Oracle® Solaris 11.1 管理：Oracle Solaris Zones、Oracle Solaris 10 Zones 和资源管理

文件号码 E36686–04
2013 年 5 月

ORACLE

130606@25097

目录

前言

本书是一套文档集的组成部分，该集合提供了 Oracle Solaris 操作系统管理信息的主要内容。本书假设您已经安装了该操作系统并且设置了计划使用的任何网络软件。

此发行版中的新增功能在第 176 页中的 “关于此版本的 Oracle Solaris Zones”中进行了介绍。

关于 Oracle Solaris Zones

Oracle Solaris Zones 产品是应用程序的完整运行时环境。区域提供从应用程序到平台资源的虚拟映射。利用区域可以使应用程序组件彼此隔离，即使这些区域共享单个 Oracle Solaris 操作系统实例也是如此。Oracle Solaris Resource Manager 产品组件（通常称为资源管理功能）使您能够分配工作负荷接收的资源数量。

区域建立资源占用（如 CPU）的边界。这些边界可以进行扩展，以适应区域中运行的应用程序不断变化的处理要求。

如需其他隔离，可以配置具有只读根目录的区域，称为不可编辑的区域。

关于 Oracle Solaris 10 Zones

Oracle Solaris 10 Zones（也称为 `solaris10` 标记非全局区域）使用 BrandZ 技术在 Oracle Solaris 11 操作系统上运行 Oracle Solaris 10 应用程序。应用程序在非全局区域所提供的安全环境中运行，不会被修改。这样，您可使用 Oracle Solaris 10 系统来开发、测试和部署应用程序。在这些标记区域内运行的工作负荷可以利用内核的增强功能以及仅适用于 Oracle Solaris 11 发行版的创新技术。

要使用该产品，请参见第 3 部分。

关于在 Oracle Solaris Trusted Extensions 系统上使用 Oracle Solaris Zones

有关在 Oracle Solaris Trusted Extensions 系统上使用区域的信息，请参见《Trusted Extensions 配置和管理》中的第 13 章 “在 Trusted Extensions 中管理区域”。请注意，只能在 Oracle Solaris Trusted Extensions 系统上引导**有标签的**标记。

Oracle Solaris Cluster 区域群集

区域群集是 Oracle Solaris Cluster 软件的一项功能。区域群集的所有节点通过 `cluster` 属性配置为非全局 `solaris` 区域。不允许其他标记类型。在区域群集上运行支持服务的方式与全局群集相同，隔离由区域提供。有关更多信息，请参见《Oracle Solaris Cluster 系统管理指南》 。

Oracle Solaris Resource Manager

使用资源管理，您可以控制应用程序如何使用可用系统资源。请参见第 1 部分。

目标读者

本书适用于负责管理一个或多个运行 Oracle Solaris 发行版的系统的所有人员。要使用本书，您应当至少具备 1 到 2 年的 UNIX 系统管理经验。

获取 Oracle 支持

Oracle 客户可以通过 My Oracle Support 获取电子支持。有关信息，请访问 `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info`，或访问 `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs`（如果您听力受损）。

印刷约定

下表介绍了本书中的印刷约定。

表 P–1 印刷约定

字体或符号	含义	示例
`AaBbCc123`	命令、文件和目录的名称；计算机屏幕输出	编辑 `.login` 文件。 使用 `ls -a` 列出所有文件。 `machine_name% you have mail.`

表 P–1 印刷约定 （续）

字体或符号	含义	示例
AaBbCc123	用户键入的内容，与计算机屏幕输出的显示不同	`machine_name% su` `Password:`
aabbcc123	要使用实名或值替换的命令行占位符	删除文件的命令为 `rm` *filename*。
AaBbCc123	保留未译的新词或术语以及要强调的词	这些称为 *Class* 选项。 **注意**：有些强调的项目在联机时以粗体显示。
新词术语强调	新词或术语以及要强调的词	**高速缓存**是存储在本地的副本。 请**勿**保存文件。
《书名》	书名	阅读《用户指南》的第 6 章。

命令中的 shell 提示符示例

下表显示了 Oracle Solaris OS 中包含的缺省 UNIX shell 系统提示符和超级用户提示符。请注意，在命令示例中显示的缺省系统提示符可能会有所不同，具体取决于 Oracle Solaris 发行版。

表 P–2 shell 提示符

shell	提示符
Bash shell、Korn shell 和 Bourne shell	`$`
Bash shell、Korn shell 和 Bourne shell 超级用户	`#`
C shell	`machine_name%`
C shell 超级用户	`machine_name#`

