 primary-vsw0 primary
  ```

  有关将虚拟交换机配置为网络接口的更多信息，请参见“启用控制/服务域与其他域之间的联网（仅限 Oracle Solaris 10）” [44]。

- Oracle Solaris 11 OS：以下命令会在名为 `net0` 的物理网络适配器上创建虚拟交换机：

  ```
  primary# ldm add-vsw net-dev=net0 primary-vsw0 primary
  ```

  以下示例使用 `ldm list-netdev -b` 命令仅显示 `svcdom` 服务域的有效虚拟交换机后端设备。

  ```
  primary# ldm list-netdev -b svcdom
  DOMAIN
  svcdom

  NAME             CLASS MEDIA STATE   SPEED OVER   LOC
  ----             ----- ----- -----   ----- ----   ---
  net0             PHYS  ETHER up      10000 ixgbe0 /SYS/MB/RISER1/PCIE
  net1             PHYS  ETHER unknown 0     ixgbe1 /SYS/MB/RISER1/PCIE4
  net2             ESTUB ETHER unknown 0     --     --
  net3             ESTUB ETHER unknown 0     --     --
  ldoms-estub.vsw0 ESTUB ETHER unknown 0     --     --
  ```

虚拟网络设备

虚拟网络设备是指在连接到虚拟交换机的域中定义的虚拟设备。虚拟网络设备由虚拟网络驱动程序管理，并且使用逻辑域通道 (logical domain channel, LDC) 通过虚拟机管理程序连接到虚拟网络。

虚拟网络设备可用作名为 `vnet`*n* 的网络接口，可以像使用任何常规网络接口一样使用该接口，并使用 Oracle Solaris 10 `ifconfig` 命令或 Oracle Solaris 11 `ipadm` 命令对其进行配置。

注 - 对于 Oracle Solaris 11，会为设备分配通用名称，因此 `vnet`*n* 将使用通用名称（例如 `net0`）。

可以使用 `ldm add-vnet`、`ldm set-vnet` 和 `ldm rm-vnet` 命令分别向域中添加虚拟网络设备、为现有的虚拟网络设备设置选项以及删除虚拟网络设备。请参见 `ldm`(1M) 手册页。

有关 Oracle Solaris 10 和 Oracle Solaris 11 中 Oracle VM Server for SPARC 联网的信息，请分别参见图 11-1 “Oracle Solaris 10 OS 的 Oracle VM Server for SPARC 网络概述” 和图 11-2 “Oracle Solaris 11 OS 的 Oracle VM Server for SPARC 网络概述”。

Inter-Vnet LDC 通道

默认情况下，Logical Domains Manager 将按以下方式分配 LDC 通道：

- 在虚拟网络设备和虚拟交换机设备之间分配一个 LDC 通道。
- 在连接到同一个虚拟交换机设备的每对虚拟网络设备之间分配一个 LDC 通道 (Inter-Vnet)。

配置 Inter-Vnet LDC 通道的目的在于，使虚拟网络设备之间可以直接通信，以实现较高的来宾域到来宾域通信性能。但是，随着虚拟交换机设备中虚拟网络设备数量的增加，所需的用于 Inter-Vnet 通信的 LDC 通道的数量也会呈指数增长。

您可以选择针对连接到给定虚拟交换机设备的所有虚拟网络设备，启用或禁用 Inter-Vnet LDC 通道分配。通过禁用此分配，可以降低 LDC 通道的使用量（用数字进行限制）。

在以下情况下禁用此分配非常有用：

- 当来宾域到来宾域通信性能不是特别重要时
- 当虚拟交换机设备中需要大量虚拟网络设备时

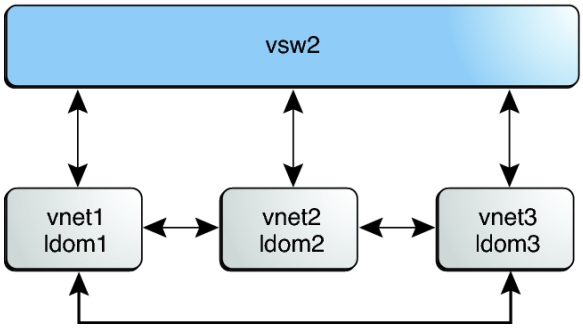
如果不分配 Inter-Vnet 通道，则可以使用更多的 LDC 通道向来宾域中添加更多的虚拟 I/O 设备。

注 - 如果来宾域到来宾域通信性能比增加系统中虚拟网络设备的数量重要，则不要禁用 Inter-Vnet LDC 通道分配。

可以使用 `ldm add-vsw` 和 `ldm set-vsw` 命令为 `inter-vnet-link` 属性指定 `on` 或 `off` 值。

下图显示的是具有三个虚拟网络设备的典型的虚拟交换机。`inter-vnet-link` 属性设置为 `on`，这意味着分配了 Inter-Vnet LDC 通道。`vnet1` 和 `vnet2` 之间的来宾域到来宾域通信是直接执行的，不需要经过虚拟交换机。

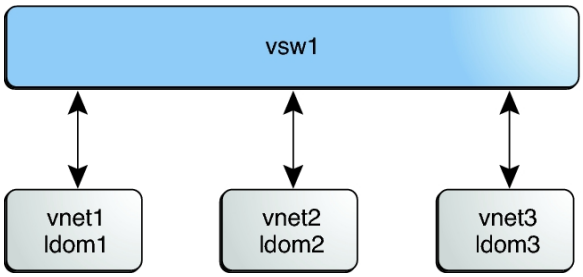
图 11-3 使用 Inter-Vnet 通道的虚拟交换机配置



下图显示与上图相同的虚拟交换机配置，但是 `inter-vnet-link` 属性设置为 `off`。不会分配 Inter-Vnet LDC 通道。与 `inter-vnet-link` 属性设置为 `on` 时相比，使用的 LDC 通道更少。在该配置下，`vnet1` 和 `vnet2` 之间的来宾域到来宾域通信必须经过 `vsw1`。

注 - 禁用 Inter-Vnet LDC 通道分配不会禁止来宾域到来宾域通信。所有的来宾域到来宾域通信都经过虚拟交换机，而不是从一个来宾域直接到达另一个来宾域。

图 11-4 不使用 Inter-Vnet 通道的虚拟交换机配置



查看网络设备配置和统计数据

使用 `ldm list-netdev` 和 `ldm list-netstat` 命令，可以分别查看系统中网络设备的信息和网络统计数据。因此，您可以集中了解给定物理域中的网络设备和统计数据。

要使用这些命令，必须至少在来宾域中运行 Oracle Solaris 11.2 SRU 1 OS。

例 11-1 列出网络设备配置信息

以下示例使用 `ldm list-netdev` 命令显示 `ldg1` 域的网络设备的短列表。

```
primary# ldm list-netdev ldg1

DOMAIN
ldg1

NAME              CLASS MEDIA STATE   SPEED OVER      LOC
----              ----- ----- -----   ----- ----      ---
net0              VNET  ETHER up      0     --        primary-vsw0/vne t0_ldg1
net3              PHYS  ETHER up      10000 --        /SYS/MB/RISER1/PCIE4
net4              VSW   ETHER up      10000 --        ldg1-vsw1
net1              PHYS  ETHER up      10000 --        /SYS/MB/RISER1/PCIE4
net5              VNET  ETHER up      0     --        ldg1-vsw1/vnet1_ldg1
net6              VNET  ETHER up      0     --        ldg1-vsw1/vnet2_ldg1
aggr2             AGGR  ETHER unknown 0     net1,net3 --
ldoms-vsw0.vport3 VNIC  ETHER unknown 0     --        ldg1-vsw1/vnet2_ldg1
ldoms-vsw0.vport2 VNIC  ETHER unknown 0     --        ldg1-vsw1/vnet1_ldg1
ldoms-vsw0.vport1 VNIC  ETHER unknown 0     --        ldg1-vsw1/vnet2_ldg3
ldoms-vsw0.vport0 VNIC  ETHER unknown 0     --        ldg1-vsw1/vnet2_ldg2
```

例 11-2 列出详细网络设备配置信息

以下示例使用 `ldm list-netdev -l` 命令显示 `ldg1` 域的网络设备的详细列表。

```
primary# ldm list-netdev -l ldg1

-------------------------------------------------------------------
DOMAIN
ldg1

NAME              CLASS    MEDIA    STATE   SPEED OVER LOC
----              -----    -----    -----   ----- ---- ---
net0              VNET     ETHER    up      0     --   primary-vsw0/vnet0_ldg1
    [/virtual-devices@100/channel-devices@200/network@0]
    MTU       : 1500 [1500-1500]
    IPADDR    : 10.129.241.200/255.255.255.0
    MAC_ADDRS : 00:14:4f:fb:9c:df

net3              PHYS     ETHER    up      10000 --   /SYS/MB/RISER1/PCIE4
    [/pci@400/pci@1/pci@0/pci@0/network@0]
    MTU       : 1500 [576-15500]
    MAC_ADDRS : a0:36:9f:0a:c5:d2

net4              VSW      ETHER    up      10000 --   ldg1-vsw1
    [/virtual-devices@100/channel-devices@200/virtual-network-switch@0]
    MTU       : 1500 [1500-1500]
    IPADDR    : 192.168.1.2/255.255.255.0
```

```
    MAC_ADDRS : 00:14:4f:fb:61:6e

net1              PHYS     ETHER    up      10000 --   /SYS/MB/RISER1/PCIE4
    [/pci@400/pci@1/pci@0/pci@0/network@0,1]
    MTU       : 1500 [576-15500]
    MAC_ADDRS : a0:36:9f:0a:c5:d2

net5              VNET     ETHER    up      0     --   ldg1-vsw1/vnet1_ldg1
    [/virtual-devices@100/channel-devices@200/network@1]
    MTU       : 1500 [1500-1500]
    IPADDR    : 0.0.0.0  /255.0.0.0
              : fe80::214:4fff:fef8:5062/ffc0::
    MAC_ADDRS : 00:14:4f:f8:50:62

net6              VNET     ETHER    up      0     --   ldg1-vsw1/vnet2_ldg1
    [/virtual-devices@100/channel-devices@200/network@2]
    MTU       : 1500 [1500-1500]
    IPADDR    : 0.0.0.0  /255.0.0.0
              : fe80::214:4fff:fef8:af92/ffc0::
    MAC_ADDRS : 00:14:4f:f8:af:92

aggr2             AGGR     ETHER    unknown  0    net1,net3 --
    MODE      : TRUNK
    POLICY    : L2,L3
    LACP_MODE : ACTIVE
    MEMBER    : net1 [PORTSTATE = attached]
    MEMBER    : net3 [PORTSTATE = attached]
    MAC_ADDRS : a0:36:9f:0a:c5:d2

ldoms-vsw0.vport3  VNIC     ETHER    unknown  0    -- ldg1-vsw1/vnet2_ldg1
    MTU       : 1500 [576-1500]
    MAC_ADDRS : 00:14:4f:f8:af:92

ldoms-vsw0.vport2  VNIC     ETHER    unknown  0    -- ldg1-vsw1/vnet1_ldg1
    MTU       : 1500 [576-1500]
    MAC_ADDRS : 00:14:4f:f8:50:62

ldoms-vsw0.vport1  VNIC     ETHER    unknown  0    -- ldg1-vsw1/vnet2_ldg3
    MTU       : 1500 [576-1500]
    MAC_ADDRS : 00:14:4f:f9:d3:88

ldoms-vsw0.vport0  VNIC     ETHER    unknown  0    -- ldg1-vsw1/vnet2_ldg2
    MTU       : 1500 [576-1500]
    MAC_ADDRS : 00:14:4f:fa:47:f4
              : 00:14:4f:f9:65:b5
              : 00:14:4f:f9:60:3f
```

例 11-3 列出网络设备统计数据

`ldm list-netstat` 命令显示系统中一个或多个域的网络统计数据。

以下示例显示系统中所有域的默认网络统计数据。

```
primary# ldm list-netstat

DOMAIN
primary

NAME               IPACKETS     RBYTES       OPACKETS     OBYTES
----               --------     ------       --------     ------
net3               0            0            0            0
net0               2.72M        778.27M      76.32K       6.01M
net4               2.72M        778.27M      76.32K       6.01M
net6               2            140          1.30K        18.17K
net7               0            0            0            0
net2               0            0            0            0
net1               0            0            0            0
aggr1              0            0            0            0
ldoms-vsw0.vport0  935.40K      74.59M       13.15K       984.43K
ldoms-vsw0.vport1  933.26K      74.37M       11.42K       745.15K
ldoms-vsw0.vport2  933.24K      74.37M       11.46K       747.66K
ldoms-vsw1.vport1  202.26K      17.99M       179.75K      15.69M
ldoms-vsw1.vport0  202.37K      18.00M       189.00K      16.24M
------------------------------------------------------------------
DOMAIN
ldg1

NAME               IPACKETS     RBYTES       OPACKETS     OBYTES
----               --------     ------       --------     ------
net0               5.19K        421.57K      68           4.70K
net3               0            0            2.07K        256.93K
net4               0            0            4.37K        560.17K
net1               0            0            2.29K        303.24K
net5               149          31.19K       78           17.00K
net6               147          30.51K       78           17.29K
aggr2              0            0            0            0
ldoms-vsw0.vport3  162          31.69K       52           14.11K
ldoms-vsw0.vport2  163          31.74K       51           13.76K
ldoms-vsw0.vport1  176          42.99K       25           1.50K
ldoms-vsw0.vport0  158          40.19K       45           4.42K
------------------------------------------------------------------
DOMAIN
ldg2

NAME               IPACKETS     RBYTES       OPACKETS     OBYTES
----               --------     ------       --------     ------
net0               5.17K        418.90K      71           4.88K
net1               2.70K        201.67K      2.63K        187.01K
net2               132          36.40K       1.51K        95.07K
------------------------------------------------------------------
DOMAIN
ldg3

NAME               IPACKETS     RBYTES       OPACKETS     OBYTES
----               --------     ------       --------     ------
net0               5.16K        417.43K      72           4.90K
net1               2.80K        206.12K      2.67K        190.36K
```

```
net2              118          35.00K       1.46K        87.78K
```

控制由虚拟网络设备使用的物理网络带宽量

使用带宽资源控制功能，您可以限制虚拟网络设备使用的物理网络带宽。至少运行 Oracle Solaris 11 OS 并且配置有虚拟交换机的服务域支持此功能。Oracle Solaris 10 服务域会忽略网络带宽设置且不做提示。此功能可确保一个来宾域不会占用所有可用的物理网络带宽而不给其他域留下任何带宽。

可以使用 `ldm add-vnet` 和 `ldm set-vnet` 命令通过为 `maxbw` 属性提供值指定带宽限制。可以使用 `ldm list-bindings` 或 `ldm list-domain -o network` 命令查看现有虚拟网络设备的 `maxbw` 属性值。最小带宽限制为 10 Mbps。

网络带宽限制

注 - 启用了混合 I/O 的虚拟网络设备不支持此功能。对于混合模式的虚拟网络，不会执行 `maxbw` 属性，因为混合 I/O 会分配一个特定的硬件资源单元，无法对该单元进行更改来限制带宽。要限制虚拟网络设备的带宽，必须禁用混合模式。

带宽资源控制仅应用于通过虚拟交换机的通信。因此，inter-vnet 通信不受此限制约束。如果您未配置物理后端设备，则可以忽略带宽资源控制。

所支持的最小带宽限制取决于服务域中的 Oracle Solaris 网络堆栈。可以将带宽限制配置为所需要的任何较高值。没有上限。带宽限制只是确保带宽不超出所配置的值。因此，可以将带宽限制配置为一个比分配给虚拟交换机的物理网络设备的链路速度更高的值。

设置网络带宽限制

可以使用 `ldm add-vnet` 命令创建虚拟网络设备并通过为 `maxbw` 属性提供值指定带宽限制。

```
primary# ldm add-vnet maxbw=limit if-name vswitch-name domain-name
```

可以使用 `ldm set-vnet` 命令为现有的虚拟网络设备指定带宽限制。

```
primary# ldm set-vnet maxbw=limit if-name domain-name
```

还可以通过为 `maxbw` 属性指定一个空值来清除带宽限制：

```
primary# ldm set-vnet maxbw= if-name domain-name
```

以下示例展示了如何使用 ldm 命令指定带宽限制。带宽是作为带单位的整数指定的。单位为 M 表示兆比特/秒，单位为 G 表示千兆比特/秒。如果未指定单位，则单位为兆比特/秒。

例 11-4 在创建虚拟网络设备时设置带宽限制

以下命令创建带宽限制为 100 Mbps 的一个虚拟网络设备 (vnet0)。

```
primary# ldm add-vnet maxbw=100M vnet0 primary-vsw0 ldg1
```

如果尝试设置低于最小值 (10 Mbps) 的带宽限制，以下命令将发出一条错误消息。

```
primary# ldm add-vnet maxbw=1M vnet0 primary-vsw0 ldg1
```

例 11-5 在现有虚拟网络设备上设置带宽限制

以下命令在现有的 vnet0 设备上设置带宽限制 200 Mbps。

带宽量可能不会达到指定的限制 200 Mbps，具体取决于实时网络通信模式。例如，带宽可能为 95 Mbps，这没有超出 200 Mbps 限制。

```
primary# ldm set-vnet maxbw=200M vnet0 ldg1
```

以下命令在现有的 vnet0 设备上设置带宽限制 2 Gbps。

因为 MAC 层中的带宽没有上限，所以可以将限制设置为 2 Gbps，即使底层物理网络速度低于 2 Gbps 也是如此。在这种情况下，带宽限制不会有任何效果。

```
primary# ldm set-vnet maxbw=2G vnet0 ldg1
```

例 11-6 清除现有虚拟网络设备上的带宽限制

以下命令清除指定的虚拟网络设备 (vnet0) 上的带宽限制。通过清除此值，虚拟网络设备将使用底层物理设备提供的最大可用带宽。

```
primary# ldm set-vnet maxbw= vnet0 ldg1
```

例 11-7 查看现有虚拟网络设备的带宽限制

ldm list-bindings 命令显示指定虚拟网络设备的 maxbw 属性的值（如果已定义）。

以下命令显示 vnet0 虚拟网络设备的带宽限制为 15 Mbps。如果未设置带宽限制，则 MAXBW 字段为空。

```
primary# ldm list-bindings
...
VSW
  NAME         MAC               NET-DEV   ID   DEVICE   LINKPROP
   primary-vsw0 00:14:4f:f9:95:97 net0      0    switch@0 1
```

```
   DEFAULT-VLAN-ID PVID VID       MTU   MODE   INTER-VNET-LINK
   1               1              1500         on

   PEER       MAC               PVID VID MTU  MAXBW  LINKPROP INTERVNETLINK
   vnet0@ldg1 00:14:4f:fb:b8:c8 1        1500 15

...

   NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
   ldg1             bound      ------  5000    8     2G

   NETWORK
      NAME             SERVICE                     ID   DEVICE
      vnet0            primary-vsw0@primary        0    network@0

      MAC               MODE   PVID VID     MTU   MAXBW      LINKPROP
      00:14:4f:fb:b8:c8 1                   1500  15

      PEER                        MAC               MODE   PVID VID
      primary-vsw0@primary        00:14:4f:f9:95:97  1

      MTU   MAXBW      LINKPROP
      1500
```

您还可以使用 `dladm show-linkprop` 命令查看 `maxbw` 属性值，如下所示：

```
primary# dladm show-linkprop -p maxbw
LINK              PROPERTY PERM VALUE  EFFECTIVE DEFAULT POSSIBLE
...
ldoms-vsw0.vport0 maxbw    rw   15     15        --      --
```

虚拟设备标识符和网络接口名称

将虚拟交换机或虚拟网络设备添加到域时，可以通过设置 `id` 属性来指定其设备编号。

```
primary# ldm add-vsw [id=switch-id] vswitch-name domain-name
primary# ldm add-vnet [id=network-id] if-name vswitch-name domain-name
```

域的每个虚拟交换机和虚拟网络设备都具有唯一的设备编号，该编号在绑定域时分配。如果使用显式设备编号（通过设置 `id` 属性）添加虚拟交换机或虚拟网络设备，将使用指定的设备编号。否则，系统将自动指定最低的可用设备编号。在这种情况下，分配的设备编号取决于将虚拟交换机或虚拟网络设备添加到系统的方式。当域被绑定时，在 `ldm list-bindings` 命令的输出中可以看到最终分配给虚拟交换机或虚拟网络设备的设备编号。

以下示例显示 `primary` 域有一个虚拟交换机 `primary-vsw0`。此虚拟交换机的设备编号是 `0` (`switch@0`)。

```
primary# ldm list-bindings primary
```

```
...
VSW
    NAME         MAC               NET-DEV DEVICE   DEFAULT-VLAN-ID PVID VID MTU MODE
    primary-vsw0 00:14:4f:fb:54:f2 nxge0   switch@0 1               1    5,6 1500
...
```

以下示例显示 ldg1 域有两个虚拟网络设备：vnet 和 vnet1。vnet 设备的设备编号是 0 (network@0)，vnet1 设备的设备编号是 1 (network@1)。

```
primary# ldm list-bindings ldg1
...
NETWORK
    NAME  SERVICE              DEVICE    MAC               MODE   PVID VID MTU
    vnet  primary-vsw0@primary network@0 00:14:4f:fb:e0:4b hybrid 1        1500
    ...
    vnet1 primary-vsw0@primary network@1 00:14:4f:f8:e1:ea        1        1500
...
```

同样，当含虚拟网络设备的域正在运行 Oracle Solaris OS 时，虚拟网络设备具有网络接口 vnet*N*。但是，虚拟网络设备的网络接口编号 *N* 不必与虚拟网络设备的设备编号 *n* 相同。

注 - 在 Oracle Solaris 11 系统上，会将 net*n* 形式的通用链路名称分配给 vsw*n* 和 vnet*n*。使用 dladm show-phys 命令确定哪些 net*n* 名称映射到哪些 vsw*n* 和 vnet*n* 设备。

注意 - Oracle Solaris OS 根据设备编号保留网络接口名称和虚拟交换机或虚拟网络设备之间的映射。如果未将设备编号显式分配给虚拟交换机或虚拟网络设备，当域被解除绑定并且之后再次绑定时，其设备编号可以更改。在这种情况下，由域中正在运行的 OS 分配的网络接口名称也可能会更改现有系统配置，并使现有系统配置无法使用。例如，在从域配置中删除虚拟交换机或虚拟网络接口时，可能会发生这种情况。

您不能使用 ldm list-* 命令直接确定与虚拟交换机或虚拟网络设备相对应的 Oracle Solaris OS 网络接口名称。但是，可以使用 ldm list -l 命令输出和 Oracle Solaris OS 的 /devices 下的条目的组合来获取此信息。

查找 Oracle Solaris OS 网络接口名称 (Oracle Solaris 11 OS)

在 Oracle Solaris 11 系统上，可以使用 ldm list-netdev 命令查找 Oracle Solaris OS 网络接口名称。有关更多信息，请参见 ldm(1M) 手册页。

以下示例显示 ldm list-netdev 和 ldm list -o network 命令。ldm list -o network 命令显示 NAME 字段中的虚拟网络设备。ldm list-netdev 输出显示 NAME 列中的对应 OS 接口名称。

```
primary# ldm list -o network ldg1
....
NETWORK
    NAME        SERVICE              ID DEVICE    MAC               MODE
    PVID VID MTU   MAXBW LINKPROP
    vnet0-ldg1  primary-vsw0@primary 0  network@0 00:14:4f:fa:eb:4e 1
             1500
    vnet1-ldg1  svcdom-vsw0@svcdom   1  network@1 00:14:4f:f8:53:45 4
             1500
           PVLAN :400,community

primary# ldm list-netdev ldg1
DOMAIN
ldg1

NAME CLASS MEDIA STATE   SPEED OVER  LOC
---- ----- ----- -----   ----- ----  ---
net0 VNET  ETHER up      0     vnet0 primary-vsw0/vnet0-ldg1
net1 VNET  ETHER up      0     vnet1 svcdom-vsw0/vnet1-ldg1
net2 VNET  ETHER unknown 0     vnet2 svcdom-vsw1/vnet2-ldg1
```

要验证 `ldm list-netdev` 输出是否正确，请从 `ldg1` 运行 `dladm show-phys` 和 `dladm show-linkprop -p mac-address` 命令：

```
ldg1# dladm show-phys
LINK     MEDIA     STATE      SPEED  DUPLEX         DEVICE
net0     Ethernet  up         0      unknown        vnet0
net1     Ethernet  up         0      unknown        vnet1
net2     Ethernet  unknown    0      unknown        vnet2

ldg1# dladm show-linkprop -p mac-address
LINK PROPERTY    PERM VALUE            EFFECTIVE        DEFAULT          POSSIBLE
net0 mac-address rw   0:14:4f:fa:eb:4e 0:14:4f:fa:eb:4e 0:14:4f:fa:eb:4e --
net1 mac-address rw   0:14:4f:f8:53:45 0:14:4f:f8:53:45 0:14:4f:f8:53:45 --
```

▼ 如何查找 Oracle Solaris OS 网络接口名称 (Oracle Solaris 10 OS)

此过程介绍如何在与 `net-c` 对应的 `ldg1` 中查找 Oracle Solaris OS 网络接口名称。此示例还显示了查找虚拟交换机而不是虚拟网络设备的网络接口名称的不同之处。在此示例过程中，来宾域 `ldg1` 包含两个虚拟网络设备：`net-a` 和 `net-c`。

1. **使用 `ldm` 命令查找 `net-c` 的虚拟网络设备编号。**

```
primary# ldm list -l ldg1
...
NETWORK
NAME         SERVICE                     DEVICE       MAC
net-a        primary-vsw0@primary        network@0    00:14:4f:f8:91:4f
```

```
net-c        primary-vsw0@primary        network@2    00:14:4f:f8:dd:68
...
```

net-c 的虚拟网络设备编号是 2 (network@2)。

要确定虚拟交换机的网络接口名称，请查找虚拟交换机设备编号，*n* 以 switch@*n* 表示。

2. 通过登录到 **ldg1** 并在 **/devices** 下查找此设备编号对应的条目来查找 **ldg1** 上的相应网络接口。

```
primary# uname -n
ldg1
primary# find /devices/virtual-devices@100 -type c -name network@2\*
/devices/virtual-devices@100/channel-devices@200/network@2:vnet1
```

网络接口名称是冒号后面的条目部分，即 vnet1。

要确定虚拟交换机的网络接口名称，请将 -name 选项的参数替换为 virtual-network-switch@*n**。然后，查找具有名称 vsw*N* 的网络接口。

3. 验证 **vnet1** 是否具有 MAC 地址 **00:14:4f:f8:dd:68**，如步骤 1 中 **net-c** 的 **ldm list -l** 输出中所示。

- Oracle Solaris 10 OS：

```
primary# ifconfig vnet1
vnet1: flags=1000842<BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 3
          inet 0.0.0.0 netmask 0
          ether 0:14:4f:f8:dd:68
```

- Oracle Solaris 11 OS：

  首先，您必须使用 dladm show-phys 命令确定为 vnet1 指定的接口名称。

```
primary# dladm show-phys |grep vnet1
net2              Ethernet             up         0      unknown   vnet1
```

  然后，使用以下命令确定 net2 的 MAC 地址。

```
primary# dladm show-linkprop -p mac-address net2
LINK PROPERTY    PERM VALUE             EFFECTIVE         DEFAULT POSSIBLE
net2 mac-address rw   00:14:4f:f8:dd:68 00:14:4f:f8:dd:68 --      --
```

自动或手动分配 MAC 地址

您必须有足够的介质访问控制 (media access control, MAC) 地址，以便分配给计划使用的一定数目的逻辑域、虚拟交换机和虚拟网络。可以让 Logical Domains Manager 自动为逻辑域、虚拟网络和虚拟交换机分配 MAC 地址，也可以从您自己

的已分配 MAC 地址池中手动分配 MAC 地址。设置 MAC 地址的 `ldm` 子命令是 `add-domain`、`add-vsw`、`set-vsw`、`add-vnet` 和 `set-vnet`。如果您未在这些子命令中指定 MAC 地址，Logical Domains Manager 会自动分配一个地址。

让 Logical Domains Manager 分配 MAC 地址的优点是，它可以使用专用于逻辑域的 MAC 地址块。此外，Logical Domains Manager 可以检测并防止 MAC 地址与同一子网中其他 Logical Domains Manager 实例发生冲突。这样，您就不必手动管理您的 MAC 地址池了。

创建逻辑域或将网络设备配置到域中时，即会分配 MAC 地址。此外，除非删除了设备或逻辑域本身，否则此地址分配是持久性的。

分配给域的 MAC 地址范围

已经为域分配了以下 512K MAC 地址块：

```
00:14:4F:F8:00:00 ~ 00:14:4F:FF:FF:FF
```

Logical Domains Manager 会使用较低的 256K 地址进行自动 MAC 地址分配，您不能手动请求此范围内的地址：

```
00:14:4F:F8:00:00 - 00:14:4F:FB:FF:FF
```

您可以使用此范围内的前一半地址来手动分配 MAC 地址：

```
00:14:4F:FC:00:00 - 00:14:4F:FF:FF:FF
```

注 - 在 Oracle Solaris 11 中，为 VNIC 分配 MAC 地址时使用的地址在这些范围之外。

自动分配算法

如果您未在创建逻辑域或网络设备时指定 MAC 地址，Logical Domains Manager 会自动为该逻辑域或网络设备分配 MAC 地址。

为了获得此 MAC 地址，Logical Domains Manager 会重复尝试选择地址，然后检查是否存在潜在冲突。从专为此目的留出的 256K 地址范围中随机选择 MAC 地址。随机选择 MAC 地址可减少重复的 MAC 地址被选作候选地址的几率。

随后将对照其他系统上的其他 Logical Domains Manager 检查所选的地址，以防止实际分配重复的 MAC 地址。“检测重复的 MAC 地址” [203]中介绍了采用的算法。如果该地址已分配，则 Logical Domains Manager 将重复操作，选择另一个地址，然后再次检查是否存在冲突。此过程将一直持续下去，直至找到尚未分配的 MAC 地址或超出 30 秒

的时间限制为止。如果达到了时间限制，则创建设备失败，并显示类似以下内容的错误消息。

```
Automatic MAC allocation failed.  Please set the vnet MAC address manually.
```

检测重复的 MAC 地址

为避免将同一个 MAC 地址分配给不同的设备，Logical Domains Manager 将与其他系统上的其他 Logical Domains Manager 进行核实，方法是通过控制域的默认网络接口发送多播消息，其中包括 Logical Domains Manager 希望分配给设备的地址。尝试分配 MAC 地址的 Logical Domains Manager 会等待一秒钟的时间，以便返回响应。如果该 MAC 地址已经分配给另一个已启用 Oracle VM Server for SPARC 的系统上的不同设备，则该系统上的 Logical Domains Manager 会发送一条响应，其中包含相关的 MAC 地址。如果发出请求的 Logical Domains Manager 收到响应，则它就会知道所选的 MAC 地址已分配，然后就会选择其他地址，并重复上述操作。

默认情况下，这些多播消息仅发送至同一子网中的其他管理器。默认生存时间 (time-to-live, TTL) 为 `1`。可以使用服务管理工具 (Service Management Facility, SMF) 属性 `ldmd/hops` 配置 TTL。

每个 Logical Domains Manager 均负责以下操作：

- 侦听多播消息
- 跟踪分配给域的 MAC 地址
- 查找重复项
- 为避免产生重复项而作出响应

如果系统上的 Logical Domains Manager 由于某种原因关闭，则在 Logical Domains Manager 关闭期间可能会产生重复的 MAC 地址。

创建逻辑域或网络设备时会执行自动 MAC 分配，而且自动 MAC 分配会一直保持到该设备或逻辑域被删除。

注 - 创建逻辑域或网络设备及启动逻辑域时，会执行检查重复 MAC 地址的检测。

将网络适配器和域结合使用

在 Oracle Solaris 10 逻辑域环境中，服务域中运行的虚拟交换机服务可以直接与符合 GLDv3 的网络适配器进行交互。虽然可以在这些系统中使用不符合 GLDv3 的网络适配器，但是虚拟交换机不能与这些网络适配器直接进行交互。有关如何使用不符合 GLDv3 的网络适配器的信息，请参见“针对 NAT 和路由配置虚拟交换机和服务域” [204]。

注 - Oracle Solaris 11 环境不存在 GLDv3 符合性问题。

有关使用链路聚合的更多信息，请参见“将链路聚合和虚拟交换机结合使用” [225]。

▼ 如何确定网络适配器是否符合 GLDv3 (Oracle Solaris 10)

此过程仅适用于 Oracle Solaris 10 域。

- **确定网络适配器是否符合 GLDv3。**
  以下示例使用 `bge0` 作为网络设备名称。

```
primary# dladm show-link bge0
bge0            type: non-vlan   mtu: 1500      device: bge0
```

`type`: 字段的值为以下其中一项：

- 符合 GLDv3 的驱动程序类型为 `non-vlan` 或 `vlan`。
- 不符合 GLDv3 的驱动程序类型为 `legacy`。

针对 NAT 和路由配置虚拟交换机和服务域

在 Oracle Solaris 10 OS 中，虚拟交换机 (`vsw`) 是第 2 层交换机，也可用作服务域中的网络设备。可以将虚拟交换机配置为仅充当不同逻辑域中的虚拟网络设备之间的交换机，但不能通过物理设备与外界网络进行连接。在这种模式下，如果将 `vsw` 作为网络设备创建并在服务域中启用 IP 路由，则虚拟网络可以通过将服务域用作路由器来与外界进行通信。如果物理网络适配器不符合 GLDv3，则要与域建立外部连接，必须使用这种操作模式。

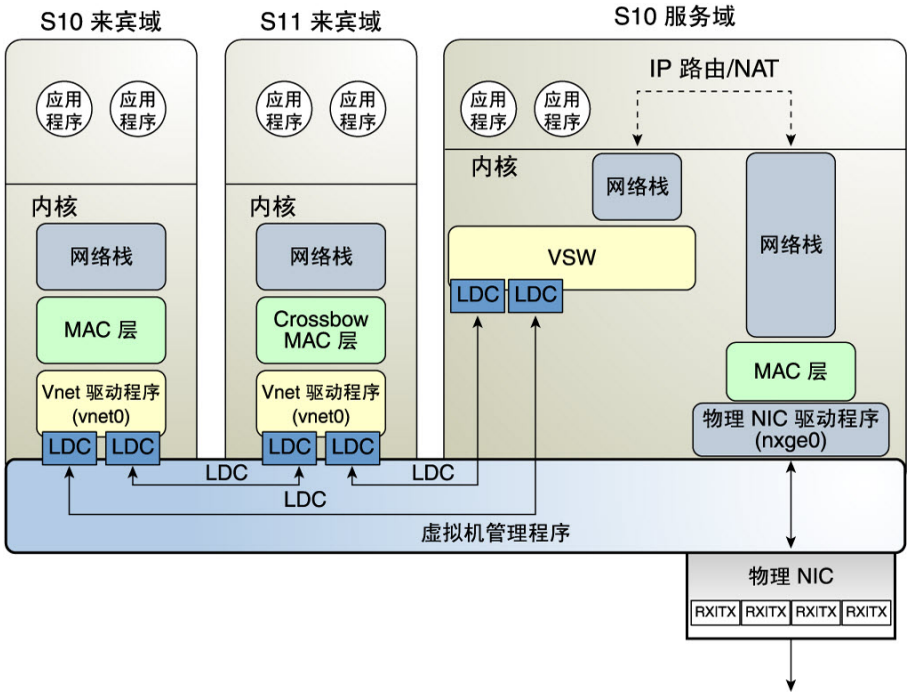
此配置的优势如下：

- 虚拟交换机不需要直接使用物理设备，即使底层设备不是符合 GLDv3 的设备，也能提供外部连接。
- 此配置可以利用 Oracle Solaris OS 的 IP 路由和过滤功能。

在 Oracle Solaris 10 系统上配置 NAT

下图显示虚拟交换机如何可用于在服务域中配置网络地址转换 (Network Address Translation, NAT)，以便为来宾域提供外部连接。

图 11-5 虚拟网络路由



▼ 如何设置虚拟交换机以为域提供外部连接 (Oracle Solaris 10)

1. **创建不具有关联的物理设备的虚拟交换机。**
   如果要分配地址，请确保虚拟交换机具有唯一的 MAC 地址。

   ```
   primary# ldm add-vsw [mac-addr=XX:XX:XX:XX:XX:XX] primary-vsw0 primary
   ```

2. **将虚拟交换机作为域所使用的物理网络设备之外的网络设备创建。**
   有关创建虚拟交换机的更多信息，请参见如何将虚拟交换机配置为主接口 [44]。

3. **如果需要，请为虚拟交换机设备配置 DHCP。**
   有关为虚拟交换机设备配置 DHCP 的更多信息，请参见如何将虚拟交换机配置为主接口 [44]。

4. **如果需要，请创建 `/etc/dhcp.vsw` 文件。**

5. 在服务域中配置 IP 路由，并在所有域中设置所需的路由表。

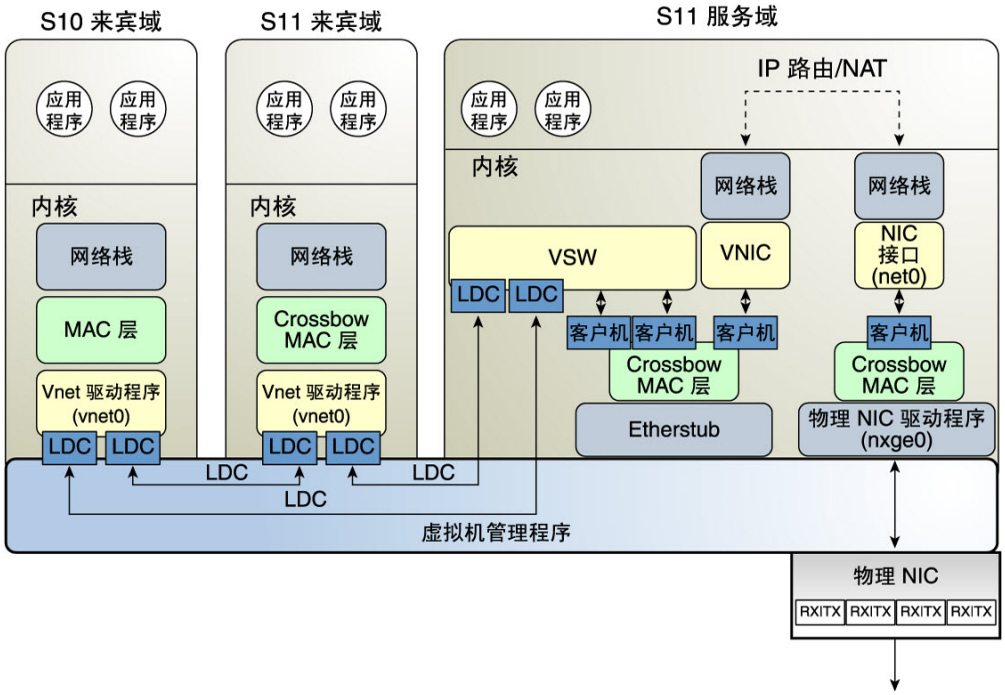
   有关 IP 路由的更多信息，请参见《Oracle Solaris Administration: IP Services 》中的“Packet Forwarding and Routing on IPv4 Networks”。

在 Oracle Solaris 11 系统上配置 NAT

Oracle Solaris 11 网络虚拟化功能包括 `etherstub`（一个伪网络设备）。此设备提供类似于物理网络设备的功能，但仅用于与其客户机进行专用通信。此伪设备可用作在虚拟网络之间提供专用通信的虚拟交换机的网络后端设备。使用 `etherstub` 设备作为后端设备时，来宾域也可与相同 `etherstub` 设备上的 VNIC 进行通信。通过以此方式使用 `etherstub` 设备，来宾域可与服务域中的区域进行通信。使用 `dladm create-etherstub` 命令可创建 `etherstub` 设备。

下图显示虚拟交换机、`etherstub` 设备和 VNIC 如何可用于在服务域中设置网络地址转换 (Network Address Translation, NAT)。

图 11-6 虚拟网络路由



可以考虑使用持久性路由。有关更多信息，请参见《Troubleshooting Network Administration Issues in Oracle Solaris 11.2 》中的“Troubleshooting Issues When Adding a Persistent Route”和《Configuring and Administering Network Components in Oracle Solaris 11.2 》中的“Creating Persistent (Static) Routes”。

▼ 如何设置虚拟交换机以为域提供外部连接 (Oracle Solaris 11)

1. 创建 Oracle Solaris 11 **etherstub** 设备。

```
primary# dladm create-etherstub stub0
```

2. 创建使用 **stub0** 作为物理后端设备的虚拟交换机。

```
primary# ldm add-vsw net-dev=stub0 primary-stub-vsw0 primary
```

3. 在 **stub0** 设备上创建 VNIC。

```
primary# dladm create-vnic -l stub0 vnic0
```

4. 配置 **vnic0** 作为网络接口。

```
primary# ipadm create-ip vnic0
primary# ipadm create-addr -T static -a 192.168.100.1/24  vnic0/v4static
```

5. 启用 IPv4 转发并创建 NAT 规则。
请参见《Connecting Systems Using Fixed Network Configuration in Oracle Solaris 11.1 》中的“Setting IP Interface Properties”和《Oracle Solaris Administration: IP Services 》中的“Packet Forwarding and Routing on IPv4 Networks”。

在 Oracle VM Server for SPARC 环境中配置 IPMP

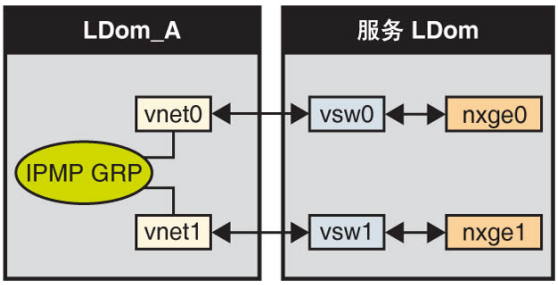
Oracle VM Server for SPARC 软件支持对虚拟网络设备使用基于链路的 IP 网络多路径 (IP network multipathing, IPMP)。配置具有虚拟网络设备的 IPMP 组时，请将该组配置为使用基于链路的检测。如果使用较早版本的 Oracle VM Server for SPARC (Logical Domains) 软件，则只能对虚拟网络设备配置基于探测的检测。

在域中将虚拟网络设备配置到 IPMP 组中

下图显示了两个虚拟网络（`vnet0` 和 `vnet1`），它们分别连接到服务域中的不同虚拟交换机实例（`vsw0` 和 `vsw1`），而这两个虚拟交换机实例又使用两个不同的物理接口。这两个物理接口在 Oracle Solaris 10 中为 `nxge0` 和 `nxge1`，在 Oracle Solaris 11 中为 `net0` 和 `net1`。图中显示了 Oracle Solaris 10 物理接口名称。

如果服务域中发生物理链路故障，则绑定到该物理设备的虚拟交换机设备会检测到链路故障。然后，虚拟交换机设备将故障传播到绑定到此虚拟交换机的相应虚拟网络设备。虚拟网络设备将此链路事件的通知发送到来宾 LDom_A 中的 IP 层，从而导致故障转移到 IPMP 组中的其他虚拟网络设备。

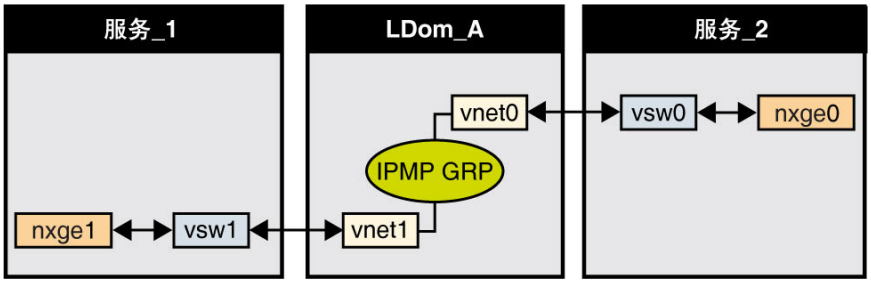
图 11-7 两个连接到不同虚拟交换机实例的虚拟网络



注 - 此图显示了 Oracle Solaris 10 系统中的配置。对于 Oracle Solaris 11 系统，只有接口名称更改为使用通用名称（例如，nxge0 和 nxge1 分别更改为 net0 和 net1）。

通过将每个虚拟网络设备（vnet0 和 vnet1）连接到不同服务域中的虚拟交换机实例，可提高逻辑域中的可靠性（如下图所示）。这种情况下，除了物理网络故障外，LDom_A 可以检测到虚拟网络故障，并在服务域崩溃或关闭后触发故障转移。

图 11-8 连接到不同服务域的各个虚拟网络设备



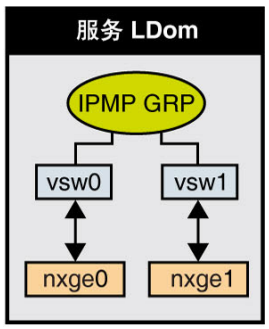
注 - 此图显示了 Oracle Solaris 10 系统中的配置。对于 Oracle Solaris 11 系统，只有接口名称更改为使用通用名称（例如，nxge0 和 nxge1 分别更改为 net0 和 net1）。

有关更多信息，请参见 Oracle Solaris 10 《Oracle Solaris Administration: IP Services》或 Oracle Solaris 11.1 Information Library（Oracle Solaris 11.1 信息库）中的 “Establishing an Oracle Solaris Network”（建立 Oracle Solaris 网络）。

在服务域中配置并使用 IPMP

可以通过将多个虚拟交换机接口配置为一个组来在服务域中配置 IPMP。下图显示了两个绑定到两个不同物理设备的虚拟交换机实例（`vsw0` 和 `vsw1`）。随后可以创建这两个虚拟交换机接口并将其配置到 IPMP 组中。如果物理链路出现故障，则绑定到此物理设备的虚拟交换机设备会检测到链路故障。然后，虚拟交换机设备将此链路事件的通知发送到服务域中的 IP 层，从而导致故障转移到 IPMP 组中的其他虚拟交换机设备。两个物理接口在 Oracle Solaris 10 中为 `nxge0` 和 `nxge1`，在 Oracle Solaris 11 中为 `net0` 和 `net1`。图中显示了 Oracle Solaris 10 物理接口名称。

图 11-9 两个配置为属于 IPMP 组的虚拟交换机接口



注 - 该图显示了 Oracle Solaris 10 系统中的配置。对于 Oracle Solaris 11 系统，只有接口名称更改为使用通用名称（例如，`nxge0` 和 `nxge1` 分别更改为 `net0` 和 `net1`）。

在 Oracle VM Server for SPARC 虚拟网络中使用基于链路的 IPMP

虚拟网络设备和虚拟交换机设备支持网络堆栈的链路状态更新。默认情况下，虚拟网络设备会报告其虚拟链路（到虚拟交换机的 LDC）的状态。默认情况下将启用此配置，不需要您执行其他配置步骤。

有时可能需要检测物理网络链路状态更改。例如，如果已将物理设备分配给虚拟交换机，即使从虚拟网络设备到其虚拟交换机设备的链路是连通的，从服务域到外部网络的物理网络链路也可能断开。在这种情况下，可能需要获取物理链路状态并将其报告给虚拟网络设备及其堆栈。

可以使用 `linkprop=phys-state` 选项为虚拟网络设备以及虚拟交换机设备配置物理链路状态跟踪。如果启用此选项，则当虚拟设备（虚拟网络或虚拟交换机）作为域中的接口创建时，它会根据物理链路状态报告其链路状态。您可以使用标准 Oracle Solaris 网络管理命令（例如 `dladm` 和 `ifconfig`）来检查链路状态。此外，链路状态也记录在 `/var/adm/messages` 文件中。

对于 Oracle Solaris 10，请参见 `dladm`(1M) 和 `ifconfig`(1M) 手册页。对于 Oracle Solaris 11，请参见 `dladm`(1M)、`ipadm`(1M) 和 `ipmpstat`(1M) 手册页。

注 - 您可以在 Oracle VM Server for SPARC 系统上同时运行不识别链路状态和识别链路状态的 `vnet` 和 `vsw` 驱动程序。但是，如果打算配置基于链路的 IPMP，则必须安装识别链路状态的驱动程序。如果打算启用物理链路状态更新，请同时将 `vnet` 和 `vsw` 驱动程序升级到 Oracle Solaris 10 1/13 OS，并且至少运行 Logical Domains Manager 1.3 版。

▼ 如何配置物理链路状态更新

此过程展示如何为虚拟网络设备启用物理链路状态更新。

也可以按照类似的步骤操作并指定 `ldm add-vsw` 和 `ldm set-vsw` 命令的 `linkprop=phys-state` 选项来启用虚拟交换机设备的物理链路状态更新。

注 - 仅当虚拟交换机设备本身作为接口创建时，才需要使用 `linkprop=phys-state` 选项。如果指定了 `linkprop=phys-state` 且物理链路断开，则虚拟网络设备会将其链路状态报告为断开，即使与虚拟交换机的连接是连通的也是如此。由于 Oracle Solaris OS 当前未提供接口来报告两种不同的链路状态，例如虚拟链路状态和物理链路状态，因此会发生这种情况。

---

1. 成为管理员。

   - 对于 Oracle Solaris 10，请参见《System Administration Guide: Security Services》中的“Configuring RBAC (Task Map)”。
   - 对于 Oracle Solaris 11.2，请参见《Securing Users and Processes in Oracle Solaris 11.2》中的第 1 章 “About Using Rights to Control Users and Processes”。

2. 启用虚拟设备的物理链路状态更新。

   您可以通过以下方式来启用虚拟网络设备的物理链路状态更新：

- 运行 `ldm add-vnet` 命令时，通过指定 `linkprop=phys-state` 创建虚拟网络设备。

  指定 `linkprop=phys-state` 选项可配置虚拟网络设备以获取物理链路状态更新并将其报告给堆栈。

注 - 如果指定了 `linkprop=phys-state` 且物理链路断开（即使与虚拟交换机的连接是连通的），则虚拟网络设备会将其链路状态报告为断开。由于 Oracle Solaris OS 当前未提供接口来报告两种不同的链路状态，例如虚拟链路状态和物理链路状态，因此会发生这种情况。

```
primary# ldm add-vnet linkprop=phys-state if-name vswitch-name domain-name
```

以下示例启用在逻辑域 `ldom1` 上连接到 `primary-vsw0` 的 `ldom1_vnet0` 的物理链路状态更新：

```
primary# ldm add-vnet linkprop=phys-state ldom1_vnet0 primary-vsw0 ldom1
```

- 运行 `ldm set-vnet` 命令时，通过指定 `linkprop=phys-state` 修改现有虚拟网络设备。

```
primary# ldm set-vnet linkprop=phys-state if-name domain-name
```

以下示例启用逻辑域 `ldom1` 上 `vnet0` 的物理链路状态更新：

```
primary# ldm set-vnet linkprop=phys-state ldom1_vnet0 ldom1
```

要禁用物理链路状态更新，请运行 `ldm set-vnet` 命令指定 `linkprop=`。

以下示例禁用逻辑域 `ldom1` 上 `ldom1_vnet0` 的物理链路状态更新：

```
primary# ldm set-vnet linkprop= ldom1_vnet0 ldom1
```

例 11-8 配置基于链路的 IPMP

以下示例说明如何在启用和不启用物理链路状态更新的情况下配置基于链路的 IPMP：

- 以下示例在域中配置两个虚拟网络设备。每个虚拟网络设备连接到服务域中的独立虚拟交换机设备，以使用基于链路的 IPMP。

注 - 未在这些虚拟网络设备上配置测试地址。此外，使用 `ldm add-vnet` 命令创建这些虚拟网络设备时，您不需要执行其他配置操作。

下列命令将虚拟网络设备添加到域。请注意，由于未指定 `linkprop=phys-state`，因此只监视与虚拟交换机链路的状态更改。

```
primary# ldm add-vnet ldom1_vnet0 primary-vsw0 ldom1
primary# ldm add-vnet ldom1_vnet1 primary-vsw1 ldom1
```

以下命令在来宾域中配置虚拟网络设备并将其分配给 IPMP 组。请注意，未在这些虚拟网络设备上配置测试地址，原因是正在使用基于链路的故障检测。

- Oracle Solaris 10 OS：使用 `ifconfig` 命令。

```
primary# ifconfig vnet0 plumb
primary# ifconfig vnet1 plumb
primary# ifconfig vnet0 group ipmp0
primary# ifconfig vnet1 group ipmp0
```

第二个和第三个命令为 `ipmp0` 接口配置 IP 地址（如果合适）。

- Oracle Solaris 11 OS：使用 `ipadm` 命令。

请注意，`net0` 和 `net1` 分别为 `vnet0` 和 `vnet1` 的 Oracle Solaris 11 虚名。

```
primary# ipadm create-ip net0
primary# ipadm create-ip net1
primary# ipadm create-ipmp ipmp0
primary# ipadm add-ipmp -i net0 -i net1 ipmp0
```

- 以下示例在域中配置两个虚拟网络设备。每个域连接到服务域中的独立虚拟交换机设备，以使用基于链路的 IPMP。虚拟网络设备也配置为获取物理链路状态更新。

```
primary# ldm add-vnet linkprop=phys-state ldom1_vnet0 primary-vsw0 ldom1
primary# ldm add-vnet linkprop=phys-state ldom1_vnet1 primary-vsw1 ldom1
```

注 - 虚拟交换机必须分配有物理网络设备，域才能成功绑定。如果域已绑定而没有为虚拟交换机分配物理网络设备，则 `ldm add-vnet` 命令将失败。

以下命令创建虚拟网络设备并将其分配给 IPMP 组：

- Oracle Solaris 10 OS：使用 `ifconfig` 命令。

```
primary# ifconfig vnet0 plumb
primary# ifconfig vnet1 plumb
primary# ifconfig vnet0 192.168.1.1/24 up
primary# ifconfig vnet1 192.168.1.2/24 up
primary# ifconfig vnet0 group ipmp0
primary# ifconfig vnet1 group ipmp0
```

- Oracle Solaris 11 OS：使用 `ipadm` 命令。

请注意，`net0` 和 `net1` 分别为 `vnet0` 和 `vnet1` 的虚名。

```
primary# ipadm create-ip net0
primary# ipadm create-ip net1
primary# ipadm create-ipmp ipmp0
primary# ipadm add-ipmp -i net0 -i net1 ipmp0
```

```
primary# ipadm create-addr -T static -a 192.168.1.1/24 ipmp0/v4addr1
primary# ipadm create-addr -T static -a 192.168.1.2/24 ipmp0/v4addr2
```

使用 VLAN 标记

Oracle VM Server for SPARC 软件支持在网络基础结构中使用 802.1Q VLAN 标记。

虚拟交换机 (vsw) 和虚拟网络 (vnet) 设备支持根据虚拟局域网 (virtual local area network, VLAN) 标识符 (identifier, ID) 交换以太网包并处理必要的以太网帧标记或取消标记操作。

您可以在来宾域中的一个虚拟网络设备上创建多个 VLAN 接口。使用 Oracle Solaris 10 `ifconfig` 命令或 Oracle Solaris 11 `dladm` 和 `ipadm` 命令，可以在虚拟网络设备上创建 VLAN 接口。创建方法与用于在任何其他物理网络设备上配置 VLAN 接口的方法相同。Oracle VM Server for SPARC 环境中的另一项要求是，必须使用 `ldm` 命令将虚拟网络分配到相应的 VLAN。请参见 `ldm`(1M) 手册页。

同样，您可以在服务域中的虚拟交换机设备上配置 VLAN 接口。从 2 到 4094 的 VLAN ID 有效；VLAN ID 1 保留为 `default-vlan-id`。

在来宾域中创建虚拟网络设备时，必须通过在 `ldm add-vnet` 命令中使用 `pvid=` 和 `vid=` 参数指定一个端口 VLAN ID 并为该虚拟网络指定零个或更多 VLAN ID，将该虚拟网络设备分配到所需的 VLAN。此信息可对虚拟交换机进行配置，使其在 Oracle VM Server for SPARC 网络中支持多个 VLAN，并在该网络中使用 MAC 地址和 VLAN ID 交换包。

同样，在将 `vsw` 设备本身创建为网络接口时应将该设备分配到的任何 VLAN 必须使用 `ldm add-vsw` 命令的 `pvid=` 和 `vid=` 参数在 `vsw` 设备中进行配置。

您可以使用 `ldm set-vnet` 或 `ldm set-vsw` 命令更改设备所属的 VLAN。

端口 VLAN ID

端口 VLAN ID (Port VLAN ID, PVID) 用于指定该虚拟网络设备必须在无标记模式下加入的 VLAN。在这种情况下，`vsw` 设备通过 PVID 指定的 VLAN 为 `vnet` 设备提供所需的帧标记或取消标记。虚拟网络中无标记的所有出站帧由虚拟交换机使用其 PVID 进行标记。使用此 PVID 标记的入站帧由虚拟交换机取消标记，然后再将其发送给 `vnet` 设备。因此，为虚拟网络分配 PVID 意味着，对于 PVID 指定的 VLAN，虚拟交换机上的相应虚拟网络端口将被标记为无标记。一个虚拟网络设备只能有一个 PVID。

在不使用 VLAN ID 而仅使用设备实例配置相应的虚拟网络接口时，接口会隐式分配给虚拟网络的 PVID 所指定的 VLAN。

例如，如果您要使用以下命令之一创建虚拟网络实例 0，并且已将 vnet 的 pvid= 参数指定为 10，则会对 vnet0 接口进行隐式分配，以使其属于 VLAN 10。请注意，以下命令显示与 Oracle Solaris 10 相关的 vnet0 接口名称。对于 Oracle Solaris 11，请改用通用名称（例如 net0）。

- Oracle Solaris 10 OS：使用 ifconfig 命令。