获取有关特权和管理权限的信息

有关角色和管理权限的更多信息，请参见《Oracle Solaris 11.1 管理：安全服务》中的第 III 部分, “角色、权限配置文件和特权”。

第 1 部 分

Oracle Solaris 资源管理

此部分介绍 Oracle Solaris 资源管理，您可以使用此软件控制应用程序使用可用系统资源的方式。

◆ ◆ ◆ 第 1 章

1 资源管理介绍

您可以利用 Oracle Solaris 资源管理功能控制应用程序如何使用可用系统资源。您可以执行以下操作：

- 分配计算资源，例如处理器时间
- 监视已分配资源使用情况，然后根据需要调整分配
- 生成用于分析、计费和容量规划的扩展记帐信息

本章包含以下主题：

资源管理概述

现代计算环境必须针对系统上不同应用程序产生的不同的工作负荷做出灵活的响应。**工作负荷**是一个或一组应用程序的所有进程的集合。如果没有使用资源管理功能，Oracle Solaris 操作系统将通过动态适应新的应用程序请求来对工作负荷需求做出响应。该缺省响应通常表示系统上的所有活动都对资源具有同等的访问权。使用资源管理功能，您可以对各个工作负荷分别进行处理。您可以执行以下操作：

- 限制访问特定资源
- 按优先级别为工作负荷提供资源
- 将工作负荷彼此隔离

最大限度地减少工作负荷之间的性能影响以及监视资源使用情况和利用率的功能，称为**资源管理**。资源管理通过一组算法来实现。算法处理应用程序在执行过程中提出的一系列功能请求。

使用资源管理功能，您可以针对不同的工作负荷修改操作系统的缺省行为。**行为**主要是指应用程序向操作系统提出一个或多个资源请求时操作系统算法所做出的一组决定。可以使用资源管理功能进行以下操作：

- 拒绝应用程序使用资源或使一个应用程序优先于其他应用程序使用超过其他方式允许的资源分配量
- 集中处理特定的分配（而不使用隔离机制）

实现使用资源管理功能的系统配置的目的有若干种。您可以执行以下操作：

- 防止应用程序毫无限制地占用资源
- 基于外部事件更改应用程序的优先级
- 根据系统利用率最大化的目标，平衡一组应用程序的资源保证

在规划资源管理配置时，主要要求如下：

- 识别系统上争用资源的工作负荷
- 将不产生冲突的工作负荷与那些性能请求会影响主工作负荷的工作负荷区分开来

识别合作的工作负荷和冲突的工作负荷后，可以在系统功能允许的范围内，创建对业务服务目标影响最小的资源配置。

通过提供控制机制、通知机制和监视机制，可在 Oracle Solaris 系统中实现有效资源管理。上述许多功能都是通过增强现有机制来提供的，例如 proc(4) 文件系统、处理器集和调度类。而其他功能是资源管理所特有的。这些功能将在后续章节中介绍。

资源分类

资源是可进行处理以更改应用程序行为的计算系统的任何方面。因此，资源就是应用程序隐式或显式请求的功能。如果拒绝或约束了此类功能，则强大的应用程序在执行时会慢很多。

相对于资源标识，可以按多种标准对资源进行分类。这些标准可以是隐式请求相对于显式请求，基于时间（例如 CPU 时间）相对于与时间无关（例如指定的 CPU 份额等）。

一般情况下，基于调度器的资源管理应用于应用程序可隐式请求的资源。例如，要继续执行，应用程序会隐式请求更多 CPU 时间。要将数据写入网络套接字，应用程序会隐式请求带宽。可针对隐式请求的资源的总使用量设置约束。

也可提供其他接口，以便显式协商带宽或 CPU 服务级别。明确请求的资源（例如请求附加线程）可以通过约束进行管理。

资源管理控制机制

Oracle Solaris 操作系统可用的三种控制机制分别为约束、调度和分区。

约束机制

使用约束，管理员或应用程序开发者可以对工作负荷所占用的特定资源设置限定。限制已知时，建立资源占用方案模型变得简单得多。也可使用限定控制不良应用程序，否则它们会通过发出无法控制的资源请求影响系统的性能或可用性。

约束确实给应用程序带来了复杂因素。它有可能会修改应用程序和系统之间的关系，导致应用程序无法再正常工作。降低这种风险的一种途径是用未知的资源行为逐渐减少对应用程序的约束。第 6 章，资源控制（概述）中讨论的资源控制提供了一种约束机制。可通过编写新的应用程序来了解其资源约束，但是并不是所有应用程序编写者都选择这样做。

调度机制

调度是指按特定间隔做出一系列分配决定。所做的决定基于可预测的算法。不需要当前分配的应用程序会将资源留给其他应用程序使用。基于调度的资源管理可确保在配置的资源充裕时全部进行利用，在配置的资源处于临界或过量使用状态时提供控制分配。底层算法定义了如何解释术语“控制”。在某些情况下，调度算法可能会保证所有应用程序都对资源具有一定的访问权限。第 8 章，公平份额调度器（概述）中介绍的公平份额调度器 (fair share scheduler, FSS) 能够以可控的方法管理应用程序对 CPU 资源的访问权限。

分区机制

分区用于将工作负荷绑定到系统可用资源的子集。该绑定保证工作负荷总是能够使用一定量的资源。使用第 12 章，资源池（概述）中介绍的资源池功能，您可以将工作负荷限定为使用计算机的特定资源部分。

使用分区的配置可避免整个系统的资源缺乏。但是，避免资源缺乏的同时，也降低了实现高利用率的能力。对于保留的资源组（例如处理器），即使其绑定的工作负荷处于闲置状态，也不能由其他工作负荷使用。

资源管理配置

部分资源管理配置可能位于网络名称服务中。该功能允许管理员在一组计算机集合中应用资源管理约束，而不是专门基于单个计算机应用。相关工作可共享一个通用标识符，可以通过记帐数据将此工作的总使用量制成表格。

第 2 章，项目和任务（概述）中更加全面地介绍了资源管理配置和面向工作负荷的标识符。第 4 章，扩展记帐（概述）中介绍了将这些标识符与应用程序资源使用相链接的扩展记帐功能。

与非全局区域交互

资源管理功能可与 Zones 结合使用来进一步完善应用程序环境。将在本指南的适当章节中介绍这些功能与 Solaris Zones 之间的交互。

何时使用资源管理

使用资源管理可以确保应用程序获得所需的响应时间。

资源管理也可增加资源利用率。通过对使用权分类和划分优先级，可在非高峰期有效使用保留资源，这样通常可避免对额外处理能力的需求。您还可以确保资源不会因负荷的改变而浪费。

服务器整合

资源管理非常适合于在单个服务器上整合多个应用程序的环境。

管理大量计算机所带来的成本和复杂性促使在更大、更具伸缩性的服务器上整合多个应用程序。您可以使用资源管理软件在一个系统上分别运行多个工作负荷，而不是通过对单独系统资源的完全访问权限，在每个单独的系统上运行一个工作负荷。使用资源管理，您可以通过在单个 Oracle Solaris 系统上运行和控制多个不同应用程序来降低总体拥有成本。

如果您提供 Internet 和应用程序服务，则可以使用资源管理来执行以下操作：

- 在单个计算机上驻留多个 Web 服务器。您可以控制每个 Web 站点的资源占用，并防止每个站点受到其他站点的可能侵入。
- 防止错误的公共网关接口 (common gateway interface, CGI) 脚本占用全部 CPU 资源。
- 阻止行为不良的应用程序泄漏所有可用虚拟内存。
- 确保用户的应用程序不受同一站点上运行的其他用户应用程序的影响。
- 在同一计算机上提供不同级别或类别的服务。
- 获取用于计费的记帐信息。

支持大规模或变动的用户群体

可以在任何拥有大规模、多样化用户基础的系统（例如教育机构）中使用资源管理功能。如果您有多个工作负荷，则可以将软件配置为赋予特定项目优先权。

例如，在大型的经纪公司里，贸易商需要不时地通过快速访问来执行查询或计算。而其他系统用户的工作负荷相对稳定。如果为贸易商的项目分配了较大比例的处理能力，则贸易商就可获得所需的响应能力。

资源管理也非常适用于支持瘦客户机系统。这些平台为无态控制台提供了帧缓存器和输入设备，例如智能卡。实际的计算在共享服务器上完成，形成了分时型环境。使用资源管理功能可以隔离服务器上的用户。这样，导致过载的用户就不会独占硬件资源并明显影响使用该系统的其他用户。

建立资源管理（任务列表）

以下任务列表高度概括了在您的系统上建立资源管理的步骤。

任务	说明	参考
识别系统上的工作负荷并按项目对每个工作负荷进行分类。	在 /etc/project 文件、NIS 映射或 LDAP 目录服务中创建项目条目。	第 37 页中的 “project 数据库”
设置系统上工作负荷的优先级。	确定哪些是关键的应用程序。这些工作负荷可能需要对资源的优先访问权。	请参阅您的业务服务目标。
监视系统上的实时活动。	使用性能工具查看系统上正在运行的工作负荷的当前资源占用。然后评估是否必须限制对给定资源的访问或将特定工作负荷与其他工作负荷隔离开来。	cpustat(1M)、iostat(1M)、mpstat(1M)、prstat(1M)、 sar(1) 和 vmstat(1M) 手册页
对系统上正在运行的工作负荷进行临时修改。	要确定可以更改哪些值，请参阅 Oracle Solaris 系统中的可用资源控制。当任务或进程正在运行时，可从命令行更新值。	第 73 页中的 “可用的资源控制”、第 79 页中的 “针对资源控制值的全局和本地操作”、第 83 页中的 “在正在运行的系统上临时更新资源控制值” 和 rctladm(1M) 和 prctl(1) 手册页。
在 project 数据库或命名服务项目数据库中为每个项目条目设置资源控制和项目属性。	/etc/project 文件或命名服务项目数据库中的每个项目条目都可包含一个或多个资源控制或属性。资源控制会约束附加到该项目上的任务和进程。对于为资源控制设置的每个阈值，您都可以关联一个或多个在达到该阈值时采取的操作。 您可以使用命令行界面来设置资源控制，	第 37 页中的 “project 数据库”、第 38 页中的 “本地 /etc/project 文件格式”、第 73 页中的 “可用的资源控制”、第 79 页中的 “针对资源控制值的全局和本地操作”和第 8 章，公平份额调度器（概述）
为项目附加的进程集所占用的物理内存资源设置上界。	资源上限执行守护进程将执行在 /etc/project 文件中为项目的 rcap.max-rss 属性定义的物理内存资源上限。	第 37 页中的 “project 数据库”和第 10 章，使用资源上限设置守护进程控制物理内存（概述）
创建资源池配置。	资源池提供了一种对系统资源（例如处理器）进行分区的途径，并在多次重新引导期间维护这些分区。可以在 /etc/project 文件中为每个条目添加一个 project.pool 属性。	第 37 页中的 “project 数据库”和第 12 章，资源池（概述）
将公平份额调度器 (fair share scheduler, FSS) 设置为缺省的系统调度器。	确保所有用户进程位于一个单独的 CPU 系统中，或者位于属于同一调度类的处理器集中。	第 107 页中的 “配置 FSS”和 dispadmin(1M) 手册页