```
# ifconfig vnet0 plumb
```

- Oracle Solaris 11 OS：使用 ipadm 命令。

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

VLAN ID

VID ID (VID) 用于指定虚拟网络设备或虚拟交换机必须在有标记模式下加入的 VLAN。虚拟网络设备通过其 VID 指定的 VLAN 发送和接收标记的帧。虚拟交换机在虚拟网络设备和外部网络之间传递使用指定的 VID 标记的所有帧。

▼ 如何为虚拟交换机和虚拟网络设备分配 VLAN

1. **将虚拟交换机 (vsw) 分配给两个 VLAN。**

   例如，将 VLAN 21 配置为无标记并将 VLAN 20 配置为标记。将虚拟网络 (vnet) 分配给三个 VLAN。将 VLAN 20 配置为无标记并将 VLAN 21 和 22 配置为标记。

```
primary# ldm add-vsw net-dev=nxge0 pvid=21 vid=20 primary-vsw0 primary
primary# ldm add-vnet pvid=20 vid=21,22 vnet01 primary-vsw0 ldom1
```

2. **创建 VLAN 接口。**

   本示例假定在域中这些设备的实例编号是 0，并且 VLAN 映射到这些子网：

VLAN 20	子网 192.168.1.0（网络掩码：255.255.255.0）
VLAN 21	子网 192.168.2.0（网络掩码：255.255.255.0）
VLAN 22	子网 192.168.3.0（网络掩码： 255.255.255.0）

   a. **在服务 (primary) 域中创建 VLAN 接口。**

      - Oracle Solaris 10 OS：使用 **ifconfig** 命令。

```
primary# ifconfig vsw0 plumb
```

```
primary# ifconfig vsw0 192.168.2.100 netmask 0xffffff00 broadcast + up
primary# ifconfig vsw20000 plumb
primary# ifconfig vsw20000 192.168.1.100 netmask 0xffffff00 broadcast + up
```

- Oracle Solaris 11 OS：使用 **dladm** 和 **ipadm** 命令。

```
primary# dladm create-vlan -l net0 -v20 vlan20
primary# ipadm create-ip vlan20
primary# ipadm create-addr -T static -a 192.168.1.100/24 vlan20/ipv4
```

b. 在来宾 (**ldom1**) 域中创建 VLAN 接口。

- Oracle Solaris 10 OS：使用 **ifconfig** 命令。

```
ldom1# ifconfig vnet0 plumb
ldom1# ifconfig vnet0 192.168.1.101 netmask 0xffffff00 broadcast + up
ldom1# ifconfig vnet21000 plumb
ldom1# ifconfig vnet21000 192.168.2.101 netmask 0xffffff00 broadcast + up
ldom1# ifconfig vnet22000 plumb
ldom1# ifconfig vnet22000 192.168.3.101 netmask 0xffffff00 broadcast + up
```

有关如何在 Oracle Solaris 10 OS 中配置 VLAN 接口的更多信息，请参阅《Oracle Solaris Administration: IP Services 》中的“Administering Virtual Local Area Networks”。

- Oracle Solaris 11 OS：使用 **dladm** 和 **ipadm** 命令。

```
ldom1# dladm create-vlan -l net0 -v21
ldom1# ipadm create-ip net0
ldom1# ipadm create-addr -T static -a 192.168.1.101/24 net0/ipv4
ldom1# ipadm create-ip net21000
ldom1# ipadm create-addr -T static -a 192.168.2.101/24 net21000/ipv4
ldom1# ipadm create-ip net22000
ldom1# ipadm create-addr -T static -a 192.168.3.101/24 net22000/ipv4
```

有关如何在 Oracle Solaris 11 OS 中配置 VLAN 接口的更多信息，请参阅《Managing Network Datalinks in Oracle Solaris 11.2 》中的第 3 章 “Configuring Virtual Networks by Using Virtual Local Area Networks”。

▼ 如何在安装服务器位于 VLAN 中时安装来宾域

使用 Oracle Solaris JumpStart 功能通过网络安装来宾域时，如果安装服务器位于 VLAN 中，请务必小心。只有 Oracle Solaris 10 系统支持此功能。

有关使用 Oracle Solaris JumpStart 功能安装来宾域的更多信息，请参见如何在 Oracle Solaris 10 来宾域上使用 Oracle Solaris JumpStart 功能 [54]。

1. 在无标记模式下初始配置网络设备。

例如，如果安装服务器位于 VLAN 21 中，则按如下所示初始配置虚拟网络：

```
primary# ldm add-vnet pvid=21 vnet01 primary-vsw0 ldom1
```

不要为该虚拟网络设备配置任何有标记的 VLAN (`vid`)。原因是 OpenBoot PROM (OBP) 不识别 VLAN 且无法处理具有 VLAN 标记的网络包。

2. 安装完成且 Oracle Solaris OS 引导后，在标记模式下配置虚拟网络。

```
primary# ldm set-vnet pvid= vid=21, 22, 23 vnet01 primary-vsw0 ldom1
```

然后，便可在有标记模式下将虚拟网络设备添加到其他 VLAN。

使用私有 VLAN

私有 VLAN (PVLAN) 机制可用于将某个常规 VLAN 划分成多个子 VLAN 以隔离网络通信。PVLAN 机制在 RFC 5517 (`http://tools.ietf.org/html/rfc5517`) 中定义。通常，一个常规 VLAN 是一个单广播域，但当配置了 PVLAN 属性时，该单广播域会被分区为更小的广播子域，同时保留现有的第 3 层配置。在配置 PVLAN 时，常规 VLAN 称为主要 *VLAN*，而子 VLAN 则称为辅助 *VLAN*。

当两个虚拟网络在某个物理链路上使用同一个 VLAN ID 时，则将在这两个虚拟网络之间传递所有广播通信。但是，在创建使用 PVLAN 属性的虚拟网络时，包转发行为可能不适用于所有情况。

下表展示了隔离的 PVLAN 和社区 PVLAN 的广播包转发规则。

表 11-1 广播包转发规则

	隔离的	社区 A	社区 B
隔离的	否	否	否
社区 A	否	是	否
社区 B	否	否	是

例如，当在 `net0` 上隔离 `vnet0` 和 `vnet1` 虚拟网络时，`net0` 不在两个虚拟网络之间传递广播通信。但是，当 `net0` 网络收到来自隔离的 VLAN 的通信时，该通信不会传递到与该 VLAN 相关的隔离端口。出现这种情况的原因在于隔离的虚拟网络仅接受主要 VLAN 的通信。

inter-vnet-links 功能支持对隔离的 PVALN 和社区 PVLAN 的通信限制。已为隔离的 PVLAN 禁用 inter-vnet-links，仅为在社区 PVLAN 的同一个社区中的虚拟网络启用 inter-vnet-links。不允许来自社区外部的其他虚拟网络的直接通信。

注 - 如果目标服务域不支持 PVLAN 功能，迁移为 PVLAN 配置的来宾域可能会失败。

PVLAN 要求

可以使用 `ldm add-vnet` 和 `ldm set-vnet` 命令配置 PVLAN。使用这些命令配置 `pvlan` 属性。请注意，还必须指定 `pvid` 属性才能成功配置 PVLAN。

此功能要求至少 Oracle Solaris 11.2.4.0.0 (SRU 4) OS。

要配置 PVLAN，必须指定以下信息：

- **主要 VLAN ID。**主要 VLAN ID 是用于配置单个虚拟网络设备的 PVLAN 的端口 VLAN ID (PVID)。此配置可确保来宾域收到 VLAN 包。请注意，不能用 PVLAN 配置 VID。此值由 `pvid` 属性表示。
- **辅助 VLAN ID。**特定 VLAN 使用辅助 VLAN ID 来提供 PVLAN 功能。将此信息指定为 `pvlan` 值的 *secondary-vid* 部分。*secondary-vid* 是介于 1-4094 范围的整数值。主要 VLAN 可以有多个辅助 VLAN，但有以下限制：
  - 主要 VLAN ID 和辅助 VLAN ID 都不能与缺省 VLAN ID 相同。
  - 主要 VLAN ID 和辅助 VLAN ID 不能有相同的隔离 PVLAN 类型和社区 PVLAN 类型值。
  - 每个主要 VLAN 仅能配置一个隔离 PVLAN。所以，不能创建使用相同主要 VLAN ID 的两个隔离 PVLAN。
  - 主要 VLAN 可以有多个社区 VLAN，但有以下限制：
    - 主要 VLAN ID 不能用作辅助 VLAN ID 来创建其他社区 PVLAN。

      例如，如果有一个主要 VLAN ID 为 `3`、辅助 VLAN ID 为 `100` 的社区 PVLAN，则无法创建使用 `3` 作为辅助 VLAN ID 的其他社区 PVLAN。
    - 辅助 VLAN ID 不能用作主要 VLAN ID 来创建社区 PVLAN。

      例如，您有一个主要 VLAN ID 为 `3`、辅助 VLAN ID 为 `100` 的社区 PVLAN，则无法创建使用 `100` 作为主要 VLAN ID 的其他社区 PVLAN。
    - 辅助 VLAN ID 不能用作常规虚拟网络或 VNIC 的 VLAN ID。

注意 - Logical Domains Manager 只能验证特定虚拟交换机上的虚拟网络的配置。如果为同一个后端设备上的 Oracle Solaris VNIC 设置了 PVLAN 配置，请确保在所有 VNIC 和虚拟网络中满足相同的要求。

- **PVLAN 类型。**将此信息指定为 `pvlan` 值的 *pvlan-type* 部分。*pvlan-type* 是以下值之一：
  - 隔离的 `PVLAN`。与隔离的 PVLAN 关联的端口与所有对等虚拟网络以及后端网络设备上的 Oracle Solaris 虚拟 NIC 隔离。包仅根据为 PVLAN 指定的值到达外部网络。

- 社区 PVLAN。与社区 PVLAN 关联的端口可以与同一个社区 PVLAN 中的其他端口通信，但与其他所有端口隔离。包根据为 PVLAN 指定的值到达外部网络。

配置 PVLAN

本节包括介绍如何创建 PVLAN 并列出 PVLAN 信息的任务。

创建 PVLAN

可以使用 `ldm add-vnet` 或 `ldm set-vnet` 命令设置 `pvlan` 属性值，从而配置 PVLAN。请参见 `ldm`(1M) 手册页。

可以使用以下命令创建或删除 PVLAN：

- 使用 `ldm add-vnet` 创建 PVLAN：

  `ldm add-vnet pvid=`*port-VLAN-ID* `pvlan=`*secondary-vid*`,`*pvlan-type* *if-name* *vswitch-name* *domain-name*

  以下命令展示了如何创建具有以下 PVLAN 的虚拟网络：主要 *vlan-id* 为 `4`，辅助 *vlan-id* 为 `200`，*pvlan-type* 为 `isolated`。

  ```
  primary# ldm add-vnet pvid=4 pvlan=200,isolated vnet1 primary-vsw0 ldg1
  ```

- 使用 `ldm set-vnet` 创建 PVLAN：

  `ldm set-vnet pvid=`*port-VLAN-ID* `pvlan=`*secondary-vid*`,`*pvlan-type* *if-name* *domain-name*

  以下命令展示了如何创建具有以下 PVLAN 的虚拟网络：主要 *vlan-id* 为 `3`，辅助 *vlan-id* 为 `300`，*pvlan-type* 为 `community`。

  ```
  primary# ldm add-vnet pvid=3 pvlan=300,community vnet1 primary-vsw0 ldg1
  ```

- 使用 `ldm set-vnet` 删除 PVLAN：

  `ldm set-vnet pvlan=` *if-name* *vswitch-name* *domain-name*

  以下命令将删除 `vnet0` 虚拟网络的 PVLAN 配置。此命令的结果是：指定的虚拟网络是一个常规 VLAN，该 VLAN 使用在配置 PVLAN 时指定的 *vlan-id*。

  ```
  primary# ldm set-vnet pvlan= vnet0 primary-vsw0 ldg1
  ```

查看 PVLAN 信息

可以通过使用多个 Logical Domains Manager 列表子命令来查看有关 PVLAN 的信息。请参见 `ldm`(1M) 手册页。

可以使用以下命令查看 PVLAN 信息。

- 使用 `ldm list-domain -o network` 列出 PVLAN 信息：

  ```
  ldm list-domain [-e] [-l] -o network [-p] [domain-name...]
  ```

  以下示例通过使用 `ldm list-domain -o network` 命令展示了有关 `ldg1` 域上的 PVLAN 配置的信息。

  - 以下 `ldm list-domain` 命令展示了有关 `ldg1` 域上的 PVLAN 配置的信息。

    ```
    primary# ldm list-domain -o network ldg1
    NAME
    ldg1

    MAC
     00:14:4f:fa:bf:0f

    NETWORK
    NAME   SERVICE              ID DEVICE    MAC
    vnet0  primary-vsw0@primary 0  network@0 00:14:4f:f8:03:ed
    MODE   PVID VID             MTU  MAXBW LINKPROP
           1    3               1500 1700
           PVLAN : 200,community
    ```

  - 以下 `ldm list-domain` 命令以可解析形式展示了 `ldg1` 域的 PVLAN 配置信息。

    ```
    primary# ldm list-domain -o network -p ldg1
    VERSION 1.13
    DOMAIN|name=ldg1|
    MAC|mac-addr=00:14:4f:fa:bf:0f
    VNET|name=vnet0|dev=network@0|service=primary-vsw0@primary
    |mac-addr=00:14:4f:f8:03:ed|mode=|pvid=1|vid=3|mtu=1500|linkprop=|id=0
    |alt-mac-addrs=|maxbw=1700|protect=|priority=|cos=|pvlan=200,community
    ```

- 使用 `ldm list-bindings` 列出 PVLAN 信息：

  ```
  ldm list-bindings [-e] [-p] [domain-name...]
  ```

  以下示例通过使用 `ldm list-bindingsnetwork` 命令展示了有关 `ldg1` 域上的 PVLAN 配置的信息。

  - 以下 `ldm list-bindings` 命令展示了有关 `ldg1` 域上的 PVLAN 配置的信息。

    ```
    primary# ldm list-bindings
    ...
    NETWORK
    NAME    SERVICE              ID DEVICE    MAC
    vnet0   primary-vsw0@primary 0  network@0 00:14:4f:f8:03:ed
    MODE   PVID VID   MTU  MAXBW LINKPROP
           1    3     1500 1700
    ```

```
       PVLAN :200,community
PEER                MAC               MODE  PVID VID  MTU  MAXBW LINKPROP
primary-vsw0@primary 00:14:4f:f8:fe:5e 1
```

- 以下 `ldm list-bindings` 命令以可解析形式展示了 `ldg1` 域的 PVLAN 配置信息。

```
primary# ldm list-bindings -p
...
VNET|name=vnet0|dev=network@0|service=primary-vsw0@primary
|mac-addr=00:14:4f:f8:03:ed|mode=|pvid=1|vid=3|mtu=1500|linkprop=
|id=0|alt-mac-addrs=|maxbw=1700|protect=|priority=|cos=|pvlan=200,community
|peer=primary-vsw0@primary|mac-addr=00:14:4f:f8:fe:5e|mode=|pvid=1|vid=
|mtu=1500|maxbw=
```

- 使用 `ldm list-constraints` 列出 PVLAN 信息：

```
ldm list-constraints [-x] [domain-name...]
```

下面展示了通过运行 `ldm list-constraints` 命令生成的输出：

```
primary# ldm list-constraints -x ldg1
...
<Section xsi:type="ovf:VirtualHardwareSection_Type">
  <Item>
    <rasd:OtherResourceType>network</rasd:OtherResourceType>
    <rasd:Address>auto-allocated</rasd:Address>
    <gprop:GenericProperty key="vnet_name">vnet0</gprop:GenericProperty>
    <gprop:GenericProperty key="service_name">primary-vsw0</gprop:GenericProperty>
    <gprop:GenericProperty key="pvid">1</gprop:GenericProperty>
    <gprop:GenericProperty key="vid">3</gprop:GenericProperty>
    <gprop:GenericProperty key="pvlan">200,community</gprop:GenericProperty>
    <gprop:GenericProperty key="maxbw">1700000000</gprop:GenericProperty>
    <gprop:GenericProperty key="device">network@0</gprop:GenericProperty>
    <gprop:GenericProperty key="id">0</gprop:GenericProperty>
  </Item>
```

优化包吞吐量性能

可以使用 `ldm add-vnet` 和 `ldm set-vnet` 命令设置以下数据链路属性值来优化包吞吐量性能：

`priority`	指定 CPU 包处理优先级
`cos`	指定链路的 IEEE 802.1p 链路服务类
`protection`	指定包通信安全类型

有关有效和默认属性值的信息，请参见 ldm(1M) 手册页。

例 11-9 设置和查看数据链路包属性

以下示例显示如何使用 ldm set-vnet 命令在单个命令中设置 priority、protection 和 cos 属性值。还可以使用 ldm add-vnet 命令添加使用指定的数据链路属性值的新虚拟网络。

```
primary# ldm set-vnet allowed-ips=192.168.100,1,192.168.100.2 \
allowed-dhcpcids=oracle1@system1.company.com, \
00:14:4f:f9:d3:88,system2,00:14:4f:fb:61:6e cos=7 priority=high \
protection=restricted,mac-nospoof,ip-nospoof,dhcp-nospoof vnet3_ldg3 ldg3
```

ldm list -o network 命令显示您使用前面的 ldm set-vnet 命令设置的 ldg3 域上的数据链路属性值。保护值是 mac-nospoof、restricted、192.168.100,1,192.168.100.2 MAC 地址的 ip-nospoof 以及 system1@company.com,00:14:4f:f9:d3:88,system2,00:14:4f:fb:61:6e 的 dhcp-nospoof。priority 设置为 high，服务类 (class of service, cos) 设置为 7。

```
primary# ldm list -o network ldg3
NAME
ldg3

MAC
00:14:4f:fa:3b:49

VSW
NAME      MAC               NET-DEV ID DEVICE   LINKPROP DEFAULT-VLAN-ID
PVID VID MTU  MODE INTER-VNET-LINK
ldg3_vsw1 00:14:4f:f9:65:b5 net3    0  switch@0 1        1
         1500      on

NETWORK
  NAME       SERVICE              ID DEVICE    MAC               MODE PVID VID
  MTU  MAXBW LINKPROP
  vnet3      primary-vsw0@primary 0  network@0 00:14:4f:fa:47:f4      1
  1500
  vnet1_ldg3 primary-vsw1@primary 1  network@1 00:14:4f:fb:90:b7      1
  1500
  vnet2_ldg3 ldg1_vsw1@ldg1       2  network@2 00:14:4f:fb:61:6e      1
  1500
  vnet3_ldg3 ldg1_vsw1@ldg1       3  network@3 00:14:4f:fb:25:70      1
                                             00:14:4f:f8:7a:8d
                                             00:14:4f:fb:6e:32
                                             00:14:4f:fb:1a:2f
  1500 1000
       PROTECTION: mac-nospoof
                   restricted
                   ip-nospoof/192.168.100,1,192.168.100.2
                   dhcp-nospoof/system1@oracle.us.oracle.com,
                   00:14:4f:f9:d3:88,system2,00:14:4f:fb:61:6e
       PRIORITY  : high
```

```
COS        : 7
```

使用 NIU 混合 I/O

虚拟 I/O 框架可实施混合 I/O 模型以改进功能和性能。混合 I/O 模型可将直接 I/O 和虚拟化 I/O 组合在一起，以便可以在虚拟机中灵活部署 I/O 资源。如果直接 I/O 无法为虚拟机提供完整功能，或者由于资源可用性或虚拟机迁移而无法为虚拟机持续提供一致的直接 I/O，则该模型尤其有用。

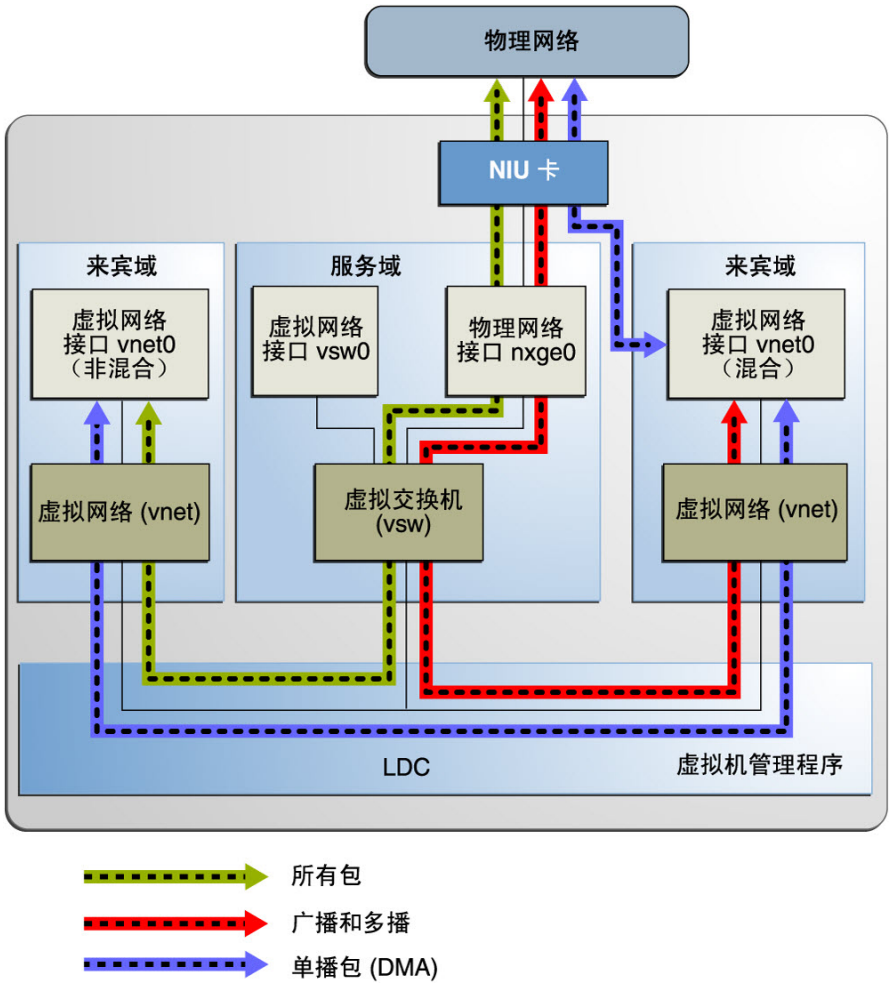
混合 I/O 体系结构完全适用于 Oracle Sun UltraSPARC T2、SPARC T3 和 SPARC T4 平台上的网络接口单元 (Network Interface Unit, NIU)。NIU 是在芯片上集成的网络 I/O 接口。使用此体系结构可以将直接内存访问 (Direct Memory Access, DMA) 资源动态分配到虚拟网络设备，从而为域中的应用程序提供稳定的性能。

注 - NIU 混合 I/O 功能已过时，而是使用 SR-IOV。Oracle VM Server for SPARC 3.2 是包括此功能的最新软件发行版。

NIU 混合 I/O 适用于 Oracle Sun UltraSPARC T2、SPARC T3 和 SPARC T4 平台。此功能由可选混合模式启用，可提供给虚拟网络 (`vnet`) 设备，其中 DMA 硬件资源已借给来宾域中的 `vnet` 设备以改进性能。在混合模式中，来宾域中的 vnet 设备可以使用 DMA 硬件资源直接在外部网络和来宾域之间发送和接收单播通信。到同一系统中其他来宾域的广播或多播通信和单播通信继续使用虚拟 I/O 通信机制进行发送。

注 - NIU 混合 I/O 不适用于 UltraSPARC T2 Plus 平台。

图 11-10 混合虚拟网络



注 - 此图显示了 Oracle Solaris 10 系统中的配置。对于 Oracle Solaris 11 系统，只有接口名称更改为使用通用名称（例如，`nxge0` 更改为 `net0`）。

混合模式仅适用于与配置为使用 NIU 网络设备的虚拟交换机 (`vsw`) 关联的 `vnet` 设备。由于可共享的 DMA 硬件资源有限，因此，在给定时间，每个 `vsw` 最多只能有三个 `vnet` 设备分配 DMA 硬件资源。如果超过三个 `vnet` 设备启用了混合模式，则根据先到先得原则完成分配。由于系统中有两个 NIU 网络设备，因此，两个不同的虚拟交换机上总共可以有六个分配了 DMA 硬件资源的 `vnet` 设备。

使用此功能时，请注意以下几点：

- 用于 vnet 设备的混合模式选项仅视为建议。因此，只有当具有 DMA 资源，并且该设备能够使用这些资源时，才会分配这些资源。
- Logical Domains Manager CLI 命令不会验证混合模式选项；也就是说，可以在任何 vnet 或任意数量的 vnet 设备上设置混合模式。
- 来宾域和服务域至少需要运行 Oracle Solaris 10 10/08 OS。
- 每个 vsw 最多只能有三个 vnet 设备可以在给定时间内借用 DMA 硬件资源。由于有两个 NIU 网络设备，因此总共可以有六个借用 DMA 硬件资源的 vnet 设备。

注 - 仅为每个 vsw 的三个 vnet 设备设置混合模式，以便保证它们分配到 DMA 硬件资源。

- 来宾域处于活动状态时不能动态更改混合模式选项。
- 仅当在来宾域中创建了活动的 vnet 设备时，才会分配 DMA 硬件资源。
- NIU 10 千兆位以太网驱动程序 (nxge) 用于 NIU 卡。同一驱动程序也用于其他 10 千兆位网卡。但是，NIU 混合 I/O 功能仅适用于 NIU 网络设备。

▼ 如何配置虚拟交换机和 NIU 网络设备

1. **确定 NIU 网络设备。**
   以下示例显示了 UltraSPARC T2 服务器上的输出。

```
primary# grep nxge /etc/path_to_inst
"/niu@80/network@0" 0 "nxge"
"/niu@80/network@1" 1 "nxge"
```

   以下示例显示了 SPARC T3-1 或 SPARC T4-1 服务器上的输出。

```
primary# grep nxge /etc/path_to_inst
"/niu@480/network@0" 0 "nxge"
"/niu@480/network@1" 1 "nxge"
```

2. **仅限 Oracle Solaris 11 OS：确定与 NIU 网络设备（例如 `nxge0`）对应的链路名称。**

```
primary# dladm show-phys -L |grep nxge0
net2              nxge0         /SYS/MB
```

3. **配置虚拟交换机。**

   - Oracle Solaris 10 OS：

```
primary# ldm add-vsw net-dev=nxge0 primary-vsw0 primary
```

   - Oracle Solaris 11 OS：

以下示例使用 net2 而不是 nxge0。

```
primary# ldm add-vsw net-dev=net2 primary-vsw0 primary
```

▼ 如何启用或禁用混合模式

默认情况下会为 vnet 设备禁用混合模式并且必须显式启用该模式。

- 使用 **ldm** 命令可启用和禁用混合模式。
  - 例如，要在创建 **vnet** 设备时为其启用混合模式，请执行以下命令：

    ```
    primary# ldm add-vnet mode=hybrid vnet01 primary-vsw0 ldom01
    ```

  - 要为 **vnet** 设备禁用混合模式，请执行以下命令：

    ```
    primary# ldm set-vnet mode= vnet01 ldom01
    ```

将链路聚合和虚拟交换机结合使用

可将虚拟交换机配置为使用链路聚合。链路聚合用作虚拟交换机的网络设备，以连接到物理网络。使用此配置，虚拟交换机可以利用 EEE 802.3ad 链路聚合标准提供的功能。此类功能包括增加带宽、负载平衡和故障转移。有关如何配置链路聚合的信息，请参见《Oracle Solaris Administration: IP Services》。

创建链路聚合后，您可以将其分配给虚拟交换机。进行此分配类似于将物理网络设备分配给虚拟交换机。使用 ldm add-vswitch 或 ldm set-vswitch 命令设置 net-dev 属性。

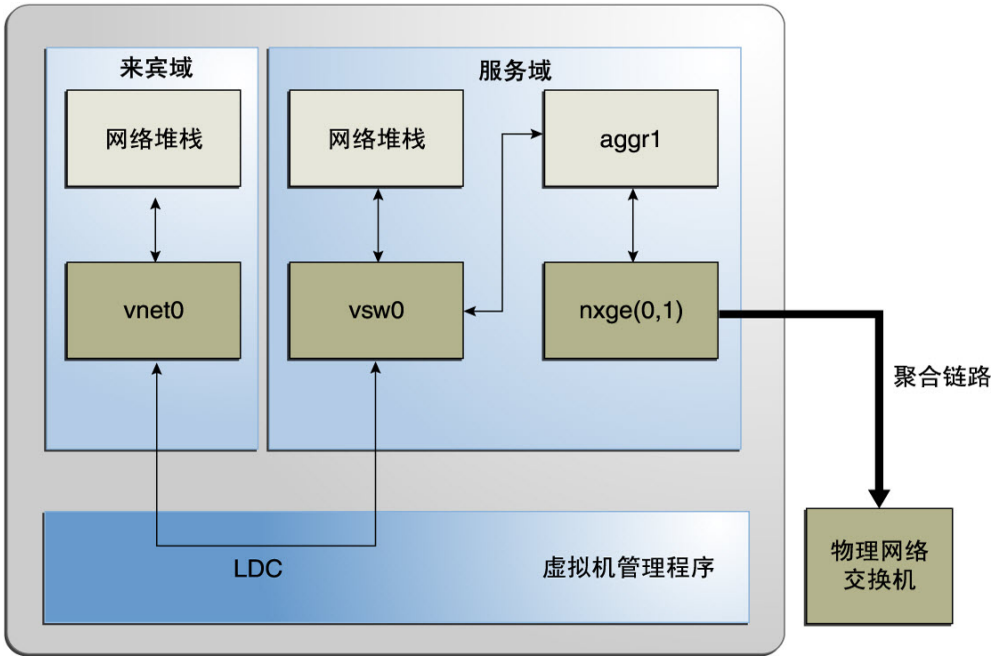
将链路聚合分配给虚拟交换机时，进出物理网络的通信会流经聚合。任何所需的负载平衡或故障转移由底层聚合框架透明处理。链路聚合对来宾域上以及绑定到使用聚合的虚拟交换机的虚拟网络 (vnet) 设备完全透明。

注 - 您不能将虚拟网络设备（vnet 和 vsw）组合到链路聚合中。

您可以在服务域中创建并使用配置为使用链路聚合的虚拟交换机。请参见如何将虚拟交换机配置为主接口 [44]。

下图显示了配置为通过物理接口 nxge0 和 nxge1 使用聚合 aggr1 的虚拟交换机。

图 11-11 配置虚拟交换机以使用链路聚合



注 - 此图显示了 Oracle Solaris 10 系统中的配置。对于 Oracle Solaris 11 系统，只有接口名称更改为使用通用名称（例如，`nxge0` 和 `nxge1` 分别更改为 `net0` 和 `net1`）。

配置巨型帧

Oracle VM Server for SPARC 虚拟交换机 (`vsw`) 和虚拟网络 (`vnet`) 设备现在可以支持有效负荷大小大于 1500 字节的以太网帧。因此，这些驱动程序现在能够提高网络吞吐量。

可通过指定虚拟交换机设备的最大传输单元 (maximum transmission unit, MTU) 启用巨型帧。在这种情况下，虚拟交换机设备和绑定到虚拟交换机设备的所有虚拟网络设备都使用指定的 MTU 值。

如果虚拟网络设备所需的 MTU 值应小于虚拟交换机支持的值，则您可以直接为虚拟网络设备指定 MTU 值。

注 - 在 Oracle Solaris 10 5/09 OS 上，必须将物理设备的 MTU 配置为与虚拟交换机的 MTU 相匹配。有关配置特定驱动程序的信息，请参见 Oracle Solaris 参考手册的 7D 部分中与此驱动程序对应的手册页。例如，要获取有关 Oracle Solaris 10 `nxge` 驱动程序的信息，请参见 `nxge(7D)` 手册页。

在少数情况下，您可以能需要使用 `ldm add-vnet` 或 `ldm set-vnet` 命令为虚拟网络设备指定与虚拟交换机的 MTU 值不同的 MTU 值。例如，如果您通过虚拟网络设备配置 VLAN 且最大的 VLAN MTU 小于虚拟交换机上的 MTU 值，则可以更改虚拟网络设备的 MTU 值。对于仅使用默认 MTU 值的域，可能不需要支持巨型帧的 `vnet` 驱动程序。但是，如果在域中虚拟网络设备已绑定到使用巨型帧的虚拟交换机，请确保 `vnet` 驱动程序支持巨型帧。

如果您在虚拟网络设备上使用 `ldm set-vnet` 命令指定 `mtu` 值，则不会将对虚拟交换机设备的 MTU 值的后续更新传播到此虚拟网络设备。要重新使虚拟网络设备从虚拟交换机设备获取 MTU 值，请运行以下命令：

```
primary# ldm set-vnet mtu= vnet-name domain-name
```

请注意，为虚拟网络设备启用巨型帧会自动为分配给此虚拟网络设备的任何混合 IO 资源启用巨型帧。

在控制域中，Logical Domains Manager 将由 `ldm set-vsw` 和 `ldm set-vnet` 命令启动的 MTU 值作为延迟重新配置操作更新。要在控制域以外的域中更新 MTU，您必须先停止域，然后再运行 `ldm set-vsw` 或 `ldm set-vnet` 命令修改 MTU 值。

▼ 如何配置虚拟网络和虚拟交换机设备以使用巨型帧

1. 登录到控制域。
2. 成为管理员。
   - 对于 Oracle Solaris 10，请参见《System Administration Guide: Security Services》中的“Configuring RBAC (Task Map)”。
   - 对于 Oracle Solaris 11.2，请参见《Securing Users and Processes in Oracle Solaris 11.2》中的第 1 章 “About Using Rights to Control Users and Processes”。
3. 确定要用于虚拟网络的 MTU 值。
   您可以指定从 1500 到 16000 字节的 MTU 值。指定的 MTU 必须匹配分配给虚拟交换机的物理网络设备的 MTU。
4. 指定虚拟交换机设备或虚拟网络设备的 MTU 值。
   执行以下操作之一：

- 通过将 MTU 指定为 `mtu` 属性的值，在服务域中的新虚拟交换机设备上启用巨型帧。

  ```
  primary# ldm add-vsw net-dev=device mtu=value vswitch-name ldom
  ```

  除配置虚拟交换机外，此命令也会更新将要绑定到此虚拟交换机的每个虚拟网络设备的 MTU 值。
- 通过将 MTU 指定为 `mtu` 属性的值，在服务域中的现有虚拟交换机设备上启用巨型帧。

  ```
  primary# ldm set-vsw net-dev=device mtu=value vswitch-name
  ```

  除配置虚拟交换机外，此命令也会更新将要绑定到此虚拟交换机的每个虚拟网络设备的 MTU 值。

例 11-10 在虚拟交换机和虚拟网络设备上配置巨型帧

- 以下示例显示了如何添加使用 MTU 值 `9000` 的新虚拟交换机设备。此 MTU 值将从虚拟交换机设备传播到所有客户机虚拟网络设备。

  首先，`ldm add-vsw` 命令创建 MTU 值为 `9000` 的虚拟交换机设备 `primary-vsw0`。请注意，网络设备 `nxge0` 的实例 0 被指定为 `net-dev` 属性的值。

  ```
  primary# ldm add-vsw net-dev=nxge0 mtu=9000 primary-vsw0 primary
  ```

  下一步，`ldm add-vnet` 命令将客户机虚拟网络设备添加到此虚拟交换机 `primary-vsw0`。请注意，虚拟网络设备的 MTU 是从此设备绑定到的虚拟交换机中隐式分配的。因此，`ldm add-vnet` 命令不需要您指定 `mtu` 属性的值。

  ```
  primary# ldm add-vnet vnet01 primary-vsw0 ldom1
  ```

  根据运行的 Oracle Solaris OS 版本，执行以下操作：

  - Oracle Solaris 10 OS：`ifconfig` 命令会在服务域 `primary` 中创建虚拟交换机接口。`ifconfig vsw0` 命令输出显示 `mtu` 属性的值是 `9000`。

    ```
    primary# ifconfig vsw0 plumb
    primary# ifconfig vsw0 192.168.1.100/24 up
    primary# ifconfig vsw0
    vsw0: flags=201000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4,CoS> mtu 9000 index 5
            inet 192.168.1.100 netmask ffffff00 broadcast 192.168.1.255
            ether 0:14:4f:fa:0:99
    ```

    `ifconfig` 命令创建来宾域 `ldom1` 中的虚拟网络接口。`ifconfig vnet0` 命令输出显示 `mtu` 属性的值是 `9000`。

    ```
    primary# ifconfig vnet0 plumb
    primary# ifconfig vnet0 192.168.1.101/24 up
    primary# ifconfig vnet0
    vnet0: flags=201000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4,CoS> mtu 9000 index 4
    ```

```
        inet 192.168.1.101 netmask ffffff00 broadcast 192.168.1.255
        ether 0:14:4f:f9:c4:13
```

- Oracle Solaris 11 OS：使用 `ipadm` 命令可查看主接口的 `mtu` 属性值。

```
# ipadm show-ifprop -p mtu net0
IFNAME PROPERTY PROTO PERM CURRENT PERSISTENT DEFAULT POSSIBLE
net0   mtu      ipv4  rw   9000    --        9000    68-9000
```

`ipadm` 命令可在来宾域 `ldom1` 中创建虚拟网络接口。`ipadm show-ifprop` 命令输出显示 `mtu` 属性的值是 `9000`。

```
primary# ipadm create-ip net0
primary# ipadm create-addr -T static -a 192.168.1.101/24 net0/ipv4
primary# ipadm show-ifprop -p mtu net0
IFNAME PROPERTY PROTO PERM CURRENT PERSISTENT DEFAULT POSSIBLE
net0   mtu      ipv4  rw   9000    --         9000    68-9000
```

- 以下示例说明如何将接口的 MTU 更改为 `4000`。

  请注意，只能将接口的 MTU 更改为小于由 Logical Domains Manager 分配的设备的 MTU。如果配置了 VLAN，并且每个 VLAN 接口都需要一个不同的 MTU，则此方法会很有用。

  - Oracle Solaris 10 OS：使用 `ifconfig` 命令。

```
primary# ifconfig vnet0 mtu 4000
primary# ifconfig vnet0
vnet0: flags=1201000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4,CoS,FIXEDMTU>
mtu 4000 index 4
        inet 192.168.1.101 netmask ffffff00 broadcast 192.168.1.255
        ether 0:14:4f:f9:c4:13
```

  - Oracle Solaris 11 OS：使用 `ipadm` 命令。

```
primary# ipadm set-ifprop -p mtu=4000 net0
primary# ipadm show-ifprop -p mtu net0
IFNAME PROPERTY PROTO PERM CURRENT PERSISTENT DEFAULT POSSIBLE
net0   mtu      ipv4  rw   4000    --         9000    68-9000
```

与 `vnet` 和 `vsw` 驱动程序的早期（不识别巨型帧）版本的兼容性 (Oracle Solaris 10)

注 - 本节仅适用于 Oracle Solaris 10 OS。

支持巨型帧的驱动程序可以与同一系统上不支持巨型帧的驱动程序交互操作。只要创建虚拟交换机时不启用巨型帧支持，就可以实现该互操作性。

注 - 如果与虚拟交换机关联的任何来宾域或服务域不使用支持巨型帧的 Oracle VM Server for SPARC 驱动程序，请不要设置 `mtu` 属性。

可通过更改虚拟交换机的 `mtu` 属性的默认值 1500 来启用巨型帧。在这种情况下，早期驱动程序版本会忽略 `mtu` 设置并继续使用默认值。请注意，`ldm list` 输出将显示您指定的 MTU 值，而不是默认值。任何大于默认 MTU 的帧均不会被发送到这些设备，并被新的驱动程序丢弃。这种情况可能会导致仍然使用早期驱动程序版本的这些来宾域的网络行为不一致。此限制既适用于客户机来宾域也适用于服务域。

因此，启用巨型帧后，请确保 Oracle VM Server for SPARC 网络中的所有虚拟设备均已升级，以便可以使用支持巨型帧的新驱动程序。您必须至少运行 Logical Domains 1.2 才能配置巨型帧。

Oracle Solaris 11 中联网特定功能的差别

在域运行 Oracle Solaris 10 OS 与 Oracle Solaris 11 OS 时，某些 Oracle VM Server for SPARC 联网功能的工作方式有所不同。在域中运行 Oracle Solaris 11 OS 时，Oracle VM Server for SPARC 虚拟网络设备和虚拟交换机的功能差别如下：

- **将 `vsw`*n* 设备配置为主网络接口，以允许服务域与来宾域进行通信**

  只有运行 Oracle Solaris 10 OS 的域才需要进行此配置。对于 Oracle Solaris 11，虚拟交换机使用 Oracle Solaris 11 网络堆栈，可自动允许其虚拟网络设备和与其后端设备相对应的网络接口（例如 `net0`）进行通信。不能再在 Oracle Solaris 11 服务域中使用 `vsw`*n* 设备自身。
- **使用 Oracle Solaris 11 `etherstub` 设备作为后端设备来创建专用虚拟交换机**

  如果未连接到后端设备，虚拟交换机仅在来宾域之间（而不是在来宾域与服务域之间）提供通信。将 `etherstub` 用作后端设备可允许来宾域与在 Oracle Solaris 11 服务域中配置的区域（包括全局区域）进行通信。通过使用连接到该 `etherstub` 的 VNIC 完成此配置。
- **对虚拟交换机和虚拟网络设备使用通用名称**

  Oracle Solaris 11 OS 会为 `vsw`*n* 和 `vnet`*n* 设备分配通用名称。确保所创建的虚拟交换机的后端设备不是另一个 `vsw` 或 `vnet` 设备。使用 `dladm show-phys` 命令可查看与通用网络设备名称相关联的实际物理设备。
- **在虚拟交换机和虚拟网络设备上使用 VNIC**

  无法在 `vsw`*n* 设备上使用 VNIC。尝试在 `vsw`*n* 上创建 VNIC 会失败。请参见《Oracle VM Server for SPARC 3.2 发行说明》中的“Oracle Solaris 11：配置有自动网络接口的区域可能无法启动”。

♦ ♦ ♦ 第 12 章 12

迁移域

本章介绍如何将域从一台主机迁移到另一台主机。

本章包括以下主题：

注 - 要使用本章描述的迁移功能，必须运行最新版本的 Logical Domains Manager、系统固件和 Oracle Solaris OS。有关使用早期版本的 Oracle VM Server for SPARC 进行迁移的信息，请参见《Oracle VM Server for SPARC 3.2 发行说明》和相关版本的管理指南。

域迁移介绍

使用域迁移，可以将来宾域从一个主机迁移到另一个主机。启动迁移的计算机为源计算机，域所迁移到的计算机为目标计算机。

在执行迁移操作的过程中，要迁移的域会从源计算机传输到目标计算机上的已迁移域。

实时迁移功能提供了性能提高特性，使活动域可以在运行的同时进行迁移。除了实时迁移外，还可以迁移绑定域或非活动域，此操作称为冷迁移。

可以使用域迁移操作执行如下任务：

- 平衡计算机之间的负荷
- 在保持来宾域继续运行的情况下执行硬件维护

要实现最佳迁移性能，请确保源计算机和目标计算机都在运行最新版本的 Logical Domains Manager。

迁移操作概述

源计算机上的 Logical Domains Manager 接受迁移域的请求，并建立与目标计算机上运行的 Logical Domains Manager 的安全网络连接。在建立此连接之后，将进行迁移。迁移操作分以下阶段执行：

阶段 1：在源计算机与目标计算机上运行的 Logical Domains Manager 连接之后，有关源计算机和要迁移的域的信息会传输到目标计算机。此信息用于执行一系列检查，以确定迁移是否可行。执行的检查基于要迁移的域的状态。例如，如果要迁移的域处于活动状态，将会执行一组与域处于绑定或非活动状态时不同的检查。

阶段 2：在通过了阶段 1 中所有的检查之后，源计算机和目标计算机就开始准备迁移了。在目标计算机上，创建一个域来接收要迁移的域。如果要迁移的域处于非活动状态或者已绑定，则迁移操作会转至阶段 5。

阶段 3：如果要迁移的域处于活动状态，则它的运行时状态信息会传输到目标计算机。要迁移的域将继续运行，Logical Domains Manager 会同时跟踪操作系统对该域进行的修改。此信息会从源计算机上的虚拟机管理程序检索出来并安装在目标计算机上的虚拟机管理程序中。

阶段 4：要迁移的域暂停。此时，所有余下的已修改状态信息都重新复制到目标计算机。这样，对域造成的中断会很小或不易察觉。中断时间取决于工作负荷。

阶段 5：从源计算机上的 Logical Domains Manager 切换到目标计算机上的 Logical Domains Manager。当迁移的域恢复执行（如果要迁移的域之前处于活动状态），而且源计算机上的域已销毁时，会进行切换。从此时起，已迁移的域将成为正在运行的域的唯一版本。

软件兼容性

要使迁移能够发生，源计算机和目标计算机都必须运行如下兼容软件：

- 在两台计算机上运行的 Logical Domains Manager 版本都必须是当前版本或以前发行的最新版本。
- 源计算机和目标计算机上安装的固件版本必须兼容，才支持实时迁移。两台计算机都必须至少运行 Oracle VM Server for SPARC 软件的此发行版支持的最低固件版本。

有关更多信息，请参见“迁移的版本限制” [237]。

迁移操作安全性

Oracle VM Server for SPARC 为迁移操作提供了以下安全功能：

- **验证。**由于迁移操作在两台计算机上执行，因此在某些情况下必须在源计算机和目标计算机上都对用户进行验证。特别是，除超级用户以外的用户必须使用 `LDoms Management` 权限配置文件。但是，如果您使用 SSL 证书执行迁移，则无需在目标计算机和源计算机上对用户进行验证，而您也无法指定其他用户。

  `ldm migrate-domain` 命令允许您选择指定要在目标计算机上验证的备用用户名。如果没有指定备用用户名，将会使用执行迁移命令的用户的用户名。请参见例 12-3 “迁移和重命名来宾域”。在这两种情况下，都会提示用户输入目标计算机的密码，除非使用 `-p` 选项启动非交互式迁移。请参见“执行非交互式迁移” [240]。
- **加密。**Oracle VM Server for SPARC 使用 SSL 对迁移流量进行加密以防止敏感数据被利用并消除对其他硬件和专用网络的需求。

  在具有加密单元的平台上，当源计算机和目标计算机上的 `primary` 域分配了加密单元时，则会提高迁移操作的速度。因为 SSL 操作可以负载转移到加密单元。

  如果平台上的 CPU 具有加密指令，则会自动提高迁移操作的速度。之所以会提高该速度，是因为 SSL 操作可通过加密指令来执行，而不是在软件中执行。
- **FIPS 140-2。**可以对系统进行配置，以执行使用 Oracle Solaris FIPS 140-2 认证的 OpenSSL 库的域迁移。请参见“用于域迁移的 FIPS 140-2 模式” [234]。

为迁移配置 SSL 证书

要执行基于证书的验证，请将 `-c` 选项与 `ldm migrate-domain` 命令配合使用。此选项与密码文件和备用用户选项互不相容。如果未指定 `-c` 选项，迁移操作将执行密码验证。

▼ 如何为迁移配置 SSL 证书

要配置 SSL 证书，必须在源计算机和目标计算机上执行此任务中的步骤。

1. **如果 `/var/opt/SUNWldm/trust` 目录尚不存在，请创建该目录。**
2. **将远程 `ldmd` 证书安全复制到本地 `ldmd` 受信任证书目录。**

远程 ldmd 证书是远程主机上的 /var/opt/SUNWldm/server.crt。本地 ldmd 受信任证书目录是 /var/opt/SUNWldm/trust。调用远程证书文件 *remote-hostname*.pem。

3. 创建从 **ldmd** 受信任证书目录中的证书到 **/etc/certs/CA** 的符号链接。
将 REMOTE 变量设置为 *remote-host*。

```
localhost# ln -s /var/opt/SUNWldm/trust/${REMOTE}.pem /etc/certs/CA/
```

4. 重新启动 **svc:/system/ca-certificates** 服务。

```
localhost# svcadm restart svc:/system/ca-certificates
```

5. 验证配置是否正常工作。

```
localhost# openssl verify  /var/opt/SUNWldm/trust/${REMOTE}.pem
/var/opt/SUNWldm/trust/remote-hostname.pem: OK
```

6. 重新启动 **ldmd** 守护进程。

```
localhost# svcadm restart ldmd
```

删除 SSL 证书

如果从 /var/opt/SUNWldm/trust 和 /etc/certs/CA 目录中删除 .pem 文件，您必须重新启动 svc:/system/ca-certificates 服务，然后重新启动 ldmd 服务。请注意，在服务重新启动之前，仍允许执行使用该 .pem 文件的任何迁移。

```
localhost# svcadm restart svc:/system/ca-certificates
localhost# svcadm restart ldmd
```

用于域迁移的 FIPS 140-2 模式

可以配置 Logical Domains Manager，以执行使用 Oracle Solaris FIPS 140-2 认证的 OpenSSL 库的域迁移。Logical Domains Manager 处于 FIPS 140-2 模式下时，仅能使用其将域迁移到具有在 FIPS 140-2 模式下运行的 Logical Domains Manager 的另一个系统。尝试迁移到非 FIPS 系统将被拒绝。如果 Logical Domains Manager 未处于 FIPS 140-2 模式，则它无法迁移到处于 FIPS 140-2 模式的 Logical Domains Manager。

要在 FIPS 140-2 模式下成功启动 Logical Domains Manager，必须启用 FIPS 中介。有关分步说明，请参见如何在 FIPS 140-2 模式下运行 Logical Domains Manager [235]。

有关更多信息以及显示如何使用支持 FIPS 140 的 OpenSSL 实现，请参见《Managing Encryption and Certificates in Oracle Solaris 11.2 》中的“How to Switch to the FIPS 140-Capable OpenSSL Implementation”和《Oracle SuperCluster M7-8 Owner’s

Guide: Overview》中的“Example of Enabling Two Applications in FIPS 140 Mode on an Oracle Solaris System”。

▼ 如何在 FIPS 140-2 模式下运行 Logical Domains Manager

开始之前 在 FIPS 140-2 模式下运行 Logical Domains Manager 之前，必须确保至少运行 Logical Domains Manager 的 3.2 版本并且 `primary` 域至少运行 Oracle Solaris 11.2 OS。

1. 安装并启用 FIPS 140-2 OpenSSL 中介。

   a. 如果需要，应安装 FIPS 140-2 OpenSSL 中介。

      安装 Oracle Solaris 11.2 OS 时，默认情况下应该安装此软件包。

      ```
      # pkg install openssl-fips-140
      ```

   b. 列出当前 OpenSSL 中介。

      ```
      # pkg mediator openssl
      MEDIATOR VER. SRC. VERSION IMPL. SRC. IMPLEMENTATION
      openssl  vendor            local      default
      ```

   c. 列出可用 OpenSSL 中介。

      ```
      # pkg mediator -a openssl
      MEDIATOR VER. SRC. VERSION IMPL. SRC. IMPLEMENTATION
      openssl  vendor            vendor     default
      openssl  system            system     fips-140
      ```

注意 - 要切换到的 OpenSSL 实现方式必须在系统中存在。如果切换到系统中不存在的实现方式，系统可能会变得无法使用。

   d. 启用 FIPS 140-2 中介。

      ```
      # pkg set-mediator -I fips-140 openssl
      ```

   e. 重新引导。

      ```
      # reboot
      ```

   f. 确认已设置 FIPS 140-2 中介。

      ```
      # pkg mediator openssl
          MEDIATOR     VER. SRC. VERSION IMPL. SRC. IMPLEMENTATION
          openssl      system            local      fips-140
      ```

2. 将 **ldmd** 守护进程配置为使用 FIPS 140-2 模式。

   a. 将 **ldmd** 守护进程置于 FIPS 140-2 模式下。

   ```
   # svccfg -s ldoms/ldmd setprop ldmd/fips1402_enabled = true
   ```

   b. 重新启动 **ldmd** 守护进程。

   ```
   # svcadm refresh ldmd
   # svcadm restart ldmd
   ```

▼ 如何将 Logical Domains Manager 从 FIPS 140-2 模式恢复为默认模式

1. 通过恢复为默认 OpenSSL 中介来停止使用 FIPS 140-2 OpenSSL 中介。
   仅当其他应用程序不需要 FIPS 140-2 中介时才执行此步骤。

   ```
   # pkg set-mediator -I default openssl
   ```

2. 重新引导。

   ```
   # reboot
   ```

3. 将 **ldmd** 守护进程配置为使用默认模式。

   ```
   # svccfg -s ldoms/ldmd setprop ldmd/fips1402_enabled = false
   ```

4. 重新启动 **ldmd** 守护进程。

   ```
   # svcadm refresh ldmd
   # svcadm restart ldmd
   ```

域迁移限制

以下各节说明域迁移的限制。Logical Domains Manager 软件和系统固件版本必须兼容才允许迁移。此外，还必须满足特定的 CPU 要求，才能确保域迁移成功。

实时迁移并非在源平台和目标平台以及系统固件版本的所有组合上都符合条件并受支持。对于无法执行实时迁移的那些组合，您可以改为执行冷迁移。

迁移的版本限制

本节介绍了执行实时迁移时的版本限制。

- Logical Domains Manager 版本。当一个系统运行最新版本的 Logical Domains Manager 而另一个系统至少运行仅低一个版本的 Logical Domains Manager 时，可以执行双向的实时迁移。
- Oracle Solaris OS 版本。可以对至少运行 Oracle Solaris 10 9/10 OS 的来宾域执行实时迁移。不能对运行 Oracle Solaris 10 10/09 OS 或更低 Oracle Solaris OS 版本的来宾域执行实时迁移。您仍可以引导这些更低 Oracle Solaris OS 版本并执行此类域的冷迁移。
- 系统固件版本。通常情况下，当源计算机和目标计算机均支持相应的最低系统固件版本时，您可以在两个系统之间执行实时迁移。

  以下列表显示支持实时迁移的平台和关联的最低系统固件版本为：

  - UltraSPARC T2 和 UltraSPARC T2 Plus 平台 – 版本 7.4.5
  - SPARC T3 和 SPARC T4 平台 – 版本 8.2.2.c
  - SPARC T5、SPARC M5 和 SPARC M6 平台 – 所有系统固件版本
  - Fujitsu M10 平台 – 所有 XCP 版本

  但是，某些特定的平台和固件组合不支持实时迁移。如果尝试将域从至少运行系统固件版本 8.4 或 XCP221 的系统中实时迁移到运行早期系统固件版本的系统，则会失败。这是由于新旧系统固件版本之间的虚拟机管理程序 API 不匹配。在这种情况下，会发出以下消息：