任务	说明	参考
激活扩展记帐功能来监视并记录任务或进程的资源占用情况。	使用扩展记帐数据可以评估当前资源控制并为将来的工作负荷规划容量要求。可以跟踪整个系统的总使用量。要获取多个系统中相关工作负荷的全部使用情况统计信息，可在多台计算机间共享项目名称。	第 64 页中的 “如何为流、进程、任务和网络组件激活扩展记帐” 和 acctadm(1M) 手册页
（可选的）如果需要对配置做其他调整，可继续从命令行更改值。您可以在任务或进程正在运行时更改值。	对现有任务所做的修改可以立即生效，无需重新启动项目。调节值，直至您对性能满意。然后，更新 /etc/project 文件或命名服务项目数据库中的当前值。	第 83 页中的 “在正在运行的系统上临时更新资源控制值”以及 rctladm(1M) 和 prctl(1) 手册页
（可选的）捕获扩展记帐数据。	针对活动的进程和任务编写扩展记帐记录。生成的文件可用于规划、分摊费用和计费。还可以使用 libexacct 的实用摘录和报告语言 (Practical Extraction and Report Language, Perl) 接口，来开发定制报告和摘录脚本。	wracct(1M) 手册页和第 59 页中的 “libexacct 的 Perl 接口”

◆ ◆ ◆ 第 2 章

2 项目和任务（概述）

本章讨论 Oracle Solaris 资源管理的**项目**和**任务**功能。项目和任务用于标记工作负荷并将它们彼此分离。

本章包含以下主题：

要使用项目和任务功能，请参见第 3 章，管理项目和任务。

项目和任务功能

要优化工作负荷响应，必须首先能够识别要分析的系统上运行的工作负荷。此信息可能很难通过单独使用纯粹面向进程或面向用户的方法来获取。在 Oracle Solaris 系统中，您可以使用两个附加功能来分离和识别工作负荷：项目和任务。**项目**为相关工作提供了网络范围内的管理标识符。**任务**将一组进程聚集成表示工作负荷组件的可管理实体。

在 `project` 名称服务数据库中指定的控制针对进程、任务和项目进行了设置。由于进程和任务控制通过 `fork` 和 `settaskid` 系统调用继承，因此，所有在项目内创建的进程和任务都可继承这些控制。有关这些系统调用的信息，请参见 `fork(2)` 和 `settaskid(2)` 手册页。

根据其项目或任务的成员关系，可以使用标准 Oracle Solaris 命令处理正在运行的进程。扩展记帐功能可以报告进程和任务的使用情况，并使用管理项目标识符标记每个记录。通过此进程，可以将脱机工作负荷分析与联机监视相互关联。项目标识符可以通过 `project` 名称服务数据库在多台计算机之间共享。这样，在（或跨）多台计算机上运行的相关工作负荷的资源占用情况最终可以在所有计算机上分析出来。

项目标识符

项目标识符是指用于标识相关工作的管理标识符。可以将项目标识符视为等同于用户标识符和组标识符的工作负荷标记。一个用户或组可以属于一个或多个项目。这些项目可用于表示允许用户（或用户组）参与的工作负荷。此成员关系然后可以作为费用分摊（例如基于使用情况或初始资源分配等）的基础。尽管必须为用户指定一个缺省项目，但是用户启动的进程可以与用户参与的任何项目关联。

确定用户的缺省项目

要登录到系统，必须为用户指定一个缺省项目。即使用户不在缺省项目中指定的用户或组列表中，此用户仍会自动成为该项目的成员。

由于系统上的每个进程都具有项目成员资格，因此，必须使用一种算法为登录或其他初始进程指定缺省项目。此算法在 `getprojent`(3C) 手册页中进行了介绍。系统按照顺序步骤确定缺省项目。如果找不到缺省项目，则会拒绝用户的登录或启动进程的请求。

系统按顺序执行以下步骤，以确定用户的缺省项目：

1. 如果用户具有在 `/etc/user_attr` 扩展用户属性数据库中定义了 `project` 属性的某个条目，则 `project` 属性的值即为缺省项目。请参见 `user_attr(4)` 手册页。
2. 如果 `project` 数据库中存在名为 `user.`*user-id* 的项目，则该项目即为缺省项目。有关更多信息，请参见 `project(4)` 手册页。
3. 如果 `project` 数据库中存在名为 `group.` *group-name* 的项目，其中 *group-name* 是在 `passwd` 文件中指定的用户缺省组的名称，则该项目即为缺省项目。有关 `passwd` 文件的信息，请参见 `passwd(4)` 手册页。
4. 如果 `project` 数据库中存在特殊的项目 `default`，则此项目即为缺省项目。

此逻辑由 `getdefaultproj()` 库函数提供。有关更多信息，请参见 `getprojent(3PROJECT)` 手册页。

使用 useradd 和 usermod 命令设置用户属性

您可以使用以下带有 `-K` 选项和 *key=value* 对的命令在本地文件中设置用户属性：

`useradd` 设置用户的缺省项目

`usermod` 修改用户信息

本地文件可包括以下内容：

- `/etc/group`
- `/etc/passwd`

- /etc/project
- /etc/shadow
- /etc/user_attr

如果正在使用某一网络命名服务（如 NIS）为本地文件补充其他条目，则这些命令不能更改该网络命名服务提供的信息。但是，这些命令确实可以根据外部**命名服务数据库**验证以下内容：

- 用户名（或角色）的唯一性
- 用户 ID 的唯一性
- 是否存在任何指定的组名

有关更多信息，请参见 useradd(1M)、usermod(1M) 和 user_attr(4) 手册页。

project 数据库

您可以将项目数据存储在本地文件、域名系统 (Domain Name System, DNS)、网络信息服务 (Network Information Service, NIS) 项目图或轻量目录访问协议 (Lightweight Directory Access Protocol, LDAP) 目录服务中。/etc/project 文件或命名服务在登录时使用，由可插拔验证模块 (pluggable authentication module, PAM) 发出的所有帐户管理请求使用它将用户绑定到缺省项目。

注 – 对项目数据库中条目的更新，无论是对 /etc/project 文件还是对网络命名服务中数据库表示形式的更新，都不会应用于当前活动的项目。使用 login 或 newtask 命令时，会将更新应用于加入项目的新任务。有关更多信息，请参见 login(1) 和 newtask(1) 手册页。

PAM 子系统

更改或设置身份的操作包括登录到系统、调用 rcp 或 rsh 命令，以及使用 ftp 或使用 su。当操作涉及更改或设置身份时，会使用一组可配置的模块来提供验证、帐户管理、证书管理和会话管理。

有关 PAM 的概述，请参见《Oracle Solaris 11.1 管理：安全服务》中的第 14 章 “使用可插拔验证模块”。

命名服务配置

资源管理支持命名服务 project 数据库。/etc/nsswitch.conf 文件中定义了 project 数据库的存储位置。缺省情况下，会先列出 files，但是源可以按任意顺序列出。

```
project: files [nis] [ldap]
```

如果列出了多个项目信息源，则 nsswitch.conf 文件会指示例程开始在列出的第一个源中搜索信息，然后搜索后续源。

有关 /etc/nsswitch.conf 文件的更多信息，请参见《Oracle Solaris Administration: Naming and Directory Services》中的第 2 章 “Name Service Switch (Overview)”和 nsswitch.conf(4)。

本地 /etc/project 文件格式

如果在 nsswitch.conf 文件中选择 files 作为 project 数据库源，则登录过程会在 /etc/project 文件中搜索项目信息。有关更多信息，请参见 projects(1) 和 project(4) 手册页。

对于系统识别的每个项目，project 文件均包含以下形式的单行条目：

```
projname:projid:comment:user-list:group-list:attributes
```

字段定义如下：

projname 项目的名称。该名称必须是由字母数字字符、下划线 (_) 字符、连字符 (-) 和句点 (.) 组成的字符串。句点是为对操作系统有特殊意义的项目保留的，只能将其用在用户的缺省项目名称中。*projname* 不能包含冒号 (:) 或换行符。

projid 系统内项目的唯一数字 ID (PROJID)。*projid* 字段的最大值为 UID_MAX (2147483647)。

comment 项目的说明。

user-list 允许参与项目的用户的列表（以逗号分隔）。

此字段中可以使用通配符。星号 (*) 允许所有用户参与项目。感叹号后跟星号 (!*) 可将所有用户排除在项目之外。感叹号 (!)后跟用户名可将指定用户排除在项目之外。

group-list 允许参与项目的用户组的列表（以逗号分隔）。

此字段中可以使用通配符。星号 (*) 允许所有组参与项目。感叹号后跟星号 (!*) 可将所有组排除在项目之外。感叹号 (!)后跟组名可将指定组排除在项目之外。

attributes 用分号分隔的名称-值对列表，如资源控制（请参见第 6 章，资源控制（概述））。*name* 是指定与对象相关的属性的任意字符串，*value* 是该属性的可选的值。

```
name[=value]
```