```
primary# ldm migrate ldg1 root@target-name
Target Password:
Domain ldg1 is using features of the system firmware that are not supported in
the version of the firmware running on the target machine.
Domain Migration of LDom ldg1 failed
```

  请注意，除非目标计算机为 SPARC M5-32 系统，否则可以将域从运行系统固件版本 8.3 的系统中实时迁移到至少运行系统固件版本 8.4 的系统中。有关更多信息，请参见《Oracle VM Server for SPARC 3.2 发行说明 》中的“误允许将域从运行系统固件 8.3 的 SPARC T4 系统迁移到 SPARC T5、SPARC M5 或 SPARC M6 系统”。

  系统固件版本 8.4、9.1 和 XCP2230 引入了对 EFI GPT 磁盘标签的支持。默认情况下，在至少运行 Oracle Solaris 11.1 OS 的系统上安装的虚拟磁盘均具有 EFI GPT 磁盘标签。在早期版本的固件（例如 9.0.*x*、8.3、7.*x* 或 XCP2221）上无法找到此磁盘标签。此情况使得无法对运行无 EFI GPT 支持的系统固件版本的系统执行实时迁移或冷迁移。请注意，在这种情况下执行冷迁移也会失败，这与先前的限制不同。

  要确定虚拟磁盘是否具有 EFI GPT 磁盘标签，请对原始设备运行 `devinfo -i` 命令。以下示例用于显示虚拟磁盘的磁盘标签是 SMI VTOC 还是 EFI GPT：

  - SMI VTOC 磁盘标签。如果虚拟磁盘具有 SMI VTOC 磁盘标签，则无论固件是否支持 EFI，均可以对固件执行迁移。

以下示例指示设备具有 VTOC 标签，因为 `devinfo -i` 命令报告了特定于设备的信息：

```
# devinfo -i /dev/rdsk/c2d0s2
/dev/rdsk/c2d0s2        0       0       73728   512     2
```

- **EFI GPT 磁盘标签**。如果虚拟磁盘具有 EFI GPT 磁盘标签，则只能对具有 EFI 支持的固件执行迁移。

  以下示例指示设备具有 EFI GPT 磁盘标签，因为 `devinfo -i` 命令报告了一个错误：

```
# devinfo -i /dev/rdsk/c1d0s0
devinfo: /dev/rdsk/c1d0s0: This operation is not supported on EFI
labeled devices
```

迁移的 CPU 限制

如果要迁移的域运行的 Oracle Solaris OS 版本低于 Oracle Solaris 10 1/13 OS，则在迁移过程中可能会显示以下消息：

```
Domain domain-name is not running an operating system that is
compatible with the latest migration functionality.
```

当运行的 OS 早于 Oracle Solaris 10 1/13 OS 时，以下 CPU 要求和限制适用：

- 必须为迁移的域分配完整核心。如果要迁移的域中的线程数少于完整核心，则在迁移的域重新引导之前，任何域都无法使用额外的线程。
- 迁移后，将会对迁移的域禁用 CPU 动态重新配置 (Dynamic Reconfiguration, DR)，直到该域重新引导。重新引导后，可以在迁移的域上使用 CPU DR。
- 目标计算机必须具有足够的空闲完整核心来提供迁移的域所需的线程数。迁移后，如果迁移的域仅使用了部分完整核心，则在迁移的域重新引导之前，任何域都无法使用额外的线程。

尝试迁移在 OpenBoot 或内核调试器中运行的域时，也适用这些限制。请参见迁移 OpenBoot PROM 中的域或在内核调试器中运行的域。

跨 CPU 迁移的版本限制

您无法在 UltraSPARC T2、UltraSPARC T2 Plus 或 SPARC T3 系统与 SPARC T5、SPARC M5 或 SPARC M6 系统之间执行实时迁移。

- SPARC T4 系统必须运行系统固件版本 8.7
- SPARC T5、SPARC M5 或 SPARC M6 系统必须运行系统固件版本 9.4
- 源计算机和目标计算机都必须运行 Oracle VM Server for SPARC 3.2 软件

设置 perf-counters 的迁移限制

对设置了 perf-counters 属性值的域执行迁移时要小心。

对将 perf-counters 属性值设置为 global 的域执行迁移之前，请确保目标计算机上没有其他域将 perf-counters 属性设置为 global。

在迁移操作过程中，根据性能访问功能是在源计算机、目标计算机还是在两者上，对 perf-counters 属性的处理会有所不同。

perf-counters 属性值的处理如下所示：

- **仅源计算机。** perf-counters 属性值不传播到目标计算机。
- **仅目标计算机。** 将要迁移的计算机上的 perf-counters 属性值更新为 perf-counters= 的等效值。
- **源计算机和目标计算机。** perf-counters 属性值从要迁移的域传播到目标计算机上的已迁移域。

有关 perf-counters 属性的更多信息，请参见“使用 Perf-Counter 属性” [283] 和 ldm(1M) 手册页。

迁移域

可以使用 ldm migrate-domain 命令启动某个域从一个主机到另一个主机的迁移。

注 - 如果迁移域，将丢弃您使用 cid 和 mblock 属性分配的任何已命名资源。相反，域使用目标系统上的匿名资源。

有关在保持活动域继续运行的情况下时对其进行迁移的信息，请参见“迁移活动域” [240]。有关迁移绑定域或非活动域的信息，请参见“迁移绑定域或非活动域” [246]。

有关迁移选项和操作数的信息，请参见 ldm(1M) 手册页。

注 - 在域迁移完成后，将新配置保存到源系统和目标系统的 SP。因此，如果源系统或目标系统经历过一次开关机循环，则迁移的域的状态正确。

执行模拟运行

如果为 `ldm migrate-domain` 命令提供 `-n` 选项，则会执行迁移检查，但不会迁移域。任何未满足的要求都会报告为错误。模拟运行结果使您可以在尝试执行实际迁移之前更正任何配置错误。

注 - 由于逻辑域的动态特性，可能会出现模拟运行成功而实际迁移失败的情况，或者相反。

执行非交互式迁移

使用 SSL 证书方法执行非交互式迁移操作。仍可以使用传统 `ldm migrate-domain -p` *filename* 命令启动非交互式迁移操作。

指定为 `-p` 选项的参数的文件名必须具有以下特征：

- 文件的第一行必须包含密码。
- 密码必须是纯文本。
- 密码的长度不得超过 256 个字符。

密码末尾的换行符和第一行之后的所有行都会被忽略。

存储目标计算机密码的文件必须得到适当保护。如果打算以这种方式存储密码，请确保已设置文件权限，以便只有 root 所有者或特权用户才可以读取或写入文件（400 或 600）。

迁移活动域

当您尝试迁移处于活动状态的域时，应考虑到对于要迁移的域、源计算机和目标计算机有某些要求和限制。有关更多信息，请参见“域迁移限制” [236]。

提示 - 可以通过向源计算机和目标计算机上的 `primary` 域添加更多的虚拟 CPU 来缩短总迁移时间。建议每个 `primary` 域中至少有两整个核心，但这并不是必需的。

迁移过程中，域会“丢失时间”。要缓解这个时间丢失问题，请将要迁移的域与外部时间源（如网络时间协议 (Network Time Protocol, NTP) 服务器）同步。如果将域配置为 NTP 客户机，则在迁移完成后，域的日期和时间会很快得以更正。

要将域配置为 Oracle Solaris 10 NTP 客户机，请参见《System Administration Guide: Network Services 》中的“Managing Network Time Protocol (Tasks)”。要将域配置为 Oracle Solaris 11 NTP 客户机，请参见《Introduction to Oracle Solaris 11 Network Services 》中的“Managing Network Time Protocol (Tasks)”。

注 - 在迁移结束时的暂停阶段，来宾域可能会发生短暂延迟。此延迟不会对网络通信造成任何明显中断，尤其是当该协议具有重试机制（如 TCP）或在应用程序级别具有重试机制（如 NFS over UDP）时。但是，如果来宾域正在运行网络敏感应用程序（如路由信息协议 (Routing Information Protocol, RIP)），则该域可能会在尝试执行某个操作时发生短暂延迟或中断。如果在暂停阶段关闭并重新创建来宾网络接口，则可能会在短时间发生此延迟。

CPU 的域迁移要求

下面是执行迁移时对 CPU 的要求和限制：

- 目标计算机必须具有足够的可用虚拟 CPU，以适应要迁移的域所使用的虚拟 CPU 数目。
- 在来宾域上设置 `cpu-arch` 属性允许您在具有不同处理器类型的系统之间迁移域。请注意，来宾域必须处于绑定或非活动状态才能更改 `cpu-arch` 值。

  受支持的 `cpu-arch` 属性值如下：

  - `native` 使用特定于 CPU 的硬件功能，以允许来宾域仅在具有相同 CPU 类型的平台之间迁移。`native` 是默认值。
  - `migration-class1` 是适用于 SPARC T4 以后的 SPARC 平台的跨 CPU 迁移系列。这些平台支持在迁移中或迁移后进行硬件加密，以使受支持的 CPU 具有下界。

    该值与 UltraSPARC T2、UltraSPARC T2 Plus、SPARC T3 平台或 Fujitsu M10 平台不兼容。
  - `sparc64-class1` 是适用于 SPARC64 平台的跨 CPU 迁移系列。由于 `sparc64-class1` 值基于 SPARC64 指令，因此，该值的指令数比 `generic` 值多。这样，与 `generic` 值相比，该值对性能不会产生影响。

    该值仅与 Fujitsu M10 服务器兼容。
  - `generic` 会使用由所有平台使用的最低通用 CPU 硬件功能，从而使来宾域能够执行与 CPU 类型无关的迁移。

  以下 `isainfo -v` 命令显示当 `cpu-arch=generic` 和 `cpu-arch=migration-class1` 时可在系统中使用的指令。

  - `cpu-arch=generic`

    ```
    # isainfo -v
    64-bit sparcv9 applications
    ```

```
        asi_blk_init vis2 vis popc
   32-bit sparc applications
        asi_blk_init vis2 vis popc v8plus div32 mul32
```

- cpu-arch=migration-class1

```
# isainfo -v
64-bit sparcv9 applications
        crc32c cbcond pause mont mpmul sha512 sha256 sha1 md5
        camellia des aes ima hpc vis3 fmaf asi_blk_init vis2
        vis popc
32-bit sparc applications
        crc32c cbcond pause mont mpmul sha512 sha256 sha1 md5
        camellia des aes ima hpc vis3 fmaf asi_blk_init vis2
        vis popc v8plus div32 mul32
```

与使用 native 值相比，使用 generic 值可能会导致来宾域性能降低。由于来宾域仅使用在所有支持的 CPU 类型上都可用的通用 CPU 功能，而不使用特定 CPU 的本机硬件功能，所以性能可能会降低。由于不使用这些功能，通过 generic 值可以在使用支持不同功能的 CPU 的系统之间灵活地迁移域。

如果在至少为 SPARC T4 的系统之间迁移域，可以设置 cpu-arch=migration-class1 以提高来宾域性能。虽然使用 generic 值可以改进性能，native 值仍提供来宾域的最佳性能。

当 cpu-arch 属性设置为 native 时，请使用 psrinfo -pv 命令确定处理器类型，如下所示：

```
# psrinfo -pv
The physical processor has 2 virtual processors (0 1)
  SPARC-T5 (chipid 0, clock 3600 MHz)
```

请注意，如果 cpu-arch 属性设置为 native 以外的值，则 psrinfo -pv 输出不会显示平台类型。此时，此命令将显示已装入 sun4v-cpu CPU 模块。

```
# psrinfo -pv
The physical processor has 2 cores and 13 virtual processors (0-12)
  The core has 8 virtual processors (0-7)
  The core has 5 virtual processors (8-12)
    sun4v-cpu (chipid 0, clock 3600 MHz)
```

内存的迁移要求

目标计算机必须具有足够的可用内存以满足域迁移的需要。此外，下列属性必须在迁移中保持不变：

- 必须能够创建相同数量的相同大小内存块。
- 内存块的物理地址不需要匹配，但是必须在迁移过程中维护相同的实际地址。

此外，目标计算机上可用内存的布局必须与要迁移的域的内存布局兼容，否则迁移将会失败。需特别指出的是，如果目标计算机上的内存分为多个小的地址范围，但是要迁移的域需要一个大的地址范围，则迁移将失败。下面的示例对这种情况进行了说明。

要迁移的域 `ldg1` 也具有 8 GB 可用内存，这些内存分布在两个内存块中。目标计算机的内存分布在三个内存块中，其中一些内存块太小。

```
source# ldm ls -o memory ldg1
NAME
ldg1

MEMORY
    RA              PA              SIZE
    0x80000000      0x400000000     2G
    0x400000000     0x880000000     6G

target# ldm ls-devices mem
MEMORY
    PA                  SIZE
    0x180880000000      5632M
    0x301f70000000      2G
    0x381b20000000      512M
```

在这种内存布局情况下，迁移将失败：

```
source# ldm migrate -n ldg1 target
Target Password:
Free memory layout and congruency requirements prevent binding the
memory block with PA 0x880000000, RA 0x400000000, and size 6G
Domain Migration of LDom ldg1 would fail if attempted
```

物理 I/O 设备的迁移要求

无法迁移可以直接访问物理设备的域。例如，您无法迁移 I/O 域。但是，可以迁移与物理设备关联的虚拟设备。

有关更多信息，请参见“PCIe 端点设备的迁移要求” [244]和“PCIe SR-IOV 虚拟功能的迁移要求” [244]。

虚拟 I/O 设备的迁移要求

要迁移的域所使用的全部虚拟 I/O 服务必须在目标计算机上可用。换而言之，必须满足下列条件：

- 要迁移的域中使用的每个虚拟磁盘后端必须在目标计算机上定义。此共享存储可以是 SAN 磁盘，也可以是可通过 NFS 或 iSCSI 协议使用的存储。您定义的虚拟磁盘后端

必须与源计算机上的后端具有相同的卷名和服务名。源计算机和目标计算机上指向后端的路径可能会有所不同，但它们必须指向同一个后端。

注意 - 即使源计算机和目标计算机上指向虚拟磁盘后端的路径不是指同一个存储，迁移也将成功。但是，目标计算机上域的行为将无法预测，该域可能将无法使用。为了纠正这种情况，请停止该域，更正配置问题，然后重新启动该域。如果不执行这些步骤，则该域可能处于不一致状态。

- 要迁移的域中的每个虚拟网络设备在目标计算机上必须具有相应的虚拟网络交换机。每个虚拟网络交换机必须与源计算机上连有该设备的虚拟网络交换机同名。

  例如，如果要迁移的域中的 `vnet0` 连接到名为 `switch-y` 的虚拟交换机服务，则目标计算机上的域必须提供名为 `switch-y` 的虚拟交换机服务。

注 - 目标计算机上的物理网络必须正确配置，以便已迁移的域能够访问它所需要的网络资源。否则，在迁移完成之后，某些网络服务可能会在已迁移的域上不可用。

例如，您可能希望确保该域能够访问正确的网络子网。您还可能希望确保网关、路由器或防火墙正确配置，以便该域能够从目标计算机访问所需的远程系统。

要迁移的域所使用的自动分配范围内的 MAC 地址必须可在目标计算机上使用。

- 虚拟控制台集中器 (`vcc`) 服务必须在目标计算机上存在，并至少具有一个空闲端口。迁移过程中会忽略显式控制台约束。已迁移域的控制台将使用已迁移域的名称作为控制台组并使用控制域中任何可用 `vcc` 设备上的任何可用端口来创建。如果控制域中没有可用端口，则控制台会通过服务域中某个可用 `vcc` 设备上的某个可用端口来创建。如果与默认组名冲突，则迁移将会失败。

PCIe 端点设备的迁移要求

无法对配置有 PCIe 端点设备的 I/O 域执行域迁移。

有关直接 I/O 功能的信息，请参见“通过分配 PCIe 端点设备创建 I/O 域” [67]。

PCIe SR-IOV 虚拟功能的迁移要求

您不能对配置有 PCIe SR-IOV 虚拟功能的 I/O 域执行域迁移。

有关 SR-IOV 功能的信息，请参见第 8 章 使用 PCIe SR-IOV 虚拟功能创建 I/O 域。

NIU 混合 I/O 的迁移要求

可以迁移使用 NIU 混合 I/O 资源的域。指定 NIU 混合 I/O 资源的约束不是域的硬性要求。如果将这样的域迁移到不具有可用 NIU 资源的计算机，则该约束将保留，但不会得到满足。

请注意，NIU 混合 I/O 功能已过时，而是使用 SR-IOV。Oracle VM Server for SPARC 3.2 是包括此功能的最新软件发行版。

加密单元的迁移要求

在具有加密单元的平台上，如果某个具有绑定加密单元的来宾域运行的是支持加密单元动态重新配置 (dynamic reconfiguration, DR) 的操作系统，则可以迁移该来宾域。

开始迁移时，Logical Domains Manager 会确定要迁移的域是否支持加密单元 DR。如果支持，Logical Domains Manager 会尝试从该域中删除所有加密单元。迁移完成后，会将加密单元重新添加到已迁移的域。

注 - 如果目标计算机无法满足对加密单元的约束，迁移操作将被阻止。在这种情况下，已迁移的域所具有的加密单元可能比迁移操作前少。

活动域中的延迟重新配置

源计算机或目标计算机上任何活动的延迟重新配置操作都会阻止迁移启动。不允许在迁移时启动延迟重新配置操作。

在活动域具有有效的电源管理弹性策略的情况下迁移

只要在源计算机或目标计算机上启用了电源管理 (power management, PM) 弹性策略，就可以执行实时迁移。

对其他域的操作

在某台计算机上运行迁移时，如果执行的任何操作可能会导致修改正在迁移的域的状态或配置，则该操作将被阻塞。对该域本身的所有操作以及对计算机上其他域的绑定或停止等操作都会被阻止。

迁移 OpenBoot PROM 中的域或在内核调试器中运行的域

执行域迁移需要在 Logical Domains Manager 和要迁移的域中运行的 Oracle Solaris OS 之间进行协调。如果要迁移的域在 OpenBoot 或内核调试器 (kmdb) 中运行，则这种协调是不可能的。因此，迁移尝试失败。

要迁移的域在 OpenBoot 中运行时，将显示以下消息：

```
primary# ldm migrate ldg1 system2
Migration is not supported while the domain ldg1 is in the 'OpenBoot Running' state
Domain Migration of LDom ldg1 failed
```

要迁移的域在内核调试器 (kmdb) 中运行时，将显示以下消息：

```
primary# ldm migrate ldg1 system2
Migration is not supported while the domain ldg1 is in the 'Solaris debugging' state
Domain Migration of LDom ldg1 failed
```

迁移绑定域或非活动域

对于绑定域或非活动域仅应用少数几个域迁移限制，这是由于这样的域在迁移时不会执行。因此，可以在不同平台类型之间进行迁移，例如，从 SPARC T3 平台迁移到 SPARC T5 平台或 Fujitsu M10 平台，因为不会在平台间复制运行时状态。

要迁移绑定域，目标计算机需满足要迁移的域的 CPU、内存和 I/O 约束。如果无法满足这些约束，则迁移将会失败。

注意 - 当您迁移绑定域时，不会检查虚拟磁盘后端 `options` 和 `mpgroup` 值，因为未与目标计算机交换运行时状态信息。当您迁移活动域时则执行此检查。

非活动域的迁移没有这样的要求。但是，在以后尝试绑定时，目标计算机必须满足已迁移的域的约束，否则，域绑定将失败。

注 - 在域迁移完成后，将新配置保存到源系统和目标系统的 SP。因此，如果源系统或目标系统经历过一次开关机循环，则迁移的域的状态正确。

虚拟 I/O 设备的迁移要求

对于非活动域，不针对虚拟 (virtual I/O, VIO) 约束执行任何检查。因此，不需要存在 VIO 服务器，迁移就可成功。与任何非活动域相同，绑定域时，VIO 服务器必须存在并处于可用状态。

PCIe 端点设备的迁移要求

无法对配置有 PCIe 端点设备的 I/O 域执行域迁移。此要求适用于绑定域，而不适用于非活动域。

有关直接 I/O (direct I/O, DIO) 功能的信息，请参见“通过分配 PCIe 端点设备创建 I/O 域” [67]。

PCIe SR-IOV 虚拟功能的迁移要求

您不能对配置有 PCIe SR-IOV 虚拟功能的 I/O 域执行域迁移。此要求适用于绑定域，而不适用于非活动域。

有关 SR-IOV 功能的信息，请参见第 8 章 使用 PCIe SR-IOV 虚拟功能创建 I/O 域。

监视正在进行的迁移

迁移过程中，正迁移的域和已迁移的域以不同的方式显示在状态输出中。`ldm list` 命令的输出指示所迁移域的状态。

`FLAGS` 字段中的第六列显示以下值之一：

- `s` – 作为迁移源的域。
- `+` – 作为迁移目标的已迁移域。
- `e` – 发生错误，需要用户干预。

以下命令显示 `ldg-src` 域是迁移的源：

```
# ldm list ldg-src
NAME       STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
ldg-src    suspended  -n---s          1     1G       0.0%  2h 7m
```

以下命令显示 ldg-tgt 域是迁移的目标：

```
# ldm list ldg-tgt
NAME       STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
ldg-tgt    bound      -----t  5000    1     1G
```

长格式的状态输出中显示有关迁移的其他信息。在源计算机上，状态输出中显示操作的完成百分比，以及目标计算机和已迁移域的名称。同样，在目标计算机上，状态输出中显示操作的完成百分比，以及源计算机和正迁移的域的名称。

以下命令显示 ldg-src 域的迁移操作的进度：

```
# ldm list -o status ldg-src
NAME
ldg-src

STATUS
    OPERATION    PROGRESS    TARGET
    migration    17%         t5-sys-2
```

取消正在进行的迁移

迁移开始之后，如果 ldm 命令由 KILL 信号中断，则迁移操作会终止。如果迁移操作被终止，则已迁移的域将被销毁，而要迁移的域会恢复（如果此前处于活动状态）。如果 ldm 命令的控制 shell 丢失，迁移将在后台继续进行。

还可以使用 ldm cancel-operation 命令从外部取消迁移操作。此命令将终止正在进行的迁移操作，正在迁移的域将恢复为活动域。ldm cancel-operation 命令必须从源计算机启动。在给定的计算机上，任何与迁移相关的命令都会影响从该计算机启动的迁移操作。目标计算机无法控制迁移操作。

从失败的迁移中恢复

当正在迁移的域已经完成向已迁移的域发送所有运行时状态信息而已迁移的域尚无法确认该域已恢复时，如果丢失网络连接，则迁移操作将终止。

您必须通过执行以下步骤来确定迁移是否成功完成：

1. 确定已迁移的域是否已成功恢复操作。已迁移的域将处于以下两种状态之一：
   - 如果迁移成功完成，已迁移的域将处于正常状态。
   - 如果迁移失败，目标计算机会清除并销毁已迁移的域。
2. 如果已迁移的域成功恢复操作，您可以放心地销毁源计算机上处于错误状态的域：但是，如果已迁移的域不存在，则源计算机上的域仍是主版本的域而且必须进行恢

复。要恢复此域，请在源计算机上运行 `ldm cancel-operation` 命令。此命令将清除错误状态，并将此域恢复到其初始状态。

迁移示例

例 12-1 使用 SSL 证书执行来宾域迁移

此示例说明如何将 `ldg1` 域迁移到名为 `t5-sys-2` 的计算机上。迁移操作开始之前，必须已经在源计算机和目标计算机上配置了 SSL 证书。请参见如何为迁移配置 SSL 证书 [233]。

```
# ldm migrate-domain -c ldg1 t5-sys-2
```

例 12-2 迁移来宾域

此示例说明如何将 `ldg1` 域迁移到名为 `t5-sys-2` 的计算机上。

```
# ldm migrate-domain ldg1 t5-sys-2
Target Password:
```

要在不提示输入目标计算机密码的情况下执行此迁移，请使用以下命令：

```
# ldm migrate-domain -p pfile ldg1 t5-sys-2
```

`-p` 选项将文件名作为参数。指定的文件包含目标计算机的超级用户密码。在此示例中，`pfile` 包含目标计算机 `t5-sys-2` 的密码。

例 12-3 迁移和重命名来宾域

此示例说明如何在迁移操作过程中重命名域。在迁移过程中，源计算机上的 `ldg-src` 域将在目标计算机 (`t5-sys-2`) 上重命名为 `ldg-tgt`。另外，`ldm-admin` 用户用于在目标计算机上进行验证。

```
# ldm migrate ldg-src ldm-admin@t5-sys-2:ldg-tgt
Target Password:
```

例 12-4 迁移故障消息

此示例说明当目标计算机不支持最新的迁移功能时，您可能看到的错误消息。

```
# ldm migrate ldg1 dt212-346
Target Password:
```

```
The target machine is running an older version of the domain
manager that does not support the latest migration functionality.

Upgrading to the latest software will remove restrictions on
a migrated domain that are in effect until it is rebooted.
Consult the product documentation for a full description of
these restrictions.

The target machine is running an older version of the domain manager
that is not compatible with the version running on the source machine.

Domain Migration of LDom ldg1 failed
```

例 12-5 获取目标计算机上域的迁移状态

此示例说明如何在迁移正在进行时获取有关已迁移域的状态。在此示例中，源计算机为 t5-sys-1。

```
# ldm list -o status ldg-tgt
NAME
ldg-tgt

STATUS
    OPERATION    PROGRESS    SOURCE
    migration    55%         t5-sys-1
```

例 12-6 获取源计算机上域的可解析迁移状态

此示例说明如何在迁移正在进行时获取有关正迁移的域的可解析状态。在此示例中，目标计算机为 t5-sys-2。

```
# ldm list -o status -p ldg-src
VERSION 1.6
DOMAIN|name=ldg-src|
STATUS
|op=migration|progress=42|error=no|target=t5-sys-2
```

♦ ♦ ♦ 第 13 章 13

管理资源

本章包含有关在 Oracle VM Server for SPARC 系统上执行资源管理的信息。

本章包括以下主题：

资源重新配置

运行 Oracle VM Server for SPARC 软件的系统可以配置资源（例如虚拟 CPU、虚拟 I/O 设备、加密单元和内存）。某些资源可以在正在运行的域上动态地进行配置，而其他一些资源则必须在停止的域上进行配置。如果无法在控制域上动态地配置资源，必须首先启动延迟重新配置。延迟重新配置会将配置活动推迟到控制域进行重新引导后。

动态重新配置

通过动态重新配置 (dynamic reconfiguration, DR)，可以在操作系统 (operating system, OS) 正在运行时添加或删除资源。对特定资源类型执行 DR 的功能取决于逻辑域中运行的 OS 是否具备相应支持。

支持对以下资源进行动态重新配置：

- **虚拟 CPU** – 在 Oracle Solaris 10 OS 和 Oracle Solaris 11 OS 的所有版本中均支持

- **虚拟 I/O 设备** – 在 Oracle Solaris 10 10/08 OS 和 Oracle Solaris 11 OS 以及更高版本中支持
- **加密单元** – 在 Oracle Solaris 10 1/13 OS 和 Oracle Solaris 11 OS 以及更高版本中支持
- **内存** – 请参见“使用内存动态重新配置” [266]
- **CPU 整体核心** – 请参见《Oracle VM Server for SPARC 3.2 安装指南 》中的“全限定 Oracle Solaris OS 版本”
- **物理 I/O 设备** – 不支持

要使用 DR 功能，Logical Domains DR 守护进程 `drd` 必须在要更改的域中运行。请参见 `drd`(1M) 手册页。

延迟重新配置

与可以立即发生的 DR 操作不同，延迟重新配置操作在以下情况下生效：

- 下次重新引导 OS 之后
- 如果未运行 OS，则在停止并启动逻辑域之后

通常，延迟重新配置操作仅限于控制域。对于所有其他域，除非可以动态重新配置资源，否则必须停止域来修改配置。

延迟的重新配置操作仅限于控制域。在根域上执行延迟重新配置期间，您可以运行有限数量的命令，以支持无法动态完成的操作。这些子命令包括 `add-io`、`set-io`、`remove-io`、`create-vf` 和 `destroy-vf`。您还可以在根域上运行 `ldm start-reconf` 命令。对于所有其他域，除非可以动态重新配置资源，否则必须停止域来修改配置。

如果正在执行延迟重新配置，则对该域的其他重新配置请求将被延迟，直到重新引导该域或者停止并启动该域为止。

`ldm cancel-reconf` 命令可以取消域上的延迟重新配置操作。有关如何使用延迟重新配置功能的更多信息，请参见 `ldm`(1M) 手册页。

注 - 如果任何其他 `ldm remove-*` 命令已对虚拟 I/O 设备执行延迟重新配置操作，则无法使用 `ldm cancel-reconf` 命令。在此情况下 `ldm cancel-reconf` 命令会失败。

可以使用延迟的重新配置减少控制域上的资源。要从控制域中删除大量 CPU，请参见《Oracle VM Server for SPARC 3.2 发行说明 》中的“从来宾域中删除大量 CPU 可能会失败”。要从控制域中删除大量内存，请参见“减少控制域的内存” [268]。

注 - 如果 `primary` 域处于延迟重新配置状态，则由 Oracle VM Server for SPARC 管理的资源只有在 `primary` 域重新引导之后才受电源管理。由 OS 直接管理的资源（例如由 Solaris Power Aware Dispatcher 管理的 CPU）不受此状态影响。

在延迟重新配置期间，仅允许执行一个 CPU 配置操作

在 `primary` 域处于延迟重新配置时，不要尝试对其执行多个 CPU 配置操作。如果尝试多个 CPU 配置请求，这些请求将被拒绝。

解决方法：执行以下操作之一：

- 取消该延迟重新配置，启动另一个延迟重新配置，然后请求自上次延迟重新配置以来丢失的配置更改。
- 重新引导具有错误 CPU 计数的控制域，然后在该域重新引导后更正分配。

资源分配

资源分配机制可通过资源分配约束在绑定时向域中分配资源。

资源分配约束是指在向域分配资源时系统必须满足的硬性要求。如果不能满足该约束，资源分配和域绑定均将失败。

注意 - 不要在两个域之间创建相互提供服务的循环依赖关系。这样的配置会产生单一故障点情况，其中一个域发生服务中断会导致另一个域变得不可用。循环依赖关系配置还会阻碍在初始绑定域后解除绑定。

Logical Domains Manager 不会阻止创建循环的域依赖关系。

如果域因为循环依赖关系而无法解除绑定，请删除导致该依赖关系的设备然后再尝试解除域绑定。

CPU 分配

从各个域的相同核心运行线程时，可能会遇到不可预测的性能降低现象。Oracle VM Server for SPARC 软件使用 CPU 关联性功能在逻辑域绑定过程中优化 CPU 分配，此操作发生在可以启动域之前。此功能尝试从分配给同一逻辑域的同一核心启动线程，因此此类分配可改善同一核心内线程间的高速缓存共享。

除非没有其他资源，否则 CPU 关联性将尝试避免在域之间共享核心。如果域已分配有部分核心并请求更多的导线束，则会首先绑定该部分核心的导线束，然后查找另一个可用核心以完成请求（如有必要）。

CPU 分配机制针对 CPU 资源使用以下约束：

- **整体核心约束。**此约束指定将 CPU 核心分配给域而不是虚拟 CPU。只要域未启用最大核心数约束，就可以分别使用 `ldm set-core` 或 `ldm set-vcpu` 命令来添加或删除

整体核心约束。域可以处于非活动、绑定或活动状态。但是，要满足应用约束的请求，必须具有足够的可用核心。以最差情况为例，如果与其他域共享核心的域需要整体核心约束，则要求空闲表中的核心可用，以便满足请求。以最佳情况为例，核心中的所有虚拟 CPU 都已在核心边界上，因此无需更改 CPU 资源即可应用约束。

- **最大核心数约束。**此约束指定可以分配给绑定域或活动域的最大核心数。

▼ 如何应用整体核心约束

在设置最大核心数约束之前，请确保域已启用了整体核心约束。

1. **在域上应用整体核心约束。**

```
primary# ldm set-core 1 domain-name
```

2. **验证域是否已启用整体核心约束。**

```
primary# ldm ls -o resmgmt domain-name
```

请注意，`max-cores` 已设置为 `unlimited`。在启用最大核心数约束之前，域无法与硬分区结合使用。

例 13-1 应用整体核心约束

此示例说明如何在 `ldg1` 域上应用整体核心约束。第一个命令用于应用约束，而第二个命令用于验证是否已启用约束。

```
primary# ldm set-core 1 ldg1
primary# ldm ls -o resmgmt ldg1
NAME
ldg1

CONSTRAINT
    cpu=whole-core
    max-cores=unlimited
```

▼ 如何应用最大核心数约束

在设置最大核心数约束之前，请确保域已启用了整体核心约束。

只能在非活动域上（而不能在绑定域或活动域上）启用、修改或禁用最大核心数约束。在控制域上更新最大核心数约束之前，必须先启动延迟重新配置。

1. **在域上启用最大核心数约束。**

```
primary# ldm set-domain max-cores=max-number-of-CPU-cores  domain-name
```

注 - 增加核心时不会影响与这些核心关联的加密单元。因此，系统不会自动向域中添加关联的加密单元。但是，只有当删除的是核心的最后一个虚拟 CPU 时，加密单元才会自动删除。此操作可防止加密单元被“孤立”。

2. 验证是否已启用整体核心约束。

```
primary# ldm ls -o resmgmt domain-name
```

3. 绑定和重新启动域。

```
primary# ldm bind domain-name
primary# ldm start domain-name
```

现在，可以将域与硬分区结合使用了。

例 13-2 应用最大核心数约束

此示例说明如何通过设置 max-cores 属性并验证约束是否已启用，将最大核心数约束为三个核心：

```
primary# ldm set-domain max-cores=3 ldg1
primary# ldm ls -o resmgmt ldg1
NAME
ldg1

CONSTRAINT
    cpu=whole-core
    max-cores=3
```

现在，可以将域与硬分区结合使用了。

以下示例会从未绑定且非活动的 ldg1 域删除最大核心数约束，但保留整体核心约束不变。

```
primary# ldm stop ldg1
primary# ldm unbind ldg1
primary# ldm set-domain max-cores=unlimited ldg1
```

或者，要同时从 ldg1 域删除最大核心数约束和整体核心约束，请按如下所示分配虚拟 CPU（而不是核心）：

```
primary# ldm set-vcpu 8 ldg1
```

在任何一种情况下，均绑定并重新启动域。

```
primary# ldm bind ldg1
primary# ldm start ldg1
```

整体核心约束和其他域功能之间的交互作用

本节介绍整体核心约束和以下功能之间的交互作用：

- “CPU 动态重新配置” [256]
- “动态资源管理” [256]

CPU 动态重新配置

整体核心约束与 CPU 动态重新配置 (dynamic reconfiguration, DR) 完全兼容。为域定义整体核心约束后，可以使用 `ldm add-core`、`ldm set-core` 或 `ldm remove-core` 命令更改活动域上的核心数。

但是，如果绑定域或活动域未处于延迟重新配置模式下，则其核心数不能超过最大核心数。最大核心数随最大核心约束进行设置，在启用整体核心约束时会自动启用该约束。任何不满足最大核心约束的 CPU DR 操作都将失败。

动态资源管理

整体核心约束与动态资源管理 (dynamic resource management, DRM) 不兼容。如果在使用整体核心约束的域上启用 DRM 策略，系统会自动禁用该策略。整体核心约束会保持启用状态。

即使在使用整体核心约束时无法启用 DRM 策略，您仍可以为域定义 DRM 策略。请注意，自动禁用某策略后，它仍保持活动状态。如果重新启动域时没有启用整体核心约束，则会自动重新启用 DRM 策略。

整体核心约束和 DRM 应按以下方式进行交互：

- 如果在域上设置整体核心约束，当您尝试在该域上启用 DRM 策略时系统将发出警告消息。
- 如果是在非活动域上使用 DRM 策略，则允许您在该域上启用整体核心约束。当该域转为活动状态且 DRM 策略处于启用状态时，系统会为该域自动禁用 DRM 策略。
- 如果在活动域或绑定域上启用 DRM 策略，则不允许您启用整体核心约束。

为系统配置硬分区

本节介绍 Oracle VM Server for SPARC 软件的硬分区，以及如何使用硬分区以符合 Oracle CPU 许可要求。

有关 Oracle 针对软件许可证的硬分区要求的信息，请参见 Partitioning: Server/Hardware Partitioning (http://www.oracle.com/us/corporate/pricing/partitioning-070609.pdf)（分区：服务器/硬件分区）。

- **CPU 核心和 CPU 线程。**Oracle VM Server for SPARC 软件在 SPARC T 系列和 SPARC M 系列平台和 Fujitsu M10 平台上运行。这些系统中使用的处理器都有多个 CPU 核心，其中每个核心都包含多个 CPU 线程。
- **硬分区和 CPU 整体核心。**从 Oracle VM Server for SPARC 2.0 发行版开始，可使用 CPU 整体核心配置强制执行硬分区。CPU 整体核心配置具有分配了 CPU 整体核心而不是各个 CPU 线程的域。默认情况下，域配置为使用 CPU 线程。

  将一个域绑定在整体核心配置中时，系统会将指定数量的 CPU 核心及其所有 CPU 线程置备到该域。使用 CPU 整体核心配置可限制可动态分配给绑定域或活动域的 CPU 核心数。
- **Oracle 硬分区许可。**为符合 Oracle 硬分区许可要求，您必须至少使用 Oracle VM Server for SPARC 2.0 发行版。您还必须按如下所示使用 CPU 整体核心：
  - 如果某个域运行的应用程序使用 Oracle 硬分区许可，则必须为该域配置 CPU 整体核心。
  - 如果域未运行使用 Oracle 硬分区许可的应用程序，则不需要为该域配置 CPU 整体核心。例如，如果您未在控制域中运行任何 Oracle 应用程序，则不需要为该域配置 CPU 整体核心。

检查域配置

您可以使用 `ldm list-o` 命令确定是否已为某个域配置 CPU 整体核心，以及如何列出分配给该域的 CPU 核心。

- 要确定域是否已配置 CPU 整体核心，请执行以下命令：

  ```
  primary# ldm list -o resmgmt domain-name
  ```

  验证整体核心约束是否显示在输出中，以及 `max-cores` 属性是否指定为域配置的最大 CPU 核心数。请参见 ldm(1M) 手册页。

  以下命令显示 `ldg1` 域已配置 CPU 整体核心且最大核心数为五个：

  ```
  primary# ldm list -o resmgmt ldg1
  NAME
  ldg1


  CONSTRAINT
     whole-core
     max-cores=5
  ```
- 绑定某域后，会为该域分配 CPU 核心。要列出分配给域的 CPU 核心，请执行以下命令：

```
primary# ldm list -o core domain-name
```

以下命令显示分配给 ldg1 域的核心：

```
primary# ldm list -o core ldg1
NAME
ldg1

CORE
CID PCPUSET
1 (8, 9, 10, 11, 12, 13, 14, 15)
2 (16, 17, 18, 19, 20, 21, 22, 23)
```

为域配置 CPU 整体核心

本节中的任务介绍如何创建具有 CPU 整体核心的新域、如何为现有域配置 CPU 整体核心，以及如何为 primary 域配置 CPU 整体核心。

注 - 在 Oracle VM Server for SPARC 2.2 发行版中，用于分配整体核心的 ldm 子命令发生了更改。

本节中的任务和示例使用 Oracle VM Server for SPARC 2.2 软件中添加的新命令。

如果您使用 2.0 或 2.1 版的 Logical Domains Manager 来为域分配整体核心，请使用 ldm add-vcpu -c、ldm set-vcpu -c 和 ldm remove-vcpu -c 命令分别代替 ldm add-core、ldm set-core 和 ldm remove-core 命令。

Oracle VM Server for SPARC 3.2 是以这种方式使用 -c 选项的最新软件发行版。

使用以下命令将域配置为使用 CPU 整体核心：

```
ldm set-core number-of-CPU-cores domain
```

此命令还为域指定最大 CPU 核心数（即 CPU 上限）。请参见 ldm(1M) 手册页。

CPU 上限和 CPU 核心分配是通过不同命令处理的。使用这些命令，您可以单独分配 CPU 核心、设置上限或进行这两个操作。即使没有 CPU 上限，也可以将分配单位设置为核心。但是，在您的 Oracle VM Server for SPARC 系统上配置硬分区时，不能在此模式下运行系统。

- 使用 add-core、set-core 或 rm-core 子命令，可将指定数量的 CPU 核心分配给域。
- 使用 create-domain 或 set-domain 子命令指定 max-cores 属性值，以设置 CPU 上限。

  如果要在 Oracle VM Server for SPARC 系统上配置硬分区，必须设置上限。

▼ 如何创建具有 CPU 整体核心的新域

注 - 只有在选择设置最大核心数约束时才需要停止域和解除域绑定。

1. **创建域。**

```
primary# ldm create domain-name
```

2. **为该域设置 CPU 整体核心数。**

```
primary# ldm set-core number-of-CPU-cores domain
```

3. **（可选）设置域的 `max-cores` 属性。**

```
primary# ldm set-domain max-cores=max-number-of-CPU-cores domain
```

4. **配置该域。**

   在进行此配置期间，请确保您使用 `ldm add-core`、`ldm set-core` 或 `ldm rm-core` 命令。

5. **绑定和启动域。**

```
primary# ldm bind domain-name
primary# ldm start domain-name
```

例 13-3 创建具有两个 CPU 整体核心的新域

此示例创建了具有两个 CPU 整体核心的域 `ldg1`。第一个命令用于创建 `ldg1` 域。第二个命令用于为 `ldg1` 域配置两个 CPU 整体核心。

此时，您可以根据如何创建具有 CPU 整体核心的新域 [259]的步骤 3 中所述的限制，对域进行进一步配置。

第三个和第四个命令显示如何绑定和启动 `ldg1` 域，此时您便可以使用 `ldg1` 域。

```
primary# ldm create ldg1
primary# ldm set-core 2 ldg1
...
primary# ldm bind ldg1
primary# ldm start ldg1
```

▼ 如何为现有域配置 CPU 整体核心

如果某个域已存在并且配置为使用 CPU 线程，则您可以更改其配置以使用 CPU 整体核心。

1. **（可选）停止并解除绑定域。**

只有当同时设置了最大核心数约束时，才需要执行此步骤。

```
primary# ldm stop domain-name
primary# ldm unbind domain-name
```

2. 为该域设置 CPU 整体核心数。

```
primary# ldm set-core number-of-CPU-cores domain
```

3. （可选）设置域的 **max-cores** 属性。

```
primary# ldm set-domain max-cores=max-number-of-CPU-cores domain
```

4. （可选）重新绑定并重新启动域。

只有当同时设置了最大核心数约束时，才需要执行此步骤。

```
primary# ldm bind domain-name
primary# ldm start domain-name
```

例 13-4 为现有域配置四个 CPU 整体核心

此示例通过为现有域 ldg1 配置四个 CPU 整体核心来更新该域的配置。

```
primary# ldm set-core 4 ldg1
```

▼ 如何为 Primary 域配置 CPU 整体核心

如果 primary 域配置为使用 CPU 线程，则您可以更改其配置以使用 CPU 整体核心。

1. （可选）将 **primary** 域置于延迟重新配置模式。

只有当需要修改 max-cores 属性时，才需要启动延迟重新配置。

```
primary# ldm start-reconf primary
```

2. 为 **primary** 域设置 CPU 整体核心数。

```
primary# ldm set-core number-of-CPU-cores primary
```

3. （可选）设置 **primary** 域的 **max-cores** 属性。

```
primary# ldm set-domain max-cores=max-number-of-CPU-cores primary
```

4. （可选）重新引导 **primary** 域。

根据系统配置，使用适当的过程重新引导 primary 域。请参见“重新引导配置了 PCIe 端点的根域” [71]。

只有当需要修改 max-cores 属性时，才需要重新引导域。

例 13-5 为控制域配置两个 CPU 整体核心

此示例在 primary 域上配置了 CPU 整体核心。第一个命令用于在 primary 域上启动延迟重新配置模式。第二个命令用于为 primary 域配置两个 CPU 整体核心。第三个命令用于将 max-cores 属性设置为 2，第四个命令用于重新引导 primary 域。

```
primary# ldm start-reconf primary
primary# ldm set-core 2 primary
primary# ldm set-domain max-cores=2 primary
primary# shutdown -i 5
```

仅当希望修改 max-cores 属性时，才需要执行可选步骤 1 和 4。

硬分区系统与其他 Oracle VM Server for SPARC 功能之间的交互

本节介绍硬分区系统如何与其他 Oracle VM Server for SPARC 功能进行交互。

CPU 动态重新配置

您可以对已配置 CPU 整体核心的域使用 CPU 动态重新配置。但是，您只能添加或删除整体 CPU 核心，而无法添加或删除个别 CPU 线程。系统的硬分区状态由 CPU 动态重新配置功能进行维护。此外，如果将 CPU 核心动态添加到域，则会强制执行最大核心数。因此，如果试图超过最大 CPU 数，则 CPU DR 命令将失败。

注 - 除非停止并解除绑定域，否则无法更改 max-cores 属性。所以，要在设置整体核心约束时所指定值的基础上增加最大核心数，您首先必须停止和解除绑定域。

使用以下命令可在绑定域或活动域中动态添加或删除 CPU 整体核心，并可为该域动态设置 CPU 整体核心数：

`ldm add-core` *number-of-CPU-cores domain*

`ldm rm-core` *number-of-CPU-cores domain*

`ldm set-core` *number-of-CPU-cores domain*

注 - 如果域处于非活动状态，这些命令还可调整域的最大 CPU 核心数。如果域为绑定域或活动域，这些命令不会影响域的最大 CPU 核心数。

例 13-6 为域动态添加两个 CPU 整体核心

此示例说明如何为 ldg1 域动态添加两个 CPU 整体核心。ldg1 域为活动域，且已配置 CPU 整体核心。第一个命令显示 ldg1 域处于活动状态。第二个命令显示 ldg1 域已配置

CPU 整体核心且最大 CPU 核心数为四个。第三个和第五个命令显示在添加两个 CPU 整体核心之前和之后分配给域的 CPU 核心。第四个命令用于为 `ldg1` 域动态添加两个 CPU 整体核心。

```
primary# ldm list ldg1
NAME    STATE   FLAGS   CONS  VCPU  MEMORY UTIL  UPTIME
ldg1    active  -n----  5000  16    2G     0.4%  5d 17h 49m
primary# ldm list -o resmgmt ldg1
NAME
ldg1

CONSTRAINT
   whole-core
        max-cores=4
primary# ldm list -o core ldg1
NAME
ldg1

CORE
CID PCPUSET
1 (8, 9, 10, 11, 12, 13, 14, 15)
2 (16, 17, 18, 19, 20, 21, 22, 23)
primary# ldm add-core 2 ldg1
primary# ldm list -o core ldg1
NAME
ldg1

CORE
CID PCPUSET
1 (8, 9, 10, 11, 12, 13, 14, 15)
2 (16, 17, 18, 19, 20, 21, 22, 23)
3 (24, 25, 26, 27, 28, 29, 30, 31)
4 (32, 33, 34, 35, 36, 37, 38, 39)
```

CPU 动态资源管理

动态资源管理 (dynamic resource management, DRM) 可用于自动管理某些域上的 CPU 资源。如果使用 DRM，则 DRM 策略不适用于配置有 CPU 整体核心的域。

DRM 策略可包括已配置 CPU 整体核心的域。但是，激活此类策略后，它会自动对该域禁用。该域将始终配置有 CPU 整体核心，除非此后为该域重新配置 CPU 线程（而不是 CPU 整体核心）。将域配置为使用 CPU 线程时，会自动为该域重新启用 DRM 策略。

电源管理

您可以为每个硬分区的域设置一个单独的电源管理 (power management, PM) 策略。

域重新引导或重新绑定

重新启动已配置 CPU 整体核心的域或重新启动整个系统时，该域将保持配置有 CPU 整体核心。在保持绑定的整个时间段内，域将使用相同的物理 CPU 核心。例如，如果重新引导某个域，则在重新引导之前和之后，该域都使用相同的物理 CPU 核心。或者，如果在域被绑定时关闭整个系统，则在再次打开该系统时，该域将配置有相同的物理 CPU 核心。如果您解除绑定某个域然后再重新绑定该域，或者使用新配置重新启动整个系统，则该域可能会使用不同的物理 CPU 核心。

为域分配物理资源

Logical Domains Manager 会自动选择要分配给域的物理资源。Oracle VM Server for SPARC 3.2 软件还允许经验丰富的管理员显式选择要分配给域或从域中删除的物理资源。

您显式分配的资源称为指定资源。自动分配的资源称为匿名资源。

注意 - 除非您是经验丰富的管理员，否则请勿分配已命名的资源。

您可以显式为控制域和来宾域分配物理资源。由于控制域保持活动状态，因此在您进行物理资源分配之前，控制域可能会选择性地处于延迟重新配置模式。或者，在您进行物理资源分配时，会自动触发延迟重新分配。请参见“管理控制域上的物理资源” [266]。有关物理资源限制的信息，请参见“针对管理域上物理资源的限制” [266]。

您可以显式为控制域和来宾域分配以下物理资源：

- 物理 CPU。通过设置 `cid` 属性，为域分配物理核心 ID。

  只有对要配置的系统的拓扑非常了解的管理员，才可使用 `cid` 属性。此高级配置功能会强制执行特定分配规则，并可能会影响系统的整体性能。

  您可以运行以下任意命令来设置此属性：

  ```
  ldm add-core cid=core-ID[,core-ID[,...]] domain-name

  ldm set-core cid=core-ID[,core-ID[,...]] domain-name

  ldm rm-core [-f] cid=core-ID[,core-ID[,...]] domain-name
  ```

  如果指定一个核心 ID 作为 `cid` 属性的值，则会显式将 *core-ID* 分配给域或从该域中删除。

注 - 不能使用 `ldm add-core` 命令将已命名的核心资源添加到已使用匿名核心资源的域。

- **物理内存。**通过设置 mblock 属性，为域分配一组连续的物理内存区域。以物理内存起始地址和大小来指定每个物理内存区域。

  只有对要配置的系统的拓扑非常了解的管理员，才可使用 mblock 属性。此高级配置功能会强制执行特定分配规则，并可能会影响系统的整体性能。

  您可以运行以下任意命令来设置此属性：

  ```
  ldm add-mem mblock=PA-start:size[,PA-start:size[,...]] domain-name
  ```

  ```
  ldm set-mem mblock=PA-start:size[,PA-start:size[,...]] domain-name
  ```

  ```
  ldm rm-mem mblock=PA-start:size[,PA-start:size[,...]] domain-name
  ```

  要分配内存块或从域中删除内存块，请设置 mblock 属性。有效值包括物理内存起始地址 (*PA-start*) 和内存块大小 (*size*)，并以冒号 (:) 进行分隔。

注 - 如果设置 mblock 或 cid 属性，则无法使用动态配置 (dynamic reconfiguration, DR) 在正运行的域之间移动内存或核心资源。要在域之间移动资源，请确保域处于绑定或非活动状态。有关管理控制域上物理资源的信息，请参见“管理控制域上的物理资源” [266]。

注 - 如果迁移域，将丢弃您使用 cid 和 mblock 属性分配的任何已命名资源。相反，域使用目标系统上的匿名资源。

您可以使用 ldm list-constraints 命令来查看针对域的资源约束。physical-bindings 约束指定哪些资源类型已通过物理方式分配给域。创建域时不会设置 physical-bindings 约束，直到将物理资源分配给该域。

physical-bindings 约束会在以下情况下设置为特定的值：

- 如果指定了 mblock 属性，则设置为 memory
- 如果指定了 cid 属性，则设置为 core
- 如果同时指定了 cid 和 mblock 属性，则设置为 core,memory

▼ 如何删除 physical-bindings 约束

要删除来宾域的物理绑定约束，必须先删除所有物理绑定的资源。

1. **取消绑定域。**

   ```
   primary# ldm unbind domain-name
   ```

2. **删除已命名的资源。**

   - 要删除指定的核心，请执行以下命令：

```
primary# ldm set-core cid=core-ID domain-name
```

- 要删除指定的内存，请执行以下命令：

```
primary# ldm set-mem mblock=PA-start:size domain-name
```

3. 添加 CPU 或内存资源。
   - 要添加 CPU 资源，请执行以下命令：

```
primary# ldm add-vcpu number domain-name
```

   - 要添加内存资源，请执行以下命令：

```
primary# ldm add-mem size[unit] domain-name
```

4. 重新绑定域。

```
primary# ldm bind domain-name
```

▼ 如何删除所有非物理绑定资源

要对没有物理绑定约束的来宾域进行约束，必须先删除所有非物理绑定的资源。

1. 取消绑定域。

```
primary# ldm unbind domain-name
```

2. 将资源数设置为 0。
   - 要设置 CPU 资源，请执行以下命令：

```
primary# ldm set-core 0 domain-name
```

   - 要设置内存资源，请执行以下命令：

```
primary# ldm set-mem 0 domain-name
```

3. 添加已物理绑定的 CPU 或内存资源。
   - 要添加 CPU 资源，请执行以下命令：

```
primary# ldm add-core cid=core-ID domain-name
```

   - 要添加内存资源，请执行以下命令：

```
primary# ldm add-mem mblock=PA-start:size domain-name
```

4. 重新绑定域。

```
primary# ldm bind domain-name
```

管理控制域上的物理资源

要从控制域约束或删除 `physical-bindings` 约束，请执行上一节中的相应步骤。但是，不是取消绑定域，而是将控制域放置在延迟重新配置中。

更改匿名资源和物理绑定命名资源之间的约束将自动触发延迟重新配置。通过使用 `ldm start-reconf primary` 命令，仍可以显式进入延迟重新配置。

与进行任何延迟重新配置更改一样，必须重新引导域（此情况下为控制域）才能完成此过程。

注 - 当控制域处于延迟重新配置模式时，您可以在控制域上使用 `ldm add-mem` 和 `ldm rm-mem` 命令来执行不受限制的内存分配。但是，您只能使用 `ldm set-core` 命令对控制域执行一次核心分配。

针对管理域上物理资源的限制

物理资源分配受到以下限制：

- 无法在相同域中进行物理和非物理内存绑定，或物理和非物理核心绑定。
- 可以在相同域中进行非物理内存绑定和物理核心绑定，或非物理核心绑定和物理内存绑定。
- 将物理资源添加到域中时，对应的资源类型会被约束为物理绑定。
- 尝试在 `physical-bindings=core` 的域中添加或删除匿名 CPU 的操作会失败。
- 对于未绑定的资源，只有在运行 `ldm bind` 命令时，才可分配和检查这些资源。
- 从域中删除物理内存时，必须删除之前添加的相同物理内存块。
- 物理内存范围不得重叠。
- 您只能使用 `ldm add-core cid=` 或 `ldm set-core cid=` 命令向域中分配物理资源。
- 如果使用 `ldm add-mem mblock=` 或 `ldm set-mem mblock=` 命令分配多个物理内存块，会立即检查地址和大小是否与其他绑定冲突。
- 在为域分配部分核心时，如果这些核心的剩余 CPU 空闲且可用，则该域可以使用整体核心语义。

使用内存动态重新配置

内存动态重新配置 (Memory Dynamic Reconfiguration, DR) 基于容量，通过此功能您可以在活动的逻辑域中添加或删除任意数量的内存。

使用内存 DR 功能的要求和限制如下：

- 您可以在任意域上执行内存 DR 操作。但是，给定时间内在一个域上只能执行一个内存 DR 操作。
- 内存 DR 功能可对给定操作中涉及的内存的地址和大小强制执行 256 MB 对齐。请参见“内存对齐” [268]。
- 不能使用内存 DR 功能向域中分配空闲内存池中的未对齐内存。请参见“添加未对齐的内存” [269]。

如果使用内存 DR 操作无法重新配置域的内存，则必须停止域才能重新配置内存。如果域是控制域，则必须先启动延迟重新配置。

在某些情况下，Logical Domains Manager 会将所请求的内存分配舍入到下一个为 8 KB 或 4 MB 倍数的最大内存量。以下示例显示了 `ldm list-domain -l` 命令的示例输出，其中的约束值小于实际分配的大小：