在名称－值对中，名称仅可包含字母、数字、下划线和句点。句点通常用作资源控制 (rctl) 的类别和子类别之间的分隔符。属性名称的第一个字符必须是字母。名称区分大小写。

可以在值中使用逗号和括号结构以便确立优先级。

分号用于分隔名称－值对。不能在值定义中使用分号。冒号用于分隔项目字段。不能在值定义中使用冒号。

注 – 如果读取此文件的例程遇到格式错误的条目，则这些例程会停止。不会指定错误条目后指定的任何项目。

以下示例显示了缺省的 `/etc/project` 文件：

```
system:0::::
user.root:1::::
noproject:2::::
default:3::::
group.staff:10::::
```

以下示例显示了在结尾添加了项目条目的缺省的 `/etc/project` 文件：

```
system:0::::
user.root:1::::
noproject:2::::
default:3::::
group.staff:10::::
user.ml:2424:Lyle Personal:::
booksite:4113:Book Auction Project:ml,mp,jtd,kjh::
```

您还可以将资源控制和属性添加到 `/etc/project` 文件：

- 要为项目添加资源控制，请参见第 86 页中的 “设置资源控制”。
- 要使用 `rcapd(1M)` 中所述的资源上限设置守护进程为项目定义物理内存资源上限，请参见第 112 页中的 “限制项目物理内存使用情况的属性”。
- 要将 `project.pool` 属性添加到项目条目，请参见第 166 页中的 “创建配置”。

NIS 的项目配置

如果正在使用 NIS，则可以在 `/etc/nsswitch.conf` 文件中进行指定，以便在 NIS 项目映射中搜索项目：

```
project: nis files
```

NIS 映射（`project.byname` 或 `project.bynumber`）与 `/etc/project` 文件具有相同的形式：

```
projname:projid:comment:user-list:group-list:attributes
```

有关更多信息，请参见《Oracle Solaris Administration: Naming and Directory Services》中的第 5 章 “Network Information Service (Overview)”。

LDAP 的项目配置

如果正在使用 LDAP，则可以在 `/etc/nsswitch.conf` 文件中进行指定，以便在 LDAP `project` 数据库中搜索项目：

```
project: ldap files
```

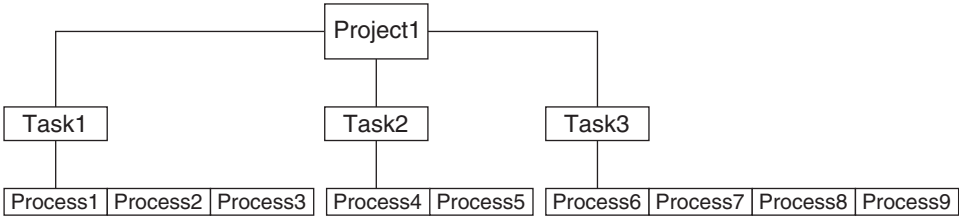
有关 LDAP 的更多信息，请参见《Oracle Solaris Administration: Naming and Directory Services》中的第 9 章 “Introduction to LDAP Naming Services (Overview)”。有关 LDAP 数据库中项目条目结构的更多信息，请参见《Oracle Solaris Administration: Naming and Directory Services》中的“Oracle Solaris Schemas”。

任务标识符

每次成功登录到项目时，都会创建一个包含登录过程的新**任务**。任务是指表示一段时间内一组工作的进程集。任务也可以视为**工作负荷组件**。会为每个任务自动指定一个任务 ID。

每个进程都是一个任务的成员，而每个任务都与一个项目关联。

图 2–1 项目和任务树



任务还支持对进程组执行的所有操作，如信号传送。您还可以将任务绑定到**处理器集**，并为任务设置调度优先级和类，优先级和类会修改任务中的所有当前进程以及后续进程。

每次加入项目时，就会创建任务。以下操作、命令和函数可创建任务：

- 登录
- `cron`

- newtask
- setproject
- su

您可以使用以下方法之一创建最终任务。所有进一步创建新任务的尝试都将失败。

- 可以使用带有 -F 选项的 newtask 命令。
- 可以在 project 命名服务数据库中为项目设置 task.final 属性。在此项目中，所有由 setproject 创建的任务都有 TASK_FINAL 标志。

有关更多信息，请参见 login(1)、newtask(1)、cron(1M)、su(1M) 和 setproject(3PROJECT) 手册页。

扩展记帐功能可以为进程提供记帐数据。此数据在任务级别聚合。

用于项目和任务的命令

下表中所示的命令提供了项目和任务功能的主要管理接口。

手册页参考	说明
projects(1)	显示用户的项目成员关系。列出 project 数据库中的项目。列显指定项目的信息。如果未提供项目名称，则显示所有项目的信息。使用带有 -l 选项的 projects 命令可列显详细输出。
newtask(1)	执行用户的缺省 shell 或指定命令，将执行命令放在指定项目拥有的新任务中。newtask 还可以用于为正在运行的进程更改任务和项目绑定。与 -F 选项一起使用，以创建最终任务。
projadd(1M)	将新项目条目添加到 /etc/project 文件。projadd 命令仅在本地系统上创建项目条目。projadd 不能更改网络命名服务提供的信息。 可用于编辑除缺省文件 /etc/project 之外的项目文件。为 project 文件提供语法检查。验证和编辑项目属性。支持标度值。
projmod(1M)	在本地系统上修改项目信息。projmod 不能更改网络命名服务提供的信息。但是，此命令确实可以根据外部命名服务验证项目名称和项目 ID 的唯一性。 可用于编辑除缺省文件 /etc/project 之外的项目文件。为 project 文件提供语法检查。验证和编辑项目属性。可用于添加新属性，向属性添加值或者删除属性。支持标度值。 与 -A 选项一起使用，可以将在项目数据库中找到的资源控制值应用于活动项目。将删除与 project 文件中定义的值不匹配的值。
projdel(1M)	从本地系统中删除项目。projdel 不能更改网络命名服务提供的信息。

手册页参考	说明
useradd(1M)	向本地文件添加缺省项目定义。与 -K *key=value* 选项一起使用，可添加或替换用户属性。
userdel(1M)	删除本地文件中的用户帐户。
usermod(1M)	修改系统上的用户登录信息。与 -K *key=value* 选项一起使用，可添加或替换用户属性。

◆ ◆ ◆ 第 3 章

管理项目和任务

本章介绍如何使用 Oracle Solaris 资源管理的项目和任务功能。

本章包含以下主题：

- 第 44 页中的“命令和命令选项示例”
- 第 46 页中的“管理项目”

有关项目和任务功能的概述，请参见第 2 章，项目和任务（概述）。

注 – 如果在安装了区域的 Oracle Solaris 系统上使用这些功能，则当这些功能命令在非全局区域 (non-global zone) 中运行时，只有同一区域中的进程才能通过使用进程 ID 的系统调用界面进行查看。

管理项目和任务（任务列表）

任务	说明	参考
查看用于项目与任务的命令和选项的示例。	显示任务和项目 ID，显示系统上当前所运行进程和项目的各种统计信息。	第 44 页中的“命令和命令选项示例”
定义项目。	在 `/etc/project` 文件中添加项目条目并修改此条目的值。	第 46 页中的“如何定义项目和查看当前项目”
删除项目。	从 `/etc/project` 文件中删除项目条目。	第 49 页中的“如何从 `/etc/project` 文件中删除项目”
验证 `project` 文件或项目数据库。	检查 `/etc/project` 文件的语法或根据外部命名服务验证项目名称和项目 ID 的唯一性。	第 49 页中的“如何验证 `/etc/project` 文件的内容”

任务	说明	参考
获取项目成员身份信息。	显示发出调用的进程的当前项目成员身份。	第 50 页中的 “如何获取项目成员身份信息”
创建新任务。	使用 newtask 命令在特定项目中创建一项新任务。	第 50 页中的 “如何创建新任务”
将正在运行的进程与不同的任务和项目进行关联。	将进程号与特定项目中的新任务 ID 进行关联。	第 50 页中的 “如何将正在运行的进程移至新任务”
添加并使用项目属性。	使用项目数据库管理命令添加、编辑、验证和删除项目属性。	第 51 页中的 “编辑和验证项目属性”

命令和命令选项示例

本节提供用于项目与任务的命令和选项的示例。

用于项目和任务的命令选项

ps 命令

使用带有 -o 选项的 ps 命令可显示任务和项目 ID。例如，要查看项目 ID，请键入以下内容：

```
# ps -o user,pid,uid,projid
USER PID   UID  PROJID
jtd  89430 124  4113
```

id 命令

使用带有 -p 选项的 id 命令可列显当前的项目 ID，以及用户和组 ID。如果提供了 *user* 操作数，则还会列显与该用户的正常登录相关联的项目：

```
#  id -p
uid=124(jtd) gid=10(staff) projid=4113(booksite)
```

pgrep 和 pkill 命令

要仅将进程与特定列表中的项目 ID 进行匹配，请使用带有 -J 选项的 pgrep 和 pkill 命令：

```
# pgrep -J projidlist
# pkill -J projidlist
```