```
Memory:
         Constraints: 1965 M
         raddr          paddr5          size
         0x1000000      0x291000000     1968M
```

添加内存

如果域处于活动状态，可以使用 `ldm add-memory` 命令向域动态添加内存。如果指定内存大小大于域的当前内存大小，也可以使用 `ldm set-memory` 命令动态添加内存。

删除内存

如果域处于活动状态，可以使用 `ldm remove-memory` 命令从域中动态删除内存。如果指定内存大小小于域的当前内存大小，也可以使用 `ldm set-memory` 命令动态删除内存。

内存删除操作可能要运行很长时间。通过对指定域运行 `ldm list -l` 命令，可以跟踪 `ldm remove-memory` 命令的进度。

通过中断 `ldm remove-memory` 命令（按 Ctrl-C）或发出 `ldm cancel-operation memdr` 命令，可以取消正在进行的删除请求。如果取消内存删除请求，则只会影响该删除请求中的未处理部分，即，仍有待从域中删除的内存量。

部分内存 DR 请求

可能只能部分完成在域中动态添加或删除内存的请求。此结果分别取决于要添加或删除的适用内存的可用性。

注 - 从域中删除内存后，在将其添加到其他域之前，会先进行清除。

重新配置控制域内存

您可以使用内存 DR 功能重新配置控制域的内存。如果无法在控制域上执行内存 DR 请求，则必须先启动延迟重新配置。

从活动域中删除大量内存时可能不适合使用内存 DR，因为内存 DR 操作可能要运行很长时间。特别要指出的是，在系统初始配置期间，应使用延迟重新配置来减少控制域中的内存。

减少控制域的内存

使用延迟重新配置而不是内存 DR 从默认出厂初始配置中减少控制域的内存。在该默认配置中，控制域拥有主机系统的所有内存。之所以不太适合使用内存 DR 功能来实现此目的，是因为并不保证活动域能够添加（或者更典型的情况下能够放弃）所有请求的内存。当然，该域中运行的 OS 会尽最大可能来完成请求。另外，内存删除操作可能要运行很长时间。涉及到大量内存操作时这些问题会扩大化，正如初始减少控制域内存的情况。

因此，请按照以下步骤使用延迟重新配置：

1. 使用 `ldm start-reconf primary` 命令将控制域置于延迟重新配置模式下。
2. 根据需要对控制域拥有的主机系统资源进行分区。
3. 如果需要，使用 `ldm cancel-reconf` 命令撤消步骤 2 中的操作，重新从头开始。
4. 重新引导控制域以使重新配置更改生效。

动态重新配置和延迟重新配置

如果控制域中存在暂挂的延迟重新配置，系统将拒绝任何其他域的内存重新配置请求。如果控制域中不存在暂挂的延迟重新配置，系统将拒绝任何不支持内存 DR 的域的内存重新配置请求。对于这些域，该请求将转换为延迟重新配置请求。

内存对齐

内存重新配置请求具有不同的对齐要求，具体取决于将应用请求的域的状态。

活动域的内存对齐

- **动态添加和删除。**内存块的地址和大小为对齐的 256 MB，以用于进行动态添加和动态删除。最小操作大小为 256 MB。

  将拒绝大于绑定大小的未对齐请求或删除请求。

  使用以下命令调整内存分配：

  - `ldm add-memory`。如果在此命令中指定 `--auto-adj` 选项，则要添加的内存量为对齐的 256 MB，这可能会增加实际添加到域的内存量。
  - `ldm remove-memory`。如果在此命令中指定 `--auto-adj` 选项，则要删除的内存量为对齐的 256 MB，这可能会减少实际从域中删除的内存量。
  - `ldm set-memory`。可将此命令视为添加或删除操作。如果指定 `--auto-adj` 选项，则要添加或删除的内存量为对齐的 256 MB，如上所述。请注意，此对齐可能会增大域的最终内存大小。
- **延迟重新配置。**内存块的地址和大小为对齐的 4 MB。如果发出的未对齐的请求，系统会将该请求舍入为对齐的 4 MB。

绑定域的内存对齐

绑定域的内存块的地址和大小为对齐的 4 MB。如果发出的未对齐的请求，系统会将该请求舍入为对齐的 4 MB。因此，所得到的域内存大小可能会超过您指定的大小。

对于 `ldm add-memory`、`ldm set-memory` 和 `ldm remove-memory` 命令，`--auto-adj` 选项会将最终内存的大小舍入为对齐的 256 MB。因此，所得到的内存可能会超过您指定的大小。

非活动域的内存对齐

对于 `ldm add-memory`、`ldm set-memory` 和 `ldm remove-memory` 命令，`--auto-adj` 选项会将最终内存的大小舍入为对齐的 256 MB。不存在对非活动域的对齐要求。“绑定域的内存对齐” [269]中所述的限制在绑定非活动域后生效。

添加未对齐的内存

内存 DR 功能对动态添加到活动域或从活动域中删除的内存的地址和大小强制执行 256 MB 内存对齐。因此，不能使用内存 DR 删除活动域中的任何未对齐内存。

此外，也不能使用内存 DR 向活动域中添加空闲内存池中的任何未对齐内存。

在所有对齐的内存都已分配后，可以使用 `ldm add-memory` 命令将剩余的未对齐内存添加到绑定域或非活动域。还可以使用此命令借助延迟重新配置操作将剩余的未对齐内存添加到控制域。

以下示例说明如何将两个剩余的 128 MB 内存块添加到 `primary` 和 `ldom1` 域中。`ldom1` 域处于绑定状态。

以下命令可在控制域上启动延迟重新配置操作。

```
primary# ldm start-reconf primary
Initiating a delayed reconfiguration operation on the primary domain.
All configuration changes for other domains are disabled until the
primary domain reboots, at which time the new configuration for the
primary domain also takes effect.
```

以下命令可向控制域中添加一个 128 MB 内存块。

```
primary# ldm add-memory 128M primary
------------------------------------------------------------------------------
Notice: The primary domain is in the process of a delayed reconfiguration.
Any changes made to the primary domain will only take effect after it reboots.
------------------------------------------------------------------------------

primary# ldm list
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
primary          active     -ndcv-  SP      8     2688M    0.1%  23d 8h 8m

primary# ldm list
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
primary          active     -n-cv-  SP      8     2560M    0.5%  23d 8h 9m
ldom1            bound      ------  5000    1     524M
```

以下命令可向 `ldom1` 域添加另外的 128 MB 内存块。

```
primary# ldm add-mem 128M ldom1
primary# ldm list
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
primary          active     -n-cv-  SP      8     2560M    0.1%  23d 8h 9m
ldom1            bound      ------  5000    1     652M
```

内存 DR 示例

以下示例说明如何执行内存 DR 操作。有关相关 CLI 命令的信息，请参见 `ldm`(1M) 手册页。

例 13-7 活动域上的内存 DR 操作

本示例说明如何动态向活动域 `ldom1` 添加内存和从中删除内存。

ldm list 的输出显示了 Memory (内存) 字段中各个域的内存。

```
primary# ldm list
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
primary          active     -n-cv-  SP      4     27392M   0.4%  1d 22h 53m
ldom1            active     -n----  5000    2     2G       0.4%  1d 1h 23m
ldom2            bound      ------  5001    2     200M
```

以下 ldm add-mem 命令将退出并显示错误，因为您必须将内存指定为 256 MB 的倍数。下一个 ldm add-mem 命令使用 --auto-adj 选项，这样，即使您将 200M 指定为要添加的内存量，系统也会将内存量舍入为 256 MB。

```
primary# ldm add-mem 200M ldom1
The size of memory must be a multiple of 256MB.

primary# ldm add-mem --auto-adj 200M ldom1
Adjusting request size to 256M.
The ldom1 domain has been allocated 56M more memory
than requested because of memory alignment constraints.

primary# ldm list
NAME            STATE   FLAGS   CONS    VCPU    MEMORY  UTIL    UPTIME
primary         active  -n-cv-  SP      4       27392M  5.0%    8m
ldom1           active  -n----  5000    2       2304M   0.5%    1m
ldom2           bound   ------  5001    2       200M
```

ldm rm-mem 命令将退出并显示错误，因为您必须将内存指定为 256 MB 的倍数。将 --auto-adj 选项添加到同一命令后，内存删除将成功，因为内存量会向下舍入到邻近的 256 MB 边界。

```
primary# ldm rm-mem --auto-adj 300M ldom1
Adjusting requested size to 256M.
The ldom1 domain has been allocated 44M more memory
than requested because of memory alignment constraints.

primary# ldm list
NAME            STATE   FLAGS   CONS    VCPU    MEMORY  UTIL    UPTIME
primary         active  -n-cv-  SP      4       27392M  0.3%    8m
ldom1           active  -n----  5000    2       2G      0.2%    2m
ldom2           bound   ------  5001    2       200M
```

例 13-8 绑定域上的内存 DR 操作

本示例说明如何向绑定域 ldom2 添加内存和从中删除内存。

ldm list 的输出显示了 Memory (内存) 字段中各个域的内存。第一个 ldm add-mem 命令可向 ldom2 域中添加 100 MB 的内存。下一个 ldm add-mem 命令中指定了 --auto-adj 选项，这可导致将额外的 112 MB 内存动态添加到 ldom2 中。

ldm rm-mem 命令可从 ldom2 域中动态删除 100 MB 的内存。如果在同一命令中指定 --auto-adj 选项以删除 300 MB 内存，内存量将向下舍入到邻近的 256 MB 边界。

```
primary# ldm list
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
primary          active     -n-cv-  SP      4     27392M   0.4%  1d 22h 53m
ldom1            active     -n----  5000    2     2G       0.4%  1d 1h 23m
ldom2            bound      ------  5001    2     200M

primary# ldm add-mem 100M ldom2

primary# ldm list
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
primary          active     -n-cv-  SP      4     27392M   0.5%  1d 22h 54m
ldom1            active     -n----  5000    2     2G       0.2%  1d 1h 25m
ldom2            bound      ------  5001    2     300M

primary# ldm add-mem --auto-adj 100M ldom2
Adjusting request size to 256M.
The ldom2 domain has been allocated 112M more memory
than requested because of memory alignment constraints.

primary# ldm list
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
primary          active     -n-cv-  SP      4     27392M   0.4%  1d 22h 55m
ldom1            active     -n----  5000    2     2G       0.5%  1d 1h 25m
ldom2            bound      ------  5001    2     512M

primary# ldm rm-mem 100M ldom2
primary# ldm list
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
primary          active     -n-cv-  SP      4     27392M   3.3%  1d 22h 55m
ldom1            active     -n----  5000    2     2G       0.2%  1d 1h 25m
ldom2            bound      ------  5001    2     412M

primary# ldm rm-mem --auto-adj 300M ldom2
Adjusting request size to 256M.
The ldom2 domain has been allocated 144M more memory
than requested because of memory alignment constraints.

primary# ldm list
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
primary          active     -n-cv-  SP      4     27392M   0.5%  1d 22h 55m
ldom1            active     -n----  5000    2     2G       0.2%  1d 1h 26m
ldom2            bound      ------  5001    2     256M
```

例 13-9 设置域内存大小

本示例说明如何使用 `ldm set-memory` 命令向域中添加内存和从中删除内存。

`ldm list` 的输出显示了 Memory (内存) 字段中各个域的内存。

```
primary# ldm list
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
primary          active     -n-cv-  SP      4     27392M   0.5%  1d 22h 55m
```

```
ldom1            active     -n----  5000    2     2G       0.2%  1d 1h 26m
ldom2            bound      ------  5001    2     256M
```

以下 `ldm set-mem` 命令会尝试将 `primary` 域的大小设置为 3400 MB。最终错误表明指定的值不在 256 MB 边界上。将 `--auto-adj` 选项添加到同一命令后，可以成功删除某些内存并保持在 256 MB 边界上。此命令还会发出警告，以指出并非所有请求的内存都可以删除，因为域可能正在使用某些内存。

```
primary# ldm set-mem 3400M primary
An ldm set-mem 3400M command would remove 23992MB, which is not a multiple
of 256MB. Instead, run ldm rm-mem 23808MB to ensure a 256MB alignment.

primary# ldm set-mem --auto-adj 3400M primary
Adjusting request size to 3.4G.
The primary domain has been allocated 184M more memory
than requested because of memory alignment constraints.
Only 9472M of memory could be removed from the primary domain
because the rest of the memory is in use.
```

下一个 `ldm set-mem` 命令会将处于绑定状态的 `ldom2` 域的内存大小设置为 690 MB。如果向同一命令中添加 `--auto-adj` 选项，将动态向 `ldom2` 中添加额外的 78 MB 内存以保持在 256 MB 边界上。

```
primary# ldm set-mem 690M ldom2
primary# ldm list
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
primary          active     -n-cv-  SP      4     17920M   0.5%  1d 22h 56m
ldom1            active     -n----  5000    2     2G       0.6%  1d 1h 27m
ldom2            bound      ------  5001    2     690M

primary# ldm set-mem --auto-adj 690M ldom2
Adjusting request size to 256M.
The ldom2 domain has been allocated 78M more memory
than requested because of memory alignment constraints.

primary# ldm list
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
primary          active     -n-cv-  SP      4     17920M   2.1%  1d 22h 57m
ldom1            active     -n----  5000    2     2G       0.2%  1d 1h 27m
ldom2            bound      ------  5001    2     768M
```

使用资源组

资源组提供另一种方式来查看系统中的资源。资源基于处理器核心、内存与 I/O 总线之间的底层物理关系进行分组。不同平台甚至相同服务器系列（例如 SPARC T5-2 和 SPARC T5-8）内的不同平台配置可以具有不同资源组，这些资源组反映硬件中的差异。使用 `ldm list-resrc-group` 命令查看资源组信息。

资源组的成员由硬件配置以静态方法进行定义。可以使用 `ldm remove-core` 和 `ldm remove-memory` 命令对特定资源组中的资源进行操作。

- `remove-core` 子命令指定要从域中删除的 CPU 核心数。使用 `-g` 选项指定资源组时，选择删除的核心全部来自该资源组。
- `remove-memory` 子命令从逻辑域中删除指定的内存量。使用 `-g` 选项指定资源组时，选择删除的内存全部来自该资源组。

有关这些命令的信息，请参见 `ldm`(1M) 手册页。

UltraSPARC T2、UltraSPARC T2 Plus、SPARC T3 和 SPARC T4 平台不支持资源组信息。`ldm list-rsrc-group` 命令不显示有关这些平台的任何信息并且 `ldm remove-core` 和 `ldm remove-memory` 命令的 `-g` 变体不起作用。

有关示例，请参见“列出资源组信息” [282]。

使用电源管理

要启用电源管理 (Power Management, PM)，首先需要在至少为 3.0 版的 ILOM 固件中设置 PM 策略。本节汇总了所需的信息，以便能够在 Oracle VM Server for SPARC 软件中使用 PM。

有关 PM 功能和 ILOM 功能的更多信息，请参见以下内容：

- 第 19 章 使用电源管理
- 《*Oracle Integrated Lights Out Manager (ILOM) 3.0 CLI* 过程指南》中的“监视功耗”
- 《*Oracle Integrated Lights Out Manager (ILOM) 3.0* 功能更新和发行说明》

使用动态资源管理

可以使用策略来确定如何自动执行 DR 活动。目前，只能创建相应策略来控制虚拟 CPU 的动态资源管理。

注意 - 以下限制会影响 CPU 动态资源管理 (dynamic resource management, DRM)：

- 在 UltraSPARC T2 和 UltraSPARC T2 Plus 平台上，如果设置了 PM 弹性策略，则无法启用 DRM。
- 在 UltraSPARC T2 和 UltraSPARC T2 Plus 平台上，如果启用了 DRM，任何从性能策略到弹性策略的更改均会延迟。
- 确保在执行域迁移操作之前禁用 CPU DRM，否则将看到错误消息。
- DRM 策略不适用于配置有整体核心约束的域。如果试图在设置了 `whole-core` 约束的域上使用 DRM，则将看到错误消息。
- 如果设置了 PM 弹性策略，则只有当固件支持标准利用率 (8.2.0) 时，才能使用 DRM。

资源管理策略用于指定可以自动在逻辑域中添加和删除虚拟 CPU 的条件。使用 `ldm add-policy`、`ldm set-policy` 和 `ldm remove-policy` 命令管理策略：

```
ldm add-policy [enable=yes|no] [priority=value] [attack=value] [decay=value]
  [elastic-margin=value] [sample-rate=value] [tod-begin=hh:mm[:ss]]
  [tod-end=hh:mm[:ss]] [util-lower=percent] [util-upper=percent] [vcpu-min=value]
  [vcpu-max=value] name=policy-name domain-name...
ldm set-policy [enable=[yes|no]] [priority=[value]] [attack=[value]] [decay=[value]]
  [elastic-margin=[value]] [sample-rate=[value]] [tod-begin=[hh:mm:ss]]
  [tod-end=[hh:mm:ss]] [util-lower=[percent]] [util-upper=[percent]] [vcpu-min=[value]]
  [vcpu-max=[value]] name=policy-name domain-name...
ldm remove-policy [name=]policy-name... domain-name
```

有关这些命令以及创建资源管理策略的信息，请参见 `ldm`(1M) 手册页。

策略在 `tod-begin` 和 `tod-end` 属性指定的时间内有效。`tod-begin` 指定的时间必须早于 `tod-end` 指定的时间（24 小时制）。默认情况下，`tod-begin` 和 `tod-end` 属性的值分别为 00:00:00 和 23:59:59。使用默认值时，此策略始终有效。

此策略使用 `priority` 属性的值为动态资源管理 (dynamic resource management, DRM) 策略指定优先级。优先级值用来确定单个域上的 DRM 策略之间的关系，以及单个系统上启用了 DRM 的域之间的关系。数值越小，优先级越高。有效值介于 1 和 9999 之间。默认值为 99。

`priority` 属性的行为取决于空闲 CPU 资源池的可用性，如下所示：

- **池中有空闲的 CPU 资源。**在这种情况下，当为单个域定义了多个重叠策略时，`priority` 属性将确定哪个 DRM 策略有效。
- **池中没有空闲的 CPU 资源。**在这种情况下，`priority` 属性指定资源能否从低优先级域动态移动到同一系统上的高优先级域。域的优先级是由该域的有效 DRM 策略指定的优先级。

  例如，高优先级域可以从另一个具有较低优先级 DRM 策略的域获取 CPU 资源。该资源获取功能仅适用于已启用 DRM 策略的域。具有相同 `priority` 值的域将不受此功能的影响。因此，如果对所有策略都使用默认优先级，则域将无法从低优先级域获取资源。为了利用此功能，请调整 `priority` 属性的值，以便它们具有不相等的值。

例如，ldg1 和 ldg2 域都具有有效的 DRM 策略。ldg1 域的 priority 属性为 1，该值的优先级比 ldg2 域的 priority 属性值 (2) 高。在以下情况下，ldg1 域可以从 ldg2 域中动态删除 CPU 资源，并将该 CPU 资源分配给它自身：

- ldg1 域需要另一个 CPU 资源。
- 空闲 CPU 资源池已经耗尽。

策略使用 util-high 和 util-low 属性值指定 CPU 利用率的阈值上限和下限。如果利用率超出 util-high 的值，将向域中添加虚拟 CPU，直到利用率的值位于 vcpu-min 和 vcpu-max 值之间。如果利用率降到 util-low 值以下，将从域中删除虚拟 CPU，直到利用率的值位于 vcpu-min 和 vcpu-max 值之间。如果达到 vcpu-min 值，将无法再动态删除任何虚拟 CPU。如果达到 vcpu-max 值，将无法再动态添加任何虚拟 CPU。

例 13-10 添加资源管理策略

例如，观察系统的典型利用率数周后，您可以设置策略，以优化资源使用情况。资源占用高峰期是每天上午 9:00 至下午 6:00（太平洋时间），资源占用低谷期是每天下午 6:00 至次日上午 9:00（太平洋时间）。

根据对系统利用率的观察，您决定基于系统总体利用率创建以下高峰期和低谷期策略：

- **高峰期**：每天上午 9:00 至下午 6:00（太平洋时间）
- **低谷期**：每天下午 6:00 至次日上午 9:00（太平洋时间）

以下 ldm add-policy 命令可在 ldom1 域上创建利用率高峰期内要使用的 high-usage 策略。

以下 high-usage 策略执行以下操作：

- 通过设置 tod-begin 和 tod-end 属性，将开始和结束时间分别指定为上午 9:00 和下午 6:00。
- 通过设置 util-lower 和 util-upper 属性，将执行策略分析的下限和上限分别指定为 25% 和 75%。
- 通过设置 vcpu-min 和 vcpu-max 属性，将最小和最大虚拟 CPU 数分别指定为 2 和 16。
- 通过设置 attack 属性，将任一资源控制周期内要添加的最大虚拟 CPU 数指定为 1。
- 通过设置 decay 属性，将任一资源控制周期内要删除的最大虚拟 CPU 数指定为 1。
- 通过设置 priority 属性，将此策略的优先级指定为 1。优先级为 1 表示即使其他策略可以生效，也强制执行此策略。
- 通过设置 name 属性，将策略文件的名称指定为 high-usage。
- 对于未指定的属性（例如 enable 和 sample-rate），使用其默认值。请参见 ldm(1M) 手册页。

```
primary# ldm add-policy tod-begin=09:00 tod-end=18:00 util-lower=25 util-upper=75 \
vcpu-min=2 vcpu-max=16 attack=1 decay=1 priority=1 name=high-usage ldom1
```

以下 ldm add-policy 命令可在 ldom1 域上创建利用率低谷期内要使用的 med-usage 策略。

以下 med-usage 策略执行以下操作：

- 通过设置 tod-begin 和 tod-end 属性，将开始和结束时间分别指定为下午 6:00 和上午 9:00。
- 通过设置 util-lower 和 util-upper 属性，将执行策略分析的下限和上限分别指定为 10% 和 50%。
- 通过设置 vcpu-min 和 vcpu-max 属性，将最小和最大虚拟 CPU 数分别指定为 2 和 16。
- 通过设置 attack 属性，将任一资源控制周期内要添加的最大虚拟 CPU 数指定为 1。
- 通过设置 decay 属性，将任一资源控制周期内要删除的最大虚拟 CPU 数指定为 1。
- 通过设置 priority 属性，将此策略的优先级指定为 1。优先级为 1 表示即使其他策略可以生效，也强制执行此策略。
- 通过设置 name 属性，将策略文件的名称指定为 high-usage。
- 对于未指定的属性（例如 enable 和 sample-rate），使用其默认值。请参见 ldm(1M) 手册页。

```
primary# ldm add-policy tod-begin=18:00 tod-end=09:00 util-lower=10 util-upper=50 \
 vcpu-min=2 vcpu-max=16 attack=1 decay=1 priority=1 name=med-usage ldom1
```

列出域资源

本节显示 ldm 子命令的语法用法，并定义一些输出项（例如，标志和利用率统计信息），同时，还会提供类似于实际显示内容的输出示例。

计算机可读的输出

如果要创建使用 ldm list 命令输出的脚本，请始终使用 -p 选项来生成计算机可读形式的输出。

要查看所有 ldm 子命令的语法用法，请使用以下命令：

```
primary# ldm --help
```

有关 ldm 子命令的更多信息，请参见 ldm(1M) 手册页。

标志定义

在域的输出中 (`ldm list`) 可以显示以下标志。如果在命令中使用可解析的长选项 (`-l -p`)，则会拼写出标志，例如，`flags=normal,control,vio-service`。否则，系统将显示字母缩写，例如，`-n-cv-`。列表标志值与位置相关。在从左至右排列的六列中的每一列中可以出现以下值。

第 1 列 – 启动或停止域

- `s` – 启动或停止

第 2 列 – 域状态

- `n` – 正常
- `t` – 转换
- `d` – 由于缺少资源而无法启动的降级域

第 3 列 – 重新配置状态

- `d` – 延迟重新配置
- `r` – 内存动态重新配置

第 4 列 – 控制域

- `c` – 控制域

第 5 列 – 服务域

- `v` – 虚拟 I/O 服务域

第 6 列 – 迁移状态

- `s` – 迁移操作中的源域
- `t` – 迁移操作中的目标域
- `e` – 迁移过程中发生错误

利用率统计信息定义

通过在 `ldm list` 命令中使用长 (`-l`) 选项，可以显示每个虚拟 CPU 的使用率统计信息 (`UTIL`)。统计信息是虚拟 CPU 代表客操作系统执行操作所用的时间的百分比。除非虚拟 CPU 被移交给虚拟机管理程序，否则将其视为代表客操作系统执行。如果客操作系统没有将虚拟 CPU 移交给虚拟机管理程序，则客操作系统中的 CPU 使用率始终显示为 `100%`。

为逻辑域报告的使用率统计信息是域中虚拟 CPU 的虚拟 CPU 使用率平均值。标准利用率统计信息 (NORM) 是虚拟 CPU 代表客操作系统执行操作时所用时间的百分比。此值将此类操作作为周期跳步考虑。仅当系统运行的系统固件版本至少为 8.2.0 时，才能使用标准虚拟化。

如果 PM 不执行周期跳步操作，则 100% 利用率等于 100% 标准利用率。如果 PM 将周期跳步调整为四个八，则 100% 利用率等于 50% 利用率，也就是说，CPU 实际上只能利用可能的周期数的一半。因此，充分利用的 CPU 具有 50% 的标准利用率。使用 `ldm list` 或 `ldm list -l` 命令可显示虚拟 CPU 和客操作系统的标准利用率。

查看各种列表

- 要查看当前安装的软件版本，请执行以下命令：

  ```
  primary# ldm -V
  ```

- 要生成所有域的短列表，请执行以下命令：

  ```
  primary# ldm list
  ```

- 要生成所有域的长列表，请执行以下命令：

  ```
  primary# ldm list -l
  ```

- 要生成所有域的扩展列表，请执行以下命令：

  ```
  primary# ldm list -e
  ```

- 要为所有域生成可解析的计算机可读列表，请执行以下命令：

  ```
  primary# ldm list -p
  ```

- 通过输入下列 *format* 选项中的一个或多个选项，可以通过资源子集的形式来生成输出。如果指定多种格式，请使用逗号分隔各项，其间不留空格。

  ```
  primary# ldm list -o resource[,resource...] domain-name
  ```

  - `console` – 输出包含虚拟控制台 (`vcons`) 和虚拟控制台集中器 (`vcc`) 服务
  - `core` – 输出包含有关已分配整体核心的域的信息
  - `cpu` – 输出包含有关虚拟 CPU (`vcpu`)、物理 CPU (`pcpu`) 和核心 ID 的信息
  - `crypto` – 加密单元输出中包含模运算单元 (Modular Arithmetic Unit, `mau`) 和任何其他受支持的加密单元，例如控制字队列 (Control Word Queue, CWQ)
  - `disk` – 输出包含虚拟磁盘 (`vdisk`) 和虚拟磁盘服务器 (`vds`)
  - `domain-name` – 输出包含变量 (`var`)、主机 ID (`hostid`)、域状态、标志、UUID 和软件状态
  - `memory` – 输出包含 `memory`
  - `network` – 输出包含介质访问控制 (`mac`) 地址、虚拟网络交换机 (`vsw`) 和虚拟网络 (`vnet`) 设备

- `physio` – 物理输入/输出包含外设部件互连 (`pci`) 和网络接口单元 (`niu`)
- `resmgmt` – 输出包含动态资源管理 (dynamic resource management, DRM) 策略信息，指出当前运行的策略并列出与整体核心配置相关的约束
- `serial` – 输出包含虚拟逻辑域通道 (`vldc`) 服务、虚拟逻辑域通道客户机 (`vldcc`)、虚拟数据平面通道客户机 (`vdpcc`) 和虚拟数据平面通道服务 (`vdpcs`)
- `stats` – 输出包含与资源管理策略相关的统计信息
- `status` – 输出包含有关正在执行的域迁移的状态

以下示例显示您可以指定的各个输出子集。

- 要列出控制域的 CPU 信息，请执行以下命令：

```
primary# ldm list -o cpu primary
```

- 要列出来宾域的域信息，请执行以下命令：

```
primary# ldm list -o domain ldm2
```

- 要列出来宾域的内存和网络信息，请执行以下命令：

```
primary# ldm list -o network,memory ldm1
```

- 要列出来宾域的 DRM 策略信息，请执行以下命令：

```
primary# ldm list -o resmgmt,stats ldm1
```

- 要显示域的变量及变量值，请执行以下命令：

```
primary# ldm list-variable variable-name domain-name
```

例如，以下命令可显示 `ldg1` 域上 `boot-device` 变量的值：

```
primary# ldm list-variable boot-device ldg1
boot-device=/virtual-devices@100/channel-devices@200/disk@0:a
```

- 要列出绑定到域的资源，请执行以下命令：

```
primary# ldm list-bindings domain-name
```

- 要列出 SP 上存储的逻辑域配置，请执行以下命令：

`ldm list-config` 命令可列出服务处理器上已存储的逻辑域配置。将此命令与 `-r` 选项一起使用，可列出允许控制域上存在自动保存文件的配置。

有关配置的更多信息，请参见“管理域配置” [287]。有关更多示例，请参见 `ldm`(1M) 手册页。

```
primary# ldm list-config
factory-default
3guests
foo [next poweron]
primary
reconfig-primary
```

配置名称右侧的标签具有以下含义：

- [current] – 最后引导的配置，仅当符合当前运行的配置时；也就是说，直到您启动重新配置。重新配置后，注释将更改为 [next poweron]。
- [next poweron] – 下次关开机循环时将使用的配置。
- [degraded] – 该配置是先前引导的配置的降级版本。

- 要列出所有服务器资源（绑定资源和非绑定资源），请执行以下命令：

```
primary# ldm list-devices -a
```

- 要列出可供分配的内存量，请执行以下命令：

```
primary# ldm list-devices mem
MEMORY
    PA                  SIZE
    0x14e000000         2848M
```

- 要确定内存的哪些部分无法用于逻辑域：

```
primary# ldm list-devices -a mem
MEMORY
    PA                  SIZE            BOUND
    0x0                 57M             _sys_
    0x3900000           32M             _sys_
    0x5900000           94M             _sys_
    0xb700000           393M            _sys_
    0x24000000          192M            _sys_
    0x30000000          255G            primary
    0x3ff0000000        64M             _sys_
    0x3ff4000000        64M             _sys_
    0x3ff8000000        128M            _sys_
    0x80000000000       2G              ldg1
    0x80080000000       2G              ldg2
    0x80100000000       2G              ldg3
    0x80180000000       2G              ldg4
    0x80200000000       103G
    0x81bc0000000       145G            primary
```

- 要列出可用的服务，请执行以下命令：

```
primary# ldm list-services
```

列出约束

对于 Logical Domains Manager，约束是您希望分配给特定域的一个或多个资源。您可能会接收到要求添加到域中的所有资源，也可能会得不到任何资源，这取决于可用资源。list-constraints 子命令可列出您要求分配给域的那些资源。

- 要列出一个域的约束，请执行以下命令：

  ```
  # ldm list-constraints domain-name
  ```

- 要以 XML 格式列出特定域的约束，请执行以下命令：

  ```
  # ldm list-constraints -x domain-name
  ```

- 要以可解析格式列出所有域的约束，请执行以下命令：

  ```
  # ldm list-constraints -p
  ```

列出资源组信息

可以使用 `ldm list-rsrc-group` 命令显示资源组信息。

以下命令显示所有资源组的信息：

```
primary# ldm list-rsrc-group
NAME                      CORE MEMORY IO
/SYS/CMU4                 12   256G   4
/SYS/CMU5                 12   256G   4
/SYS/CMU6                 12   128G   4
/SYS/CMU7                 12   128G   4
```

与其他 `ldm list-*` 命令类似，可以指定选项来显示可解析的输出、详细输出以及有关特定资源组和域的信息。有关更多信息，请参见 ldm(1M) 手册页。

以下示例使用 `-l` 选项来显示有关 `/SYS/CMU5` 资源组的详细信息。

```
primary# ldm list-rsrc-group -l /SYS/CMU5
NAME                                  CORE   MEMORY    IO
/SYS/CMU5                             12     256G      4

CORE
    CID                                          BOUND
    192, 194, 196, 198, 200, 202, 208, 210       primary
    212, 214, 216, 218                           primary

MEMORY

    PA              SIZE              BOUND
    0xc0000000000   228M              ldg1
    0xc0030000000   127G              primary
    0xc1ffc000000   64M               _sys_
    0xd0000000000   130816M           primary
    0xd1ffc000000   64M               _sys_

IO
    DEVICE          PSEUDONYM         BOUND
    pci@900         pci_24            primary
    pci@940         pci_25            primary
    pci@980         pci_26            primary
```

```
pci@9c0         pci_27            primary
```

使用 Perf-Counter 属性

使用性能寄存器访问权限控制功能，可以获取、设置以及取消设置域对特定性能寄存器组的访问权限。

使用 `ldm add-domain` 和 `ldm set-domain` 命令为 `perf-counters` 属性指定值。如果未指定 `perf-counters` 值，则值为 `htstrand`。请参见 `ldm`(1M) 手册页。

可以为 `perf-counters` 属性指定以下值：

`global`	授予域对其已分配资源可以访问的全局性能计数器的访问权限。一次仅一个域可以访问全局性能计数器。可以单独指定此值或者与 `strand` 或 `htstrand` 值一起指定。
`strand`	授予域对分配给该域的 CPU 上存在的导线束性能计数器的访问权限。不能将此值与 `htstrand` 值一起指定。
`htstrand`	与 `strand` 值的行为相同并且允许检测分配给域的 CPU 上的超级特权模式事件。不能将此值与 `strand` 值一起指定。

要禁用对任何性能计数器的所有访问，请指定 `perf-counters=`。

如果虚拟机管理程序没有性能访问功能，尝试设置 `perf-counters` 属性将失败。

`ldm list -o domain` 和 `ldm list -e` 命令显示 `perf-counters` 属性的值。如果不支持性能访问功能，`perf-counters` 值将不显示在输出中。

例 13-11 创建域并指定其性能寄存器访问权限

创建对 `global` 寄存器集有访问权限的新 `ldg0` 域：

```
primary# ldm add-domain perf-counters=global ldg0
```

例 13-12 为域指定性能寄存器访问权限

指定 `ldg0` 域可以访问 `global` 和 `strand` 寄存器集：

```
primary# ldm set-domain perf-counters=global,strand ldg0
```

例 13-13 指定域不具有任何寄存器集访问权限

指定 `ldg0` 域不具有任何寄存器集访问权限：

```
primary# ldm set-domain perf-counters= ldg0
```

例 13-14　查看性能访问信息

以下示例显示如何使用 `ldm list -o domain` 命令查看性能访问信息。

- 以下 `ldm list -o domain` 命令显示在 `ldg0` 域上指定了 `global` 和 `htstrand` 性能值：

```
primary# ldm list -o domain ldg0
NAME     STATE     FLAGS     UTIL
NORM
ldg0     active    -n----    0.0% 0.0%

SOFTSTATE
Solaris running

UUID
    062200af-2de2-e05f-b271-f6200fd3eee3

HOSTID
    0x84fb315d

CONTROL
    failure-policy=ignore
    extended-mapin-space=on
    cpu-arch=native
    rc-add-policy=
    shutdown-group=15
    perf-counters=global,htstrand

DEPENDENCY
    master=

PPRIORITY    4000

VARIABLES
    auto-boot?=false
    boot-device=/virtual-devices@100/channel-devices@200/disk@0:a
      /virtualdevices@100/channel@200/disk@0
    network-boot-arguments=dhcp,hostname=solaris,
      file=http://10.129.241.238:5555/cgibin/wanboot-cgi
    pm_boot_policy=disabled=0;ttfc=2000;ttmr=0;
```

- 以下 `ldm list -p -o domain` 命令显示与先前示例中相同的信息，但是使用可解析形式：

```
primary# ldm list -p -o domain ldg0
```

```
VERSION 1.12
DOMAIN|name=ldg0|state=active|flags=normal|util=|norm_util=
UUID|uuid=4e8749b9-281b-e2b1-d0e2-ef4dc2ce5ce6
HOSTID|hostid=0x84f97452
CONTROL|failure-policy=reset|extended-mapin-space=on|cpu-arch=native|rc-add-policy=|
shutdown-group=15|perf-counters=global,htstrand
DEPENDENCY|master=
VARIABLES
|auto-boot?=false
|boot-device=/virtual-devices@100/channel-devices@200/disk@0
|pm_boot_policy=disabled=0;ttfc=2500000;ttmr=0;
```

♦ ♦ ♦ 第 14 章

14 管理域配置

本章包含有关管理域配置的信息。

本章包括以下主题：

管理域配置

域配置是单个系统中所有域及其资源分配的完整说明。可以在服务处理器 (service processor, SP) 上保存和存储配置，以供以后使用。

在 SP 上保存配置可使该配置在系统关开机循环之后仍保持有效。可以保存多个配置，并指定在下次尝试打开电源时要引导的配置。

启动系统时，SP 将引导选定的配置。系统将运行该配置中指定的同一组域，并使用该配置中指定的同一虚拟化和分区资源分配。默认配置是最近保存的配置。您也可以使用 `ldm set-spconfig` 命令或相应的 ILOM 命令显式请求另一配置。

注意 - 请始终以 XML 形式将您的稳定配置保存至 SP。通过以这种方式保存配置，您便可在电源出现故障后恢复系统配置，并在日后使用已保存的配置。请参见“保存域配置” [290]。

每当您将配置保存至 SP 时，就会在控制域中保存一份 SP 配置和 Logical Domains 约束数据库的本地副本。此本地副本称为引导集。引导集用于在系统进行关开机循环时装入对应的 Logical Domains 约束数据库。

在 SPARC T5、SPARC M5 和 SPARC M6 系统上，控制域上的引导集即为配置的主副本。在启动时，Logical Domains Manager 会自动将所有配置与 SP 进行同步，以确保 SP 上的配置始终与存储在控制域中的引导集相同。

注 - 由于此引导集包含关键系统数据，因此，请确保控制域的文件系统使用诸如磁盘镜像或 RAID 之类的技术来减少磁盘故障所产生的影响。

physical domain（物理域）是指由一个 Oracle VM Server for SPARC 实例管理的资源范围。物理域可以是一个完整的物理系统，例如，使用受支持的 SPARC T 系列平台时。它可以是整个系统，也可以是该系统的一个子集，例如，使用受支持的 SPARC M 系列平台时。

可用配置恢复方法

Oracle VM Server for SPARC 支持以下配置恢复方法：

- 自动保存方法，在 SP 上没有可用配置时使用。

  可能会在以下情形之一出现此情况：

  - 存放已保存配置的硬件被更换
  - 该配置不是最新配置，原因是您由于疏忽而未将最新配置更改保存至 SP 或发生意外的关开机循环
- `ldm add-domain` 方法，当域的一个子集需要恢复其配置时使用
- `ldm init-system` 方法，只应作为最后一项措施来使用。只有当 SP 上的配置以及控制域中的自动保存信息全部丢失时才使用此方法。

使用自动保存恢复配置

每次更改域配置时，都会在控制域上自动保存当前配置的副本。此自动保存操作不会将配置显式保存到 SP。

自动保存操作会立即发生，即使处于以下情况也是如此：

- 当新配置未显式保存在 SP 上时
- 当受影响的域重新引导后才会实施配置更改时

当保存在 SP 上的配置丢失时，通过此自动保存操作，可以恢复配置。当系统执行关开机循环后未将当前配置显式保存到 SP 时，也可以通过此操作恢复配置。这些情况下，如果该配置比为下次引导标记的配置新，Logical Domains Manager 可以在重新启动时恢复该配置。

注 - 电源管理、FMA 和 ASR 事件不会导致对自动保存文件进行更新。

可以自动或手动将自动保存文件恢复到新的或现有的配置。默认情况下，当自动保存配置比相应的运行中配置新时，会将一条消息写入 Logical Domains 日志。因此，必须使用 `ldm add-spconfig -r` 命令手动更新现有配置或根据自动保存数据创建新配置。请注意，必须在使用此命令完成手动恢复后关闭并重新打开电源。

注 - 当延迟的重新配置处于暂挂状态时，将立即自动保存配置更改。因此，如果运行 `ldm list-config -r` 命令，自动保存配置将显示为比当前配置新。

有关如何使用 `ldm *-spconfig` 命令管理配置和手动恢复自动保存文件的信息，请参见 `ldm`(1M) 手册页。

有关如何选择要引导的配置的信息，请参见“将 Oracle VM Server for SPARC 与服务处理器结合使用” [306]。您也可以使用 `ldm set-spconfig` 命令（如 `ldm`(1M) 手册页所述）。

自动恢复策略

自动恢复策略指定当自动保存在控制域上的一个配置比相应的运行中配置新时如何处理配置的恢复。自动恢复策略是通过设置 `ldmd` SMF 服务的 `autorecovery_policy` 属性指定的。此属性可以具有以下值：

- `autorecovery_policy=1` – 自动保存配置比相应的运行中配置新时，记录警告消息。这些消息记录在 `ldmd` SMF 日志文件中。必须手动执行任何配置恢复。这是默认策略。
- `autorecovery_policy=2` – 如果自动保存配置比相应的运行中配置新，则显示通知消息。每次重新启动 Logical Domains Manager 之后，首次发出 `ldm` 命令时，此通知消息将显示在所有 `ldm` 命令的输出中。必须手动执行任何配置恢复。
- `autorecovery_policy=3` – 如果自动保存配置比相应的运行中配置新，将自动更新该配置。此操作会覆盖将在下次关开机循环期间使用的 SP 配置。要使此配置可用，必须再执行一次关闭并重新打开电源操作。将使用保存在控制域上的较新配置更新此配置。此操作不会影响当前运行的配置。它只会影响要在下次关开机循环期间使用的配置。同时还会记录一条消息，指出已在 SP 上保存了较新的配置，并会在系统下次执行关开机循环时对其进行引导。这些消息记录在 `ldmd` SMF 日志文件中。

▼ 如何修改自动恢复策略

1. 登录到控制域。
2. 成为管理员。
   - 对于 Oracle Solaris 10，请参见《System Administration Guide: Security Services》中的“Configuring RBAC (Task Map)”。
   - 对于 Oracle Solaris 11.2，请参见《Securing Users and Processes in Oracle Solaris 11.2》中的第 1 章 “About Using Rights to Control Users and Processes”。

3. 查看 **autorecovery_policy** 属性值。

```
# svccfg -s ldmd listprop ldmd/autorecovery_policy
```

4. 停止 **ldmd** 服务。

```
# svcadm disable ldmd
```

5. 更改 **autorecovery_policy** 属性值。

```
# svccfg -s ldmd setprop ldmd/autorecovery_policy=value
```

例如，要将策略设置为执行自动恢复，则将属性值设置为 3：

```
# svccfg -s ldmd setprop ldmd/autorecovery_policy=3
```

6. 刷新并重新启动 **ldmd** 服务。

```
# svcadm refresh ldmd
# svcadm enable ldmd
```

例 14-1 从日志修改自动恢复策略以自动恢复

以下示例显示如何查看 autorecovery_policy 属性的当前值并将其更改为新值。此属性的原始值为 1，这意味着会记录自动保存更改。svcadm 命令用于停止并重新启动 ldmd 服务，svccfg 命令用于查看和设置属性值。

```
# svccfg -s ldmd listprop ldmd/autorecovery_policy
ldmd/autorecovery_policy integer 1
# svcadm disable ldmd
# svccfg -s ldmd setprop ldmd/autorecovery_policy=3
# svcadm refresh ldmd
# svcadm enable ldmd
```

保存域配置

可以保存系统上单个域或所有域的域配置。

除了指定的物理资源以外，以下方法不保留实际绑定。然而，该方法会保留用于创建这些绑定的约束。保存和恢复配置后，这些域将具有相同的虚拟资源，但不一定绑定到相同的物理资源。按照管理员的指定，绑定指定的物理资源。

- 要保存单个域的配置，请创建一个包含该域的约束的 XML 文件。

```
# ldm list-constraints -x domain-name >domain-name.xml
```

以下示例显示如何创建 XML 文件 ldg1.xml，该文件包含 ldg1 域的约束：

```
# ldm list-constraints -x ldg1 >ldg1.xml
```

- 要保存系统上所有域的配置，请创建一个包含所有域的约束的 XML 文件。

  ```
  # ldm list-constraints -x >file.xml
  ```

  以下示例显示如何创建 XML 文件 config.xml，该文件包含系统上所有域的约束：

  ```
  # ldm list-constraints -x >config.xml
  ```

恢复域配置

本节介绍如何从一个 XML 文件恢复来宾域和控制 (primary) 域的域配置。

- 要恢复来宾域的域配置，请使用 ldm add-domain -i 命令（如如何从 XML 文件恢复域配置 (ldm add-domain) [291]中所述）。尽管您可以将 primary 域的约束保存到 XML 文件，但不能使用该文件作为此命令的输入。
- 要恢复 primary 域的域配置，请使用 ldm init-system 命令以及 XML 文件中的资源约束来重新配置 primary 域。您还可以使用 ldm init-system 命令重新配置 XML 文件中所述的其他域，但在配置完成后，这些域可能会保持非活动状态。请参见如何从 XML 文件恢复域配置 (ldm init-system) [292]。

▼ 如何从 XML 文件恢复域配置 (ldm add-domain)

此过程适用于来宾域，而不适用于控制 (primary) 域。如果要恢复 primary 域或 XML 文件中描述的其他域的配置，请参见如何从 XML 文件恢复域配置 (ldm init-system) [292]。

1. **通过使用创建的 XML 文件作为输入创建域。**

   ```
   # ldm add-domain -i domain-name.xml
   ```

2. **绑定域。**

   ```
   # ldm bind-domain [-fq] domain-name
   ```

   -f 选项强制绑定域，即使检测到无效的后端设备也是如此。-q 选项禁用对后端设备的验证，以便该命令更快地运行。

3. **启动域。**

   ```
   # ldm start-domain domain-name
   ```

例 14-2 从 XML 文件恢复单个域

以下示例显示如何恢复单个域。首先，从 XML 文件恢复 ldg1 域。然后，绑定并重新启动所恢复的 ldg1 域。

```
# ldm add-domain -i ldg1.xml
# ldm bind ldg1
# ldm start ldg1
```

▼ 如何从 XML 文件恢复域配置 (ldm init-system)

此过程介绍了如何使用 ldm init-system 命令与 XML 文件重新创建以前保存的配置。

注意 - ldm init-system 命令可能无法正确恢复已使用物理 I/O 命令的配置。此类命令为 ldm add-io、ldm set-io、ldm remove-io、ldm create-vf 和 ldm destroy-vf。有关更多信息，请参见《Oracle VM Server for SPARC 3.2 发行说明 》中的“ldm init-system 命令可能无法正确恢复已执行了物理 I/O 更改的域配置”。

开始之前 您应该已通过运行 ldm list-constraints -x 命令创建了 XML 配置文件。该文件应描述一个或多个域配置。

1. 登录到 **primary** 域。

2. 检验系统是否为 **factory-default** 配置。

```
primary# ldm list-config | grep "factory-default"
factory-default [current]
```

如果系统不在 factory-default 配置中，请参见《Oracle VM Server for SPARC 3.2 安装指南 》中的“如何恢复出厂默认配置”。

3. 成为管理员。

   - 对于 Oracle Solaris 10，请参见《System Administration Guide: Security Services 》中的“Configuring RBAC (Task Map)”。
   - 对于 Oracle Solaris 11.2，请参见《Securing Users and Processes in Oracle Solaris 11.2 》中的第 1 章 “About Using Rights to Control Users and Processes”。

4. 从 XML 文件还原一个或多个域配置。

```
# ldm init-system [-frs] -i filename.xml
```

必须重新引导 primary 域，才能使配置生效。-r 选项将在配置后重新引导 primary 域。如果不指定 -r 选项，则必须手动执行重新引导。

-s 选项仅恢复虚拟服务配置（vds、vcc 和 vsw），执行时可能不必重新引导。

-f 选项会跳过出厂默认配置检查并继续操作，而不考虑系统上已有的配置。使用 -f 选项时一定要格外小心。ldm init-system 命令假定系统采用出厂默认配置，因此会直接应用

由 XML 文件指定的更改。如果在系统采用非出厂默认配置时使用 -f，有可能会导致系统未按照 XML 文件所指定的那样进行配置。一个或多个更改可能无法应用于系统，具体情况取决于 XML 文件中的更改和初始配置这一组合。

primary 域将按此文件中的说明进行重新配置。任何在 XML 文件中具有配置的非 primary 域都会进行重新配置，但会保持非活动状态。

例 14-3 从 XML 配置文件恢复域

以下示例显示如何使用 ldm init-system 命令以 factory-default 配置恢复 primary 域和系统上的所有域。

- **恢复 primary 域。**-r 选项用于在配置完成后重新引导 primary 域。primary.xml 文件包含先前保存的 XML 域配置。

```
primary# ldm init-system -r -i primary.xml
```

- **恢复系统中的所有域。**将系统中的域恢复为 config.xml XML 文件中的配置。config.xml 文件包含先前保存的 XML 域配置。primary 域将由 ldm init-system 命令自动重新启动。所有其他域将进行恢复，但不会进行绑定和重新启动。

```
# ldm init-system -r -i config.xml
```

系统重新引导之后，以下命令将绑定和重新启动 ldg1 和 ldg2 域：

```
# ldm bind ldg1
# ldm start ldg1
# ldm bind ldg2
# ldm start ldg2
```

解决服务处理器连接问题

如果想要使用 ldm 命令来管理由于与 SP 的错误通信而导致的域配置失败，请执行适用于您的平台的以下恢复步骤：

- **SPARC T3、SPARC T4、SPARC T5、SPARC M5 和 SPARC M6 系统。**重新启动 ldmd 服务。

```
primary# svcadm enable ldmd
```

如果重新启动 ldmd 守护进程未能恢复通信，请重新启动 SP。

♦ ♦ ♦ 第 15 章

15 处理硬件错误

本章包含有关 Oracle VM Server for SPARC 如何处理硬件错误的信息。

本章包括以下主题：

硬件错误处理概述

从 SPARC T5 和 SPARC M5 开始，Oracle VM Server for SPARC 软件为 SPARC 企业级平台增加了以下 RAS 功能：

- **故障管理体系结构 (Fault Management Architecture, FMA) 黑名单。**当 FMA 检测到有故障的 CPU 或内存资源后，Oracle VM Server for SPARC 会将其放入黑名单。不能将列入黑名单的有故障资源重新分配给任何域，除非 FMA 将其标记为已修复。
- **恢复模式。**自动恢复因资源出现故障或缺少资源而无法引导的域配置。

虽然 Fujitsu M10 平台不支持将有故障的资源列入黑名单，但 Fujitsu M10 平台自动替换功能可提供类似的功能。

使用 FMA 将有故障的资源列入黑名单或取消其配置

FMA 会在检测到有故障的资源后联系 Logical Domains Manager。然后，Logical Domains Manager 会尝试在所有正在运行的域中停止使用该资源。为确保将来无法将有故障的资源分配给域，FMA 会将该资源添加到黑名单。

Logical Domains Manager 仅支持将 CPU 和内存资源列入黑名单，不支持将 I/O 资源列入黑名单。

如果有故障的资源未在使用中，则 Logical Domains Manager 会将该资源从 `ldm list-devices` 输出中显示的可用资源列表中删除。此时，该资源会在内部标记为“已列入黑名单”，这样，将来就无法将其分配给域了。

如果有故障的资源正在使用中，则 Logical Domains Manager 会尝试清除该资源。为避免正在运行的域发生服务中断，Logical Domains Manager 会先尝试使用 CPU 或内存动态重新配置来清除有故障的资源。如果某个核心可用作目标，则 Logical Domains Manager 会重新映射有故障的核心。如果此“实时清除”成功，则有故障的资源会在内部标记为已列入黑名单，并且不会显示在 `ldm list-devices` 输出中，这样，该资源将来就不会分配给域了。

如果实时清除失败，则 Logical Domains Manager 会在内部将故障资源标记为 "evacuation pending"。资源在 `ldm list-devices` 输出中显示为正常，因为资源在运行中的域上仍在使用中，直到受影响的来宾域重新引导或停止。

受影响的来宾域停止或重新引导后，Logical Domains Manager 会尝试清除有故障的资源并在内部将其标记为已列入黑名单，这样将来就不会分配该资源了。此类设备不会显示在 `ldm` 输出中。待清除完成后，Logical Domains Manager 会尝试启动来宾域。但是，如果因没有足够的可用资源而无法启动来宾域，则来宾域会标记为“已降级”，并记录以下警告消息，以便用户能够进行干预来执行手动恢复。

```
primary# ldm ls
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  NORM  UPTIME
primary          active     -n-cv-  UART    368   2079488M 0.1%  0.0%  16h 57m
gd0              bound      -d----  5000    8

warning: Could not restart domain gd0 after completing pending evacuation.
The domain has been marked degraded and should be examined to see
if manual recovery is possible.
```

系统进行关开机循环后，FMA 会对仍有故障的资源重复清除请求，而 Logical Domains Manager 会通过清除这些有故障的资源并在内部将其标记为已列入黑名单来处理这些请求。

在提供 FMA 黑名单支持之前，如果来宾域因资源故障而发生紧急情况，则可能会陷入紧急情况-重新引导的死循环。通过在重新引导来宾域后使用资源清除和黑名单功能，可以避免这一紧急情况-重新引导循环，并防止将来尝试使用有故障的资源。

在检测到有故障的资源或缺少的资源后恢复域

如果 SPARC T5、SPARC M5、SPARC M6 或 Fujitsu M10 系统在打开电源时检测到资源故障或缺少资源，则 Logical Domains Manager 会尝试使用剩余可用资源恢复已配置的域。进行恢复时，系统（或 SPARC M 系列上的物理域）将视为处于恢复模式。只有在启用恢复模式后才会尝试进行恢复。请参见“启用恢复模式” [299]。

打开电源时，如果在以下任意情况下无法引导最后选择的电源打开配置，则系统固件将恢复为出厂默认配置：

- 所用配置中每个 PCIe 交换机内的 I/O 拓扑与最后选择的电源打开配置中的 I/O 拓扑不匹配
- 最后选择的电源打开配置中的 CPU 资源或内存资源不再位于系统中

如果启用了恢复模式，则 Logical Domains Manager 会从最后选择的电源打开配置恢复所有活动域和绑定域。所得到的运行中配置称为降级配置。降级配置将保存至 SP 并会始终用作活动配置，直到保存新的 SP 配置或物理域进行关开机循环为止。

注 - 由于降级配置已经成为运行中配置，因此在恢复后，该物理域无需关开机循环即可激活该配置。

如果物理域进行关开机循环，则系统固件会首先尝试引导最后一个原始电源打开配置。这样，如果已替换缺少的硬件或有故障的硬件，则系统便可引导原始正常配置。如果无法引导最后选择的电源打开配置，则该固件会接着尝试引导相关联的降级配置（如果有）。如果降级配置无法引导或不存在，则会引导出厂默认配置并调用恢复模式。

恢复操作将按以下顺序进行：

- **控制域。**Logical Domains Manager 通过恢复 CPU、内存、I/O 配置以及虚拟 I/O 服务来恢复控制域。

  如果所有可恢复的域所需的 CPU 或内存量超过剩余可用量，则会根据其他域的大小相应减少 CPU 或核心数或内存量。例如，在一个包含四个域的系统中，如果每个域分配有 25% 的 CPU 和内存，则所得到的降级配置仍会向每个域分配 25% 的 CPU 和内存。如果 `primary` 域最初最多包含两个核心（16 个虚拟 CPU）和 8 GB 的内存，则控制域的大小不会减小。

  分配给其他域的根联合体和 PCIe 设备将从控制域中被删除。属于控制域的根联合体上的虚拟功能将重新创建。分配给控制域的任何缺失根联合体、PCIe 设备、物理功能或虚拟功能都会标记为已清除。然后，Logical Domains Manager 将重新引导控制域，以使更改生效。
- **根域。**重新引导控制域后，Logical Domains Manager 将恢复根域。如果需要，CPU 和内存量将根据其他可恢复的域相应地减少。如果根联合体在物理层面不再位于系统中，则会标记为已清除。恢复操作期间，此根联合体不会配置到域中。只要分配给根域的根联合体至少有一个可用，此根域就会进行恢复。如果任何根联合体均不可用，则此根域不会进行恢复。Logical Domains Manager 会引导根域，并在属于根域的物理功能上重新创建虚拟功能。所有缺少的 PCIe 插槽、物理功能和虚拟功能都会标记为已清除。如有可能，此域提供的所有虚拟 I/O 服务将重新创建。

注 - 此时无法恢复由非 `primary` 根域借出 PCIe 插槽的配置。因此，必须在恢复完成后将这些插槽手动移动到 I/O 域。

- **I/O 域。**Logical Domains Manager 可恢复所有 I/O 域。系统中缺少的所有 PCIe 插槽和虚拟功能都会标记为已清除。如果所需的 I/O 设备均不存在，则不会恢复此域，

此时，其 CPU 和内存资源可供其他域使用。如有可能，此域提供的所有虚拟 I/O 服务将重新创建。

- **来宾域。**只有当至少恢复了为来宾域提供服务的一个服务域之后，此来宾域才会进行恢复。如果此来宾域无法进行恢复，则其 CPU 和内存资源可供其他来宾域使用。

如有可能，系统会向一个域分配与原始配置相同的 CPU 数和内存量。如果此 CPU 数或内存量不可用，则会相应地减少这些资源，以使用其余可用资源。

注 - 系统处于恢复模式时，只能执行 `ldm list-*` 命令。在恢复操作完成之前，所有其他 `ldm` 命令都将禁用。

Logical Domains Manager 只会尝试恢复绑定域和活动域。未绑定的任何域的现有资源配置将按原样复制到新配置中。

恢复操作期间，可用资源可能会比先前引导的配置要少。因此，Logical Domains Manager 可能只能恢复一部分先前配置的域。此外，恢复后的域可能并不包含原始配置中的所有资源。例如，恢复后的绑定域所具有的 I/O 资源可能比先前的配置要少。如果某个域的 I/O 设备不再存在或其父服务域无法恢复，则可能无法恢复该域。

恢复模式会将其步骤记录到 Logical Domains Manager SMF 日志 `/var/svc/log/ldoms-ldmd:default.log` 中。当 Logical Domains Manager 启动恢复、重新引导控制域以及恢复完成时，会在系统控制台中写入消息。

注意 - 恢复后的域并不保证能够完全正常运行。该域可能不包含运行 OS 实例或应用程序所需的资源。例如，恢复后的域可能只有网络资源，而没有磁盘资源。或者，恢复后的域可能缺少运行应用程序所需的文件系统。在域中使用多路径 I/O 可减少因缺少 I/O 资源而产生的影响。

恢复模式硬件和软件要求

- **硬件要求** – SPARC T5、SPARC M5、SPARC M6 和 Fujitsu M10 平台支持恢复模式功能。
- **固件要求** – SPARC T5、SPARC M5 和 SPARC M6 系统至少需要 9.1.0.a 版本的系统固件。Fujitsu M10 服务器至少需要 XCP2230 版本的系统固件
- **软件要求** – 所有域必须至少运行 Oracle Solaris 11.1.10.5.0 OS 或 Oracle Solaris 10 1/13 OS 以及《Oracle VM Server for SPARC 3.2 安装指南 》中的“全限定 Oracle Solaris OS 版本”中必需的修补程序。

降级配置

每个物理域只能将一个降级配置保存至 SP。如果已经有一个降级配置，则该配置会被新创建的降级配置所替换。

您不能直接与降级配置进行交互。必要时，系统固件会以透明方式引导下一个电源打开配置的降级版本。通过此透明引导，系统便可在关开机循环后缺少的资源重新出现时引导原始配置。如果活动配置为降级配置，则该配置会在 `ldm list-spconfig` 输出中标记为 `[degraded]`。

如果活动配置为降级配置，则自动保存功能将被禁用。如果在降级配置处于活动状态时将新配置保存至 SP，则新配置将为正常非降级配置。

注 - 如果先前缺少的资源在后续关开机循环时重新出现，则不会对正常配置的内容产生任何影响。但是，如果此后选择了触发恢复模式的配置，则 SP 会引导原始非降级配置，因为其所有硬件均可用。

启用恢复模式

`ldmd/recovery_mode` SMF 属性用于控制恢复模式行为。

要对 Logical Domains Manager 进行配置，以便在系统进入恢复模式时自动开始恢复过程，必须首先启用恢复模式。要启用恢复模式，请将 `ldmd/recovery_mode` 属性值设置为 `auto`，然后刷新 `ldmd` SMF 服务。

```
primary# svccfg -s ldmd setprop ldmd/recovery_mode = astring: auto
primary# svcadm refresh ldmd
primary# svcadm restart ldmd
```

默认情况下，`ldmd/recovery_mode` 属性不存在。如果此属性不存在或设置为 `never`，则 Logical Domains Manager 会退出恢复模式而不执行任何操作，此时，物理域将运行出厂默认配置。

注 - 如果在系统固件请求恢复模式时该模式尚未启用，请在发出该请求后执行以下命令来启用恢复模式：

```
primary# svccfg -s ldmd setprop ldmd/recovery_mode = astring: auto
primary# svcadm refresh ldmd
primary# svcadm restart ldmd
```

在此情况下，只有当未对系统进行任何更改（即，系统仍采用出厂默认配置）时，恢复模式才会立即启动。

将域标记为已降级

如果使用 FMA 将某一资源列入黑名单，从而使域因资源不足而无法启动，则该域将标记为已降级。此时，该域仍处于绑定状态，从而可以防止分配给该域的其余资源重新分配给其他域。

将 I/O 资源标记为已清除

在恢复模式下检测到的缺失 I/O 资源会通过在 `ldm` 列表输出中显示一个星号 (*) 来标记为已清除。

♦ ♦ ♦ 第 16 章

16 执行其他管理任务

本章包含有关使用 Oracle VM Server for SPARC 软件的信息以及前面章节未介绍的任务。

本章包括以下主题：

在 CLI 中输入名称

以下各节介绍了在 Logical Domains Manager CLI 中输入名称的限制。

- 文件名 (*file*) 和变量名 (*var-name*)
  - 第一个字符必须是字母、数字或正斜杠 (*/*)。
  - 后面的字符必须是字母、数字或标点。
- 虚拟磁盘服务器 *backend* 和虚拟交换机设备名称

名称必须包含字母、数字或标点。

- 配置名称 (*config-name*)

  为存储在服务处理器 (service processor, SP) 上的配置指定的逻辑域配置名称 (*config-name*) 不得超过 64 个字符。

- 所有其他名称

  其余名称，例如逻辑域名称 (*domain-name*)、服务名称（*vswitch-name*、*service-name*、*vdpcs-service-name* 和 *vcc-name*）、虚拟网络名称 (*if-name*) 以及虚拟磁盘名称 (*disk-name*)，必须采用以下格式：

  - 第一个字符必须是字母或数字。
  - 后面的字符必须是字母、数字或以下任一字符：`-_+#.:;~()`。

更新 /etc/system 文件中的属性值

不要手动更改 /etc/system 文件。当 /etc/system.d 目录中的文件指定了优化属性值时，会在重新引导时自动生成此文件。

▼ 如何添加或修改优化属性值

1. **在现有的 /etc/system 文件以及在 /etc/system.d 文件中搜索优化属性名称。**
   例如，要指定 vds:vd_volume_force_slice 属性的值，请确定是否已设置该属性。

   ```
   # grep 'vds:vd_volume_force_slice' /etc/system /etc/system.d/*
   ```

2. **更新属性值：**

   - **如果在其中一个 /etc/system.d 文件中找到了该属性，则在现有文件中更新属性值。**

   - **如果在 /etc/system 文件中找到该属性或者未找到该属性，则在 /etc/system.d 目录中创建一个具有如下示例中所示名称的文件：**

     /etc/system.d/com.*company-name*:ldoms-config

通过网络连接到来宾控制台

如果在 vntsd(1M) SMF 清单中将 listen_addr 属性设置为控制域的 IP 地址，则可以通过网络连接到来宾控制台。例如：

```
$ telnet hostname 5001
```

注 - 启用对控制台的网络访问会有安全隐患。任何用户都可以连接到控制台，因此默认情况下它处于禁用状态。

服务管理工具清单是一个描述服务的 XML 文件。有关创建 SMF 清单的更多信息，请参阅 Oracle Solaris 10 System Administrator Documentation (http://download.oracle.com/docs/cd/E18752_01/index.html)。

注 - 要通过控制台访问来宾域中的非英语 OS，控制台终端必须采用该 OS 所需的语言环境。

使用控制台组

通过虚拟网络终端服务器守护进程 vntsd，您可以使用一个 TCP 端口访问多个域控制台。创建域时，Logical Domains Manager 通过为该域的控制台创建新的默认组，来为每个控制台分配唯一的 TCP 端口。该 TCP 端口随后被分配给控制台组，而非控制台本身。可以使用 set-vcons 子命令将控制台绑定到现有的组。

▼ 如何将多个控制台组成一个组

1. **将域的控制台绑定到一个组。**

   以下示例说明将三个不同域（ldg1、ldg2 和 ldg3）的控制台绑定到同一控制台组 (group1)。

   ```
   primary# ldm set-vcons group=group1 service=primary-vcc0 ldg1
   primary# ldm set-vcons group=group1 service=primary-vcc0 ldg2
   primary# ldm set-vcons group=group1 service=primary-vcc0 ldg3
   ```

2. **连接到关联的 TCP 端口（此示例中为 `localhost` 的端口 `5000`）。**

   ```
   # telnet localhost 5000
   primary-vnts-group1: h, l, c{id}, n{name}, q:
   ```

   系统会提示您选择一个域控制台。

3. **通过选择 `l`（列表）列出组中的域。**

   ```
   primary-vnts-group1: h, l, c{id}, n{name}, q: l
   DOMAIN ID          DOMAIN NAME                 DOMAIN STATE
   0                  ldg1                        online
   1                  ldg2                        online
   ```

```
2                 ldg3                        online
```

注 - 要将控制台重新分配到不同的组或 vcc 实例，必须将域解除绑定；即，域必须处于非活动状态。有关配置和使用 SMF 来管理 vntsd 以及使用控制台组的更多信息，请参阅 Oracle Solaris 10 OS vntsd(1M) 手册页。

停止高负载的域会超时

ldm stop-domain 命令会在域完成关闭操作之前超时。如果发生这种情况，Logical Domains Manager 会返回一个类似以下内容的错误。

```
LDom ldg8 stop notification failed
```

但是，域可能仍在处理关闭请求。使用 ldm list-domain 命令可以检验域的状态。例如：

```
# ldm list-domain ldg8
NAME         STATE   FLAGS  CONS   VCPU MEMORY  UTIL UPTIME
ldg8         active  s----  5000   22   3328M   0.3% 1d 14h 31m
```

上面的列表显示域处于活动状态，但 s 标志指示域正在停止过程中。此状态应该是短暂状态。

以下示例说明域现在已停止。

```
# ldm list-domain ldg8
NAME         STATE   FLAGS  CONS   VCPU MEMORY  UTIL UPTIME
ldg8         bound   -----  5000   22   3328M
```

ldm stop 命令可使用 shutdown 命令停止域。执行关闭序列所需的时间通常比执行快速停止所需的时间要长得多；快速停止是通过运行 ldm stop -q 命令执行的。请参见 ldm(1M) 手册页。

长关闭序列可能会生成以下超时消息：

domain-name stop timed out. The domain might still be in the process of shutting down. Either let it continue, or specify -f to force it to stop.

在运行此关闭序列期间，还会显示域的 s 标志。

操作具有 Oracle VM Server for SPARC 的 Oracle Solaris OS

本节介绍实例化由 Logical Domains Manager 创建的配置之后，在使用 Oracle Solaris OS 方面所发生的行为变化。

Oracle Solaris OS 启动后 OpenBoot 固件不可用

OpenBoot 固件在 Oracle Solaris OS 启动之后不可用，因为它已从内存中删除。

要从 Oracle Solaris OS 访问 `ok` 提示符，必须使用 Oracle Solaris OS `halt` 命停止域。

执行服务器关开机循环

每当在运行 Oracle VM Server for SPARC 软件的系统上执行需要对服务器进行关开机循环的任何维护操作时，您都必须先将当前逻辑域配置保存至 SP。

要将当前域配置保存至 SP，请使用以下命令：

```
# ldm add-config config-name
```

在 Oracle Solaris OS 中发生中断的结果

可以按照以下方式启动 Oracle Solaris OS 中断：

1. 输入设备设置为 `keyboard` 时按 L1-A 键序。
2. 虚拟控制台处于 `telnet` 提示符下时输入 `send break` 命令。

在启动此类中断时，Oracle Solaris OS 将发出以下提示：

```
c)ontinue, s)ync, r)eset, h)alt?
```

键入代表发出这些类型的中断后希望系统执行的操作的字母。

重新引导控制域的结果

下表显示了重新引导控制 (`primary`) 域的预期行为。

表 16-1 重新引导控制域的预期行为

命令	是否配置其他域？	行为
`reboot`	未配置	在非正常关机但未关闭电源的情况下重新引导控制域。
	已配置	在非正常关机但未关闭电源的情况下重新引导控制域。
`shutdown -i 5`	未配置	主机在正常关机后关闭电源，保持关机状态直到 SP 电源打开。
	已配置	在正常关机但未关闭电源的情况下重新引导。

有关重新引导具有根域角色的域的后果信息，请参见“重新引导配置了 PCIe 端点的根域” [71]。

将 Oracle VM Server for SPARC 与服务处理器结合使用

本节介绍有关将 Integrated Lights Out Manager (ILOM) 服务处理器 (service processor, SP) 与 Logical Domains Manager 结合使用的信息。有关使用 ILOM 软件的更多信息，请参见有关您具体平台的文档，网址为：http://www.oracle.com/technetwork/documentation/sparc-tseries-servers-252697.html。

现有的 ILOM 命令增加了一个 `config` 选项。

```
-> set /HOST/bootmode config=config-name
```

使用该选项可以在下次打开电源时将配置设置为其他配置，包括 `factory-default` 出厂配置。

无论主机的电源是打开还是关闭，您都可以调用此命令。它将在下次主机复位或打开电源时生效。

要重置逻辑域配置，请将该选项设置为 `factory-default`。

```
-> set /HOST/bootmode config=factory-default
```

此外，也可以选择已创建（在 Logical Domains Manager 中使用 `ldm add-config` 命令）且存储在服务处理器 (service processor, SP) 上的其他配置。执行 ILOM `bootmode` 命令时，可以使用在 Logical Domains Manager `ldm add-config` 命令中指定的名称来选择配置。例如，假设您使用名称 `ldm-config1` 存储了配置。

```
-> set /HOST/bootmode config=ldm-config1
```

现在，必须执行系统关开机循环以装入新配置。

有关 `ldm add-config` 命令的更多信息，请参见 `ldm`(1M) 手册页。

配置域依赖关系

可以使用 Logical Domains Manager 建立域之间的依赖关系。如果域具有一个或多个依赖于它的域，则该域称为主域。如果域依赖于其他域，该域称为从属域。

通过设置 `master` 属性，每个从属域最多可以指定四个主域。例如，`pine` 从属域在以下用逗号分隔的列表中指定其四个主域：

```
# ldm add-domain master=alpha,beta,gamma,delta pine
```

alpha、beta、gamma 和 delta 主域全都指定故障策略 stop。

每个主域都可以指定在主域发生故障时对其从属域产生何种影响。例如，如果主域发生故障，它可能会要求其从属域发生紧急情况。如果某个从属域有多个主域，则每个主域都必须有相同的故障策略。因此，第一个主域发生故障将触发其所有从属域上定义的故障策略。

主域的故障策略是通过将以下任一值设置为 failure-policy 属性来控制的：

- ignore 将忽略所有从属域
- panic 使所有从属域都发生紧急情况（类似于运行 ldm panic 命令）
- reset 立即停止，然后重新启动所有从属域（类似于运行 ldm stop -f，然后运行 ldm start 命令）
- stop 停止所有从属域（类似于运行 ldm stop -f 命令）

在此示例中，主域按如下所示指定其故障策略：

```
primary# ldm set-domain failure-policy=ignore apple
primary# ldm set-domain failure-policy=panic lemon
primary# ldm set-domain failure-policy=reset orange
primary# ldm set-domain failure-policy=stop peach
primary# ldm set-domain failure-policy=stop alpha
primary# ldm set-domain failure-policy=stop beta
primary# ldm set-domain failure-policy=stop gamma
primary# ldm set-domain failure-policy=stop delta
```

可以使用此机制创建域之间的显式依赖关系。例如，来宾域隐式依赖于服务域以提供其虚拟设备。当来宾域所依赖的服务域未启动并运行时，来宾域的 I/O 将被阻止。通过将来宾域定义为其服务域的从属域，可以指定来宾域在其服务域关闭时的行为。如果未建立此类依赖关系，来宾域只会等待其服务域返回到服务状态。

注 - Logical Domains Manager 不允许创建会产生依赖关系循环的域关系。有关更多信息，请参见“依赖关系循环” [309]。

有关域依赖关系 XML 示例，请参见例 22-6 “ldom_info XML 输出示例”。

域依赖关系示例

以下示例说明如何配置域依赖关系。

例 16-1 使用域依赖关系配置故障策略

第一个命令创建名为 twizzle 的主域。此命令使用 failure-policy=reset 来指定如果 twizzle 域发生故障，从属域将重置。第二个命令修改名为 primary 的主域。此命令使用 failure-policy=reset 来指定如果 primary 域发生故障，从属域将重置。第三个命令

创建名为 chocktaw 的从属域，该从属域依赖于 twizzle 和 primary 两个主域。从属域使用 master=twizzle,primary 来指定其主域。如果 twizzle 或 primary 域发生故障，则 chocktaw 域将重置。

```
primary# ldm add-domain failure-policy=reset twizzle
primary# ldm set-domain failure-policy=reset primary
primary# ldm add-domain master=twizzle,primary chocktaw
```

例 16-2 修改域以分配主域

本示例说明如何使用 ldm set-domain 命令修改 orange 域，以将 primary 分配为主域。第二个命令使用 ldm set-domain 命令将 orange 和 primary 分配为 tangerine 域的主域。第三个命令列出有关所有这些域的信息。

```
primary# ldm set-domain master=primary orange
primary# ldm set-domain master=orange,primary tangerine
primary# ldm list -o domain
NAME             STATE      FLAGS   UTIL
primary          active     -n-cv-  0.2%

SOFTSTATE
Solaris running

HOSTID
    0x83d8b31c

CONTROL
    failure-policy=ignore

DEPENDENCY
    master=

------------------------------------------------------------------------------
NAME             STATE      FLAGS   UTIL
orange           bound      ------

HOSTID
    0x84fb28ef

CONTROL
    failure-policy=ignore

DEPENDENCY
    master=primary

------------------------------------------------------------------------------
NAME             STATE      FLAGS   UTIL
tangerine        bound      ------

HOSTID
    0x84f948e9
```

```
CONTROL
    failure-policy=ignore

DEPENDENCY
    master=orange,primary
```

例 16-3 显示可解析的域列表

以下显示了一个列有可解析输出的示例：

```
primary# ldm list -o domain -p
```

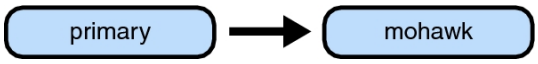
依赖关系循环

Logical Domains Manager 不允许创建会产生依赖关系循环的域关系。依赖关系循环是指两个或多个域之间的一种关系，其中，从属域可能会依赖于其本身或主域可能会依赖于其从属域之一。

Logical Domains Manager 在添加依赖关系之前确定是否存在依赖关系循环。Logical Domains Manager 自从属域开始沿着主阵列指定的所有路径进行搜索，直到到达路径的终点。沿途所发现的任何依赖关系循环都将被报告为错误。

以下示例说明了如何会产生依赖关系循环。第一个命令创建名为 `mohawk` 的从属域，该从属域将其主域指定为 `primary`。因此，在下图所示的依赖关系链中，`mohawk` 依赖于 `primary`。

图 16-1 单个域依赖关系

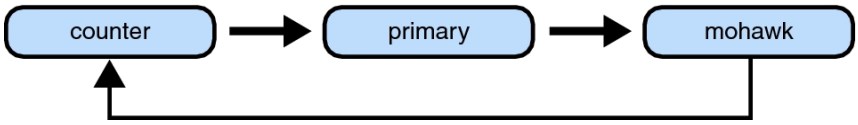


第二个命令创建名为 `primary` 的从属域，该从属域将其主域指定为 `counter`。因此，在下图所示的依赖关系链中，`mohawk` 依赖于 `primary`，而 primary 又依赖于 `counter`。

图 16-2 多个域依赖关系

第三个命令将尝试创建 counter 和 mohawk 域之间的依赖关系，这种依赖关系将产生下图所示的依赖关系循环。

图 16-3 域依赖关系循环



ldm set-domain 命令将失败，并显示以下错误消息：

```
# ldm add-domain master=primary mohawk
# ldm set-domain master=counter primary
# ldm set-domain master=mohawk counter
Dependency cycle detected: LDom "counter" indicates "primary" as its master
```

通过映射 CPU 和内存地址来确定出错位置

本节介绍如何能够将 Oracle Solaris 故障管理体系结构 (Fault Management Architecture, FMA) 报告的信息与标记为出现故障的逻辑域资源相关联。

FMA 以物理 CPU 编号形式报告 CPU 错误，以物理内存地址形式报告内存错误。

如果您想要确定发生错误的逻辑域，以及该域中相应的虚拟 CPU 编号或实际内存地址，则必须执行映射。

CPU 映射

您可以查找与给定的物理 CPU 编号对应的域和虚拟 CPU 编号。

首先，使用以下命令为所有域生成一个可解析的长列表：

```
primary# ldm list -l -p
```

在列表的 VCPU 部分中查找 pid 字段等于物理 CPU 编号的条目。

- 如果找到了这样的条目，则 CPU 所在的域就是其下方列出此条目的域，该域中的虚拟 CPU 编号由此条目的 vid 字段指定。
- 如果未找到这样的条目，则 CPU 不在任何域中。

内存映射

您可以查找域以及该域中的实际内存地址（与给定物理内存地址 (PA) 对应）。

首先，为所有域生成一个可解析的长列表。

```
primary# ldm list -l -p
```

请在该列表的 MEMORY 部分中查找 PA 介于 *pa* 和 (*pa* + *size* - 1) 闭合区间内的行；即 *pa* ≤ PA ≤ (*pa* + *size* - 1)。*pa* 和 *size* 对应于该行中相应字段的值。

- 如果找到了这样的条目，则 PA 所在的域就是其下方列出此条目的域，该域中的相应实际地址由 *ra* + (PA - *pa*) 指定。
- 如果未找到这样的条目，则 PA 不在任何域中。

CPU 和内存映射示例

例 16-4 确定域配置

以下命令将生成一个可解析的逻辑域配置长列表。

```
primary# ldm list -l -p
VERSION 1.6
DOMAIN|name=primary|state=active|flags=normal,control,vio-service|
cons=SP|ncpu=4|mem=1073741824|util=0.6|uptime=64801|
softstate=Solaris running
VCPU
|vid=0|pid=0|util=0.9|strand=100
|vid=1|pid=1|util=0.5|strand=100
|vid=2|pid=2|util=0.6|strand=100
|vid=3|pid=3|util=0.6|strand=100
MEMORY
|ra=0x8000000|pa=0x8000000|size=1073741824
IO
|dev=pci@780|alias=bus_a
|dev=pci@7c0|alias=bus_b
...
DOMAIN|name=ldg1|state=active|flags=normal|cons=5000|
ncpu=2|mem=805306368|util=29|uptime=903|
softstate=Solaris running
VCPU
|vid=0|pid=4|util=29|strand=100
|vid=1|pid=5|util=29|strand=100
MEMORY
|ra=0x8000000|pa=0x48000000|size=805306368
...
DOMAIN|name=ldg2|state=active|flags=normal|cons=5001|
```

```
ncpu=3|mem=1073741824|util=35|uptime=775|
softstate=Solaris running
VCPU
|vid=0|pid=6|util=35|strand=100
|vid=1|pid=7|util=34|strand=100
|vid=2|pid=8|util=35|strand=100
MEMORY
|ra=0x8000000|pa=0x78000000|size=1073741824
...
```

例 16-5 确定与物理 CPU 编号对应的虚拟 CPU

该逻辑域配置如例 16-4 “确定域配置” 所示。此示例介绍如何确定与物理 CPU 编号 `5` 对应的域和虚拟 CPU 以及与物理地址 `0x7e816000` 对应的域和实际地址。

在列表中浏览 `VCPU` 条目以查找 `pid` 字段等于 `5` 的条目，可以发现以下条目位于逻辑域 `ldg1` 之下。

```
|vid=1|pid=5|util=29|strand=100
```

因此，物理 CPU 编号 `5` 位于域 `ldg1` 中，在该域中对应的虚拟 CPU 编号为 `1`。

在列表中浏览 `MEMORY` 条目，可以发现以下条目位于域 `ldg2` 之下。

```
ra=0x8000000|pa=0x78000000|size=1073741824
```

其中 0x78000000 <= 0x7e816000 <= (0x78000000 + 1073741824 - 1)；即，*pa* <= PA <= (*pa* + *size* - 1)。因此，PA 位于域 `ldg2` 中，相应的实际地址为 0x8000000 + (0x7e816000 - 0x78000000) = 0xe816000。

使用通用唯一标识符

每个域会分配有一个通用唯一标识符 (universally unique identifier, UUID)。UUID 在创建域时分配。对于传统域，UUID 是在 `ldmd` 守护进程初始化时分配的。

注 - 如果使用 `ldm migrate-domain -f` 命令将域迁移到运行较早版本 Logical Domains Manager 的目标计算机，UUID 会丢失。从运行较早版本 Logical Domains Manager 的源计算机中迁移域时，会在迁移过程中为域分配新的 UUID。否则，会迁移 UUID。

可以通过运行 `ldm list -l`、`ldm list-bindings` 或 `ldm list -o domain` 命令来获取域的 UUID。以下示例显示了 `ldg1` 域的 UUID：

```
primary# ldm create ldg1
primary# ldm ls -l ldg1
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
```

```
ldg1             inactive   ------

UUID
    6c908858-12ef-e520-9eb3-f1cd3dbc3a59

primary# ldm ls -l -p ldg1
VERSION 1.6
DOMAIN|name=ldg1|state=inactive|flags=|cons=|ncpu=|mem=|util=|uptime=
UUID|uuid=6c908858-12ef-e520-9eb3-f1cd3dbc3a59
```

虚拟域信息命令和 API

使用 `virtinfo` 命令可以收集有关运行的虚拟域的信息。还可以使用虚拟域信息 API 创建程序以收集与虚拟域相关的信息。

以下列表显示了一些可以通过使用命令或 API 收集的有关虚拟域的信息。

- 域类型（实施、控制、来宾、I/O、服务、根）
- 由虚拟域管理器确定的域名
- 域的通用唯一标识符 (universally unique identifier, UUID)
- 域的控制域的网络节点名称
- 运行域的机箱序列号

有关 `virtinfo` 命令的信息，请参见 `virtinfo`(1M) 手册页。有关 API 的信息，请参见 `libv12n`(3LIB) 和 `v12n`(3EXT) 手册页。

使用逻辑域通道

Oracle VM Server for SPARC 使用逻辑域通道 (logical domain channel, LDC) 来实现所有通信，例如控制台、虚拟 I/O 和控制通信。LDC 是用来在两个端点之间实现通信的方法。虽然各个端点通常在不同的域中，但是端点也可以位于同一个域中以启用回送通信。

Oracle VM Server for SPARC 3.2 软件和系统固件提供了一个大型 LDC 端点池，可用于控制域和来宾域。此 LDC 端口池仅可用于 SPARC T4、SPARC T5、SPARC M5、SPARC M6 和 Fujitsu M10 平台。池中的 LDC 数量基于如下平台类型：

- SPARC T4 – 每个来宾域 1984 个 LDC 端点，总共 98304 个 LDC 端点
- SPARC T5 – 每个来宾域 1984 个 LDC 端点，总共 98304 个 LDC 端点
- SPARC M5 – 每个来宾域 1984 个 LDC 端点，每个物理域 98304 个 LDC 端点
- SPARC M6 – 每个来宾域 1984 个 LDC 端点，每个物理域 98304 个 LDC 端点
- Fujitsu M10 – 每个来宾域 1984 个 LDC 端点，总共 196608 个 LDC 端点

支持 LDC 端点池所需的系统固件为 8.5.2.b（针对 SPARC T4）和 9.2.1.b（针对 SPARC T5、SPARC M5 和 SPARC M6）以及 XCP2240（针对 Fujitsu M10）。

如果在支持的平台上或者在 UltraSPARC T2、UltraSPARC T2 Plus 或 SPARC T3 上运行较旧的系统固件版本，则 LDC 端点仍适用以下限制：

- UltraSPARC T2 系统 – 512 个 LDC 端点
- UltraSPARC T2 Plus、SPARC T3、SPARC T4、SPARC T5、SPARC M5、SPARC M6 和 Fujitsu M10 平台 – 768 个 LDC 端点

该限制可能会在控制域中造成问题，因为可能需要大量的 LDC 端点用于虚拟 I/O 数据通信和对其他域进行 Logical Domains Manager 控制。

在尝试添加服务或绑定域时，如果任何一个域的 LDC 端点的数量超过了上限，该操作会失败，并显示类似于以下内容的错误消息：

```
13 additional LDCs are required on guest primary to meet this request,
but only 9 LDCs are available
```

以下准则可用于正确规划 LDC 端点的使用并了解在控制域中为何可能会遇到 LDC 功能溢出：

- 控制域使用大约 15 个 LDC 端点来支持各种通信，包括虚拟机管理程序 (hypervisor)、故障管理体系结构 (Fault Management Architecture, FMA) 和系统处理器 (system processor, SC)，该数量与所配置的其他域的数量无关。控制域使用的 LDC 端点的数量取决于平台和所使用的软件版本。
- Logical Domains Manager 针对每个域（包括控制域自身）向控制域分配一个 LDC 端点以用于控制通信。
- 控制域上的每个虚拟 I/O 服务使用一个 LDC 端点来支持该服务的每个已连接客户机。每个域需要至少一个虚拟网络、一个虚拟磁盘和一个虚拟控制台。

以下公式根据这些准则来确定控制域所需的 LDC 端点数：

> 15 + *number-of-domains* + (*number-of-domains* x *number-of-virtual-services*) = *total-LDC-endpoints*

number-of-domains 是包括控制域在内的域总数，*number-of-virtual-services* 是由此域提供服务的虚拟 I/O 设备的数量。

以下示例展示了存在一个控制域和八个额外的域时如何使用此公式确定 LDC 端点的数量：

15 + 9 + (8 x 3) = 48 个 LDC 端点

以下示例有 45 个来宾域并且每个域包括五个虚拟磁盘、两个虚拟网络和一个虚拟控制台。计算将得到以下结果：

15 + 46 + 45 x 8 = 421 个 LDC 端点

Logical Domains Manager 将根据您的平台支持的 LDC 端点数接受或拒绝配置。

如果控制域中的 LDC 端点数不够用，请考虑创建服务域或 I/O 域来为来宾域提供虚拟 I/O 服务。此操作可以在 I/O 域和服务域上（而非在控制域上）创建 LDC 端点。

来宾域也可能会用完 LDC 端点。`inter-vnet-link` 属性设置为 `on` 可能会导致此情况，因为这需要向来宾域分配额外的 LDC 端点以直接连接到对方。

以下公式用于确定 `inter-vnet-link=off` 时来宾域需要的 LDC 端点数：

2 + *number-of-vnets* + *number-of-vdisks* = *total-LDC-endpoints*

2 表示虚拟控制台和控制通信，*number-of-vnets* 是分配给来宾域的虚拟网络设备的总数，*number-of-vdisks* 是分配给来宾域的虚拟磁盘的总数。

以下示例展示了如何使用公式来确定 `inter-vnet-link=off` 且有两个虚拟磁盘和两个虚拟网络时每个来宾域的 LDC 端点数：

2 + 2 + 2 = 6 个 LDC 端点

以下公式用于确定 `inter-vnet-link=on` 时来宾域需要的 LDC 端点数：

2 + [[(*number-of-vnets-from-vswX* x *number-of-vnets-in-vswX*)] ...] + *number-of-vdisks* = *total-LDC-endpoints*

2 表示虚拟控制台和控制通信，*number-of-vnets-from-vswX* 是从 *vswX* 虚拟交换机分配给来宾域的虚拟设备的总数，*number-of-vnets-in-vswX* 是 *vswX* 虚拟交换机上的虚拟网络设备的总数，*number-of-virtual-disks* 是分配给来宾域的虚拟磁盘的总数。

以下示例展示了如何使用公式来确定 `inter-vnet-link=on` 且有两个虚拟磁盘和两台虚拟交换机时每个来宾域的 LDC 端点数：第一台虚拟交换机有八个虚拟网络并且将其中四个分配给域。第二台虚拟交换机将其全部八个虚拟网络分配给域：

2 + (4 x 8) + (8 x 8) + 2 = 100 个 LDC 端点

要解决来宾域的 LDC 端点用完这一问题，请考虑使用 `ldm add-vsw` 或 `ldm set-vsw` 命令将 `inter-vnet-link` 属性设置为 `off`。此操作可减少具有虚拟网络设备的域的 LDC 端点数。不过，`off` 属性值不影响具有虚拟交换机的服务域，因为服务域仍然需要到每个虚拟网络设备的 LDC 连接。此属性设置为 `off` 时，不会为 inter-vnet 通信使用 LDC 通道。在此情况下，仅为虚拟网络设备和虚拟交换机设备之间的通信分配 LDC 通道。请参见 `ldm`(1M) 手册页。

注 - 尽管禁用 inter-vnet 链路分配可以减少 LDC 端点的数量，但这可能会对来宾域到来宾域的网络性能产生负面影响。之所以会发生此性能下降是因为，所有的来宾-来宾通信都经过虚拟交换机，而不是从一个来宾域直接到达另一个来宾域。

引导大量域

可以引导以下数量的域（具体值取决于所使用的平台）：

- 对于 Fujitsu M10 服务器：每个物理分区最多 256 个
- 对于 SPARC M6 系统：每个物理域最多 128 个
- 对于 SPARC M5 系统：每个物理域最多 128 个
- 对于 SPARC T5 系统：最多 128 个
- 对于 SPARC T4 服务器：最多 128 个
- 对于 SPARC T3 服务器：最多 128 个
- 对于 UltraSPARC T2 Plus 服务器：最多 128 个
- 对于 UltraSPARC T2 服务器：最多 64 个

如果存在未分配的虚拟 CPU，请将它们指定给服务域以帮助处理虚拟 I/O 请求。在创建 32 个以上的域时，为服务域分配 4 到 8 个虚拟 CPU。如果最大域配置是服务域中只有一个 CPU，则在配置和使用该域时不要为这个唯一的 CPU 施加不必要的压力。虚拟交换机 (`vsw`) 服务应当分布到计算机中的所有网络适配器上。例如，如果在 Sun SPARC Enterprise T5240 服务器上引导 128 个域，请创建 4 个 `vsw` 服务，每个服务为 32 个虚拟网络 (`vnet`) 实例提供服务。为每个 `vsw` 服务分配超过 32 个 `vnet` 实例可能导致服务域中发生硬挂起。

要运行最大配置，计算机需要足够的内存量来支持来宾域。内存量取决于所使用的平台和 OS。请参见有关所使用平台的文档、《Oracle Solaris 10 8/11 Installation Guide: Planning for Installation and Upgrade》、《Installing Oracle Solaris 11 Systems》和《Installing Oracle Solaris 11.1 Systems》。

如果来宾域所使用的 `vsw` 服务为多个域中的多个虚拟网络提供服务，则该域中的内存和交换空间使用量会增加。这一使用量增加是由于与 `vsw` 相连接的所有 `vnet` 实例之间的对等链路造成的。对于服务域来说，内存越多越好。在运行 64 个以上的域时，建议的最小值为四 GB。分组启动域（一组最多包含 10 个域）并等到它们引导之后再启动下一批。在域上安装操作系统时适用同样的建议。可以通过禁用 inter-vnet 链路来减少链路数量。请参见Inter-Vnet LDC 通道。

彻底关闭 Oracle VM Server for SPARC 系统并对该系统执行关开机循环

如果自上次将配置保存到 SC 以来对配置进行了任何更改，请务必在关闭 Oracle VM Server for SPARC 系统或对其执行关开机循环之前保存您希望保留的最新配置。

▼ 如何关闭具有多个活动域的系统

1. 关闭、停止并解除绑定所有的非 I/O 域。
2. 关闭、停止并解除绑定所有处于活动状态的 I/O 域。

3. 将域更改为初始状态 5。

```
primary# shutdown -i 5
```

还可以不使用 `shutdown` 命令，而改用 `init 5` 命令。

▼ 如何对系统执行关开机循环

1. 关闭系统电源。
   请参见如何关闭具有多个活动域的系统 [316]。
2. 使用 SP 为系统通电。

Logical Domains 变量持久性

变量更新在重新引导之后会保留，但在执行关开机循环之后不会保留，除非通过控制域上的 OpenBoot 固件启动变量更新，或者在执行变量更新之后将配置保存到 SC。

请注意以下条件：

- 当控制域重新引导时，如果没有绑定的来宾域而且未在进行延迟重新配置，SC 会对系统执行关开机循环。
- 当控制域重新引导时，如果已绑定来宾域或来宾域处于活动状态（或者控制域正在进行延迟重新配置），则 SC 不会对系统执行关开机循环。

可以使用以下任一方法来指定域的 Logical Domains 变量：

- 在 OpenBoot 提示符下。
- 使用 Oracle Solaris OS `eeprom(1M)` 命令。
- 使用 Logical Domains Manager CLI (`ldm`)。
- 在有限方式下，从系统控制器 (system controller, SC) 使用 `bootmode` 命令。此方法只能用于某些变量，并且只能在 `factory-default` 配置中使用。

使用上述任何方法进行的变量更新始终应当在域重新引导之后保留。变量更新还始终会应用于保存到 SC 的任何后续域配置。

在 Oracle VM Server for SPARC 3.2 软件中，少数情况下变量更新不会按所预期的那样保留：

- 所有变量更新方法所做的更新在该域重新引导之后均会保留。但是，除非随后将逻辑域配置保存到 SC，否则在对系统执行关开机循环之后，更新不会保留。

  但在控制域中，使用 OpenBoot 固件命令或 `eeprom` 命令 *do* 所做的更新会在系统循环开关机后保留，即使后续没有将新逻辑域配置保存到 SC 时也是如此。`eeprom` 在

SPARC T5、SPARC M5 和 SPARC M6 系统上以及在至少运行 8.2.1 版本的系统固件的 SPARC T3 和 SPARC T4 系统上支持此行为。

如果您希望保留 Logical Domains 变量更改，请执行以下操作之一：

- 将系统置于 `ok` 提示符下，然后更新变量。
- 在 Logical Domains Manager 处于禁用状态时更新变量：

```
# svcadm disable ldmd
update variables
# svcadm enable ldmd
```

- 在运行 Live Upgrade 时，执行以下步骤：

```
# svcadm disable -t ldmd
# luactivate be3
# init 6
```

如果您修改了逻辑域上的时间或日期（例如，使用 `ntpdate` 命令），则所做的更改在该域重新引导之后会保留，而在对相应主机执行关开机循环之后不会保留。要确保对时间所做的更改可以保留，请将包含时间更改的配置保存到 SP 并从该配置进行引导。

为解决这些问题，已记录了以下错误号：15375997、15387338、15387606 和 15415199。

调整中断限制

硬件提供了有限数目的中断，因此，Oracle Solaris 会限制每个设备可以使用的中断次数。默认限制应与典型系统配置的需求匹配，但可能需要为某些系统配置调整此值。

如果对 PCIe 总线启用 I/O 虚拟化，则会将中断硬件资源分配给每个 I/O 域。分配给每个域的这些资源的数量有限，这可能导致一些中断分配问题。这种情况仅影响 UltraSPARC T2、UltraSPARC T2 Plus、SPARC T3、SPARC T4、SPARC T5、SPARC M5 和 SPARC M6 平台。

Oracle Solaris 控制台上的以下警告表示在附加 I/O 设备驱动程序时中断供应已耗尽：

```
WARNING: ddi_intr_alloc: cannot fit into interrupt pool
```

具体而言，以下情况下可能需要调整此限制：当系统划分为多个逻辑域并且当向任意来宾域分配了太多的 I/O 设备时。Oracle VM Server for SPARC 会将中断总数划分为较小中断集并将其分配给来宾域。如果向某个来宾域分配了太多的 I/O 设备，则其供应可能太少以致于无法向每个设备分配默认的中断限制。因此，来宾域在完全附加所有驱动程序之前会耗尽其中断供应。

某些驱动程序提供了一个可选的回调例程，允许 Oracle Solaris OS 自动调整其中断。默认限制不适用于这些驱动程序。

使用 ::irmpools 和 ::irmreqs MDB 宏确定如何使用中断。::irmpools 宏显示划分为池的总体中断供应。::irmreqs 宏显示哪些设备映射到每个池。对于每个设备，::irmreqs 显示是否由一个可选的回调例程强制执行默认限制、每个驱动程序请求了多少中断，以及每个驱动程序拥有多少中断。

虽然这些宏不显示有关未能附加的驱动程序的信息，但可使用该信息计算可调整默认限制的限度。对于使用了多个中断且没有提供回调例程的任何设备，都可以通过调整默认限制来强制其使用较少的中断。对于此类设备，减少默认限制可以释放中断供其他设备使用。

要调整默认限制，请在 /etc/system 文件中将 ddi_msix_alloc_limit 属性设置为从 1 到 8 的某个值。然后，重新引导系统以使更改生效。

有关正确创建或更新 /etc/system 属性值的信息，请参见“更新 /etc/system 文件中的属性值” [302]。

为了最大限度地提高性能，请先分配较大的值，然后以较小的增量减小该值，直至系统成功引导且不出现任何警告。可使用 ::irmpools 和 ::irmreqs 宏来度量调整对所附加的所有驱动程序的影响。

例如，假设在引导来宾域中的 Oracle Solaris OS 时发出了以下警告：

```
WARNING: emlxs3: interrupt pool too full.
WARNING: ddi_intr_alloc: cannot fit into interrupt pool
```

::irmpools 和 ::irmreqs 宏显示以下信息：

```
# echo "::irmpools" | mdb -k
ADDR             OWNER   TYPE   SIZE  REQUESTED  RESERVED
00000400016be970 px#0    MSI/X  36    36         36

# echo "00000400016be970::irmreqs" | mdb -k
ADDR             OWNER   TYPE   CALLBACK NINTRS NREQ NAVAIL
00001000143acaa8 emlxs#0 MSI-X  No       32     8    8
00001000170199f8 emlxs#1 MSI-X  No       32     8    8
000010001400ca28 emlxs#2 MSI-X  No       32     8    8
0000100016151328 igb#3   MSI-X  No       10     3    3
0000100019549d30 igb#2   MSI-X  No       10     3    3
0000040000e0f878 igb#1   MSI-X  No       10     3    3
000010001955a5c8 igb#0   MSI-X  No       10     3    3
```

在此示例中，默认限制是每个设备 8 个中断，此中断限制不足以支持将最后的 emlxs3 设备附加到系统。假设所有 emlxs 实例的行为方式相同，则 emlxs3 可能请求了 8 个中断。

从总的池大小 36 个中断中减去由所有 igb 设备使用的 12 个中断，还剩下 24 个中断可供 emlxs 设备使用。将 24 个中断除以 4，这表明每个设备 6 个中断将能够使所有 emlxs 设备都可以附加，且具有相同的性能。因此，在 /etc/system 文件中添加以下调整：

```
set ddi_msix_alloc_limit = 6
```

有关正确创建或更新 /etc/system 属性值的信息，请参见“更新 /etc/system 文件中的属性值” [302]。

当系统成功引导且未出现任何警告时，::irmpools 和 ::irmreqs 宏会显示以下更新的消息：

```
primary# echo "::irmpools" | mdb -k
ADDR             OWNER   TYPE   SIZE  REQUESTED  RESERVED
00000400018ca868 px#0    MSI/X  36    36         36

# echo "00000400018ca868::irmreqs" | mdb -k
ADDR             OWNER   TYPE   CALLBACK NINTRS NREQ NAVAIL
0000100016143218 emlxs#0 MSI-X  No       32     8    6
0000100014269920 emlxs#1 MSI-X  No       32     8    6
000010001540be30 emlxs#2 MSI-X  No       32     8    6
00001000140cbe10 emlxs#3 MSI-X  No       32     8    6
00001000141210c0 igb#3   MSI-X  No       10     3    3
0000100017549d38 igb#2   MSI-X  No       10     3    3
0000040001ceac40 igb#1   MSI-X  No       10     3    3
000010001acc3480 igb#0   MSI-X  No       10     3    3
```

列出域 I/O 依赖关系

某个域的 I/O 操作经常由另一个域提供，如服务域或 I/O 域。例如，服务域可以导出虚拟设备，或者引导域可以提供对物理设备的直接访问。

请注意这些隐含的 I/O 依赖关系，因为服务域或引导域中的中断将导致依赖的域的服务也中断。

可以使用 ldm list-dependencies 命令查看域之间的 I/O 依赖关系。除了列出域的依赖关系以外，还可以反转输出以显示特定域的依赖项。

以下列表显示了可通过使用 ldm list-dependencies 命令查看的 I/O 依赖关系的类型：

VDISK
: 在虚拟磁盘连接到由虚拟磁盘服务器导出的虚拟磁盘后端时创建的依赖关系

VNET
: 在虚拟网络设备连接到虚拟交换机时创建的依赖关系

IOV
: 在 SR-IOV 虚拟功能与 SR-IOV 物理功能关联时创建的依赖关系

以下 ldm list-dependencies 命令显示了可用于查看域依赖关系信息的一些方法：

- 要显示详细的域依赖关系信息，请使用 -l 选项。

```
primary# ldm list-dependencies -l
DOMAIN         DEPENDENCY      TYPE      DEVICE
primary
```

```
svcdom
ldg0            primary         VDISK     primary-vds0/vdisk0
                                VNET      primary-vsw0/vnet0
                svcdom          VDISK     svcdom-vds0/vdisk1
                                VNET      svcdom-vsw0/vnet1
ldg1            primary         VDISK     primary-vds0/vdisk0
                                VNET      primary-vsw0/vnet0
                                IOV       /SYS/MB/NET0/IOVNET.PF0.VF0
                svcdom          VDISK     svcdom-vds0/vdisk1
                                VNET      svcdom-vsw0/vnet1
                                IOV       /SYS/MB/NET2/IOVNET.PF0.VF0
```

- 要显示有关按依赖关系分组的依赖项的详细信息，请同时使用 `-l` 和 `-r` 选项。

```
primary# ldm list-dependencies -r -l
DOMAIN          DEPENDENT       TYPE      DEVICE
primary         ldg0            VDISK     primary-vds0/vdisk0
                                VNET      primary-vsw0/vnet0
                ldg1            VDISK     primary-vds0/vdisk0
                                VNET      primary-vsw0/vnet0
                                IOV       /SYS/MB/NET0/IOVNET.PF0.VF0
svcdom          ldg0            VDISK     svcdom-vds0/vdisk1
                                VNET      svcdom-vsw0/vnet1
                ldg1            VDISK     svcdom-vds0/vdisk1
                                VNET      svcdom-vsw0/vnet1
                                IOV       /SYS/MB/NET2/IOVNET.PF0.VF0
```

部分 II

可选的 Oracle VM Server for SPARC 软件

本部分介绍可以与 Oracle VM Server for SPARC 3.2 软件一起使用的可选软件和功能。

♦ ♦ ♦ 第 17 章

17 Oracle VM Server for SPARC 物理机到虚拟机转换工具

本章包括以下主题：

Oracle VM Server for SPARC P2V 工具概述

Oracle VM Server for SPARC 物理机到虚拟机 (Physical-to-Virtual, P2V) 转换工具可以自动将现有物理系统转换为在芯片多线程 (chip multithreading, CMT) 系统上的逻辑域中运行 Oracle Solaris 10 OS 的虚拟系统。您可以从运行 Oracle Solaris 10 OS 或 Oracle Solaris 11 OS 的控制域运行 `ldmp2v` 命令，以将以下其中一个源系统转换为逻辑域：

- 运行版本至少为 Solaris 8、Solaris 9 或 Oracle Solaris 10 OS 的任何基于 `sun4u` SPARC 的系统
- 运行 Oracle Solaris 10 OS 但不在逻辑域中运行的任何 `sun4v` 系统

注 - `ldmp2v` 命令不支持任何基于 SPARC 且运行 Oracle Solaris 10 OS（包含 ZFS 根）或 Oracle Solaris 11 OS 的系统。

可在以下各阶段执行从物理系统到虚拟系统的转换：

- **收集阶段**。在物理源系统上运行。在 `collect` 阶段，会基于收集到的有关源系统的配置信息创建源系统的文件系统映像。
- **准备阶段**。在目标系统的控制域上运行。在 `prepare` 阶段，会基于 `collect` 阶段收集到的配置信息在目标系统上创建逻辑域。文件系统映像将恢复到一个或多个虚拟磁盘。可以使用 P2V 工具在纯文本文件或 ZFS 卷上创建虚拟磁盘。还可以在物理磁盘、LUN 或您创建的卷管理器卷上创建虚拟磁盘。映像会被修改，以允许它作为逻辑域运行。

- **转换阶段。**在目标系统的控制域上运行。在 `convert` 阶段，通过使用标准的 Oracle Solaris 升级过程将创建的逻辑域转换为运行 Oracle Solaris 10 OS 的逻辑域。

有关 P2V 工具的信息，请参见 `ldmp2v`(1M) 手册页。

以下几节介绍如何执行从物理系统到虚拟系统的转换。

收集阶段

收集阶段在要转换的系统上运行。要创建一致的文件系统映像，请确保系统尽可能以静默方式运行，且所有应用程序都已停止。`ldmp2v` 命令会创建所有已挂载 UFS 文件系统的备份，因此，请确保要移动到逻辑域的所有文件系统都已挂载。可以排除不需要移动的已挂载文件系统，例如 SAN 存储上的文件系统或将以其他方式移动的文件系统。可使用 `-x` 选项排除此类文件系统。使用 `-x` 选项排除的文件系统不会在来宾域上重新创建。可以使用 `-0` 选项排除文件和目录。

无需在源系统上进行任何更改。唯一要求是 `ldmp2v` 脚本已安装在控制域上。确保源系统上存在 `flarcreate` 实用程序。

准备阶段

准备阶段使用收集阶段收集到的数据创建与源系统相当的逻辑域。

可通过以下方式之一使用 `ldmp2v prepare` 命令：

- **自动模式。**此模式会自动创建虚拟磁盘并恢复文件系统数据。
  - 创建逻辑域以及与源系统上大小相同的所需虚拟磁盘。
  - 对磁盘进行分区并恢复文件系统。

    如果 `/`、`/usr` 和 `/var` 文件的总大小小于 10 GB，将自动调整这些文件系统的大小来适应 Oracle Solaris 10 OS 的更大磁盘空间要求。可以使用 `-x no-auto-adjust-fs` 选项禁用自动调整大小，或者使用 `-m` 选项手动调整文件系统的大小。
  - 修改逻辑域的 OS 映像，以使用适用于逻辑域的版本替换对物理硬件的所有引用。然后，可以使用常规 Oracle Solaris 升级过程将系统升级到 Oracle Solaris 10 OS。所做修改包括更新 `/etc/vfstab` 文件以包含新的磁盘名称。任何 Oracle Solaris Volume Manager 或 Veritas 卷管理器 (Veritas Volume Manager, VxVM) 封装的引导磁盘都会自动在此过程中取消封装。磁盘会在取消封装时转换为普通磁盘分片。如果 VxVM 安装在源系统上，则 P2V 进程会在创建的来宾域上禁用 VxVM。
- **非自动模式。**必须手动创建虚拟磁盘并恢复文件系统数据。通过此模式，您能够更改磁盘的大小和数量、分区和文件系统布局。此模式下的准备阶段仅会在文件系统上运行逻辑域创建和 OS 映像修改步骤。
- **清除模式。**删除逻辑域和由 `ldmp2v` 创建的所有底层后端设备。

转换阶段

在转换阶段，逻辑域使用 Oracle Solaris 升级过程升级到 Oracle Solaris 10 OS。升级操作会删除所有现有软件包并安装 Oracle Solaris 10 `sun4v` 软件包，该软件包会自动执行 `sun4u` 到 `sun4v` 的转换。`convert` 阶段可使用 Oracle Solaris DVD ISO 映像或网络安装映像。在 Oracle Solaris 10 系统上，您也可以使用 Oracle Solaris JumpStart 功能执行完全自动化的升级操作。

后端设备

可以基于以下多种后端类型为来宾域创建虚拟磁盘：文件 (`file`)、ZFS 卷 (`zvol`)、物理磁盘或 LUN (`disk`) 或卷管理器卷 (`disk`)。如果通过以下方式之一指定 `file` 或 `zvol` 作为后端类型，`ldmp2v` 命令将自动创建相应大小的文件或 ZFS 卷：

- 通过使用 `-b` 选项
- 通过在 `/etc/ldmp2v.conf` 文件中指定 `BACKEND_TYPE` 参数的值

通过 `disk` 后端类型，您可以将物理磁盘、LUN 或卷管理器卷（Oracle Solaris Volume Manager 和 Veritas 卷管理器 (Veritas Volume Manager, VxVM)）用作虚拟磁盘的后端设备。必须在开始 `prepare` 阶段之前创建相应大小的磁盘或卷。对于物理磁盘或 LUN，请指定后端设备作为块的分片 2 或磁盘的字符设备，例如 `/dev/dsk/c0t3d0s2`。对于卷管理器卷，请指定块或字符设备，例如为 Oracle Solaris Volume Manager 指定 `/dev/md/dsk/d100`，或为 VxVM 指定 `/dev/vx/dsk/ldomdg/vol1`。

除非使用 `-B` *backend*:*volume*:*vdisk* 选项指定卷和虚拟磁盘名称，否则为来宾域创建的卷和虚拟磁盘会采用给定的默认名称。

- *backend* 用于指定要使用的后端的名称。必须为 `disk` 后端类型指定 *backend*。对于 `file` 和 `zvol` 后端类型，*backend* 是可选的，可以用于为 `ldmp2v` 创建的文件或 ZFS 卷设置非默认名称。默认名称为 `$BACKEND_PREFIX/`*guest-name*`/disk`*N*。
- *volume* 对于所有后端类型都是可选的，可指定为来宾域创建的虚拟磁盘服务器卷的名称。如果没有指定，则 *volume* 为 *guest-name*`-vol`*N*。
- *vdisk* 对于所有后端类型都是可选的，可指定来宾域中卷的名称。如果没有指定，则 *vdisk* 为 `disk`*N*。

注 - 在转换过程中，虚拟磁盘临时命名为 *guest-name*`-disk`*N* 以确保该名称在控制域中是唯一的。

要为 *backend*、*volume* 或 *vdisk* 指定一个空值，请仅包括冒号分隔符。例如，指定 `-B ::vdisk001` 会将虚拟磁盘名称设置为 `vdisk001`，并使用后端和卷的默认名称。如果没有指定 *vdisk*，可省略结尾冒号分隔符。例如，`-B /ldoms/ldom1/vol001:vol001` 将后端文件

的名称指定为 `/ldoms/ldom1/vol001`，将卷名称指定为 `vol001`。默认的虚拟磁盘名称为 `disk0`。

安装 Oracle VM Server for SPARC P2V 工具

只能在目标系统的控制域上安装和配置 Oracle VM Server for SPARC P2V 工具软件包。不需要在源系统上安装该软件包。您可以直接将 `/usr/sbin/ldmp2v` 脚本从目标系统复制到源系统。

注 - 在 Oracle Solaris 10 系统上从 `SUNWldmp2v` 软件包安装 `ldmp2v`，而默认情况下在 Oracle Solaris 11 系统上从 `ldomsmanager` 软件包安装 `ldmp2v`。

使用 SPARC P2V 工具的先决条件

运行 Oracle VM Server for SPARC P2V 工具之前，请确保符合以下条件：

- 源系统上已安装有下列 Flash 归档文件修补程序：
  - 对于 Solaris 8 OS：修补程序 ID 109318-34 或更高版本
  - 对于 Solaris 9 OS：修补程序 ID 113434-06 或更高版本
- 目标系统在以下操作系统上至少运行 Logical Domains 1.1：
  - Oracle Solaris 10 10/08 OS
  - 带有相应 Logical Domains 1.1 修补程序的 Oracle Solaris 10 5/08 OS
- 来宾域至少运行 Oracle Solaris 10 5/08 OS
- 源系统至少运行 Solaris 8 OS

除了这些先决条件外，还请配置一个要由源系统和目标系统共享的 NFS 文件系统。此文件系统应可由 `root` 写入。但是，如果共享文件系统不可用，请使用足够大的本地文件系统来存放源系统在源系统和目标系统上的文件系统转储。

SPARC P2V 工具的使用限制

Oracle VM Server for SPARC P2V 工具具有以下限制：

- 仅支持 UFS 文件系统。
- 源系统仅支持普通磁盘 (`/dev/dsk/c0t0d0s0`)、Oracle Solaris Volume Manager 元设备 (`/dev/md/dsk/d`*NNN*) 和 VxVM 封装的引导磁盘。
- 在 P2V 进程执行期间，每个来宾域都只可具有一个虚拟交换机和虚拟磁盘服务器。P2V 转换后，可以将多个虚拟交换机和虚拟磁盘服务器添加到域。

- 在已封装的引导磁盘上，对 VxVM 卷的支持仅限于以下卷：`rootvol`、`swapvol`、`usr`、`var`、`opt` 和 `home`。这些卷的原始分片必须仍在引导磁盘上存在。在 Oracle Solaris 10 OS 上，P2V 工具支持 Veritas Volume Manager 5.x。不过，您也可以使用 P2V 工具转换使用 VxVM 的 Solaris 8 和 Solaris 9 操作系统。
- 如果在运行 `ldmp2v collect` 操作之前使用 `zoneadm detach` 命令对区域进行分离，可以转换具有区域的 Oracle Solaris 10 系统。P2V 转换完成之后，请使用 `zoneadm attach` 命令重新连接已在来宾域上创建的区域。有关在来宾域上执行这些步骤的信息，请参见《System Administration Guide: Oracle Solaris Containers-Resource Management and Oracle Solaris Zones》中的第 23 章 “Moving and Migrating Non-Global Zones (Tasks)”。

注 - P2V 工具不会更新任何区域配置（如区域路径或网络接口），也不会为区域路径移动或配置存储。您必须在来宾域上手动更新区域配置和移动区域路径。请参见《System Administration Guide: Oracle Solaris Containers-Resource Management and Oracle Solaris Zones》中的第 23 章 “Moving and Migrating Non-Global Zones (Tasks)”。

▼ 如何安装 Oracle VM Server for SPARC P2V 工具

此过程介绍了如何使用 `SUNWldmp2v` 软件包在 Oracle Solaris 10 系统上安装 `ldmp2v` 命令。

在 Oracle Solaris 11 系统上，默认情况下会在安装 `ldomsmanager` 软件包时安装 `ldmp2v` 命令。

1. 从 **`http://www.oracle.com/virtualization/index.html`** 上的 Oracle VM Server for SPARC 下载页面中，下载 P2V 软件包 **`SUNWldmp2v`**。

   `SUNWldmp2v` 软件包包括在 Oracle VM Server for SPARC zip 文件中。

2. 成为管理员。

   - 对于 Oracle Solaris 10，请参见《System Administration Guide: Security Services》中的“Configuring RBAC (Task Map)”。
   - 对于 Oracle Solaris 11.2，请参见《Securing Users and Processes in Oracle Solaris 11.2》中的第 1 章 “About Using Rights to Control Users and Processes”。

3. 使用 **`pkgadd`** 命令安装 **`SUNWldmp2v`** 软件包。

   ```
   # pkgadd -d . SUNWldmp2v
   ```

4. **创建 /etc/ldmp2v.conf 文件并配置下列默认属性：**

   - VDS – 虚拟磁盘服务的名称，例如 VDS="primary-vds0"
   - VSW – 虚拟交换机的名称，例如 VSW="primary-vsw0"
   - VCC – 虚拟控制台集中器的名称，例如 VCC="primary-vcc0"
   - BACKEND_TYPE – zvol、file 或 disk 的后端类型
   - BACKEND_SPARSE – 确定是将后端设备创建为稀疏卷或文件 (BACKEND_SPARSE="yes") 还是非稀疏卷或文件 (BACKEND_SPARSE="no")
   - BACKEND_PREFIX – 用于创建虚拟磁盘后端设备的位置

     BACKEND_TYPE="zvol" 时，将 BACKEND_PREFIX 值指定为 ZFS 数据集名称。BACKEND_TYPE="files" 时，BACKEND_PREFIX 值将解释为相对于 / 的目录的路径名称。

     例如，BACKEND_PREFIX="tank/ldoms" 将导致在 tank/ldoms/*domain-name* 数据集中创建 ZVOL，在 /tank/ldoms/*domain-name* 子目录中创建文件。

     BACKEND_PREFIX 属性不适用于 disk 后端。
   - BOOT_TIMEOUT – Oracle Solaris OS 引导的超时时间（以秒为单位）

   有关更多信息，请参见 ldmp2v.conf.sample 配置文件，该配置文件包含在可下载软件包中。

使用 ldmp2v 命令

本节介绍三阶段转换示例。

例 17-1 收集阶段示例

以下示例说明如何使用 ldmp2v collect 命令。

- **共享已挂载 NFS 的文件系统。**以下示例说明了执行 collect 步骤的最简单方法，其中源系统和目标系统共享一个已挂载 NFS 的文件系统。

  作为超级用户，确保所有所需的 UFS 文件系统都已挂载。

```
volumia# df -k
Filesystem            kbytes    used   avail capacity  Mounted on
/dev/dsk/c1t1d0s0    16516485  463289 15888032     3%    /
/proc                      0       0       0     0%    /proc
fd                         0       0       0     0%    /dev/fd
mnttab                     0       0       0     0%    /etc/mnttab
/dev/dsk/c1t1d0s3    8258597    4304 8171708     1%    /var
swap                 4487448      16 4487432     1%    /var/run
swap                 4487448      16 4487432     1%    /tmp
```

```
/dev/dsk/c1t0d0s0    1016122       9  955146     1%    /u01
vandikhout:/u1/home/dana
                     6230996752 1051158977 5179837775    17%    /home/dana
```

以下示例显示源系统和目标系统共享一个已挂载 NFS 的文件系统时如何运行收集工具：

```
volumia# ldmp2v collect -d /home/dana/volumia
Collecting system configuration ...
Archiving file systems ...
Determining which filesystems will be included in the archive...
Creating the archive...
895080 blocks
Archive creation complete.
```

- **没有共享已挂载 NFS 的文件系统。**如果源系统和目标系统没有共享已挂载 NFS 的文件系统，可将文件系统映像写入本地存储，稍后再将其复制到控制域。Flash 归档文件实用程序会自动排除它所创建的归档文件。

```
volumia# ldmp2v collect -d /var/tmp/volumia
Collecting system configuration ...
Archiving file systems ...
Determining which filesystems will be included in the archive...
Creating the archive...
895080 blocks
Archive creation complete.
```

将 Flash 归档文件和 manifest 文件从 /var/tmp/volumia 目录复制到目标系统。

提示 - 在某些情况下，ldmp2v 可能会显示 cpio 命令错误。最为常见的是，这些错误生成诸如 File size of etc/mnttab has increased by 435 之类的消息。您可以忽略与日志文件相关的消息，也可以忽略与反映系统状态的文件相关的消息。请确保仔细检查所有的错误消息。

- **跳过文件系统备份步骤。**如果您已使用第三方备份工具（如 NetBackup）为系统创建备份，则可以使用 none 归档方法跳过文件系统备份步骤。使用此选项时，将仅创建系统配置清单。

```
volumia# ldmp2v collect -d home/dana/p2v/volumia -a none
Collecting system configuration ...
The following file system(s) must be archived manually: / /u01 /var
```

请注意，如果源系统和目标系统没有共享由 -d 指定的目录，您必须将该目录的内容复制到控制域。必须在准备阶段之前将目录内容复制到控制域。

例 17-2 准备阶段示例

以下示例说明如何使用 ldmp2v prepare 命令。

- 以下示例使用 /etc/ldmp2v.conf 中配置的默认值创建名为 volumia 的逻辑域，同时保留物理系统的 MAC 地址：

```
# ldmp2v prepare -d /home/dana/p2v/volumia -o keep-mac volumia
Creating vdisks ...
Creating file systems ...
Populating file systems ...
Modifying guest domain OS image ...
Removing SVM configuration ...
Unmounting guest file systems ...
Creating domain volumia ...
Attaching vdisks to domain volumia ...
```

- 以下命令显示了有关 volumia 逻辑域的信息：

```
# ldm list -l volumia
NAME             STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  UPTIME
volumia          inactive   ------          2     4G

NETWORK
    NAME    SERVICE                DEVICE     MAC               MODE   PVID VID
    vnet0   primary-vsw0                       00:03:ba:1d:7a:5a        1

DISK
    NAME    DEVICE  TOUT MPGROUP        VOLUME                      SERVER
    disk0                                       volumia-vol0@primary-vds0
    disk1                                       volumia-vol1@primary-vds0
```

- 以下示例说明如何使用 -C 选项彻底删除域及其后端设备。

```
# ldmp2v prepare -C volumia
Cleaning up domain volumia ...
Removing vdisk disk0 ...
Removing vdisk disk1 ...
Removing domain volumia ...
Removing volume volumia-vol0@primary-vds0 ...
Removing ZFS volume tank/ldoms/volumia/disk0 ...
Removing volume volumia-vol1@primary-vds0 ...
Removing ZFS volume tank/ldoms/volumia/disk1 ...
```

- 以下示例说明如何通过使用 -m 选项指定挂载点和新大小来在 P2V 过程中调整一个或多个文件系统的大小。

```
# ldmp2v prepare -d /home/dana/p2v/volumia -m /:8g volumia
Resizing file systems ...
```

```
Creating vdisks ...
Creating file systems ...
Populating file systems ...
Modifying guest domain OS image ...
Removing SVM configuration ...
Modifying file systems on SVM devices ...
Unmounting guest file systems ...
Creating domain volumia ...
Attaching vdisks to domain volumia ...
```

例 17-3 转换阶段示例

以下示例说明如何使用 ldmp2v convert 命令。

- **使用网络安装服务器。**ldmp2v convert 命令可使用指定的虚拟网络接口通过网络引导域。必须在安装服务器上运行 setup_install_server 和 add_install_client 脚本。

  在 Oracle Solaris 10 系统上，您可以使用 Oracle Solaris JumpStart 功能执行完全自动化的转换。此功能需要您在 JumpStart 服务器上为客户机创建和配置相应的 sysidcfg 文件和配置文件。该配置文件应包含以下行：

  ```
  install_type    upgrade
  root_device     c0d0s0
  ```

  sysidcfg 文件仅用于升级操作，因此如下配置应已足够：

  ```
  name_service=NONE
  root_password=uQkoXlMLCsZhI
  system_locale=C
  timeserver=localhost
  timezone=Europe/Amsterdam
  terminal=vt100
  security_policy=NONE
  nfs4_domain=dynamic
  auto_reg=disable
  network_interface=PRIMARY {netmask=255.255.255.192
          default_route=none protocol_ipv6=no}
  ```

  有关使用 JumpStart 的更多信息，请参见《Oracle Solaris 10 1/13 Installation Guide: JumpStart Installations》。

注 - 示例 sysidcfg 文件包括 auto_reg 关键字，此关键字是在 Oracle Solaris 10 9/10 发行版中引入的。只有当您至少运行 Oracle Solaris 10 9/10 发行版时，才需要使用此关键字。

```
# ldmp2v convert -j -n vnet0 -d /p2v/volumia volumia
```

```
LDom volumia started
Waiting for Solaris to come up ...
Using Custom JumpStart
Trying 0.0.0.0...
Connected to 0.
Escape character is '^]'.

Connecting to console "volumia" in group "volumia" ....
Press ~? for control options ..
SunOS Release 5.10 Version Generic_137137-09 64-bit
Copyright (c) 1983-2010, Oracle and/or its affiliates.  All rights reserved.
Configuring devices.
Using RPC Bootparams for network configuration information.
Attempting to configure interface vnet0...
Configured interface vnet0
Reading ZFS config: done.
Setting up Java. Please wait...
Serial console, reverting to text install
Beginning system identification...
Searching for configuration file(s)...
Using sysid configuration file
  129.159.206.54:/opt/SUNWjet/Clients/volumia/sysidcfg
Search complete.
Discovering additional network configuration...
Completing system identification...
Starting remote procedure call (RPC) services: done.
System identification complete.
Starting Solaris installation program...
Searching for JumpStart directory...
Using rules.ok from 129.159.206.54:/opt/SUNWjet.
Checking rules.ok file...
Using begin script: Clients/volumia/begin
Using profile: Clients/volumia/profile
Using finish script: Clients/volumia/finish
Executing JumpStart preinstall phase...
Executing begin script "Clients/volumia/begin"...
Begin script Clients/volumia/begin execution completed.
Searching for SolStart directory...
Checking rules.ok file...
Using begin script: install_begin
Using finish script: patch_finish
Executing SolStart preinstall phase...
Executing begin script "install_begin"...
Begin script install_begin execution completed.
WARNING: Backup media not specified.  A backup media (backup_media)
  keyword must be specified if an upgrade with disk space reallocation
```

```
  is required

Processing profile

Loading local environment and services

Generating upgrade actions
Checking file system space: 100% completed
Space check complete.

Building upgrade script

Preparing system for Solaris upgrade

Upgrading Solaris: 10% completed
[...]
```

- **使用 ISO 映像。**`ldmp2v convert` 命令可将 Oracle Solaris DVD ISO 映像附加到逻辑域并从中进行引导。要进行升级，请回答所有 `sysid` 提示，然后选择“升级”。

注意 - 在转换来宾域之前需执行安全检查。此检查可确保初始系统的 IP 地址不处于活动状态，以防止网络上出现重复的活动 IP 地址。可以使用 `-x skip-ping-test` 选项跳过此安全检查。跳过此检查可加快转换过程。只有当您确信不存在重复的 IP 地址（例如，初始主机未处于活动状态）时，才能使用此选项。

对 `sysid` 问题的回答仅会在升级过程期间使用。此数据不会应用至磁盘上的现有 OS 映像。执行转换的最快速、最简单的方法是选择“非联网”。指定的 `root` 密码不需要与源系统的 `root` 密码匹配。升级操作会保留系统的原始标识，该标识将在升级后的重新引导后生效。执行升级所需的时间取决于原始系统上安装的 Oracle Solaris Cluster。

```
# ldmp2v convert -i /tank/iso/s10s_u5.iso -d /home/dana/p2v/volumia volumia
Testing original system status ...
LDom volumia started
Waiting for Solaris to come up ...

        Select 'Upgrade' (F2) when prompted for the installation type.
        Disconnect from the console after the Upgrade has finished.

Trying 0.0.0.0...
Connected to 0.
Escape character is '^]'.

Connecting to console "volumia" in group "volumia" ....
Press ~? for control options ..
```

```
Configuring devices.
Using RPC Bootparams for network configuration information.
Attempting to configure interface vnet0...
Extracting windowing system. Please wait...
Beginning system identification...
Searching for configuration file(s)...
Search complete.
Discovering additional network configuration...
Configured interface vnet0
Setting up Java. Please wait...

Select a Language

   0. English
   1. French
   2. German
   3. Italian
   4. Japanese
   5. Korean
   6. Simplified Chinese
   7. Spanish
   8. Swedish
   9. Traditional Chinese

Please make a choice (0 - 9), or press h or ? for help:
[...]
- Solaris Interactive Installation ----------------------------------------------

  This system is upgradable, so there are two ways to install the Solaris
  software.

  The Upgrade option updates the Solaris software to the new release, saving
  as many modifications to the previous version of Solaris software as
  possible.  Back up the system before using the Upgrade option.

  The Initial option overwrites the system disks with the new version of
  Solaris software.  This option allows you to preserve any existing file
  systems.  Back up any modifications made to the previous version of Solaris
  software before starting the Initial option.

  After you select an option and complete the tasks that follow, a summary of
  your actions will be displayed.

-------------------------------------------------------------------------------
      F2_Upgrade    F3_Go Back    F4_Initial    F5_Exit    F6_Help
```

Oracle VM Server for SPARC 物理机到虚拟机转换工具的已知问题

ldmp2v convert 命令：引导期间出现 VxVM 警告消息

在 Oracle Solaris 10 OS 上，对于 Oracle VM Server for SPARC P2V 工具来说，Veritas Volume Manager (VxVM) 5.*x* 是唯一受支持（经过测试）的版本。在 Solaris 8 和 Solaris 9 操作系统上，早期版本的 VxVM（如 3.*x* 和 4.*x*）可能也可以正常工作。在这些情况下，在运行 `ldmp2v convert` 命令之后首次引导时可能会显示来自 VxVM 驱动程序的警告消息。可以忽略这些消息。在来宾域引导之后，可以删除旧的 `VRTS*` 软件包。

```
Boot device: disk0:a File and args:
SunOS Release 5.10 Version Generic_139555-08 64-bit
Copyright 1983-2009 Sun Microsystems, Inc. All rights reserved.
Use is subject to license terms.
Hostname: normaal
Configuring devices.
/kernel/drv/sparcv9/vxdmp: undefined symbol 'romp'
WARNING: mod_load: cannot load module 'vxdmp'
WARNING: vxdmp: unable to resolve dependency, module 'misc/ted' not found
/kernel/drv/sparcv9/vxdmp: undefined symbol 'romp'
WARNING: mod_load: cannot load module 'vxdmp'
WARNING: vxdmp: unable to resolve dependency, module 'misc/ted' not found
/kernel/drv/sparcv9/vxio: undefined symbol 'romp'
WARNING: mod_load: cannot load module 'vxio'
WARNING: vxio: unable to resolve dependency, module 'drv/vxdmp' not found
WARNING: vxspec : CANNOT INITIALIZE vxio DRIVER
WARNING: VxVM vxspec V-5-0-0 vxspec: vxio not loaded. Aborting vxspec load
WARNING: vxspec : CANNOT INITIALIZE vxio DRIVER
WARNING: VxVM vxspec V-5-0-0 vxspec: vxio not loaded. Aborting vxspec load
WARNING: vxspec : CANNOT INITIALIZE vxio DRIVER
WARNING: VxVM vxspec V-5-0-0 vxspec: vxio not loaded. Aborting vxspec load
WARNING: vxspec : CANNOT INITIALIZE vxio DRIVER
WARNING: VxVM vxspec V-5-0-0 vxspec: vxio not loaded. Aborting vxspec load
WARNING: vxspec : CANNOT INITIALIZE vxio DRIVER
WARNING: VxVM vxspec V-5-0-0 vxspec: vxio not loaded. Aborting vxspec load
WARNING: vxspec : CANNOT INITIALIZE vxio DRIVER
WARNING: VxVM vxspec V-5-0-0 vxspec: vxio not loaded. Aborting vxspec load
WARNING: vxspec : CANNOT INITIALIZE vxio DRIVER
NOTICE: VxVM not started
```

使用 `ldmp2v prepare -R` 时不显示 "Upgrade"（升级）选项

如果未将保存根 (/) 文件系统的分片的分区标记设为 `root`，Oracle Solaris 安装程序将不会显示 "Upgrade"（升级）选项。如果在标记来宾的引导磁盘时未显式设置该标记，则会发生此情况。可以按如下所示使用 `format` 命令设置分区标记：

```
AVAILABLE DISK SELECTIONS:
0. c0d0 <SUN-DiskImage-10GB cyl 282 alt 2 hd 96 sec 768>
  /virtual-devices@100/channel-devices@200/disk@0
1. c4t2d0 <SUN146G cyl 14087 alt 2 hd 24 sec 848>
  /pci@400/pci@0/pci@1/scsi@0/sd@2,0
2. c4t3d0 <SUN146G cyl 14087 alt 2 hd 24 sec 848>
  /pci@400/pci@0/pci@1/scsi@0/sd@3,0
Specify disk (enter its number)[0]: 0
selecting c0d0
[disk formatted, no defect list found]
format> p


PARTITION MENU:
0      - change `0' partition
1      - change `1' partition
2      - change `2' partition
3      - change `3' partition
4      - change `4' partition
5      - change `5' partition
6      - change `6' partition
7      - change `7' partition
select - select a predefined table
modify - modify a predefined partition table
name   - name the current table
print  - display the current table
label  - write partition map and label to the disk
!<cmd> - execute <cmd>, then return
quit

partition> 0
Part      Tag    Flag     Cylinders       Size            Blocks
0 unassigned    wm       0              0         (0/0/0)          0

Enter partition id tag[unassigned]: root
Enter partition permission flags[wm]:
Enter new starting cyl[0]: 0
Enter partition size[0b, 0c, 0e, 0.00mb, 0.00gb]: 8g
partition> label
Ready to label disk, continue? y

partition>
```

ldmp2v 命令：不再使用 ufsdump 归档方法

恢复受 UFS 文件系统上的文件支持的虚拟磁盘上的 ufsdump 归档可能会导致系统挂起。在这种情况下，ldmp2v prepare 命令将退出。如果虚拟磁盘是 UFS 文件系统上的文件，则在手动恢复 ufsdump 归档以便为 ldmp2v prepare -R /altroot 命令做准备时，可能会遇到此问题。为了与以前创建的 ufsdump 归档兼容，仍可以使用 ldmp2v prepare 命令恢复不受 UFS 文件系统上的文件支持的虚拟磁盘上的 ufsdump 归档。不过，建议不要使用 ufsdump 归档。

♦ ♦ ♦ 第 18 章

18 Oracle VM Server for SPARC Configuration Assistant (Oracle Solaris 10)

Oracle VM Server for SPARC Configuration Assistant（`ldmconfig` 命令）将引导您完成通过设置基本属性配置逻辑域的过程。它在基于芯片多线程 (chip multithreading, CMT) 的系统上运行。

收集配置数据之后，Configuration Assistant 将创建适合于作为逻辑域引导的配置。还可以使用 Configuration Assistant 选定的默认值创建可用的系统配置。

注 - `ldmconfig` 命令仅在 Oracle Solaris 10 系统上受支持。

除了本章，还可参见 `ldmconfig`(1M) 手册页。

使用 Configuration Assistant (`ldmconfig`)

`ldmconfig` 命令通过与用户界面屏幕相对应的一系列操作工作。最终结果是创建可部署到逻辑域的配置。

以下各节说明了如何安装 `ldmconfig` 命令以及 Configuration Assistant 工具的一些功能。

安装 Configuration Assistant

Configuration Assistant 作为 `SUNWldm` 软件包的一部分提供。

安装 `SUNWldm` 软件包之后，可以在 `/usr/sbin` 目录中找到 `ldmconfig` 命令。出于传统目的，该命令同时安装于 `/opt/SUNWldm/bin` 目录。

运行 Configuration Assistant 的先决条件

安装并运行 Configuration Assistant 之前，请确保满足以下条件：

- 目标系统必须至少运行 Logical Domains 1.2 软件。
- 您的终端窗口必须至少 80 个字符宽、24 行长。

Configuration Assistant 的限制和已知问题

Configuration Assistant 有以下限制：

- 使用 ldmconfig 时调整终端大小可能会导致输出乱码
- 仅支持 UFS 磁盘文件作为虚拟磁盘
- 仅可在不存在现有逻辑域配置的系统上工作
- 虚拟控制台集中器端口从 5000 到 5100
- 无法更改用于来宾域、服务和设备的默认名称

ldmconfig 功能

ldmconfig 命令通过与用户界面屏幕相对应的一系列操作工作。可以在这些屏幕中向后（上一步）和向前（下一步）导航，直到到达最后一步。最后一步将生成配置。任何时候都可以退出 Configuration Assistant 或重置配置以使用默认值。在最终屏幕中，可以将配置部署到逻辑域。

首先，Configuration Assistant 自动检查系统以根据最佳实践确定最合适的默认属性值，然后显示控制部署所需的这些属性。请注意，这不是详尽的列表。可以设置其他属性以进一步定制配置。

有关使用 ldmconfig 工具的信息，请参见 ldmconfig(1M) 手册页。

您可以调整以下属性：

- **来宾域数量**。指定要创建的应用程序来宾域数。最少为一个来宾域。最大值由 VCPU 资源的可用性确定。例如，在 64 线程的 CMT 系统上，最多可以创建 60 个来宾域（每个域一个线程，保留四个线程用于控制域）。如果选择了最佳实践，则每个来宾域的最小 VCPU 资源数是单个核心。因此，在已选择最佳实践的 8 核心、每核心 8 线程的系统上，最多可以创建七个来宾域，每个域一个核心。此外，将一个核心分配给控制域。

  Configuration Assistant 显示可为此系统配置的域的最大数目。

  Configuration Assistant 执行以下任务来创建域：

  - 对于所有域：
    - 在端口 5000 到 5100 上创建虚拟终端服务
    - 创建虚拟磁盘服务
    - 在指定的网络适配器上创建虚拟网络交换机
    - 启用虚拟终端服务器守护进程
  - 对于每个域：

    - 创建逻辑域
    - 配置分配给域的 VCPU
    - 配置分配给域的内存
    - 创建用作虚拟磁盘的 UFS 磁盘文件
    - 为磁盘文件创建虚拟磁盘服务器设备 (vdsdev)
    - 将磁盘文件指定为域的虚拟磁盘 vdisk0
    - 添加连接到指定的网络适配器上的虚拟交换机的虚拟网络适配器
    - 设置 OBP 属性 auto-boot?=true
    - 设置 OBP 属性 boot-device=vdisk0
    - 绑定域
    - 启动域
- **默认网络**。指定新域将用于虚拟网络的网络适配器。该适配器必须存在于系统中。Configuration Assistant 将突出显示系统当前用作默认适配器的适配器，以及具有活动链路状态的适配器（已连线的适配器）。
- **虚拟磁盘大小**。为每个新域创建虚拟磁盘。这些虚拟磁盘是根据位于本地文件系统中的磁盘文件创建的。此属性控制每个虚拟磁盘的大小（以 GB 为单位）。最小大小 8 GB 基于包含 Oracle Solaris 10 OS 所需的近似大小，最大大小为 100 GB。

  如果 Configuration Assistant 找不到具有足够空间来包含所有域的磁盘文件的文件系统，将显示错误屏幕。这种情况下，可能需要在重新运行应用程序之前执行以下操作：

    - 减小虚拟磁盘的大小
    - 减少域数
    - 添加更多更大容量的文件系统
- **虚拟磁盘目录**。指定具有足够容量的文件系统，用于存储将作为新域的虚拟磁盘创建的文件。该目录基于选定的域数和虚拟磁盘的大小。每次更改这些属性值时，必须重新计算该值并选择目标目录。Configuration Assistant 提供具有足够空间的文件系统列表。指定文件系统名称之后，Configuration Assistant 在此文件系统中创建名为 /ldoms/disks 的目录，在该目录中创建磁盘映像。
- **最佳实践**。指定是否对属性值使用最佳实践。
    - 值为 yes 时，Configuration Assistant 将为几个配置属性值使用最佳实践。它将强制使用每个域一个核心的最小值，包括系统域。此设置会将来宾域的最大数量限制为系统中存在的核心总数减去用于系统域的一个核心。例如，对于每路八个核心的双路SPARC EnterpriseT5140，来宾域的最大数是 15（加上系统域）。
    - 值为 no 时，Configuration Assistant 允许创建最少有一个线程的域，但至少会保留四个线程用于系统域。

接下来，Configuration Assistant 将概述要创建的部署配置，其中包括以下信息：

- 域数量
- 分配给每个来宾域的 CPU
- 分配给每个来宾域的内存

- 虚拟磁盘的大小和位置
- 将用于来宾域的虚拟网络服务的网络适配器
- 系统将用于服务的 CPU 和内存量
- 如果识别出有效的 Oracle Solaris OS DVD，将使用它来创建共享虚拟 CD-ROM 设备，以允许来宾域安装 Oracle Solaris OS

最后，Configuration Assistant 将配置系统，以创建指定的 Oracle VM Server for SPARC 部署。它还会说明要采取的操作并显示要运行的命令，以配置系统。此信息可帮助您了解如何使用配置系统所需的 `ldm` 命令。

注意 - 不要与此配置步骤进行交互，也不要中断此过程，否则可能会导致系统配置不完整。

命令成功完成后，重新引导系统以使更改生效。

♦ ♦ ♦ 第 19 章

19 使用电源管理

本章包含有关在 Oracle VM Server for SPARC 系统上使用电源管理的信息。

使用电源管理

要启用电源管理 (Power Management, PM)，需要先在 Oracle Integrated Lights Out Manager (ILOM) 3.0 固件中设置 PM 策略。本节汇总了所需的信息，以便能够在 Oracle VM Server for SPARC 软件中使用 PM。

有关 ILOM 的更多信息，请参见以下内容：

- 《*Oracle Integrated Lights Out Manager (ILOM) 3.0 CLI* 过程指南》中的“监视功耗”
- 《*Oracle Integrated Lights Out Manager (ILOM) 3.0* 功能更新和发行说明》

电源策略可在任意时间点管理系统的用电。支持以下电源策略，并假定底层平台已实现 PM 功能：

- **禁用**。系统可以使用所有可用功率。
- **性能**。启用下列一项或多项 PM 功能，这些功能对性能的影响微乎其微：
  - CPU 核心自动禁用
  - CPU 时钟周期跳步
  - CPU 动态电压和频率调节 (dynamic voltage and frequency scaling, DVFS)
  - 一致性链路调节
  - Oracle Solaris 电源感知分配器 (Power Aware Dispatcher, PAD)
- **弹性**。使用性能一节中描述的 PM 功能将系统电源使用率调至当前利用率水平。例如，利用率降低，资源的电源状态也将降级。

电源管理功能

PM 功能如下：

- **CPU 核心自动禁用。**启用弹性或性能策略后，如果某个 CPU 核心上的所有硬件线程（导线束）均未绑定到某个域，则 Logical Domains Manager 将自动禁用该核心。此功能仅适用于 UltraSPARC T2、UltraSPARC T2 Plus、SPARC T3 和 SPARC T4 平台。
- **CPU 时钟周期跳步。**弹性策略生效后，Logical Domains Manager 将在绑定到域的以下 CPU 资源上自动调整执行指令的时钟周期数：
  - 处理器（在运行 Oracle Solaris 10 或 Oracle Solaris 11 OS 的域上的 SPARC T3 或 SPARC T4）
  - 核心（仅限在运行 Oracle Solaris 10 OS 的域上的 SPARC M5）
  - 核心对（仅限在运行 Oracle Solaris 10 OS 的域上的 SPARC T5 或 SPARC M6）

  如果处理器、核心或核心对没有绑定导线束，则 Logical Domains Manager 也会应用周期跳步。
- **CPU 动态电压和频率调节 (dynamic voltage and frequency scaling, DVFS)。**弹性策略生效后，Logical Domains Manager 会自动调整与运行 Oracle Solaris 10 OS 的域绑定的处理器的时钟频率。Logical Domains Manager 还会降低未绑定导线束的 SPARC T5、SPARC M5 和 SPARC M6 处理器的时钟频率。此功能只在 SPARC T5、SPARC M5 和 SPARC M6 系统上可用。
- **一致性链路调节。**弹性策略生效后，Logical Domains Manager 将使虚拟机管理程序自动调整正在使用的一致性链路数。此功能仅在 SPARC T5-2 系统中提供。
- **功率极限。**您可以在 SPARC T3、SPARC T4、SPARC T5、SPARC M5 和 SPARC M6 平台上设置功率极限，以限制系统的功率消耗。如果功率消耗超过功率极限，则 PM 将采用一些方法来降低功率。可以使用 ILOM 服务处理器 (service processor, SP) 设置功率极限。

  请参见以下文档：

  - 《*Oracle Integrated Lights Out Manager (ILOM) 3.0 CLI* 过程指南》
  - 《*Oracle Integrated Lights Out Manager (ILOM) 3.0* 功能更新和发行说明》

  可以使用 ILOM 界面设置功率极限、宽限期和违规操作。如果宽限期过后仍超出功率极限，将执行违规操作。

  如果当前功率消耗超出功率极限，则系统会尝试对 CPU 的电源状态进行降级。如果功率消耗降到功率极限以下，则允许升级这些资源的电源状态。如果系统具有有效的弹性策略，则根据利用率级别来确定是否升级资源的电源状态。
- **Solaris 电源感知分配器 (Power Aware Dispatcher, PAD)。**运行 Oracle Solaris 11.1 OS 的来宾域在 SPARC T5、SPARC M5 和 SPARC M6 系统上使用电源感知分配器 (power-aware dispatcher, PAD) 来最大程度减少空闲或未充分利用的资源的功耗。PAD 取代了 Logical Domains Manager，用于调整 CPU 时钟周期跳步级别以及 DVFS 级别。

有关使用 ILOM 3.0 固件 CLI 配置电源策略的说明，请参见《*Oracle Integrated Lights Out Manager (ILOM) 3.0 CLI* 过程指南》中的“监视功耗”。

查看功耗数据

Oracle VM Server for SPARC 3.1 软件提供电源管理 (Power Management, PM) 可观察性模块和 `ldmpower` 命令，允许您查看域的 CPU 线程功耗数据。

`ldmd/pm_observability_enabled` 服务管理工具 (Service Management Facility, SMF) 属性设置为 `true` 时，PM 可观察性模块默认情况下处于启用状态。请参见 `ldmd`(1M) 手册页。

`ldmpower` 命令具有以下选项和操作数，使用这些选项和操作数可以定制功耗报告数据：

```
ldmpower [-ehiprstvx | -o hours | -m minutes] | -c resource [-l domain-name[,domain-
name[,...]]]
  [interval [count]]
```

有关选项的信息，请参见 `ldmpower`(1M) 手册页。

要以非特权用户身份运行此命令，您必须分配有 `LDoms Power Mgmt Observability` 权限配置文件。如果您已分配有 `LDoms Management` 或 `LDoms Review` 权限配置文件，则您将自动具有运行 `ldmpower` 命令的权限。

有关 Oracle VM Server for SPARC 如何使用权限的信息，请参见“Logical Domains Manager 配置文件内容” [26]。

这些权限配置文件可以直接分配给用户或之后会分配给用户的角色。如果将其中一种配置文件直接分配给用户，您必须使用 `pfexec` 命令或配置文件 shell（例如 `pfbash` 或 `pfksh`），以成功使用 `ldmpower` 命令查看 CPU 线程功耗数据。请参见“通过使用权限委托管理逻辑域” [23]。

以下示例显示如何启用 PM 可观察性模块以及收集分配给域的 CPU 功耗数据的方式。

例 19-1 启用电源管理可观察性模块

以下命令通过将 `ldmd/pm_observability_enabled` 属性设置为 `true`（如果该属性当前设置为 `false`）来启用 PM 可观察性模块。

```
# svccfg -s ldmd setprop ldmd/pm_observability_enabled=true
# svcadm refresh ldmd
# svcadm restart ldmd
```

例 19-2 使用配置文件 Shell 通过角色和权限配置文件获取 CPU 线程功耗数据

- 以下示例显示如何创建具有 `LDoms Power Mgmt Observability` 权限配置文件的 `ldmpower` 角色，该角色允许您运行 `ldmpower` 命令。

  ```
  primary# roleadd -P "LDoms Power Mgmt Observability" ldmpower
  primary# passwd ldmpower
  ```

```
New Password:
Re-enter new Password:
passwd: password successfully changed for ldmpower
```

此命令将 ldmpower 角色分配给 sam 用户。

```
primary# usermod -R ldmpower sam
```

用户 sam 承担 ldmpower 角色，可以使用 ldmpower 命令。例如：

```
$ id
uid=700299(sam) gid=1(other)
$ su ldmpower
Password:
$ pfexec ldmpower
Processor Power Consumption in Watts
DOMAIN  15_SEC_AVG  30_SEC_AVG  60_SEC_AVG
primary 75          84          86
gdom1   47          24          19
gdom2   10          24          26
```

- 以下示例说明如何使用权限配置文件运行 ldmpower 命令。
  - Oracle Solaris 10：将权限配置文件分配给 *username*。

    ```
    primary# usermod -P "All,Basic Solaris User,LDoms Power Mgmt Observability" \
    username
    ```

    以下命令显示如何验证用户是否为 sam 以及 All、Basic Solaris User 和 LDoms Power Mgmt Observability 权限配置文件是否有效。

    ```
    $ id
    uid=702048(sam) gid=1(other)
    $ profiles
    All
    Basic Solaris User
    LDoms Power Mgmt Observability
    $ pfexec ldmpower
    Processor Power Consumption in Watts
    DOMAIN  15_SEC_AVG  30_SEC_AVG  60_SEC_AVG
    primary 75          84          86
    gdom1   47          24          19
    gdom2   10          24          26
    ```

  - Oracle Solaris 11：将权限配置文件分配给用户。

    ```
    primary# usermod -P +"LDoms Power Mgmt Observability" sam
    ```

    以下命令显示如何验证用户是否为 sam 以及 All、Basic Solaris User 和 LDoms Power Mgmt Observability 权限配置文件是否有效。

```
$ id
uid=702048(sam) gid=1(other)
$ profiles
All
Basic Solaris User
LDoms Power Mgmt Observability
$ pfexec ldmpower
Processor Power Consumption in Watts
DOMAIN  15_SEC_AVG  30_SEC_AVG  60_SEC_AVG
primary 75          84          86
gdom1   47          24          19
gdom2   10          24          26
```

例 19-3 查看处理器功耗数据

以下示例说明如何使用 ldmpower 报告域的处理器功耗数据。

- 以下命令显示所有域的 15 秒、30 秒和 60 秒处理器功耗数据移动平均值：

```
primary# ldmpower
Processor Power Consumption in Watts
DOMAIN  15_SEC_AVG  30_SEC_AVG  60_SEC_AVG
primary 75          84          86
gdom1   47          24          19
gdom2   10          24          26
```

- 以下命令显示所有域的推测功耗数据：primary、gdom1 和 gdom2。

```
primary# ldmpower -x
System Power Consumption in Watts
DOMAIN  15_SEC_AVG  30_SEC_AVG  60_SEC_AVG
primary 585/57.47%  701/68.96%  712/70.22%
gdom1   132/12.97%  94/9.31%    94/9.30%
gdom2   298/29.27%  218/21.47%  205/20.22%
```

- 以下命令显示 gdom2 和 gdom5 域的瞬时处理器功耗数据。它每隔十秒报告一次数据，共报告了五次。

```
primary# ldmpower -itl gdom2,gdom5 10 5
Processor Power Consumption in Watts
DOMAIN          TIMESTAMP               INSTANT
gdom2           2013.05.17 11:14:45     13
gdom5           2013.05.17 11:14:45     24

gdom2           2013.05.17 11:14:55     18
gdom5           2013.05.17 11:14:55     26
```

```
gdom2           2013.05.17 11:15:05     9
gdom5           2013.05.17 11:15:05     16

gdom2           2013.05.17 11:15:15     15
gdom5           2013.05.17 11:15:15     19

gdom2           2013.05.17 11:15:25     12
gdom5           2013.05.17 11:15:25     18
```

- 以下命令显示所有域过去 12 小时的平均功耗数据。以一小时为间隔显示自上次请求按小时进行计算以来的数据。

```
primary# ldmpower -eto 12
Per domain MINIMUM and MAXIMUM power consumption ever recorded:
primary         2013.05.17 08:53:06     3               Min Processors
primary         2013.05.17 08:40:44     273             Max Processors
gdom1           2013.05.17 09:56:35     2               Min Processors
gdom1           2013.05.17 08:53:06     134             Max Processors
gdom2           2013.05.17 10:31:55     2               Min Processors
gdom2           2013.05.17 08:56:35     139             Max Processors

primary         2013.05.17 08:53:06     99              Min Memory
primary         2013.05.17 08:40:44     182             Max Memory
gdom1           2013.05.17 09:56:35     13              Min Memory
gdom1           2013.05.17 08:53:06     20              Max Memory
gdom2           2013.05.17 10:31:55     65              Min Memory
gdom2           2013.05.17 08:56:35     66              Max Memory

Processor Power Consumption in Watts
12 hour's worth of data starting from 2013.05.16 23:17:02
DOMAIN          TIMESTAMP               1 HOUR AVG
primary         2013.05.17 09:37:35     112
gdom1           2013.05.17 09:37:35     15
gdom2           2013.05.17 09:37:35     26

primary         2013.05.17 10:37:35     96
gdom1           2013.05.17 10:37:35     12
gdom2           2013.05.17 10:37:35     21

primary         2013.05.17 11:37:35     85
gdom1           2013.05.17 11:37:35     11
gdom2           2013.05.17 11:37:35     23
...
```

♦ ♦ ♦ 第 20 章

使用 Oracle VM Server for SPARC 管理信息库软件

Oracle VM Server for SPARC 管理信息库 (Management Information Base, MIB) 使第三方系统管理应用程序能够使用简单网络管理协议 (Simple Network Management Protocol, SNMP) 对域执行远程监视并启动和停止逻辑域（简称域）。

在控制域上只能运行 Oracle VM Server for SPARC MIB 软件的一个实例。控制域应当至少运行 Solaris 10 11/06 OS 和 Oracle VM Server for SPARC 2.2 软件。

本章包括以下主题：

Oracle VM Server for SPARC 管理信息库概述

本节包括以下主题：

相关产品和功能

为了成功使用 Oracle VM Server for SPARC MIB，必须了解如何使用以下软件产品和软件：

- Oracle Solaris OS
- Oracle VM Server for SPARC 软件
- Simple Network Management Protocol, SNMP（简单网络管理协议）
- SNMP 管理信息库 (Management Information Base, MIB)
- Oracle Solaris SNMP 代理
- SNMP 版本 1 (SNMPv1)、SNMP 版本 2 (SNMPv2c) 和 SNMP 版本 3 (SNMPv3) 协议
- 管理信息结构 (Structure of Management Information, SMI) 版本 1 和版本 2
- 管理信息库 (Management Information Base, MIB) 结构
- 抽象语法表示法 1 (Abstract Syntax Notation, ASN.1)

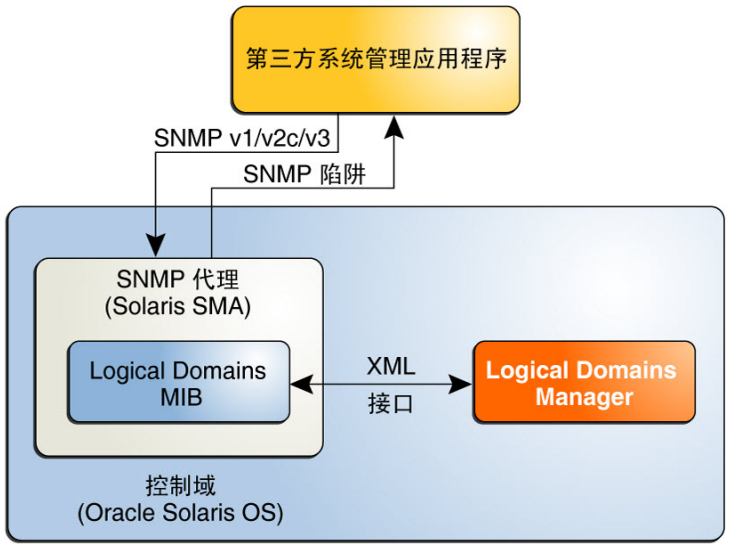
软件组件

Oracle VM Server for SPARC MIB 软件包 `SUNWldmib.v` 包含以下软件组件：

- `SUN-LDOM-MIB.mib` 是文本文件形式的 SNMP MIB。此文件定义 Oracle VM Server for SPARC MIB 中的对象。
- `ldomMIB.so` 是共享库形式的系统管理代理扩展模块。此模块使 Oracle Solaris SNMP 代理 能够响应 Oracle VM Server for SPARC MIB 中指定的信息请求并生成陷阱。

下图说明 Oracle VM Server for SPARC MIB、Oracle Solaris SNMP 代理、Logical Domains Manager 和第三方系统管理应用程序之间的交互。下图中显示的交互在“系统管理代理” [353]和“Logical Domains Manager 和 Oracle VM Server for SPARC MIB” [354]中有说明。

图 20-1 Oracle VM Server for SPARC MIB 与 Oracle Solaris SNMP 代理、Logical Domains Manager 和第三方系统管理应用程序之间的交互



系统管理代理

Oracle Solaris SNMP 代理可执行以下功能：

- 侦听第三方系统管理应用程序的请求以获取或设置由 Oracle VM Server for SPARC MIB 提供的数据。此代理侦听标准的 SNMP 端口 `161`。
- 通过使用 SNMP 通知的标准端口 `162` 来向所配置的系统管理应用程序发出陷阱。

Oracle VM Server for SPARC MIB 由控制域上的 Oracle Solaris OS 默认 Oracle Solaris SNMP 代理导出。

Oracle Solaris SNMP 代理 支持 SNMP 版本 v1、v2c 和 v3 的 get、set 和 trap 功能。大多数 Oracle VM Server for SPARC MIB 对象都是只读的，用于监视目的。但是，要启动或停止域，必须向 `ldomTable` 表的 `ldomAdminState` 属性中写入一个值。请参见表 20-1 “域表 (ldomTable)”。

Logical Domains Manager 和 Oracle VM Server for SPARC MIB

域是一个容器，由来宾操作系统的一组虚拟资源组成。Logical Domains Manager 提供用来创建、配置和管理域的命令行界面 (command-line interface, CLI)。Logical Domains Manager 和 Oracle VM Server for SPARC MIB 支持以下虚拟资源：

- CPU
- 内存
- 磁盘、网络和控制台 I/O
- 加密单元

解析基于 XML 的控制接口

Logical Domains Manager 将基于 XML 的控制接口导出到 Oracle VM Server for SPARC MIB。Oracle VM Server for SPARC MIB 解析 XML 接口并填充 MIB。Oracle VM Server for SPARC MIB 仅提供对控制域的支持。

提供 SNMP 陷阱

Oracle VM Server for SPARC MIB 定期轮询 Logical Domains Manager 看是否有更新或状态更改，然后向系统管理应用程序发出 SNMP 陷阱。

提供故障和恢复信息

如果 Oracle VM Server for SPARC MIB 无法再分配所需的资源，则 MIB 会通过 SNMP 代理向系统管理应用程序返回一个一般错误。SNMP 陷阱传送机制不对此错误进行确认。Oracle VM Server for SPARC MIB 中未实现任何特定的状态或检查点。具有 Oracle VM Server for SPARC MIB 的 Oracle Solaris SNMP 代理 由 `init` 进程和服务管理工具 (Service Management Facility, SMF) 启动和监视。如果 Oracle Solaris SNMP 代理 失败并退出，SMF 会自动重新启动该进程，随后该新进程会动态重新启动 Oracle VM Server for SPARC MIB 模块。

Oracle VM Server for SPARC MIB 对象树

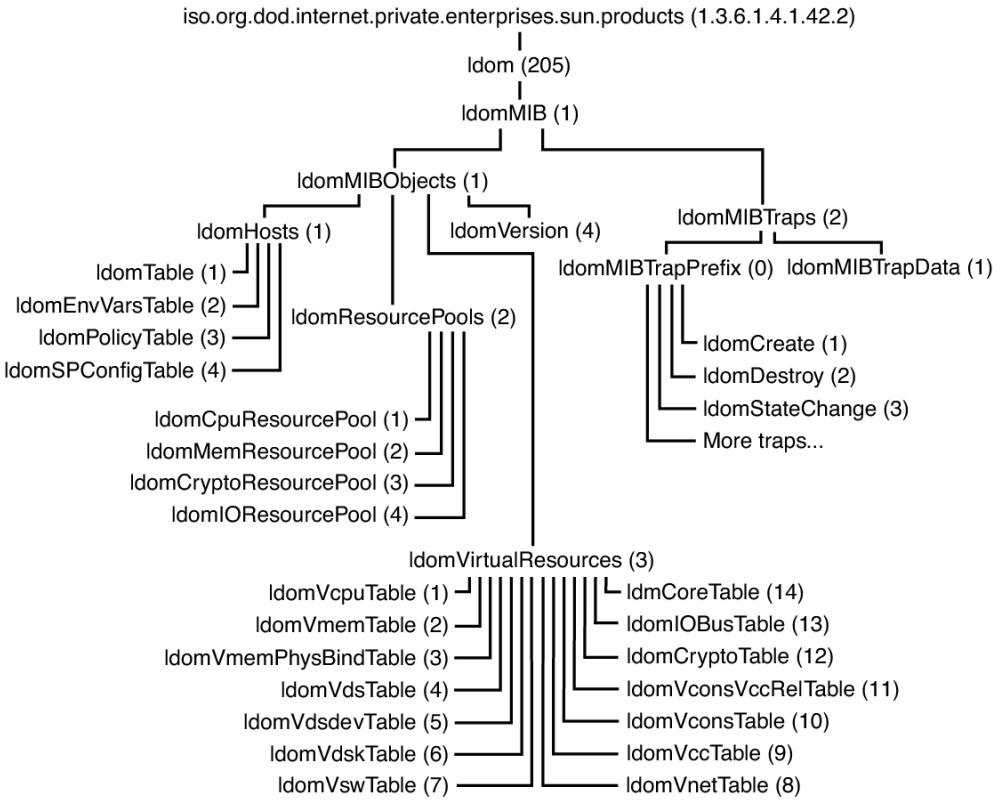
由 SNMP 管理的对象按树状层次结构进行组织。对象标识符 (object identifier, OID) 由一系列基于树中节点的整数组成，整数之间用点分隔。每个管理对象都有一个数值 OID

和一个相关的文本名称。Oracle VM Server for SPARC MIB 在对象树的这一部分注册为 `ldom (205)` 分支：

```
iso(1).org(3).dod(6).internet(1).private(4).enterprises(1).sun(42).products(2)
```

下图显示 Oracle VM Server for SPARC MIB 下面的主要子树：

图 20-2 Oracle VM Server for SPARC MIB 树



安装和配置 Oracle VM Server for SPARC MIB 软件

本节介绍如何在 Oracle Solaris 10 和 Oracle Solaris 11 系统上安装和配置 Oracle VM Server for SPARC MIB 软件。有关管理 SNMP 的信息，请参见 `snmpd.conf`(4) 或 `snmpd.conf`(5) 手册页。

安装和配置 Oracle VM Server for SPARC MIB 软件

下表列出了可用来安装和配置 Oracle VM Server for SPARC MIB 软件的任务。

任务	说明	有关指导
在 `primary` 域上安装 Oracle VM Server for SPARC MIB 软件包。	使用 `pkgadd` 命令可在 Oracle Solaris 10 系统上安装 `SUNWldmib.v` 软件包。或者，可使用 `pkg install` 命令在 Oracle Solaris 11 系统上安装 `system/ldoms/mib` 软件包。	如何安装 Oracle VM Server for SPARC MIB 软件包 [356]
将 Oracle VM Server for SPARC MIB 模块装入 Oracle Solaris SNMP 代理 以查询 Oracle VM Server for SPARC MIB。	修改 SNMP 配置文件以装入 `ldomMIB.so` 模块。	如何将 Oracle VM Server for SPARC MIB 模块装入 Oracle Solaris SNMP 代理 中 [357]
将 Oracle VM Server for SPARC MIB 软件包从 `primary` 域中删除。	使用 `pkgrm` 命令从 Oracle Solaris 10 系统中删除 `SUNWldmib` 软件包。或者，使用 `pkg remove` 命令从 Oracle Solaris 11 系统中删除 `system/ldoms/mib` 软件包。	如何删除 Oracle VM Server for SPARC MIB 软件包 [358]

▼ 如何安装 Oracle VM Server for SPARC MIB 软件包

此过程描述如何在 Oracle Solaris 10 和 Oracle Solaris 11 系统上安装 Oracle VM Server for SPARC MIB 软件包。Oracle VM Server for SPARC MIB 软件包包含在 Oracle VM Server for SPARC 3.2 软件中。

Oracle VM Server for SPARC MIB 软件包包括以下文件：

- `/opt/SUNWldmib/lib/mibs/SUN-LDOM-MIB.mib`
- `/opt/SUNWldmib/lib/ldomMIB.so`

开始之前 下载和安装 Oracle VM Server for SPARC 3.2 软件。请参见《Oracle VM Server for SPARC 3.2 安装指南 》中的第 2 章 “安装和启用软件”。

1. 确定系统运行的是 Oracle Solaris 10 OS 还是 Oracle Solaris 11 OS。

```
# uname -r
```

2. 在 **`primary`** 域上安装 Oracle VM Server for SPARC MIB 软件。

   - Oracle Solaris 10：安装 Oracle VM Server for SPARC MIB 软件包 **`SUNWldmib`**。

     ```
     # pkgadd -d . SUNWldmib.v
     ```

- Oracle Solaris 11：安装 Oracle VM Server for SPARC MIB 软件包 **system/ldoms/mib**。

  ```
  # pkg install -v -g IPS-package-directory/ldoms.repo mib
  ```

接下来的步骤 在安装此软件包之后，可以将系统配置为动态装入 Oracle VM Server for SPARC MIB 模块。请参见如何将 Oracle VM Server for SPARC MIB 模块装入 Oracle Solaris SNMP 代理 中 [357]。

▼ 如何将 Oracle VM Server for SPARC MIB 模块装入 Oracle Solaris SNMP 代理 中

必须将 Oracle VM Server for SPARC MIB 模块 `ldomMIB.so` 装入 Oracle Solaris SNMP 代理 中才能查询 Oracle VM Server for SPARC MIB。动态装入 Oracle VM Server for SPARC MIB 模块的目的在于让该模块包括在 SNMP 代理中，而不需要您重新编译和重新链接代理库。

此过程描述如何将系统配置为动态装入 Oracle VM Server for SPARC MIB 模块。《*Solaris System Management Agent Developer's Guide*》中提供了在不重新启动 Oracle Solaris SNMP 代理 的情况下动态装入模块的说明。有关 Oracle Solaris SNMP 代理 的更多信息，请参见《*Solaris System Management Administration Guide*》。

1. 确定系统运行的是 Oracle Solaris 10 OS 还是 Oracle Solaris 11 OS。

   ```
   # uname -r
   ```

2. 更新 SNMP 配置文件。

   - Oracle Solaris 10：
     将下面一行追加到 `/etc/sma/snmp/snmpd.conf` 配置文件中：

     ```
     dlmod ldomMIB /opt/SUNWldmib/lib/ldomMIB.so
     ```

   - Oracle Solaris 11：
     将下面这行内容附加到 `/etc/net-snmp/snmp/snmpd.conf` 配置文件末尾：

     ```
     dlmod ldomMIB /opt/SUNWldmib/lib/ldomMIB.so
     ```

3. 重新启动 SMF 服务。

   - Oracle Solaris 10：

     ```
     # svcadm restart svc:/application/management/sma:default
     ```

   - Oracle Solaris 11：

```
# svcadm restart svc:/application/management/net-snmp:default
```

▼ 如何删除 Oracle VM Server for SPARC MIB 软件包

此过程描述如何从 Oracle Solaris 10 或 Oracle Solaris 11 系统删除 Oracle VM Server for SPARC MIB 软件包并卸载 Oracle VM Server for SPARC MIB 模块。

1. 停止 SMF 服务。

   - Oracle Solaris 10：

     ```
     # svcadm disable svc:/application/management/sma:default
     ```

   - Oracle Solaris 11：

     ```
     # svcadm disable svc:/application/management/net-snmp:default
     ```

2. 将 Oracle VM Server for SPARC MIB 软件包从 **primary** 域中删除。

   - Oracle Solaris 10：

     ```
     # pkgrm SUNWldmib
     ```

   - Oracle Solaris 11：

     ```
     # pkg uninstall system/ldoms/mib
     ```

3. 更新 SNMP 配置文件。

   - Oracle Solaris 10：
     删除在安装期间添加到 `/etc/sma/snmp/snmpd.conf` 文件中的行。

     ```
     dlmod ldomMIB /opt/SUNWldmib/lib/ldomMIB.so
     ```

   - Oracle Solaris 11：
     删除在安装期间添加到 `/etc/net-snmp/snmp/snmpd.conf` 文件中的行。

     ```
     dlmod ldomMIB /opt/SUNWldmib/lib/ldomMIB.so
     ```

4. 重新启动 SMF 服务。

   - Oracle Solaris 10：

     ```
     # svcadm restart svc:/application/management/sma:default
     ```

   - Oracle Solaris 11：

```
# svcadm restart svc:/application/management/net-snmp:default
```

管理安全性

本节描述如何创建新的简单网络管理协议 (Simple Network Management Protocol, SNMP) 版本 3 (v3) 用户以提供对 Oracle Solaris SNMP 代理 的安全访问。对于 SNMP 版本 1 (v1) 和版本 2 (v2c)，访问控制机制是团体字符串 (community string)，它定义 SNMP 服务器和其客户机之间的关系。团体字符串 (community string) 控制客户机对服务器的访问，这与密码控制用户对系统的访问相似。请参见《*Solaris System Management Agent Administration Guide*》。

注 - 通过创建 `snmpv3` 用户，您可以将 SNMP 中的 Oracle Solaris SNMP 代理 与 Oracle VM Server for SPARC MIB 结合使用。此类用户不会与可能已使用 Oracle Solaris 中用于 Logical Domains Manager 的权限功能配置的用户进行交互或发生冲突。

▼ 如何创建初始 `snmpv3` 用户

此过程描述如何在 Oracle Solaris 10 或 Oracle Solaris 11 系统上创建初始 `snmpv3` 用户。

可以通过克隆此初始用户来创建其他用户。克隆使后续用户能够继承初始用户的验证和安全类型。您以后可以更改这些类型。

克隆初始用户时，可以为新用户设置私钥数据。您必须知道您为初始用户和后续用户配置的密码。一次只能克隆一个用户，并从初始用户开始克隆。请参见所用 Oracle Solaris OS 版本的《*Solaris System Management Agent Administration Guide*》中的“To Create Additional SNMPv3 Users with Security”。

1. 停止 Oracle Solaris SNMP 代理。

   - Oracle Solaris 10：

     ```
     # svcadm disable -t svc:/application/management/sma:default
     ```

   - Oracle Solaris 11：

     ```
     # svcadm disable svc:/application/management/net-snmp:default
     ```

2. 创建初始用户。
   此步骤创建具有您所选密码 *my-password* 的用户 *initial-user*，并向 `/etc/sma/snmp/snmpd.conf` 文件中添加一个相应条目。此条目赋予初始用户对代理的读写访问权限。

注 - 密码中必须至少包含八个字符。

- Oracle Solaris 10：

```
# /usr/sfw/bin/net-snmp-config --create-snmpv3-user -a my-password initial-user
```

- Oracle Solaris 11：

```
# /usr/bin/net-snmp-config --create-snmpv3-user -a my-password initial-user
```

3. 启动 Oracle Solaris SNMP 代理。

- Oracle Solaris 10：

```
# svcadm enable svc:/application/management/sma:default
```

- Oracle Solaris 11：

```
# svcadm enable svc:/application/management/net-snmp:default
```

4. 验证初始用户是否已创建。

```
# snmpget -v 3 -u initial-user -l authNoPriv -a MD5 -A my-password localhost sysUpTime.0
```

监视域

本节描述如何通过查询 Oracle VM Server for SPARC MIB 来监视逻辑域（简称域）。本节还提供各种类型的 MIB 输出的描述。

本节包括以下主题：

设置环境变量

在查询 Oracle VM Server for SPARC MIB 之前，必须为所用 shell 设置 `PATH`、`MIBDIRS` 和 `MIBS` 环境变量。Oracle Solaris 10 和 Oracle Solaris 11 的值不同。

- Oracle Solaris 10：
  - 对于 C shell 用户：

```
% setenv PATH /usr/sfw/bin:$PATH
% setenv MIBDIRS /opt/SUNWldmib/lib/mibs:/etc/sma/snmp/mibs
% setenv MIBS +SUN-LDOM-MIB
```

- 对于 Bourne 和 Korn shell 用户：

```
$ PATH=/usr/sfw/bin:$PATH; export PATH
$ MIBDIRS=/opt/SUNWldmib/lib/mibs:/etc/sma/snmp/mibs; export MIBDIRS
$ MIBS=+SUN-LDOM-MIB; export MIBS
```

- Oracle Solaris 11：
  - 对于 C shell 用户：

```
% setenv PATH /usr/bin:$PATH
% setenv MIBDIRS /opt/SUNWldmib/lib/mibs:/etc/net-snmp/snmp/mibs
% setenv MIBS +SUN-LDOM-MIB
```

  - 对于 Bourne 和 Korn shell 用户：

```
$ PATH=/usr/bin:$PATH; export PATH
$ MIBDIRS=/opt/SUNWldmib/lib/mibs:/etc/net-snmp/snmp/mibs; export MIBDIRS
$ MIBS=+SUN-LDOM-MIB; export MIBS
```

查询 Oracle VM Server for SPARC MIB

当系统上有大量域时，SNMP 代理可能还没来得及响应 SNMP 请求就已超时。要增大超时值，请使用 `-t` 选项指定更长的超时值。例如，下面的 `snmpwalk` 命令将超时值设置为 20 秒：

```
# snmpwalk -t 20 -v1 -c public localhost SUN-LDOM-MIB::ldomTable
```

还可以使用 `-t` 选项为 `snmpget` 和 `snmptable` 命令指定超时值。

- 检索单个 MIB 对象：

  `# snmpget -v` *version* `-c` *community-string host MIB-object*

- 检索一系列 MIB 对象：

  使用 `snmpwalk` 或 `snmptable` 命令。

  `# snmpwalk -v` *version* `-c` *community-string host MIB-object*
  `# snmptable -v` *version* `-c` *community-string host MIB-object*

注 - 使用带 `-v2c` 或 `-v3` 选项的 `snmptable` 命令查询 Oracle VM Server for SPARC MIB 2.1 软件时将收到空的 SNMP 表。使用 `-v1` 选项时 `snmptable` 命令可以按预期工作。

要解决此问题，请使用 `-CB` 选项以仅使用 `GETNEXT`（而不是 `GETBULK`）请求来检索数据。请参见“查询 Oracle VM Server for SPARC MIB” [361]。

例 20-1 检索单个 Oracle VM Server for SPARC MIB 对象 (snmpget)

下面的 snmpget 命令查询 ldomVersionMajor 对象的值。此命令为 localhost 主机指定 snmpv1 (-v1) 和一个团体字符串 (-c public)。

```
# snmpget -v1 -c public localhost SUN-LDOM-MIB::ldomVersionMajor.0
SUN-LDOM-MIB::ldomVersionMajor.0 = INTEGER: 1
```

例 20-2 从 ldomTable 检索对象值 (snmpwalk)

以下示例说明如何使用 snmpwalk 命令从 ldomTable 检索对象值。

- 以下 snmpwalk -v1 命令将返回 ldomTable 表中所有对象的值。

```
# snmpwalk -v1 -c public localhost SUN-LDOM-MIB::ldomTable
SUN-LDOM-MIB::ldomName.1 = STRING: primary
SUN-LDOM-MIB::ldomName.2 = STRING: LdomMibTest_1
SUN-LDOM-MIB::ldomAdminState.1 = INTEGER: 0
SUN-LDOM-MIB::ldomAdminState.2 = INTEGER: 0
SUN-LDOM-MIB::ldomOperState.1 = INTEGER: active(1)
SUN-LDOM-MIB::ldomOperState.2 = INTEGER: bound(6)
SUN-LDOM-MIB::ldomNumVCpu.1 = INTEGER: 8
SUN-LDOM-MIB::ldomNumVCpu.2 = INTEGER: 4
SUN-LDOM-MIB::ldomMemSize.1 = INTEGER: 3360
SUN-LDOM-MIB::ldomMemSize.2 = INTEGER: 256
SUN-LDOM-MIB::ldomMemUnit.1 = INTEGER: megabytes(2)
SUN-LDOM-MIB::ldomMemUnit.2 = INTEGER: megabytes(2)
SUN-LDOM-MIB::ldomNumCrypto.1 = INTEGER: 1
SUN-LDOM-MIB::ldomNumCrypto.2 = INTEGER: 0
SUN-LDOM-MIB::ldomNumIOBus.1 = INTEGER: 2
SUN-LDOM-MIB::ldomNumIOBus.2 = INTEGER: 0
SUN-LDOM-MIB::ldomUUID.1 = STRING: 5f8817d4-5d2e-6f7d-c4af-91b5b34b5723
SUN-LDOM-MIB::ldomUUID.2 = STRING: 11284146-87ca-4877-8d80-cd0f60d5ec26
SUN-LDOM-MIB::ldomMacAddress.1 = STRING: 00:14:4f:46:47:d6
SUN-LDOM-MIB::ldomMacAddress.2 = STRING: 00:14:4f:f8:d5:6c
SUN-LDOM-MIB::ldomHostID.1 = STRING: 0x844647d6
SUN-LDOM-MIB::ldomHostID.2 = STRING: 0x84f8d56c
SUN-LDOM-MIB::ldomFailurePolicy.1 = STRING: ignore
SUN-LDOM-MIB::ldomFailurePolicy.2 = STRING: ignore
SUN-LDOM-MIB::ldomMaster.1 = STRING:
SUN-LDOM-MIB::ldomMaster.2 = STRING:
SUN-LDOM-MIB::ldomExtMapinSpace.1 = STRING: off
SUN-LDOM-MIB::ldomExtMapinSpace.2 = STRING: off
SUN-LDOM-MIB::ldomWholeCore.1 = INTEGER: 0
SUN-LDOM-MIB::ldomWholeCore.2 = INTEGER: 0
SUN-LDOM-MIB::ldomCpuArch.1 = STRING: native
```

```
SUN-LDOM-MIB::ldomCpuArch.2 = STRING: native
SUN-LDOM-MIB::ldomShutdownGroup.1 = INTEGER: 0
SUN-LDOM-MIB::ldomShutdownGroup.2 = INTEGER: 15
SUN-LDOM-MIB::ldomPerfCounters.1 = STRING: htstrand
SUN-LDOM-MIB::ldomPerfCounters.2 = STRING: global,htstrand
```

- 以下 `snmpwalk` 命令使用 `snmpv2c` 和 `snmpv3` 检索 `ldomTable` 的内容：

```
# snmpwalk -v2c -c public localhost SUN-LDOM-MIB::ldomTable
# snmpwalk -v 3 -u test -l authNoPriv -a MD5 -A testpassword localhost \
SUN-LDOMMIB::ldomTable
```

例 20-3 从 `ldomTable` 以表格形式检索对象值 (`snmptable`)

以下示例说明如何使用 `snmptable` 命令以表格形式从 `ldomTable` 中检索对象值。

- 以下 `snmptable -v1` 命令将以表格形式显示 `ldomTable` 的内容。

```
# snmptable -v1 -c public localhost SUN-LDOM-MIB::ldomTable
```

- 以下 `snmptable` 命令使用 `snmpv2c` 以表格形式显示 `ldomTable` 的内容。

  请注意，对于 `v2c` 或 `v3` `snmptable` 命令，使用 `-CB` 选项仅指定 `GETNEXT`（而非 `GETBULK`）请求来检索数据。

```
# snmptable -v2c -CB -c public localhost SUN-LDOM-MIB::ldomTable
```

检索 Oracle VM Server for SPARC MIB 信息

本节描述可以表格或标量对象形式从 Oracle VM Server for SPARC MIB 中检索的信息。

域表 (`ldomTable`)

`ldomTable` 用于代表系统中的每个域。信息包括虚拟 CPU、内存、加密单元和 I/O 总线的资源约束。表中还包括其他域信息，如通用唯一标识符 (universally unique identifier, UUID)、MAC 地址、主机 ID、故障策略和主域。

表 20-1 域表 (`ldomTable`)

名称	数据类型	访问	说明
`ldomIndex`	整数	不可访问	用作此表的索引的整数
`ldomName`	显示字符串	只读	域的名称
`ldomAdminState`	整数	读/写	启动或停止域以进行有效管理：

名称	数据类型	访问	说明
			■ 值为 1 将启动域 ■ 值为 2 将停止域
ldomOperState	整数	只读	域的当前状态，可以为下列值之一： ■ 1 表示“活动”状态 ■ 2 表示“正在停止”状态 ■ 3 表示“非活动”状态 ■ 4 表示“正在绑定”状态 ■ 5 表示“正在解除绑定”状态 ■ 6 表示“绑定”状态 ■ 7 表示“正在启动”状态
ldomNumVCPU	整数	只读	使用的虚拟 CPU 的数量。如果域处于非活动状态，则该值为请求的虚拟 CPU 数量。
ldomMemSize	整数	只读	使用的虚拟内存量。如果域处于非活动状态，则该值为请求的内存大小。
ldomMemUnit	整数	只读	下列内存单位之一： ■ 1 表示 KB ■ 2 表示 MB ■ 3 表示 GB ■ 4 表示字节 如果未指定，则单位值为字节。
ldomNumCrypto	整数	只读	使用的加密单元的数量。如果域处于非活动状态，则该值为请求的加密单元的数量。
ldomNumIOBus	整数	只读	使用的物理 I/O 设备的数量
ldomUUID	显示字符串	只读	域的 UUID
ldomMacAddress	显示字符串	只读	域的 MAC 地址
ldomHostID	显示字符串	只读	域的主机 ID
ldomFailurePolicy	显示字符串	只读	主域的故障策略，可以为 `ignore`、`panic`、`reset` 或 `stop` 之一
ldomMaster	显示字符串	只读	从属域的主域（最多四个）名称
ldomExtMapinSpace	显示字符串	只读	域的扩展 mapin 空间。扩展 mapin 空间是指增加的 LDC 共享内存空间。需要此内存空间来支持使用直接映射共享内存的大量虚拟 I/O 设备。虚拟网络设备也使用此空间来提高性能和可伸缩性。默认值为 `off`。
ldomThreading	显示字符串	只读	为域指定较高的每周期指令数 (instructions per cycle, IPC) 线程控制。动态线程控制每个核心激活的硬件线程数量。有效值包括： ■ `max-throughput`，这意味着每个核心的所有导线束都处于活动状态（默认值） ■ `max-ipc`，这意味着每个核心有一个导线束处于活动状态
ldomWholeCore	整数	只读	约束域仅使用整体核心。如果未启用整体核心约束，则该值为 `0`。否则，该值将显示最大核心数。

名称	数据类型	访问	说明
ldomCpuArch	显示字符串	只读	域的 CPU 体系结构。CPU 体系结构指定是否可将域迁移到其他 sun4v CPU 体系结构。有效值包括： ■ native，这意味着仅允许将域迁移到具有相同 sun4v CPU 体系结构的平台（默认值） ■ generic，这意味着允许将域迁移到所有可兼容的 sun4v CPU 体系结构
ldomShutdownGroup	整数	只读	来宾域的关闭组号。在 SPARC64-X 系统上，SP 启动的按顺序关闭将按域的关闭组号降序（从 15 到 0）来关闭域。默认值为 15。
ldomPerfCounters	字符串	只读	来宾域的性能寄存器访问信息。值可以是 global（一次仅在一个域上），并且可以是以下项之一：htstrand 或 strand。默认值为 htstrand。

环境变量表 (ldomEnvVarsTable)

ldomEnvVarsTable 描述由所有的域使用的 OpenBoot PROM 环境变量。

表 20-2 环境变量表 (ldomEnvVarsTable)

名称	数据类型	访问	说明
ldomEnvVarsLdomIndex	整数	只读	用作 ldomTable 的索引的整数，表示包含 OpenBoot PROM 环境变量的域
ldomEnvVarsIndex	整数	只读	用于对该表中的 OpenBoot PROM 环境变量编制索引的整数
ldomEnvVarsName	显示字符串	只读	OpenBoot PROM 变量的名称
ldomEnvVarsValue	显示字符串	只读	OpenBoot PROM 变量的值

域策略表 (ldomPolicyTable)

ldomPolicyTable 描述应用于所有域的动态资源管理 (dynamic resource management, DRM) 策略。

表 20-3 域策略表 (ldomPolicyTable)

名称	数据类型	访问	说明
ldomPolicyLdomIndex	整数	只读	用作 ldomTable 的索引的整数，表示包含 DRM 策略的域

名称	数据类型	访问	说明
ldomPolicyIndex	整数	不可访问	用于对该表中的 DRM 策略编制索引的整数
ldomPolicyName	显示字符串	只读	策略名称
ldomPolicyStatus	显示字符串	只读	策略状态
ldomPolicyPriority	整数	只读	用于确定在策略重叠时选择哪个 DRM 策略的优先级
ldomPolicyVcpuMin	整数	只读	域的虚拟 CPU 的最小数量
ldomPolicyVcpuMax	整数	只读	域的虚拟 CPU 的最大数量。值为 unlimited 表示将使用最大整数值 2147483647。
ldomPolicyUtilLower	整数	只读	触发策略分析的最低利用率级别
ldomPolicyUtilUpper	整数	只读	触发策略分析的最高利用率级别
ldomPolicyTodBegin	显示字符串	只读	策略的有效启动时间，格式为 *hh*:*mm*:[*ss*]
ldomPolicyTodEnd	显示字符串	只读	策略的有效停止时间，格式为 *hh*:*mm*:[*ss*]
ldomPolicySampleRate	整数	只读	资源回收时间，以秒为单位
ldomPolicyElasticMargin	整数	只读	缓冲量，介于 util-lower 属性 (ldomPolicyUtilLower) 和空闲虚拟 CPU 数量之间，避免在虚拟 CPU 数量较少时发生振荡
ldomPolicyAttack	整数	只读	在任一资源控制周期内添加的最大资源量。值为 unlimited 表示将使用最大整数值 2147483647。
ldomPolicyDecay	整数	只读	在任一资源控制周期内删除的最大资源量

服务处理器配置表 (ldomSPConfigTable)

ldomSPConfigTable 描述所有域的服务处理器 (service processor, SP) 配置。

表 20-4 服务处理器配置表 (ldomSPConfigTable)

名称	数据类型	访问	说明
ldomSPConfigIndex	整数	不可访问	用于对该表中的 SP 配置编制索引的整数
ldomSPConfigName	显示字符串	只读	SP 配置名称
ldomSPConfigStatus	显示字符串	只读	SP 配置状态

域的资源池和标量变量

可以为域分配以下资源：

- 虚拟 CPU (vcpu)
- 内存 (mem)
- 加密单元 (mau)

- 虚拟交换机 (vsw)
- 虚拟网络 (vnet)
- 虚拟磁盘服务器 (vds)
- 虚拟磁盘服务器设备 (vdsdev)
- 虚拟磁盘 (vdisk)
- 虚拟控制台集中器 (vcc)
- 虚拟控制台 (vcons)
- 物理 I/O 设备 (io)

以下标量 MIB 变量用来代表资源池及其属性。

表 20-5 CPU 资源池的标量变量

名称	数据类型	访问	说明
ldomCpuRpCapacity	整数	只读	ldomCpuRpCapacityUnit 中的资源池允许的最大预留空间
ldomCpuRpReserved	整数	只读	当前从资源池中预留的累计 CPU 处理器时钟速度，以 MHz 为单位
ldomCpuRpCapacityUnit 和 ldomCpuRpReservedUnit	整数	只读	下列 CPU 分配单位之一： ■ 1 表示 MHz ■ 2 表示 GHz 默认值为 MHz。

表 20-6 内存资源池的标量变量

名称	数据类型	访问	说明
ldomMemRpCapacity	整数	只读	MemRpCapacityUnit 中的资源池允许的最大预留空间
ldomMemRpReserved	整数	只读	MemRpReservedUnit 中当前从资源池中预留的内存量
ldomMemRpCapacityUnit 和 ldomMemRpReservedUnit	整数	只读	下列内存分配单位之一： ■ 1 表示 KB ■ 2 表示 MB ■ 3 表示 GB ■ 4 表示字节 如果未指定，则单位值为字节。

表 20-7 加密资源池的标量变量

名称	数据类型	访问	说明
ldomCryptoRpCapacity	整数	只读	资源池允许的最大预留空间
ldomCryptoRpReserved	整数	只读	当前从资源池中预留的加密单元的数量

表 20-8 I/O 总线资源池的标量变量

名称	数据类型	访问	说明
ldomIOBusRpCapacity	整数	只读	池允许的最大预留空间
ldomIOBusRpReserved	整数	只读	当前从资源池中预留的 I/O 总线的数量

虚拟 CPU 表 (ldomVcpuTable)

ldomVcpuTable 描述由所有的域使用的虚拟 CPU。

表 20-9 虚拟 CPU 表 (ldomVcpuTable)

名称	数据类型	访问	说明
ldomVcpuLdomIndex	整数	只读	用作 ldomTable 的索引的整数，表示包含虚拟 CPU 的域
ldomVcpuIndex	整数	不可访问	用于对该表中的虚拟 CPU 编制索引的整数
ldomVcpuDeviceID	显示字符串	只读	虚拟 CPU 的标识符 (Identifier of the virtual CPU, VID)
ldomVcpuOperationalStatus	整数	只读	下列 CPU 状态之一： 1=未知 2=其他 3=正常 4=降级 5=有压力 6=故障预警 7=错误 8=不可恢复的错误 9=正在启动 10=正在停止 11=已停止 12=服务中 13=无联系 14=失去通信 15=已中止 16=暂停 17=错误的支持实体

名称	数据类型	访问	说明
			18=已完成 19=电源模式 默认值为 1（未知），因为 Logical Domains Manager 不提供 CPU 状态。
ldomVcpuPhysBind	显示字符串	只读	物理绑定 (Physical binding, PID)。包含分配给此虚拟 CPU 的硬件线程（导线束）的标识符。此标识符唯一地标识核心和芯片。
ldomVcpuPhysBindUsage	整数	只读	指示该虚拟 CPU 所使用的线程的总容量（以 MHz 为单位）。例如，假定一个线程最多可以在 1 GHz 的速度下运行。如果只为该虚拟 CPU 分配此容量的一半（即线程的 50%），则属性值为 500。
ldomVcpuCoreID	显示字符串	只读	核心的标识符（核心 ID）。
ldomVcpuUtilPercent	显示字符串	只读	指示虚拟 CPU 的利用率百分比。

虚拟内存表

域的内存空间称为实际内存，即虚拟内存。虚拟机管理程序检测到的主机平台内存空间称为物理内存。虚拟机管理程序映射物理内存块以形成由域使用的实际内存块。

以下示例说明所请求的内存大小可以在两个内存块之间进行分割，而不是分配给单个大内存块。假定域请求 521 MB 的实际内存。可以使用 {物理地址，实际地址，大小} 格式，为主机系统上的内存分配两个 256 MB 的块作为物理内存。

```
{0x1000000, 0x1000000, 256}, {0x2000000, 0x2000000,256}
```

在一个域中，最多可以将 64 个物理内存段分配给来宾域。因此，系统会使用一个辅助表（而不是显示字符串）来存放每个内存段。显示字符串有最多包含 255 个字符这一限制。

虚拟内存表 (ldomVmemTable)

ldomVmemTable 描述由域使用的虚拟内存的属性。

表 20-10 虚拟内存表 (ldomVmemTable)

名称	数据类型	访问	说明
ldomVmemLdomIndex	整数	只读	用作 ldomTable 的索引的整数，表示包含虚拟内存的域
ldomVmemIndex	整数	不可访问	用于对该表中的虚拟内存编制索引的整数
ldomVmemNumberofBlocks	整数	只读	虚拟内存块的数量

虚拟内存物理绑定表 (ldomVmemPhysBindTable)

ldomVmemPhysBindTable 是一个辅助表，其中包含所有域的物理内存段。

表 20-11 虚拟内存物理绑定表 (ldomVmemPhysBindTable)

名称	数据类型	访问	说明
ldomVmemPhysBindLdomIndex	整数	只读	用作 ldomTable 的索引的整数，表示包含物理内存段的域
ldomVmemPhysBind	显示字符串	只读	采用如下格式映射到此虚拟内存块的物理内存列表：{物理地址，实际地址，大小}

虚拟磁盘表

虚拟磁盘服务 (vds) 和它所映射到的物理设备 (vdsdev) 为 Oracle VM Server for SPARC 技术提供虚拟磁盘功能。虚拟磁盘服务导出许多本地卷（物理磁盘或文件系统）。如果指定了虚拟磁盘服务，则会包括以下内容：

- 后备设备 (vdsdev) 的完整 /dev 路径
- 要添加到服务中的设备的唯一名称（卷名）

可以向单个磁盘服务绑定一个或多个磁盘、磁盘分片和文件系统。每个磁盘都有一个唯一的名称和卷名。在将磁盘绑定到服务时使用卷名。Logical Domains Manager 从虚拟磁盘服务及其逻辑卷创建虚拟磁盘客户机 (vdisk)。

虚拟磁盘服务表 (ldomVdsTable)

ldomVdsTable 描述所有域的虚拟磁盘服务。

表 20-12 虚拟磁盘服务表 (ldomVdsTable)

名称	数据类型	访问	说明
ldomVdsLdomIndex	整数	只读	用作 ldomTable 的索引的整数，表示包含虚拟磁盘服务的域
ldomVdsIndex	整数	不可访问	用于对该表中的虚拟磁盘服务编制索引的整数
ldomVdsServiceName	显示字符串	只读	虚拟磁盘服务的服务名称。属性值为 ldm add-vds 命令所指定的 *service-name*。
ldomVdsNumofAvailVolume	整数	只读	由该虚拟磁盘服务导出的逻辑卷的数量
ldomVdsNumofUsedVolume	整数	只读	用于（绑定到）该虚拟磁盘服务的逻辑卷的数量

虚拟磁盘服务设备表 (ldomVdsdevTable)

ldomVdsdevTable 描述由所有的虚拟磁盘服务使用的虚拟磁盘服务设备。

表 20-13 虚拟磁盘服务设备表 (ldomVdsdevTable)

名称	数据类型	访问	说明
ldomVdsdevVdsIndex	整数	只读	用于对 ldomVdsTable 编制索引的整数，表示包含虚拟磁盘设备的虚拟磁盘服务
ldomVdsdevIndex	整数	不可访问	用于对该表中的虚拟磁盘服务设备编制索引的整数
ldomVdsdevVolumeName	显示字符串	只读	虚拟磁盘服务设备的卷名。此属性为要添加到虚拟磁盘服务的设备指定唯一的名称。虚拟磁盘服务出于添加此设备的目的将此名称导出到客户机。属性值为 ldm add-vdsdev 命令所指定的 *volume-name*。
ldomVdsdevDevPath	显示字符串	只读	物理磁盘设备的路径名。属性值为 *ldm add-vdsdev* 命令所指定的 backend。
ldomVdsdevOptions	显示字符串	只读	磁盘设备的一个或多个选项：ro、slice 或 excl
ldomVdsdevMPGroup	显示字符串	只读	磁盘设备的多路径组名

虚拟磁盘表 (ldomVdiskTable)

ldomVdiskTable 描述所有域的虚拟磁盘。

表 20-14 虚拟磁盘表 (ldomVdiskTable)

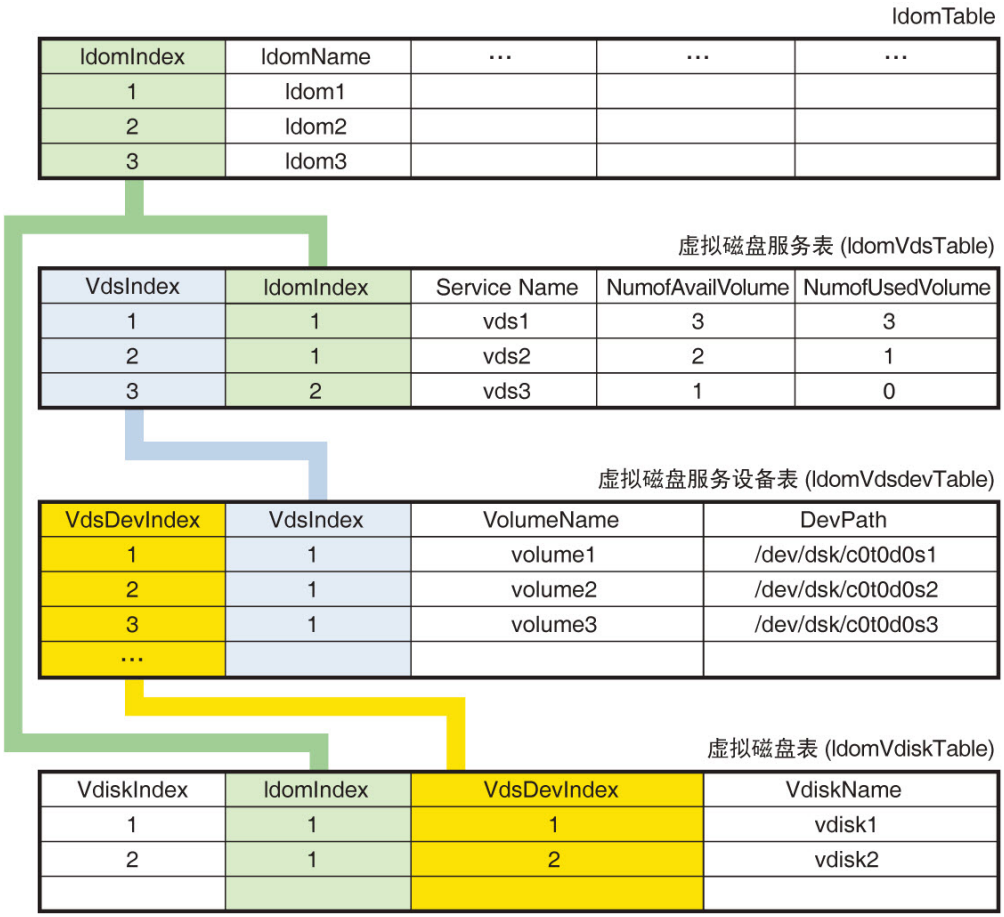
名称	数据类型	访问	说明
ldomVdiskLdomIndex	整数	只读	用作 ldomTable 的索引的整数，表示包含虚拟磁盘设备的域
ldomVdiskVdsDevIndex	整数	只读	用于对 ldomVdsdevTable 编制索引的整数，表示虚拟磁盘服务设备
ldomVdiskIndex	整数	不可访问	用于对该表中的虚拟磁盘编制索引的整数
ldomVdiskName	显示字符串	只读	虚拟磁盘的名称。属性值为 *ldm add-vdisk* 命令所指定的 disk-name。
ldomVdiskTimeout	整数	只读	在虚拟磁盘客户机和虚拟磁盘服务器之间建立连接时的超时值，以秒为单位
ldomVdiskID	显示字符串	只读	虚拟磁盘的标识符

下图说明如何使用索引来定义虚拟磁盘表和域表之间的关系。索引的使用方式如下：

- ldomVdsTable 和 ldomVdiskTable 中的 ldomIndex 指向 ldomTable。

- `ldomVdsdevTable` 中的 `VdsIndex` 指向 `ldomVdsTable`。
- `ldomVdiskTable` 中的 `VdsDevIndex` 指向 `ldomVdsdevTable`。

图 20-3 虚拟磁盘表和域表之间的关系



虚拟网络表

Oracle VM Server for SPARC 虚拟网络支持使来宾域能够相互通信并通过物理以太网设备与外部主机通信。虚拟网络包含以下主要组件：

- 虚拟交换机 (`vsw`)
- 虚拟网络设备 (`vnet`)

在服务域上创建虚拟交换机之后，可以将物理网络设备绑定到虚拟交换机。在此之后，可以为使用虚拟交换机服务进行通信的域创建一个虚拟网络设备。虚拟交换机服务通过连接到同一个虚拟交换机来与其他域通信。如果物理设备绑定到虚拟交换机，则虚拟交换机服务与外部主机通信。

虚拟交换机服务表 (ldomVswTable)

ldomVswTable 描述所有域的虚拟交换机服务。

表 20-15 虚拟交换机服务表 (ldomVswTable)

名称	数据类型	访问	说明
ldomVswLdomIndex	整数	只读	用作 ldomTable 的索引的整数，表示包含虚拟交换机服务的域
ldomVswIndex	整数	不可访问	用于对该表中的虚拟交换机设备编制索引的整数
ldomVswServiceName	显示字符串	只读	虚拟交换机服务的名称
ldomVswMacAddress	显示字符串	只读	由虚拟交换机使用的 MAC 地址
ldomVswPhysDevPath	显示字符串	只读	虚拟网络交换机的物理设备路径。如果没有物理设备绑定到虚拟交换机，则此属性的值为 null。
ldomVswMode	显示字符串	只读	对于正在运行的群集节点，值为 mode=sc
ldomVswDefaultVlanID	显示字符串	只读	虚拟交换机的默认 VLAN ID
ldomVswPortVlanID	显示字符串	只读	虚拟交换机的端口 VLAN ID
ldomVswVlanID	显示字符串	只读	虚拟交换机的 VLAN ID
ldomVswLinkprop	显示字符串	只读	值为 linkprop=phys-state 表示将基于物理网络设备报告链路状态
ldomVswMtu	整数	只读	虚拟交换机设备的最大传输单元 (Maximum transmission unit, MTU)
ldomVswID	显示字符串	只读	虚拟交换机设备的标识符
ldomVswInterVnetLink	显示字符串	只读	Inter-VNet 通信的 LDC 通道分配状态。值为 on 或 off。

虚拟网络设备表 (ldomVnetTable)

ldomVnetTable 描述所有域的虚拟网络设备。

表 20-16 虚拟网络设备表 (ldomVnetTable)

名称	数据类型	访问	说明
ldomVnetLdomIndex	整数	只读	用作 ldomTable 的索引的整数，表示包含虚拟网络设备的域

名称	数据类型	访问	说明
ldomVnetVswIndex	整数	只读	用于对虚拟交换机服务表编制索引的整数
ldomVnetIndex	整数	不可访问	用于对该表中的虚拟网络设备编制索引的整数
ldomVnetDevName	显示字符串	只读	虚拟网络设备的名称。属性值为 ldm add-vnet 命令所指定的 net-dev 属性。
ldomVnetDevMacAddress	显示字符串	只读	此网络设备的 MAC 地址。属性值为 ldm add-vnet 命令所指定的 mac-addr 属性。
ldomVnetMode	显示字符串	只读	值为 mode=hybrid 表示将使用虚拟网络设备上的 NIU 混合 I/O
ldomVnetPortVlanID	显示字符串	只读	虚拟网络设备的端口 VLAN ID
ldomVnetVlanID	显示字符串	只读	虚拟网络设备的 VLAN ID
ldomVnetLinkprop	显示字符串	只读	值为 linkprop=phys-state 表示将基于物理网络设备报告链路状态
ldomVnetMtu	整数	只读	虚拟网络设备的 MTU
ldomVnetID	显示字符串	只读	虚拟网络设备的标识符

虚拟控制台表

Oracle VM Server for SPARC 服务域提供虚拟网络终端服务 (vNTS)。vNTS 提供一个名为虚拟控制台集中器 (virtual console concentrator, vcc) 的虚拟控制台服务，以及端口号范围。每个虚拟控制台集中器都有多个控制台组 (vcons)，而且每个组都分配有一个端口号。每个组都可以包含多个域。

虚拟控制台集中器表 (ldomVccTable)

ldomVccTable 描述所有域的虚拟控制台集中器。

表 20-17 虚拟控制台集中器表 (ldomVccTable)

名称	数据类型	访问	说明
ldomVccLdomIndex	整数	只读	用作 ldomTable 的索引的整数，表示包含虚拟控制台服务的域
ldomVccIndex	整数	不可访问	用于对该表中的虚拟控制台集中器编制索引的整数
ldomVccName	显示字符串	只读	虚拟控制台集中器的名称。属性值为 *ldm add-vcc* 命令所指定的 vcc-name。
ldomVccPortRangeLow	整数	只读	虚拟控制台集中器使用的 TCP 端口范围的下限。属性值为 ldm add-vcc 命令所指定的 port-range 的 *x* 部分。

名称	数据类型	访问	说明
ldomVccPortRangeHigh	整数	只读	虚拟控制台集中器使用的 TCP 端口范围的上限。属性值为 *ldm add-vcc* 命令所指定的 port-range 的 y 部分。

虚拟控制台组表 (ldomVconsTable)

ldomVconsTable 描述所有虚拟控制台服务的虚拟控制台组。此表还会显示控制台日志记录在每个域上已启用还是已禁用。

表 20-18 虚拟控制台组表 (ldomVconsTable)

名称	数据类型	访问	说明
ldomVconsIndex	整数	不可访问	用于对该表中的虚拟组编制索引的整数
ldomVconsGroupName	显示字符串	只读	虚拟控制台所连接的组名。属性值为 ldm set-vcons 命令所指定的 group。
ldomVconsLog	显示字符串	只读	控制台日志记录状态。属性值为 ldm set-vcons 命令所指定的字符串 on 或 off。 如果某个组包含多个域，则该属性将显示 ldm set-vcons 命令最近修改过的域的控制台日志记录状态。
ldomVconsPortNumber	整数	只读	分配给该组的端口号。属性值为 ldm set-vcons 命令所指定的 port。

虚拟控制台关系表 (ldomVconsVccRelTable)

ldomVconsVccRelTable 包含用来说明域表、虚拟控制台集中器表和控制台组表之间关系的索引值。

表 20-19 虚拟控制台关系表 (ldomVconsVccRelTable)

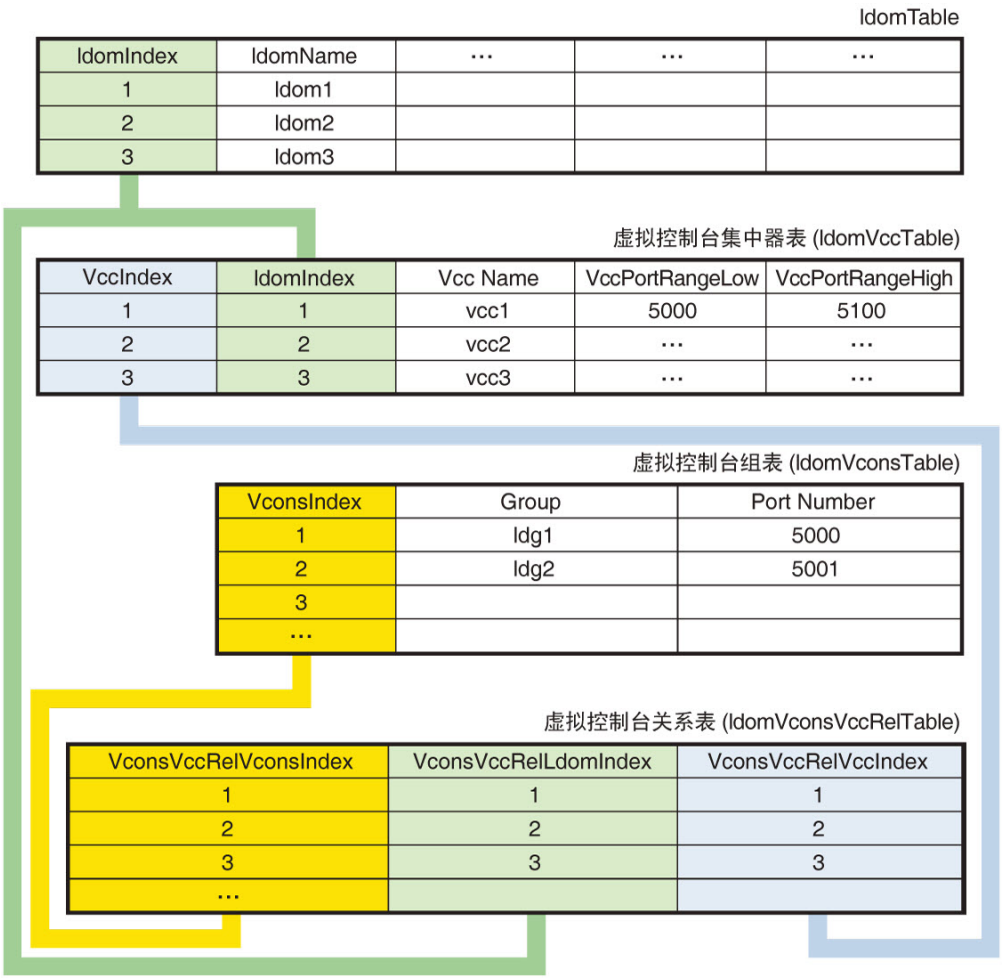
名称	数据类型	访问	说明
ldomVconsVccRelVconsIndex	整数	只读	ldomVconsTable 中 ldomVconsIndex 的值
ldomVconsVccRelLdomIndex	整数	只读	ldomTable 中 ldomIndex 的值
ldomVconsVccRelVccIndex	整数	只读	ldomVccTable 中 ldomVccIndex 的值

下图说明如何使用索引来定义虚拟控制台表和域表之间的关系。索引的使用方式如下：

- ldomVccTable 和 ldomVconsVccRelTable 中的 ldomIndex 指向 ldomTable。

- `ldomVconsVccRelTable` 中的 `VccIndex` 指向 `ldomVccTable`。
- `ldomVconsVccRelTable` 中的 `VconsIndex` 指向 `ldomVconsTable`。

图 20-4 虚拟控制台表和域表之间的关系



加密单元表 (`ldomCryptoTable`)

`ldomCryptoTable` 描述由所有的域使用的加密单元。加密单元有时称为模运算单元 (modular arithmetic unit, MAU)。

表 20-20 加密单元表 (ldomCryptoTable)

名称	数据类型	访问	说明
ldomCryptoLdomIndex	整数	只读	用作 ldomTable 的索引的整数，表示包含加密单元的域
ldomCryptoIndex	整数	不可访问	用于对该表中的加密单元编制索引的整数
ldomCryptoCpuSet	显示字符串	只读	映射到 MAU-unit cpuset 的 CPU 列表。例如，{0, 1, 2, 3}。

I/O 总线表 (ldomIOBusTable)

ldomIOBusTable 描述所有的域使用的物理 I/O 设备和 PCI 总线。

表 20-21 I/O 总线表 (ldomIOBusTable)

名称	数据类型	访问	说明
ldomIOBusLdomIndex	整数	只读	用作 ldomTable 的索引的整数，表示包含 I/O 总线的域
ldomIOBusIndex	整数	不可访问	用于对该表中的 I/O 总线编制索引的整数
ldomIOBusName	显示字符串	只读	物理 I/O 设备名称
ldomIOBusPath	显示字符串	只读	物理 I/O 设备路径
ldomIOBusOptions	显示字符串	只读	物理 I/O 设备选件

核心表 (ldomCoreTable)

ldomCoreTable 描述所有域的核心信息，如 core-id 和 cpuset。

表 20-22 核心表 (ldomCoreTable)

名称	数据类型	访问	说明
ldomCoreLdomIndex	整数	只读	用作 ldomTable 的索引的整数，表示包含核心的域
ldomCoreIndex	整数	不可访问	用于对该表中的核心编制索引的整数
ldomCoreID	显示字符串	只读	核心的标识符（核心 ID）
ldomCoreCpuSet	显示字符串	只读	映射到核心 cpuset 的 CPU 列表

域版本信息的标量变量

Logical Domains Manager 协议支持域版本（由主版本号和次要版本号组成）。Oracle VM Server for SPARC MIB 具有描述域版本信息的标量变量。

表 20-23 域版本信息的标量变量

名称	数据类型	访问	说明
ldomVersionMajor	整数	只读	主版本号
ldomVersionMinor	整数	只读	次要版本号

ldomVersionMajor 和 ldomVersionMinor 的值等于 ldm list -p 命令所显示的版本。例如：

```
$ ldm ls -p
VERSION 1.6
...

$ snmpget -v1 -c public localhost SUN-LDOM-MIB::ldomVersionMajor.0
SUN-LDOM-MIB::ldomVersionMajor.0 = INTEGER: 1

$ snmpget -v1 -c public localhost SUN-LDOM-MIB::ldomVersionMinor.0
SUN-LDOM-MIB::ldomVersionMinor.0 = INTEGER: 5
```

使用 SNMP 陷阱

本节描述如何将系统设置为发送和接收陷阱。本节还描述可用来接收逻辑域（简称域）更改通知的陷阱，以及其他可供您使用的陷阱。

Oracle VM Server for SPARC MIB 为 Oracle Solaris 10 和 Oracle Solaris 11 提供相同的 SNMP 陷阱。但是，snmptrapd 守护进程不再自动接受 Oracle Solaris 11 的所有传入陷阱。必须使用授权的 SNMP v1 和 v2c 团体字符串和/或 SNMPv3 用户配置该守护进程。未授权的陷阱或通知将被丢弃。请参见 snmptrapd.conf(4) 和 snmptrapd.conf(5) 手册页。

使用 Oracle VM Server for SPARC MIB 模块陷阱

Oracle Solaris 11 MIB 与 Oracle Solaris 10 MIB 提供相同的 SNMP 陷阱。但是，net-snmp 版本不同，必须以不同方式对其进行配置。在 Oracle Solaris 10 MIB 中，snmptrapd 接受所有传入通知并自动记录它们。在 Oracle Solaris 11 MIB 中，会对传入通知应用访问控制检查。如果在运行 snmptrapd 时未使用合适的配置文件或等效的访问控制设置，则不会处理此类陷阱。请参见 snmptrapd.conf(4) 和 snmptrapd.conf(5) 手册页。

▼ 如何发送陷阱

1. 配置陷阱。

- Oracle Solaris 10：

  编辑 `/etc/sma/snmp/snmpd.conf` 文件，以添加用来定义陷阱、信息版本和目标的指令。

  ```
  trapcommunity string --> define community string to be used when sending traps
  trapsink host[community [port]] --> to send v1 traps
  trap2sink host[community [port]] --> to send v2c traps
  informsink host[community [port]] --> to send informs
  ```

  有关更多信息，请参见 `snmpd.conf`(4) 和 `snmpd.conf`(5) 手册页。

- Oracle Solaris 11：

  编辑 `/etc/net-snmp/snmp/snmpd.conf` SNMP 配置文件，以添加用来定义陷阱、信息版本和目标的指令。

  必须使用 `pfedit` 命令编辑 `/etc/net-snmp/snmp/snmpd.conf` 文件。

  ```
  trapcommunity string --> define community string to be used when sending traps
  trapsink host[community [port]] --> to send v1 traps
  trap2sink host[community [port]] --> to send v2c traps
  informsink host[community [port]] --> to send informs
  ```

  有关更多信息，请参见 `snmpd.conf`(4) 和 `snmpd.conf`(5) 手册页。

  例如，以下指令将在发送陷阱时使用 `public` 字符串作为团体字符串，并指示将 `v1` 陷阱发送到 `localhost` 目标：

  ```
  trapcommunity public
  trapsink localhost
  ```

2. 通过创建或编辑 **`/usr/etc/snmp/snmptrapd.conf`** SNMP **`trapd`** 配置文件配置访问控制设置。

   必须使用 `pfedit` 命令编辑 `/etc/net-snmp/snmp/snmpd.conf` 文件。

   以下示例显示授权发送陷阱的用户 (`public`) 以及应如何处理传入陷阱 (`log,execute,net`)。请参见 `snmptrapd.conf`(4) 和 `snmptrapd.conf`(5) 手册页。

   ```
   authCommunity log,execute,net public
   ```

3. 要接收 SNMP 陷阱消息，请启动 SNMP 陷阱守护进程实用程序 **`snmptrapd`**。

例 20-4 发送 SNMP `v1` 和 `v2c` 陷阱

此示例向同一主机上运行的 SNMP 陷阱守护进程发送 `v1` 和 `v2c` 陷阱。使用以下指令更新 Oracle Solaris 10 `/etc/sma/snmp/snmpd.conf` 文件或 Oracle Solaris 11 `/etc/net-snmp/snmp/snmpd.conf` 文件：

```
trapcommunity public
```

```
trapsink localhost
trap2sink localhost
```

▼ 如何接收陷阱

- 启动 SNMP 陷阱守护进程实用程序。

  - Oracle Solaris 10：

    有关输出格式选项的信息，请参见 snmptrapd(1M) 手册页。

    snmptrapd 实用程序是一种 SNMP 应用程序，可用来接收和记录 SNMP TRAP 消息。例如，下面的 snmptrapd 命令显示创建了一个名为 ldg2 (ldomName = ldg2) 的新域 (ldomTrapDesc = Ldom Created)。

    ```
    # /usr/sfw/sbin/snmptrapd -P -F \
    "TRAP from %B on %m/%l/%y at %h:%j:%k Enterprise=%N Type=%w SubType=%q\n
    with Varbinds: %v\nSecurity info:%P\n\n" localhost:162
    TRAP from localhost on 5/18/2007 at 16:30:10 Enterprise=. Type=0 SubType=0
    with Varbinds: DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (47105)
    0:07:51.05 SNMPv2-MIB::snmpTrapOID.0 = OID: SUN-LDOM-MIB::ldomCreate
    SUN-LDOM-MIB::ldomIndexNotif = INTEGER: 3 SUN-LDOM-MIB::ldomName = STRING: ldg2
    SUN-LDOM-MIB::ldomTrapDesc = STRING: Ldom Created
    Security info:TRAP2, SNMP v2c, community public
    ```

    请注意，为方便阅读，已将 -F 选项参数字符串分为两行。

  - Oracle Solaris 11：

    有关输出格式选项的信息，请参见 snmptrapd(1M) 手册页。

    snmptrapd 实用程序是一种 SNMP 应用程序，可用来接收和记录 SNMP TRAP 消息。例如，下面的 snmptrapd 命令显示创建了一个名为 ldg2 (ldomName = ldg2) 的新域 (ldomTrapDesc = Ldom Created)。

    ```
    # /usr/sbin/snmptrapd -f -Le -F \
    "TRAP from %B on %m/%l/%y at %h:%j:%k Enterprise=%N Type=%w SubType=%q\n
    with Varbinds: %v\nSecurity info:%P\n\n" localhost:162
    NET-SNMP version 5.4.1
    TRAP from localhost on 6/27/2012 at 12:13:48
    Enterprise=SUN-LDOM-MIB::ldomMIBTraps Type=6 SubType=SUN-LDOM-MIB::ldomCreate
    with Varbinds: SUN-LDOM-MIB::ldomIndexNotif = INTEGER: 3
    SUN-LDOM-MIB::ldomName = STRING: ldg2   SUN-LDOM-MIB::ldomTrapDesc = STRING:
    Ldom Created
    Security info:TRAP, SNMP v1, community public
    ```

    请注意，为方便阅读，已将 -F 选项参数字符串分为两行。

Oracle VM Server for SPARC MIB 陷阱描述

本节描述可供您使用的 Oracle VM Server for SPARC MIB 陷阱。

域创建 (ldomCreate)

此陷阱会在有任何域创建出来时通知您。

表 20-24 域创建陷阱 (ldomCreate)

名称	数据类型	说明
ldomIndexNotif	整数	ldomTable 的索引
ldomName	显示字符串	域的名称
ldomTrapDesc	显示字符串	陷阱的描述

域销毁 (ldomDestroy)

此陷阱会在有任何域被销毁时通知您。

表 20-25 域销毁陷阱 (ldomDestroy)

名称	数据类型	说明
ldomIndexNotif	整数	ldomTable 的索引
ldomName	显示字符串	域的名称
ldomTrapDesc	显示字符串	陷阱的描述

域状态更改 (ldomStateChange)

此陷阱会通知您任何域操作状态更改。

表 20-26 域状态更改陷阱 (ldomStateChange)

名称	数据类型	说明
ldomIndexNotif	整数	ldomTable 的索引
ldomName	显示字符串	域的名称
ldomOperState	整数	域的新状态
ldomStatePrev	整数	域的先前状态

名称	数据类型	说明
ldomTrapDesc	显示字符串	陷阱的描述

虚拟 CPU 更改 (ldomVCpuChange)

此陷阱会在域中虚拟 CPU 的数量发生更改时通知您。

表 20-27 域虚拟 CPU 更改陷阱 (ldomVCpuChange)

名称	数据类型	说明
ldomIndexNotif	整数	ldomTable 的索引
ldomName	显示字符串	包含虚拟 CPU 的域的名称
ldomNumVCPU	整数	域中虚拟 CPU 的新数量
ldomNumVCPUPrev	整数	域中虚拟 CPU 的先前数量
ldomTrapDesc	显示字符串	陷阱的描述

虚拟内存更改 (ldomVMemChange)

此陷阱会在域中虚拟内存量发生更改时通知您。

表 20-28 域虚拟内存更改陷阱 (ldomVMemChange)

名称	数据类型	说明
ldomIndexNotif	整数	ldomTable 的索引
ldomName	显示字符串	包含虚拟内存的域的名称
ldomMemSize	整数	域的虚拟内存量
ldomMemSizePrev	整数	域的先前虚拟内存量
ldomMemUnit	整数	虚拟内存的内存单位，为下列值之一： ■ 1 表示 KB ■ 2 表示 MB ■ 3 表示 GB ■ 4 表示字节 如果未指定，则单位值为字节。
ldomMemUnitPrev	整数	先前虚拟内存的内存单位，为下列值之一： ■ 1 表示 KB ■ 2 表示 MB ■ 3 表示 GB ■ 4 表示字节

名称	数据类型	说明
		如果未指定，则单位值为字节。
ldomTrapDesc	显示字符串	陷阱的描述

虚拟磁盘服务更改 (ldomVdsChange)

此陷阱在域的虚拟磁盘服务发生更改时通知您。

表 20-29 域虚拟磁盘服务更改陷阱 (ldomVdsChange)

名称	数据类型	说明
ldomIndexNotif	整数	ldomTable 的索引
ldomName	显示字符串	包含虚拟磁盘服务的域的名称
ldomVdsServiceName	显示字符串	已更改的虚拟磁盘服务的名称
ldomChangeFlag	整数	指示虚拟磁盘服务发生的下列更改之一： ■ 1 表示“添加” ■ 2 表示“修改” ■ 3 表示“删除”
ldomTrapDesc	显示字符串	陷阱的描述

虚拟磁盘更改 (ldomVdiskChange)

此陷阱在域的虚拟磁盘发生更改时通知您。

表 20-30 虚拟磁盘更改陷阱 (ldomVdiskChange)

名称	数据类型	说明
ldomIndexNotif	整数	ldomTable 的索引
ldomName	显示字符串	包含虚拟磁盘设备的域的名称
ldomVdiskName	显示字符串	发生更改的虚拟磁盘设备的名称
ldomChangeFlag	整数	指示虚拟磁盘服务发生的下列更改之一： ■ 1 表示“添加” ■ 2 表示“修改” ■ 3 表示“删除”
ldomTrapDesc	显示字符串	陷阱的描述

虚拟交换机更改 (ldomVswChange)

此陷阱在域的虚拟交换机发生更改时通知您。

表 20-31 虚拟交换机更改陷阱 (ldomVswChange)

名称	数据类型	说明
ldomIndexNotif	整数	ldomTable 的索引
ldomName	显示字符串	包含虚拟交换机服务的域的名称
ldomVswServiceName	显示字符串	已更改的虚拟交换机服务的名称
ldomChangeFlag	整数	指示虚拟交换机服务发生的下列更改之一： ■ 1 表示“添加” ■ 2 表示“修改” ■ 3 表示“删除”
ldomTrapDesc	显示字符串	陷阱的描述

虚拟网络更改 (ldomVnetChange)

此陷阱在域的虚拟网络发生更改时通知您。

表 20-32 虚拟网络更改陷阱 (ldomVnetChange)

名称	数据类型	说明
ldomIndexNotif	整数	ldomTable 的索引
ldomName	显示字符串	包含虚拟网络设备的域的名称
ldomVnetDevName	显示字符串	域的虚拟网络设备的名称
ldomChangeFlag	整数	指示虚拟磁盘服务发生的下列更改之一： ■ 1 表示“添加” ■ 2 表示“修改” ■ 3 表示“删除”
ldomTrapDesc	显示字符串	陷阱的描述

虚拟控制台集中器更改 (ldomVccChange)

此陷阱在域的虚拟控制台集中器发生更改时通知您。

表 20-33 虚拟控制台集中器更改陷阱 (ldomVccChange)

名称	数据类型	说明
ldomIndexNotif	整数	ldomTable 的索引
ldomName	显示字符串	包含虚拟控制台集中器的域的名称
ldomVccName	显示字符串	发生更改的虚拟控制台集中器服务的名称

名称	数据类型	说明
ldomChangeFlag	整数	指示虚拟控制台集中器发生的下列更改之一： ■ 1 表示“添加” ■ 2 表示“修改” ■ 3 表示“删除”
ldomTrapDesc	显示字符串	陷阱的描述

虚拟控制台组更改 (ldomVconsChange)

此陷阱在域的虚拟控制台组发生更改时通知您。

表 20-34 虚拟控制台组更改陷阱 (ldomVconsChange)

名称	数据类型	说明
ldomIndexNotif	整数	ldomTable 的索引
ldomName	显示字符串	包含虚拟控制台组的域的名称
ldomVconsGroupName	显示字符串	发生更改的虚拟控制台组的名称
ldomChangeFlag	整数	指示虚拟控制台组发生的下列更改之一： ■ 1 表示“添加” ■ 2 表示“修改” ■ 3 表示“删除”
ldomTrapDesc	显示字符串	陷阱的描述

启动和停止域

本节描述可用来停止和启动域的有效管理操作。您可以通过为域表 ldomTable 的 ldomAdminState 属性设置值来控制这些有效管理操作。请参见表 20-1 “域表 (ldomTable)”。

▼ 如何启动域

此过程描述如何启动现有的绑定域。如果具有指定域名的域不存在或尚未绑定，此操作将会失败。

1. 确认 *domain-name* 域存在且已绑定。

```
# ldm list domain-name
```

2. 标识 **ldomTable** 中的 *domain-name*。

```
# snmpwalk -v1 -c public localhost SUN-LDOM-MIB::ldomTable
SUN-LDOM-MIB::ldomName.1 = STRING: primary
SUN-LDOM-MIB::ldomName.2 = STRING: LdomMibTest_1
SUN-LDOM-MIB::ldomAdminState.1 = INTEGER: 0
SUN-LDOM-MIB::ldomAdminState.2 = INTEGER: 0
SUN-LDOM-MIB::ldomOperState.1 = INTEGER: active(1)
SUN-LDOM-MIB::ldomOperState.2 = INTEGER: bound(6)
SUN-LDOM-MIB::ldomNumVCpu.1 = INTEGER: 8
SUN-LDOM-MIB::ldomNumVCpu.2 = INTEGER: 4
SUN-LDOM-MIB::ldomMemSize.1 = INTEGER: 3360
SUN-LDOM-MIB::ldomMemSize.2 = INTEGER: 256
SUN-LDOM-MIB::ldomMemUnit.1 = INTEGER: megabytes(2)
SUN-LDOM-MIB::ldomMemUnit.2 = INTEGER: megabytes(2)
SUN-LDOM-MIB::ldomNumCrypto.1 = INTEGER: 1
SUN-LDOM-MIB::ldomNumCrypto.2 = INTEGER: 0
SUN-LDOM-MIB::ldomNumIOBus.1 = INTEGER: 2
SUN-LDOM-MIB::ldomNumIOBus.2 = INTEGER: 0
SUN-LDOM-MIB::ldomUUID.1 = STRING: 5f8817d4-5d2e-6f7d-c4af-91b5b34b5723
SUN-LDOM-MIB::ldomUUID.2 = STRING: 11284146-87ca-4877-8d80-cd0f60d5ec26
SUN-LDOM-MIB::ldomMacAddress.1 = STRING: 00:14:4f:46:47:d6
SUN-LDOM-MIB::ldomMacAddress.2 = STRING: 00:14:4f:f8:d5:6c
SUN-LDOM-MIB::ldomHostID.1 = STRING: 0x844647d6
SUN-LDOM-MIB::ldomHostID.2 = STRING: 0x84f8d56c
SUN-LDOM-MIB::ldomFailurePolicy.1 = STRING: ignore
SUN-LDOM-MIB::ldomFailurePolicy.2 = STRING: ignore
SUN-LDOM-MIB::ldomMaster.1 = STRING:
SUN-LDOM-MIB::ldomMaster.2 = STRING:
SUN-LDOM-MIB::ldomExtMapinSpace.1 = STRING: off
SUN-LDOM-MIB::ldomExtMapinSpace.2 = STRING: off
SUN-LDOM-MIB::ldomWholeCore.1 = INTEGER: 0
SUN-LDOM-MIB::ldomWholeCore.2 = INTEGER: 0
SUN-LDOM-MIB::ldomCpuArch.1 = STRING: native
SUN-LDOM-MIB::ldomCpuArch.2 = STRING: native
SUN-LDOM-MIB::ldomShutdownGroup.1 = INTEGER: 0
SUN-LDOM-MIB::ldomShutdownGroup.2 = INTEGER: 15
SUN-LDOM-MIB::ldomPerfCounters.1 = STRING: htstrand
SUN-LDOM-MIB::ldomPerfCounters.2 = STRING: global,htstrand
```

3. 启动 *domain-name* 域。

   使用 `snmpset` 命令，可以通过将 `ldomAdminState` 属性的值设置为 `1` 来启动域。*n* 指定要启动的域。

```
# snmpset -v version -c community-string hostname \
SUN-LDOM-MIB::ldomTable.1.ldomAdminState.n = 1
```

4. 使用以下命令之一验证 *domain-name* 域是否处于活动状态：

   - `# ldm list` *domain-name*

- `# snmpget -v` *version* `-c` *community-string hostname* `SUN-LDOM-MIB::ldomOperState.`*n*

例 20-5 启动来宾域

此示例在将 `ldomAdminState` 属性设置为 `1` 之前，先验证 `LdomMibTest_1` 域是否存在且已绑定。最后，`ldm list LdomMibTest_1` 命令验证 `LdomMibTest_1` 域是否处于活动状态。

```
# ldm list LdomMibTest_1
# snmpset -v1 -c private localhost SUN-LDOM-MIB::ldomTable.1.ldomAdminState.2 = 1
# ldm list LdomMibTest_1
```

您还可以使用 `snmpget` 命令（而不是使用 `ldm list` 命令）检索 `LdomMibTest_1` 域的状态。

```
# snmpget -v1 -c public localhost SUN-LDOM-MIB::ldomOperState.2
```

请注意，如果在使用 `snmpset` 启动域时域处于非活动状态，则域将首先绑定然后再启动。

▼ 如何停止域

此过程描述如何停止已启动的域。由该域托管的任何操作系统实例都将停止。

1. 标识 **`ldomTable`** 中的 *domain-name*。

```
# snmpwalk -v1 -c public localhost SUN-LDOM-MIB::ldomTable
SUN-LDOM-MIB::ldomName.1 = STRING: primary
SUN-LDOM-MIB::ldomName.2 = STRING: LdomMibTest_1
SUN-LDOM-MIB::ldomAdminState.1 = INTEGER: 0
SUN-LDOM-MIB::ldomAdminState.2 = INTEGER: 0
SUN-LDOM-MIB::ldomOperState.1 = INTEGER: active(1)
SUN-LDOM-MIB::ldomOperState.2 = INTEGER: bound(6)
SUN-LDOM-MIB::ldomNumVCpu.1 = INTEGER: 8
SUN-LDOM-MIB::ldomNumVCpu.2 = INTEGER: 4
SUN-LDOM-MIB::ldomMemSize.1 = INTEGER: 3360
SUN-LDOM-MIB::ldomMemSize.2 = INTEGER: 256
SUN-LDOM-MIB::ldomMemUnit.1 = INTEGER: megabytes(2)
SUN-LDOM-MIB::ldomMemUnit.2 = INTEGER: megabytes(2)
SUN-LDOM-MIB::ldomNumCrypto.1 = INTEGER: 1
SUN-LDOM-MIB::ldomNumCrypto.2 = INTEGER: 0
SUN-LDOM-MIB::ldomNumIOBus.1 = INTEGER: 2
SUN-LDOM-MIB::ldomNumIOBus.2 = INTEGER: 0
SUN-LDOM-MIB::ldomUUID.1 = STRING: 5f8817d4-5d2e-6f7d-c4af-91b5b34b5723
SUN-LDOM-MIB::ldomUUID.2 = STRING: 11284146-87ca-4877-8d80-cd0f60d5ec26
SUN-LDOM-MIB::ldomMacAddress.1 = STRING: 00:14:4f:46:47:d6
SUN-LDOM-MIB::ldomMacAddress.2 = STRING: 00:14:4f:f8:d5:6c
SUN-LDOM-MIB::ldomHostID.1 = STRING: 0x844647d6
SUN-LDOM-MIB::ldomHostID.2 = STRING: 0x84f8d56c
```

```
SUN-LDOM-MIB::ldomFailurePolicy.1 = STRING: ignore
SUN-LDOM-MIB::ldomFailurePolicy.2 = STRING: ignore
SUN-LDOM-MIB::ldomMaster.1 = STRING:
SUN-LDOM-MIB::ldomMaster.2 = STRING:
SUN-LDOM-MIB::ldomExtMapinSpace.1 = STRING: off
SUN-LDOM-MIB::ldomExtMapinSpace.2 = STRING: off
SUN-LDOM-MIB::ldomWholeCore.1 = INTEGER: 0
SUN-LDOM-MIB::ldomWholeCore.2 = INTEGER: 0
SUN-LDOM-MIB::ldomCpuArch.1 = STRING: native
SUN-LDOM-MIB::ldomCpuArch.2 = STRING: native
SUN-LDOM-MIB::ldomShutdownGroup.1 = INTEGER: 0
SUN-LDOM-MIB::ldomShutdownGroup.2 = INTEGER: 15
SUN-LDOM-MIB::ldomPerfCounters.1 = STRING: htstrand
SUN-LDOM-MIB::ldomPerfCounters.2 = STRING: global,htstrand
```

2. 停止 *domain-name* 域。

   使用 `snmpset` 命令，可以通过将 `ldomAdminState` 属性的值设置为 `2` 来停止域。*n* 指定要停止的域。

   ```
   # snmpset -v version -c community-string hostname \
   SUN-LDOM-MIB::ldomTable.1.ldomAdminState.n = 2
   ```

3. 使用以下命令之一验证 *domain-name* 域是否已绑定：

   - `# ldm list` *domain-name*
   - `# snmpget -v` *version* `-c` *community-string hostname* `SUN-LDOM-MIB::ldomOperState.`*n*

例 20-6 停止来宾域

此示例将 `ldomAdminState` 属性设置为 `2` 以停止来宾域，然后使用 `ldm list LdomMibTest_1` 命令验证 `LdomMibTest_1` 域是否已绑定。

```
# snmpset -v1 -c private localhost SUN-LDOM-MIB::ldomTable.1.ldomAdminState.2 = 2
# ldm list LdomMibTest_1
```

♦ ♦ ♦ 第 21 章

21 Logical Domains Manager 发现

本章提供了有关发现在子网的系统中运行的 Logical Domains Manager 的信息。

发现运行 Logical Domains Manager 的系统

通过使用多播消息，可在子网上发现 Logical Domains Manager。`ldmd` 守护进程能够在网络上侦听特定的多播包。如果此多播消息为某种特定类型，`ldmd` 将回复调用方。这样，就可以在运行 Oracle VM Server for SPARC 的系统上发现 `ldmd`。

多播通信

此发现机制使用 `ldmd` 守护进程所使用的多播网络，以检测自动分配 MAC 地址时所产生的冲突。要配置多播套接字，必须提供以下信息：

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

默认情况下，在计算机已连接到的子网上只能发送多播包。可以通过设置 `ldmd` 守护进程的 `ldmd/hops` SMF 属性来更改以上行为。

消息格式

发现消息必须清晰标记，才能与其他消息区分开。以下多播消息格式确保发现侦听进程可以区分发现消息：

```
#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;
```

▼ 如何发现在子网上运行的 Logical Domains Manager

1. **打开多播套接字。**
   确保您使用的是“多播通信” [389]中指定的端口和组信息。

2. **通过套接字发送 `multicast_msg_t` 消息。**
   此消息应包含以下内容：

   - `version_no` 的有效值，由 `MAC_MULTI_VERSION` 定义，其值为 `1`。
   - `magic_no` 的有效值，由 `MAC_MULTI_MAGIC_NO` 定义，其值为 `92792004`。
   - `LDMD_DISC_SEND` 的 `msg_type`

3. **侦听多播套接字，以获取来自 Logical Domains Manager 的响应。**
   这些响应必须是包含以下内容的 `multicast_msg_t` 消息：

   - `version_no` 的有效值
   - `magic_no` 的有效值
   - 已设置为 `LDMD_DISC_RESP` 的 `msg_type`

- 有效荷载包含 `ldmd_discovery_t` 结构，其中包含以下信息：
  - `ldmd_version` – 在系统上运行的 Logical Domains Manager 的版本
  - `hostname` – 系统的主机名称
  - `ip_address` – 系统的 IP 地址
  - `port_no` – Logical Domains Manager 用于通信的端口号，应该是 XMPP 端口 `6482`

当侦听来自 Logical Domains Manager 的响应时，请确保已放弃任何自动分配 MAC 冲突检测包。

♦ ♦ ♦ 第 22 章

22 将 XML 接口与 Logical Domains Manager 结合使用

本章介绍可扩展标记语言 (Extensible Markup Language, XML) 通信机制，通过该机制，可连接外部用户程序和 Oracle VM Server for SPARC 软件。本章包含下列基本主题：

XML 传输

外部程序可利用可扩展消息处理现场协议 (Extensible Messaging and Presence Protocol, XMPP – RFC 3920) 与 Logical Domains Manager 进行通信。XMPP 受本地和远程连接支持，且默认情况下处于启用状态。要禁用远程连接，请将 `ldmd/xmpp_enabled` SMF 属性设置为 `false` 并重新启动 Logical Domains Manager。

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

注 - 禁用 XMPP 服务器还会阻止域迁移和内存动态重新配置。

XMPP 服务器

Logical Domains Manager 实现了一个能够与许多可用 XMPP 客户机应用程序和库进行通信的 XMPP 服务器。Logical Domains Manager 使用下列安全机制：

- 传输层安全 (Transport Layer Security, TLS)，用于保护客户机与其自身之间的信道。
- 简单身份验证和安全层 (Simple Authentication and Security Layer, SASL)，用于身份验证。`PLAIN` 是唯一受支持的 SASL 机制。您必须将用户名和密码发送到服务器，以便该服务器可以为您授权以执行监视或管理操作。

本地连接

Logical Domains Manager 会检测用户客户机是否与其在同一域中运行，如果是，则与该客户机进行最小 XMPP 握手。具体地说，通过 TLS 设置安全通道后的 SASL 身份验证步骤将被跳过。身份验证和授权的完成都将基于实现客户机接口的进程的凭据。

客户机可以选择实现完全 XMPP 客户机，也可以选择仅运行进行流化处理的 XML 解析器，例如 `libxml2` XML 简单 API (Simple API for XML, SAX) 解析器。无论采用哪种方式，客户机在进行 TLS 协商时都必须处理 XMPP 握手。有关所需次序，请参阅 XMPP 规范。

XML 协议

完成通信初始化后，接下来会发送 Oracle VM Server for SPARC 定义的 XML 消息。XML 消息有两种常规类型：

- 请求和响应消息，使用 `<LDM_interface>` 标记。此类型的 XML 消息用于传送命令并从 Logical Domains Manager 获取返回的结果，类似于使用命令行界面 (command-line interface, CLI) 执行命令。此标记还用于事件的注册和注销。
- 事件消息，使用 `<LDM_event>` 标记。此类型的 XML 消息用于异步报告 Logical Domains Manager 发布的事件。

请求和响应消息

到 Oracle VM Server for SPARC 的 XML 接口有两种不同的格式：

- 一种格式用于向 Logical Domains Manager 发送命令
- 一种格式用于使 Logical Domains Manager 响应传入消息的状态和在该消息中请求的操作。

这两种格式共享许多共同的 XML 结构，但为了更好地理解它们之间的差异，在此分别对它们进行讨论。

请求消息

向 Logical Domains Manager 发出的最基本传入 XML 请求包括对单个对象执行的单个命令的描述。更加复杂的请求可以处理多个命令以及每个命令对应的多个对象。以下示例显示基本 XML 命令的结构。

例 22-1 在单个对象上执行的单个命令的格式

```
<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>
          <!-- Note: More Sections sections can be placed here -->
        </Content>
      </Envelope>
    </data>
    <!-- Note: More Data sections can be placed here -->
  </cmd>
  <!-- Note: More Commands sections can be placed here -->
</LDM_interface>
```

<LDM_interface> 标记

发送给 Logical Domains Manager 的所有命令都必须以 <LDM_interface> 标记开头。发送到 Logical Domains Manager 的任何文档内必须仅包含一个 <LDM_interface> 标记。<LDM_interface> 标记必须包含如例 22-1 “在单个对象上执行的单个命令的格式” 中所示的版本属性。

<cmd> 标记

在 <LDM_interface> 标记内，文档必须至少包含一个 <cmd> 标记。每个 <cmd> 段必须仅包含一个 <action> 标记。<action> 标记用于描述要运行的命令。每个 <cmd> 标记必须至少包含一个 <data> 标记，用于描述要在其上执行命令的对象。

<cmd> 标记还可以包含一个 <options> 标记，用于与某些命令关联的选项和标志。下列命令使用选项：

- `ldm remove-domain` 命令可使用 `-a` 选项。
- `ldm bind-domain` 命令可使用 `-f` 选项。
- `ldm add-vdsdev` 命令可使用 `-f` 选项。
- `ldm cancel-operation` 命令可使用 `migration` 或 `reconf` 选项。
- `ldm add-spconfig` 命令可使用 `-r` *autosave-name* 选项。
- `ldm remove-spconfig` 命令可使用 `-r` 选项。
- `ldm list-spconfig` 命令可使用 `-r` [*autosave-name*] 选项。
- `ldm stop-domain` 命令可以使用以下标记设置命令参数：
  - <force> 表示 `-f` 选项。
  - <halt> 表示 `-h` 选项。
  - <message> 表示 `-m` 选项。
  - <quick> 表示 `-q` 选项。
  - <reboot> 表示 `-r` 选项。
  - <timeout> 表示 `-t` 选项。

  请注意，标记不能包含任何内容值。但是，`-t` 和 `-m` 选项必须具有一个非 null 值，例如 <timeout>10</timeout> 或 <message>Shutting down now</message>。

以下 XML 示例片段说明如何通过重新引导消息向 `ldm stop-domain` 命令传递重新引导请求：

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

<data> 标记

每个 <data> 段都包含与指定命令相关的对象的描述。<data> 段的格式以开放虚拟化格式 (Open Virtualization Format, OVF) 规范草案中的 XML 模式部分为基础。该模式定义了一个 <Envelope> 段（其中包含一个 <References> 标记，未被 Oracle VM Server for SPARC 使用）以及 <Content> 和 <Section> 段。

对于 Oracle VM Server for SPARC，<Content> 段用于标识和描述特定域。<Content> 节点的 id= 属性中的域名用于标识域。<Content> 段中有一个或多个 <Section> 段，它们根据特定命令需要来描述域的资源。

如果您只需要标识一个域名，则无需使用任何 <Section> 标记。相反，如果命令不需要任何域标识符，则需要提供一个 <Section> 段，用于描述命令所需的资源，该段位于 <Content> 段之外，但仍在 <Envelope> 段之内。

在可以推断出对象信息的情况下，<data> 段不需要包含 <Envelope> 标记。此情况主要适用于对监控适用于操作的所有对象的请求以及事件注册和注销请求。

通过两种额外的 OVF 类型，可以使用 OVF 规范的模式来正确定义所有类型的对象：

- <gprop:GenericProperty> 标记
- <Binding> 标记

<gprop:GenericProperty> 标记用于处理任何对象中未在 OVF 规范中定义的属性。属性名称在节点的 key= 属性中定义，属性的值是节点的内容。<binding> 标记会在 ldm list-bindings 命令输出使用，用于定义绑定到其他资源的资源。

响应消息

从包含的命令和对象上来讲，传出 XML 响应的结构与传入请求的结构很相似，只是添加了针对指定的每个对象和命令的 <Response> 段以及针对请求的总体 <Response> 段。<Response> 段可提供状态和消息信息。以下示例显示了基本 XML 请求的响应结构。

例 22-2 对单个对象上执行的单个命令的响应的格式

```
<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> sections can be placed here -->
        </Content>
      </Envelope>
      <response>
        <status>success or failure</status>
        <resp_msg>Reason for failure</resp_msg>
      </response>
    </data>
    <!-- Note: More Data sections can be placed here -->
    <response>
      <status>success or failure</status>
      <resp_msg>Reason for failure</resp_msg>
    </response>
  </cmd>
  <!-- Note: More Command sections can be placed here -->
  <response>
    <status>success or failure</status>
    <resp_msg>Reason for failure</resp_msg>
  </response>
</LDM_interface>
```

总体响应

此 <response> 段是 <LDM_interface> 段的直接子级，用于指示整个请求的整体成功或失败。除非传入 XML 文档格式错误，否则，<response> 段仅包含一个 <status> 标记。如果此响应状态指示成功，则说明所有对象上的所有命令都已成功。如果此响应状态为失败且没有 <resp_msg> 标记，则说明原始请求中的一个命令失败。<resp_msg> 标记仅用于描述 XML 文档自身的一些问题。

命令响应

<cmd> 段下的 <response> 段通知用户特定命令是成功还是失败。<status> 标记用于显示命令成功还是失败。与总响应一样，如果命令失败，<response> 段仅包含一个 <resp_msg> 标记（如果请求的 <cmd> 段的内容格式错误）。否则，失败状态表示执行该命令的某个对象导致发生故障。

对象响应

最后，<cmd> 段中的每个 <data> 段还包含一个 <response> 段。此段可显示在此特定对象上运行的命令是成功还是失败。如果响应的状态是 SUCCESS，则 <response> 段中不会出现 <resp_msg> 标记。如果状态为 FAILURE，则 <response> 字段会显示一个或多个 <resp_msg> 标记，具体取决于对该对象运行该命令时遇到的错误。运行命令时出现问题、对象格式错误或未知均可导致对象错误。

除 <response> 段外，<data> 段还可以包含其他信息。此信息的格式与传入 <data> 字段的格式相同，用于描述导致失败的对象。请参见“<data> 标记” [396]。此附加信息在以下情况下非常有用：

- 当某命令针对特定 <data> 段失败而针对所有其他 <data> 段成功时
- 当空的 <data> 段传入命令并对一些域失败，而对其他域成功时

事件消息

您可以订阅以接收发生的某些状态更改的事件通知，而不使用轮询。可以单独订阅或集体订阅的事件类型有三种。有关完整的详细信息，请参见“事件类型” [400]。

注册和注销

使用 <LDM_interface> 消息可注册事件。请参见“<LDM_interface> 标记” [395]。<action> 标记详细说明了要注册或注销的事件类型，<data> 段保留为空。

例 22-3 事件注册请求消息示例

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

Logical Domains Manager 通过表示注册或注销是否成功的 <LDM_interface> 响应消息进行响应。

例 22-4 事件注册响应消息示例

```
<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>
```

在事件子段中会列出每个类型的事件的操作字符串。

<LDM_event> 消息

事件消息除其开始标记是 <LDM_event> 之外，其格式与传入 <LDM_interface> 消息相同。消息的 <action> 标记是为触发事件而执行的操作。消息的 <data> 段用于描述与事件关联的对象；其详细信息取决于所发生事件的类型。

例 22-5 <LDM_event> 通知示例

```
<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>
```

事件类型

您可以订阅以下事件类型：

- 域事件
- 硬件事件
- 进度事件
- 资源事件

所有事件都与 ldm 子命令相对应。

域事件

域事件用于描述可直接对域执行的操作。可以在 <LDM_event> 消息的 <action> 标记中指定以下域事件：

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

这些事件在 OVF <data> 段中始终仅包含一个 <Content> 标记，用于描述发生事件的域。要注册域事件，请发送 <LDM_interface> 消息，其中，<action> 标记设置为 reg-domain-events。要取消注册这些事件，请发送 <LDM_interface> 消息，其中，<action> 标记设置为 unreg-domain-events。

硬件事件

硬件事件与更改物理系统硬件相关。如果使用 Oracle VM Server for SPARC 软件，则唯一的硬件更改是，在添加、删除或设置服务处理器 (service processor, SP) 配置时对 SP 进行的更改。当前，适用于此情况的仅有三种事件：

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

硬件事件在 OVF <data> 段中始终仅包含一个 <Section> 标记，用于描述发生事件的 SP 配置。要注册这些事件，请发送 <LDM_interface> 消息，其中，<action> 标记设置为 reg-hardware-events。要取消注册这些事件，请发送 <LDM_interface> 消息，其中，<action> 标记设置为 unreg-hardware-events。

进度事件

进度事件是针对长时间运行的命令（例如，域迁移）发布的。这些事件可报告在运行命令期间完成的进展程度。目前，仅报告 migration-process 事件。

进度事件在 OVF <data> 段中始终仅包含一个 <Section> 标记，用于描述受事件影响的 SP 配置。要注册这些事件，请发送 <LDM_interface> 消息，其中，<action> 标记设置为 reg-hardware-events。要取消注册这些事件，请发送 <LDM_interface> 消息，其中，<action> 标记设置为 unreg-hardware-events。

进度事件的 <data> 段包含一个 <content> 段，用于描述受影响的域。此 <content> 段使用 ldom_info <Section> 标记来更新进度。下列常规属性显示于 ldom_info 段中：

- --progress – 命令执行的进度百分比
- --status – 命令状态，可以是 ongoing、failed 或 done 中的一种。
- --source – 报告进度的计算机

资源事件

在任意域中添加、删除或更改资源时，会发生资源事件。其中某些事件的 <data> 段包含 <Content> 标记，该标记下的 <Section> 标记用于在 OVF <data> 段提供服务名称。

可以在 <LDM_event> 消息的 <action> 标记中指定以下事件：

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

以下资源事件在 OVF <data> 段中始终仅包含 <Content> 标记，用于描述发生事件的域。

- add-vcpu
- add-crypto
- add-memory
- add-io
- add-variable
- add-vconscon
- add-vdisk
- add-vdiskserver

- add-vnet
- add-vswitch
- add-vdpcs
- add-vdpcc
- set-vcpu
- set-crypto
- set-memory
- set-variable
- set-vnet
- set-vconsole
- set-vdisk
- remove-vcpu
- remove-crypto
- remove-memory
- remove-io
- remove-variable
- remove-vdisk
- remove-vnet
- remove-vdpcc

要注册资源事件，请发送 <LDM_interface> 消息，其中，<action> 标记设置为 reg-resource-events。要注销这些事件，需要使用 <LDM_interface> 消息，其中，<action> 标记设置为 unreg-resource-events。

所有事件

您还可以注册以侦听所有三种类型的事件，而无需单独注册每个事件。要同时注册所有这三种事件，需要发送 <LDM_interface> 消息，其中，<action> 标记设置为 reg-all-events。要注销这些事件，需要使用 <LDM_interface> 消息，其中，<action> 标记设置为 unreg-all-events。

Logical Domains Manager 操作

<action> 标记中指定的命令（*-*-events 命令除外）对应于 ldm 命令行界面中的那些命令。有关 ldm 子命令的详细信息，请参见 ldm(1M) 手册页。

注 - XML 接口不支持 Logical Domains Manager CLI 所支持的动词或命令别名。

下面是 <action> 标记中支持的字符串：

- add-domain
- add-io
- add-mau
- add-memory
- add-spconfig
- add-variable
- add-vconscon
- add-vcpu
- add-vdisk
- add-vdiskserver
- add-vdiskserverdevice
- add-vdpcc
- add-vdpcs
- add-vnet
- add-vswitch
- bind-domain
- cancel-operation
- list-bindings
- list-constraints
- list-devices
- list-domain
- list-services
- list-spconfig
- list-variable
- 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-vdpcc
- remove-vdpcs
- remove-vnet
- remove-vswitch
- set-domain
- set-mau
- set-memory
- set-spconfig
- set-variable
- set-vconscon
- set-vconsole
- set-vcpu
- 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 资源和属性

本节提供 Logical Domains Manager 资源以及可为每种资源定义的属性的示例。这些资源和属性在 XML 示例中以**粗体**显示。下列示例显示了未绑定输出的资源。可以使用约束输出为 Logical Domains Manager 操作创建输入，但域迁移输出除外。请参见“域迁移” [421]。每个资源都定义在 `<Section>` OVF 段中并由 `<rasd:OtherResourceType>` 标记指定。

域信息 (ldom_info) 资源

以下示例显示 ldom_info 资源的可选属性：

例 22-6 ldom_info XML 输出示例

以下示例显示为多个 ldom_info 属性指定的值，例如，uuid、hostid 和 Address。

```
<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 资源始终包含在 <Content> 段中。ldom_info 资源中的下列属性是可选属性：

- <uuid> 标记（指定域的 UUID）。
- <rasd:Address> 标记，指定要分配给域的 MAC 地址。
- <gprop:GenericProperty key="extended-mapin-space"> 标记，用于指定为域启用 (on) 还是禁用 (off) 扩展 mapin 空间。默认值为 off。
- <gprop:GenericProperty key="failure-policy"> 标记，用于指定从属域在主域发生故障时应表现出的行为。默认值为 ignore。下面是有效的属性值：
  - ignore 忽略主域故障（从属域不受影响）。
  - panic 在主域发生故障时使所有从属域都发生紧急情况。
  - reset 在主域发生故障时重置所有从属域。
  - stop 在主域发生故障时停止所有从属域。
- <gprop:GenericProperty key="hostid"> 标记，用于指定要分配给域的主机 ID。
- <gprop:GenericProperty key="master"> 标记，用于最多指定四个以逗号分隔的主域名称。

- <gprop:GenericProperty key="progress"> 标记，用于指定命令执行的进度百分比。
- <gprop:GenericProperty key="source"> 标记，用于指定报告命令进度的计算机。
- <gprop:GenericProperty key="status"> 标记，用于指定命令的状态（done、failed 或 ongoing）。
- <gprop:GenericProperty key="rc-add-policy"> 标记，用于指定对于可能添加到指定域的根联合体是启用还是禁用直接 I/O 和 SR-IOV I/O 虚拟化操作。有效值为 iov 和空值 (rc-add-policy=)。
- <gprop:GenericProperty key="perf-counters"> 标记，用于指定要访问的性能寄存器集（global、htstrand、strand）。

  如果平台没有性能访问功能，将忽略 perf-counters 属性值。

CPU (cpu) 资源

add-vcpu、set-vcpu 和 remove-vcpu XML 请求操作的等效操作是按照如下方式设置 <gpropGenericProperty key="wcore"> 标记的值：

- 如果使用 -c 选项，请将 wcore 属性设置为指定的全体核心数。
- 如果未使用 -c 选项，请将 wcore 属性设置为 0。

请注意，cpu 资源的分配单位属性 <rasd:AllocationUnits> 始终指定虚拟 CPU 数，而非核心数。

cpu 资源始终包含在 <Content> 段中。

例 22-7 cpu XML 示例

以下示例显示 ldm add-vcpu -c 1 ldg1 命令的等效 XML 请求。

Oracle VM Server for SPARC 3.2 是包括 -c 选项和 ldm add-vcpu、ldm set-vcpu 和 ldm remove-vcpu 命令的最新软件发行版。

```
<?xml version="1.0"?>
<LDM_interface version="1.3"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="./schemas/combined-v3.xsd"
xmlns:ovf="./schemas/envelope"
xmlns:rasd="./schemas/CIM_ResourceAllocationSettingData"
xmlns:vssd="./schemas/CIM_VirtualSystemSettingData"
xmlns:gprop="./schemas/GenericProperty"
xmlns:bind="./schemas/Binding">
  <cmd>
    <action>add-vcpu</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>cpu</rasd:OtherResourceType>
              <rasd:AllocationUnits>8</rasd:AllocationUnits>
              <gprop:GenericProperty key="wcore">1</gprop:GenericProperty>
            </Item>
          </Section>
        </Content>
      </Envelope>
    </data>
  </cmd>
</LDM_interface>
```

例 22-8 `ldm list-bindings` 命令的 `cpu` XML 部分输出

以下示例使用 `ldm list-bindings` 命令显示 `<cpu>` 段的 XML 输出。

```
<?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>
              <threading>8</threading>
              <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 22-9 `ldm list-domain` 命令的 `cpu` XML 部分输出

以下示例使用 `ldm list-domain` 命令显示 `<cpu>` 段的 XML 输出。

```
<?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) 资源

mau 资源始终包含在 <Content> 段中。唯一属性为 <rasd:AllocationUnits> 标记，用于指出 MAU 数或其他加密单元数。

注 - mau 资源是受支持服务器上的任意受支持的加密单元。当前，存在两个受支持的加密单元：模运算单元 (Modular Arithmetic Unit, MAU) 和控制字队列 (Control Word Queue, CWQ)。

例 22-10 mau XML 示例

```
<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>
```

内存 (memory) 资源

内存资源始终包含在 <Content> 段中。唯一属性为 <rasd:AllocationUnits> 标记，用于指出内存大小。

例 22-11 memory XML 示例

```
<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>
```

虚拟磁盘服务器 (vds) 资源

虚拟磁盘服务器 (vds) 资源可以在 <Content> 段中作为域描述的一部分，也可以独自出现于 <Envelope> 段中。唯一属性为 <gprop:GenericProperty> 标记，该标记具有 service_name 键，其中包含所描述的 vds 资源的名称。

例 22-12 vds XML 示例

```
<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>
```

虚拟磁盘服务器卷 (vds_volume) 资源

vds_volume 资源可以在 <Content> 段中作为域描述的一部分，也可以独自出现于 <Envelope> 段中。它必须含有带有以下项的 <gprop:GenericProperty> 标记：

- vol_name – 卷名
- service_name – 此卷要绑定到的虚拟磁盘服务器的名称
- block_dev – 要与此卷关联的文件或设备名称

（可选）vds_volume 资源还可以具有以下属性：

- vol_opts – 下列一项或多项，用逗号分隔，在一个字符串内：{ro,slice,excl}
- mpgroup – 多路径（故障转移）组的名称

例 22-13 vds_volume XML 示例

```
<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>
```

磁盘 (disk) 资源

disk 资源始终包含在 <Content> 段中。它必须含有带有以下项的 <gprop:GenericProperty> 标记：

- vdisk_name – 虚拟磁盘的名称
- service_name – 此虚拟磁盘要绑定到的虚拟磁盘服务器的名称
- vol_name – 此虚拟磁盘要关联到的虚拟磁盘服务设备

（可选）disk 资源还可以具有 timeout 属性，它是在虚拟磁盘客户机 (vdc) 与虚拟磁盘服务器 (vds) 之间建立连接时的超时值（秒）。如果具有多个虚拟磁盘 (vdisk) 路径，则 vdc 可以尝试连接到其他 vds。超时设置可以确保在指定时间段内与任何 vds 建立连接。

例 22-14 disk XML 示例

```
<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>
```

虚拟交换机 (vsw) 资源

vsw 资源既可以在 <Content> 段中作为域描述的一部分，也可以独自出现于 <Envelope> 段中。它必须具有带 service_name 键的 <gprop:GenericProperty> 标记，该键是要分配给虚拟交换机的名称。

（可选）vsw 资源还可以具有以下属性：

- <rasd:Address> – 向虚拟交换机分配 MAC 地址
- default-vlan-id – 指定在标记模式下，需要包含虚拟网络设备或虚拟交换机的默认虚拟局域网 (virtual local area network, VLAN)。第一个 VLAN ID (*vid1*) 是为 default-vlan-id 预留的。
- dev_path – 要与此虚拟交换机关联的网络设备的路径
- id – 指定新虚拟交换机设备的 ID。默认情况下将自动生成 ID 值，如果需要匹配 OS 中的现有设备名称，则设置此属性。
- inter_vnet_link – 指定是否为 Inter-Vnet 通信分配 LDC 通道。默认值为 on。
- linkprop – 指定虚拟设备是否应获取物理链路状态更新。当值为 phys-state 时，虚拟设备可获取物理链路状态更新。如果值为空，则虚拟设备不会获取物理链路状态更新（默认设置）。
- mode – Oracle Solaris Cluster 心跳支持的 sc。
- pvid – 端口的虚拟局域网 (virtual local area network, VLAN) 标识符 (identifier, ID)，表示虚拟网络需要以无标记模式成为其成员的 VLAN。
- mtu – 指定虚拟交换机或/和绑定到虚拟交换机的虚拟网络设备的最大传输单元 (maximum transmission unit, MTU)。有效值是 1500 到 16000。如果指定了无效值，ldm 命令会发出错误。
- vid – 虚拟局域网 (virtual local area network, VLAN) 标识符 (identifier, ID)，表示虚拟网络和虚拟交换机需要以标记模式成为其成员的 VLAN。

例 22-15 vsw XML 示例

```
<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">on</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>
```

网络 (network) 资源

network 资源始终包含在 <Content> 段中。它必须含有带有以下项的 <gprop:GenericProperty> 标记：

- linkprop – 指定虚拟设备是否应获取物理链路状态更新。当值为 phys-state 时，虚拟设备可获取物理链路状态更新。如果值为空，则虚拟设备不会获取物理链路状态更新（默认设置）。
- vnet_name – 虚拟网络 (vnet) 的名称
- service_name – 此虚拟网络要绑定到的虚拟交换机 (vswitch) 的名称

（可选）network 资源还可以具有以下属性：

- <rasd:Address> – 向虚拟交换机分配 MAC 地址
- pvid – 端口的虚拟局域网 (virtual local area network, VLAN) 标识符 (identifier, ID)，表示虚拟网络需要以无标记模式成为其成员的 VLAN。
- vid – 虚拟局域网 (virtual local area network, VLAN) 标识符 (identifier, ID)，表示虚拟网络和虚拟交换机需要以标记模式成为其成员的 VLAN。
- mode – 用于为该虚拟网络启用混合 I/O 的 hybrid。

注 - NIU 混合 I/O 功能已过时，而是使用 SR-IOV。Oracle VM Server for SPARC 3.2 是包括此功能的最新软件发行版。

例 22-16 network XML 示例

```
<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="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>
```

虚拟控制台集中器 (vcc) 资源

vcc 资源既可以在 <Content> 段中作为域描述的一部分，也可以独自出现于 <Envelope> 段中。它可以含有带有以下项的 <gprop:GenericProperty> 标记：

- service_name – 要分配给虚拟控制台集中器服务的名称
- min_port – 要与此 vcc 关联的最小端口号
- max_port – 要与此 vcc 关联的最大端口号

例 22-17 vcc XML 示例

```
<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>
```

变量 (var) 资源

var 资源始终包含在 <Content> 段中。它可以含有带有以下项的 <gprop:GenericProperty> 标记：

- name – 变量的名称
- value – 变量的值

例 22-18 var XML 示例

```
<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 设备 (physio_device) 资源

physio_device 资源始终包含在 <Content> 段中。可以使用 add-io、set-io、remove-io、create-vf、destroy-vf 和 set-domain 子命令来修改此资源。

例 22-19 physio_device XML 示例

以下示例说明如何对虚拟功能、物理功能和根联合体执行操作。

- 以下 XML 示例片段说明如何使用 ldm add-io 命令将 /SYS/MB/NET0/IOVNET.PF0.VF0 虚拟功能添加到 ldg1 域。

```
<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>
```

- 以下 XML 示例片段说明如何使用 ldm set-io 命令将 pci_1 根联合体的 iov_bus_enable_iov 属性值设置为 on。

```
<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>
```

- 以下 XML 示例片段说明如何使用 `ldm set-io` 命令将 `/SYS/MB/NET0/IOVNET.PF1` 物理功能的 `unicast-slots` 属性值设置为 `6`。

```
<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>
```

- 以下 XML 示例片段说明如何使用 `ldm create-vf` 命令创建具有以下属性值的 `/SYS/MB/NET0/IOVNET.PF1.VF0` 虚拟功能。
  - `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>
```

- 以下 XML 示例片段说明如何使用 `ldm create-vf` 命令创建 `iov_pf_repeat_count_str` 值 (3) 指定的虚拟功能数以及 `/SYS/MB/NET0/IOVNET.PF1` 物理功能。使用 `iov_pf_repeat_count_str` 属性创建多个虚拟功能时，不能指定其他属性值。

```
<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 配置 (spconfig) 资源

服务处理器 (service processor, SP) 配置 (`spconfig`) 资源始终独自出现于 `<Envelope>` 段中。它可以含有带有以下项的 `<gprop:GenericProperty>` 标记：

- `spconfig_name` – 要在 SP 上存储的配置的名称。

- `spconfig_status` – 特定 SP 配置的当前状态。在 `ldm list-spconfig` 命令的输出中会使用此属性。

例 22-20 `spconfig` XML 示例

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

DRM 策略配置 (`policy`) 资源

DRM 策略 (`policy`) 资源显示在 `<Envelope>` 段，可以包含带以下键的 `<gprop:GenericProperty>` 标记：

- `policy_name` – DRM 策略的名称
- `policy_enable` – 指定 DRM 策略启用还是禁用
- `policy_priority` – DRM 策略的优先级
- `policy_vcpu_min` – 域的虚拟 CPU 资源的最小数量
- `policy_vcpu_max` – 域的虚拟 CPU 资源的最大数量
- `policy_util_lower` – 触发策略分析的最低利用率级别
- `policy_util_upper` – 触发策略分析的最高利用率级别
- `policy_tod_begin` – DRM 策略的有效启动时间
- `policy_tod_end` – DRM 策略的有效停止时间
- `policy_sample_rate` – 抽样率，这是以秒为单位的周期时间
- `policy_elastic_margin` – 介于 CPU 利用率上下限之间的缓冲量
- `policy_attack` – 在任一资源控制周期内可添加的最大资源量
- `policy_decay` – 在任一资源控制周期内可删除的最大资源量

例 22-21 `policy` XML 示例

```
<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>
```

虚拟数据平面通道服务 (vdpcs) 资源

只有 Netra DPS 环境会涉及此资源。vdpcs 资源既可以在 <Content> 段中作为域描述的一部分，也可以独自出现于 <Envelope> 段中。唯一属性为 <gprop:GenericProperty> 标记，其中含有 service_name 属性值项，该项为所描述的虚拟数据平面通道服务 (vdpcs) 资源的名称。

例 22-22 vdpcs XML 示例

```
<Envelope>
  <References/>
  <Content xsi:type="ovf:VirtualSystem_Type" id="ldg1">
    <Section xsi:type="ovf:VirtualHardwareSection_Type">
      <Item>
        <rasd:OtherResourceType>vdpcs</rasd:OtherResourceType>
        <gprop:GenericProperty key="service_name">dg1-vdpcs</gprop:GenericProperty>
      </Item>
    </Section>
  </Content>
</Envelope>
```

虚拟数据平面通道客户机 (vdpcc) 资源

只有 Netra DPS 环境会涉及此资源。虚拟数据平面通道客户机资源始终包含在 <Content> 段中。它可以含有带有以下项的 <gprop:GenericProperty> 标记：

- vdpcc_name – 虚拟数据平面通道客户机 (vdpcc) 的名称
- service_name – 要将此 vdpcc 绑定到的虚拟数据平面通道服务 vdpcs 的名称

例 22-23 vdpcc XML 示例

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

控制台 (console) 资源

console 资源始终包含在 <Content> 段中。它可以含有带有以下项的 <gprop:GenericProperty> 标记：

- port – 此虚拟控制台 (console) 要更改为的端口
- service_name – 此 console 要绑定到的虚拟控制台集中器 (vcc) 服务
- group – 此 console 要绑定到的组的名称
- enable-log – 对此控制台启用或禁用虚拟控制台日志记录

例 22-24 console XML 示例

```
<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>
```

域迁移

以下示例显示了 ldm migrate-domain 命令的 <data> 段中包含的内容。

- 第一个 <Content> 节点（不含 <ldom_info> 段）是要迁移的源域。
- 第二个 <Content> 节点（含有 <ldom_info> 段）是要迁移到的目标域。源域和目标域的名称可以相同。
- 目标域的 <ldom_info> 段描述要迁移到的计算机以及迁移到该计算机所需的详细信息：
  - target-host 是要迁移到的目标计算机。
  - user-name 是目标计算机的登录用户名，该用户名必须采用 SASL 64 位编码。
  - password 是目标计算机的登录密码，该密码必须采用 SASL 64 位编码。

注 - Logical Domains Manager 使用 sasl_decode64() 对目标用户名和密码进行解码，使用 sasl_encode64() 对这些值进行编码。SASL 64 编码等同于 base64 编码。

例 22-25 migrate-domain <data> 段示例

```
<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 模式

由 Logical Domains Manager 使用的 XML 模式位于 /opt/SUNWldm/bin/schemas 目录中。文件名如下：

- cim-common.xsd – cim-common.xsd 模式
- cim-rasd.xsd – cim-rasd.xsd 模式
- cim-vssd.xsd – cim-vssd.xsd 模式
- cli-list-constraint-v3.xsd – cli-list-constraint-v3.xsd 模式
- combined-v3.xsd – LDM_interface XML 模式
- event-v3.xsd – LDM_Event XML 模式
- ldmd-binding.xsd – Binding_Type XML 模式
- ldmd-property.xsd – GenericProperty XML 模式

- `ovf-core.xsd` – `ovf-core.xsd` 模式
- `ovf-envelope.xsd` – `ovf-envelope.xsd` 模式
- `ovf-section.xsd` – `ovf-section.xsd` 模式
- `ovf-strings.xsd` – `ovf-strings.xsd` 模式
- `ovfenv-core.xsd` – `ovfenv-core.xsd` 模式
- `ovfenv-section.xsd` – `ovfenv-section.xsd` 模式

词汇表

本词汇表定义了 Oracle VM Server for SPARC 文档中的术语、缩写和首字母缩略词。

A

API	Application Programming Interface（应用编程接口）。
ASN	Abstract Syntax Notation（抽象语法表示法）。
auditing（审计）	跟踪系统更改并确定实施更改的用户。
auditreduce	用于从审计迹文件中合并和选择审计记录的命令（请参见 auditreduce(1M) 手册页）。
authorization（授权）	一种用于通过使用 Oracle Solaris OS 权限确定谁有权执行任务和访问数据的方式。

B

bge	用于 Broadcom BCM57xx 设备的 Broadcom 千兆位以太网驱动程序。
BSM	Basic Security Module（基本安全模块）。
bsmconv	用于启用 BSM 的命令（请参见 bsmconv(1M) 手册页）。
bsmunconv	用于禁用 BSM 的命令（请参见 bsmunconv(1M) 手册页）。

C

CMT	Chip Multithreading（芯片多线程）。
compliance（符合性）	确定系统配置是否符合预定义的安全配置文件。

configuration（配置）	保存在服务处理器上的逻辑域配置的名称。
constraints（约束）	对于 Logical Domains Manager，约束是您希望分配给特定域的一个或多个资源。您可能会接收到要求添加到域中的所有资源，也可能会得不到任何资源，这取决于可用资源。
control domain（控制域）	通过使用 Logical Domains Manager 创建和管理其他逻辑域和服务的特权域。
CWQ	Control Word Queue（控制字队列），一种加密单元。

D

DHCP	Dynamic Host Configuration Protocol（动态主机配置协议）。
DIO	Direct I/O（直接 I/O）。
DMA	Direct Memory Access（直接内存访问），能够不通过 CPU 在内存和设备（例如，网卡）之间传输数据。
DMP	Dynamic Multipathing（动态多路径）(Veritas)。
domain（域）	请参见 logical domain（逻辑域）。
DPS	Data Plane Software（数据平面软件）。
DR	Dynamic Reconfiguration（动态重新配置）。
drd	用于 Logical Domains Manager 的 Oracle Solaris 10 OS 动态重新配置守护进程（请参见 drd(1M) 手册页）。
DRM	Dynamic Resource Management（动态资源管理）。
DS	Domain Service（域服务）模块 (Oracle Solaris 10 OS)。
DVD	Digital Versatile Disc（数字多功能光盘）。

E

EFI	Extensible Firmware Interface（可扩展固件接口）。
ETM	Encoding Table Management（编码表管理）模块 (Oracle Solaris 10 OS)。

F

FC_AL	Fiber Channel Arbitrated Loop（光纤通道仲裁环路）。
FMA	Fault Management Architecture（故障管理体系结构）。
fmd	Oracle Solaris 10 OS 故障管理器守护进程（请参见 `fmd`(1M) 手册页）。
fmthard	用于填充硬盘上的标签的命令（请参见 `fmthard`(1M) 手册页）。
format	磁盘分区和维护实用程序（请参见 `format`(1M) 手册页）。

G

Gb	Gigabit（千兆位）。
GLDv3	Generic LAN Driver Version 3（Generic LAN Driver 版本 3）。
guest domain（来宾域）	它使用来自 I/O 域和服务域的服务，并由控制域进行管理。

H

hardening（强化）	修改 Oracle Solaris OS 配置以提高安全性。
hypervisor（虚拟机管理程序）	在操作系统和硬件层之间插入的固件层。

I

I/O	输入/输出设备，例如内部磁盘和 PCIe 控制器及其连接的适配器和设备。
I/O domain（I/O 域）	此域对物理 I/O 设备具有直接所有权和直接访问权，而且它与其他逻辑域共享以虚拟设备形式出现的那些设备。
IB	Infiniband。
IDE	Integrated Drive Electronics（集成驱动器电子装置）。
IDR	Interim Diagnostics Release（临时诊断版）。

ILOM	Integrated Lights Out Manager，一种专用硬件和支持软件系统，可用于独立于操作系统来管理服务器。
ioctl	Input/Output Control Call（输入/输出控制调用）。
IPMP	Internet Protocol Network Multipathing（Internet 协议网络多路径）。

K

kaio	Kernel Asynchronous Input/Output（内核异步输入/输出）。
KB	Kilobyte（千字节）。
KU	Kernel Update（内核更新）。

L

LAN	Local-Area Network（局域网）。
LDAP	Lightweight Directory Access Protocol（轻量目录访问协议）。
LDC	Logical Domain Channel（逻辑域通道）。
ldm	Logical Domains Manager 实用程序（请参见 ldm(1M) 手册页）。
ldmd	Logical Domains Manager 守护进程。
lofi	Loopback file（回送文件）。
logical domain（逻辑域）	一种由资源的离散逻辑分组构成的虚拟机，在一个计算机系统中有其自身的操作系统和标识。又称为 *domain*（域）。
Logical Domains Manager	用于创建和管理逻辑域并将资源分配给域的 CLI。

M

MAC	介质访问控制 (media access control, MAC) 地址；Logical Domains Manager 可以自动分配 MAC 地址，您也可以手动分配 MAC 地址。
MAU	Modular Arithmetic Unit（模运算单元）。
MB	Megabyte（兆字节）。
MD	服务器数据库中的机器描述 (Machine Description, MD)。

mem、memory 内存单元默认大小（字节），也可以指定千兆字节 (G)，千字节 (K) 或兆字节 (M)。可分配给来宾域的服务器虚拟内存。

metadb 用于创建和删除 Solaris Volume Manager 元设备状态数据库副本的命令（请参见 metadb(1M) 手册页）。

metaset 用于配置磁盘集的命令（请参见 metaset(1M) 手册页）。

mhd 用于执行多主机磁盘控制操作的命令（请参见 mhd(7i) 手册页）。

MIB Management Information Base（管理信息库）。

minimizing（最小化） 安装所必需的最少数量的核心 Oracle Solaris OS 软件包。

MMF Multimode Fiber（多模光纤）。

MMU Memory Management Unit（内存管理单元）。

mpgroup 虚拟磁盘故障转移的多路径组 (multipathing group, mpgroup) 名称。

mtu Maximum Transmission Unit（最大传输单元）。

N

ndpsldcc Netra DPS Logical Domain Channel Client（Netra DPS 逻辑域通道客户机）。另请参见 vdpcc。

ndpsldcs Netra DPS Logical Domain Channel Service（Netra DPS 逻辑域通道服务）。另请参见 vdpcs。

NIS Network Information Services（网络信息服务）。

NIU Network Interface Unit（网络接口单元）（Oracle 的 Sun SPARC Enterprise T5120 和 T5220 服务器）。

NTS Network Terminal Server（网络终端服务器）。

NVRAM Non-Volatile Random-Access Memory（非易失性随机存取存储器）。

nxge 用于 NIU 10Gb 以太网适配器的驱动程序。

O

OID Object Identifier（对象标识符），用来唯一标识 MIB 中每个对象的一系列数字。

OVF	Open Virtualization Format（开放虚拟化格式）。

P

P2V	Oracle VM Server for SPARC 物理机到虚拟机转换工具。
PA	Physical Address（物理地址）。
PCI	外设部件互连 (peripheral component interconnect, PCI) 总线。
PCI-X	PCI 扩展总线。
PCIe	PCI EXPRESS 总线。
pcpu	Physical CPU（物理 CPU）。
physical domain（物理域）	由一个 Oracle VM Server for SPARC 实例管理的资源范围。物理域可以是一个完整的物理系统，例如，使用受支持的 SPARC T 系列平台时。它可以是整个系统，也可以是该系统的一个子集，例如，使用受支持的 SPARC M 系列平台时。
physical function（物理功能）	支持 SR-IOV 规范中定义的 SR-IOV 功能的 PCI 功能。物理功能包含 SR-IOV 功能结构，用于管理 SR-IOV 功能。物理功能是全面的 PCIe 功能，可以像其他任何 PCIe 设备一样发现、管理和处理。物理功能拥有完全配置资源，可以用于配置或控制 PCIe 设备。
physio	Physical Input/Output（物理输入/输出）。
PICL	Platform Information and Control Library（平台信息和控制库）。
picld	PICL 守护进程（请参见 `picld`(1M) 手册页）。
PM	虚拟 CPU 和内存的电源管理 (Power management, PM)。
praudit	用于输出审计迹文件内容的命令（请参见 `praudit`(1M) 手册页）。
PRI	Priority（优先级）。

R

RA	Real Address（实际地址）。
RAID	Redundant Array of Inexpensive Disk（廉价磁盘冗余阵列），可用于将多个独立磁盘组合为一个逻辑单元。
RPC	Remote Procedure Call（远程过程调用）。

S

SASL	Simple Authentication and Security Layer（简单验证和安全层）。
SAX	XML 简单 API (Simple API for XML, SAX) 解析程序，用于遍历 XML 文档。SAX 解析程序以事件为基础，通常用于对数据进行流化处理。
service domain（服务域）	向其他逻辑域提供诸如虚拟交换机、虚拟控制台连接器和虚拟磁盘服务器等设备的逻辑域。
service processor, SP（服务处理器）	SP 也称为系统控制器 (system controller, SC)，用于监视和运行物理计算机。
SMA	System Management Agent（系统管理代理）。
SMF	Service Management Facility（服务管理工具）。
SMI	Structure of Management Information（管理信息结构），用于定义管理对象并对这些对象进行分组，以供 MIB 使用。
SNMP	Simple Network Management Protocol（简单网络管理协议）。
SR-IOV	Single Root I/O Virtualization（单一根 I/O 虚拟化）。
SSH	Secure Shell（安全 Shell）。
ssh	安全 Shell 命令（请参见 ssh(1) 手册页）。
sshd	安全 Shell 守护进程（请参见 sshd(1M) 手册页）。
SunVTS	Sun Validation Test Suite（Sun 验证测试套件）。
svcadm	处理服务实例（请参见 svcadm(1M) 手册页）。
system controller, SC（系统控制器）	另请参见服务处理器。

T

TLS	Transport Layer Security（传输层安全）。

U

UDP	User Diagram Protocol（用户图表协议）。

unicast（单点传送）	在单个发送方和单个接收方之间发生的网络通信。
uscsi	User SCSI Command Interface（用户 SCSI 命令接口）（请参见 `uscsi`(7I) 手册页）。
UTP	Unshielded Twisted Pair（非屏蔽双绞线）。

V

var	变量。
VBSC	Virtual Blade System Controller（虚拟刀片系统控制器）。
vcc、**vconscon**	虚拟控制台集中器服务，带有分配给来宾域的特定端口范围。
vcons、**vconsole**	用于访问系统级消息的虚拟控制台。通过连接控制域中特定端口上的 `vconscon` 服务来实现连接。
vcpu	Virtual Central Processing Unit（虚拟中央处理器）。服务器中的每个核心都表示为一个虚拟 CPU。
vdc	Virtual Disk Client（虚拟磁盘客户机）。
vdisk	虚拟磁盘是与各种类型的物理设备、卷或文件关联的通用块设备。
vdpcc	Netra DPS 环境中的虚拟数据平面通道客户机 (virtual data plane channel client, vdpcc)。
vdpcs	Netra DPS 环境中的虚拟数据平面通道服务 (virtual data plane channel service, vdpcs)。
vds、**vdiskserver**	通过虚拟磁盘服务器可以将虚拟磁盘导入到逻辑域中。
vdsdev、**vdiskserverdevice**	虚拟磁盘服务器设备是由虚拟磁盘服务器导出的。该设备可以是整个磁盘、一个磁盘分片、一个文件或一个磁盘卷。
virtual function（虚拟功能）	与物理功能关联的 PCI 功能。虚拟功能是一种轻量 PCIe 功能，可以与物理功能以及与同一物理功能关联的其他虚拟功能共享一个或多个物理资源。仅允许虚拟功能拥有用于其自身行为的配置资源。
vldc	虚拟逻辑域通道 (virtual logical domain channel, vldc) 服务。
vldcc	Virtual Logical Domain Channel Client（虚拟逻辑域通道客户机）。
vnet	虚拟网络设备可实现虚拟以太网设备，还可以通过使用虚拟网络交换机 (`vswitch`) 与系统中的其他 `vnet` 设备进行通信。

VNIC	Virtual Network Interface Card（虚拟网络接口卡），是可以从物理网络设备创建并分配给某区域的物理网络设备虚拟实例。
vNTS	Virtual Network Terminal Service（虚拟网络终端服务）。
vntsd	域控制台的 Oracle Solaris 10 OS 虚拟网络终端服务器守护进程（请参见 `vntsd`(1M) 手册页）。
volfs	卷管理文件系统（请参见 `volfs`(7FS) 手册页）。
vsw、**vswitch**	虚拟网络交换机，它可将虚拟网络设备连接到外部网络，还可以在虚拟网络设备和外部网络之间交换包。
VTOC	Volume Table of Content（卷目录）。
VxDMP	Veritas Dynamic Multipathing（Veritas 动态多路径）。
VxVM	Veritas Volume Manager（Veritas 卷管理器）。

X

XFP	eXtreme Fast Path（极速路径）。
XML	Extensible Markup Language（可扩展标记语言）。
XMPP	Extensible Messaging and Presence Protocol（可扩展消息处理现场协议）。

Z

ZFS	Zettabyte File System（Zettabyte 文件系统） (Oracle Solaris 10 OS)。
zpool	ZFS 存储池（请参见 `zpool`(1M) 手册页）。
ZVOL	ZFS Volume Emulation Driver（ZFS 卷模拟驱动程序）。

索引

O

P

T

V

W

X