要仅将进程与特定列表中的任务 ID 进行匹配，请使用带有 -T 选项的 pgrep 和 pkill 命令：

```
# pgrep -T taskidlist
# pkill -T taskidlist
```

prstat 命令

要显示系统上当前运行的进程和项目的各种统计信息，请使用带有 -J 选项的 prstat 命令：

```
% prstat -J
   PID USERNAME  SIZE   RSS STATE  PRI NICE      TIME  CPU PROCESS/NLWP
 12905 root     4472K 3640K cpu0    59    0   0:00:01 0.4% prstat/1
   829 root       43M   33M sleep   59    0   0:36:23 0.1% Xorg/1
   890 gdm        88M   26M sleep   59    0   0:22:22 0.0% gdm-simple-gree/1
   686 root     3584K 2756K sleep   59    0   0:00:34 0.0% automountd/4
     5 root       0K    0K sleep   99  -20   0:02:43 0.0% zpool-rpool/138
  9869 root       44M   17M sleep   59    0   0:02:06 0.0% poold/9
   804 root     7104K 5968K sleep   59    0   0:01:28 0.0% intrd/1
   445 root     7204K 4680K sleep   59    0   0:00:38 0.0% nscd/33
   881 gdm      7140K 5912K sleep   59    0   0:00:06 0.0% gconfd-2/1
   164 root     2572K 1648K sleep   59    0   0:00:00 0.0% pfexecd/3
   886 gdm      7092K 4920K sleep   59    0   0:00:00 0.0% bonobo-activati/2
    45 netcfg   2252K 1308K sleep   59    0   0:00:00 0.0% netcfgd/2
   142 daemon   7736K 5224K sleep   59    0   0:00:00 0.0% kcfd/3
    43 root     3036K 2020K sleep   59    0   0:00:00 0.0% dlmgmtd/5
   405 root     6824K 5400K sleep   59    0   0:00:18 0.0% hald/5
PROJID    NPROC  SWAP   RSS MEMORY      TIME  CPU PROJECT
     1        4 4728K   19M   0.9%   0:00:01 0.4% user.root
     0      111  278M  344M    17%   1:15:02 0.1% system
    10        2 1884K 9132K   0.4%   0:00:00 0.0% group.staff
     3        3 1668K 6680K   0.3%   0:00:00 0.0% default

Total: 120 processes, 733 lwps, load averages: 0.01, 0.00, 0.00
```

要显示系统上当前运行的进程和任务的各种统计信息，请使用带有 -T 选项的 prstat 命令：

```
% prstat -T
       PID USERNAME  SIZE   RSS STATE  PRI NICE      TIME  CPU PROCESS/NLWP
 12907 root     4488K 3588K cpu0    59    0   0:00:00 0.3% prstat/1
   829 root       43M   33M sleep   59    0   0:36:24 0.1% Xorg/1
   890 gdm        88M   26M sleep   59    0   0:22:22 0.0% gdm-simple-gree/1
  9869 root       44M   17M sleep   59    0   0:02:06 0.0% poold/9
     5 root       0K    0K sleep   99  -20   0:02:43 0.0% zpool-rpool/138
   445 root     7204K 4680K sleep   59    0   0:00:38 0.0% nscd/33
   881 gdm      7140K 5912K sleep   59    0   0:00:06 0.0% gconfd-2/1
   164 root     2572K 1648K sleep   59    0   0:00:00 0.0% pfexecd/3
   886 gdm      7092K 4920K sleep   59    0   0:00:00 0.0% bonobo-activati/2
    45 netcfg   2252K 1308K sleep   59    0   0:00:00 0.0% netcfgd/2
   142 daemon   7736K 5224K sleep   59    0   0:00:00 0.0% kcfd/3
    43 root     3036K 2020K sleep   59    0   0:00:00 0.0% dlmgmtd/5
   405 root     6824K 5400K sleep   59    0   0:00:18 0.0% hald/5
   311 root     3488K 2512K sleep   59    0   0:00:00 0.0% picld/4
   409 root     4356K 2768K sleep   59    0   0:00:00 0.0% hald-addon-cpuf/1
TASKID    NPROC  SWAP   RSS MEMORY      TIME  CPU PROJECT
  1401        2 2540K 8120K   0.4%   0:00:00 0.3% user.root
    94       15   84M  162M   7.9%   0:59:37 0.1% system
```

```
   561        1   37M   24M   1.2%   0:02:06 0.0% system
     0        2    0K    0K   0.0%   0:02:47 0.0% system
    46        1 4224K 5524K   0.3%   0:00:38 0.0% system
Total: 120 processes, 733 lwps, load averages: 0.01, 0.00, 0.00
```

注 – `-J` 和 `-T` 选项不能一起使用。

将 cron 和 su 用于项目和任务

cron 命令

`cron` 命令将发出 `settaskid`，以确保每个 `cron`、`at` 和 `batch` 作业都是在单独的任务中执行，并对提交用户使用了适当的缺省项目。`at` 和 `batch` 命令也会捕获当前项目 ID，以确保在运行 `at` 作业时恢复项目 ID。

su 命令

作为模拟登录的一部分，`su` 命令将通过创建新任务加入目标用户的缺省项目。

要使用 `su` 命令切换用户的缺省项目，请键入以下内容：

```
# su - user
```

管理项目

▼ 如何定义项目和查看当前项目

此示例说明如何使用 `projadd` 命令添加项目条目，以及如何使用 `projmod` 命令修改此条目。

1. **成为 root 用户或承担等效角色。**
2. **使用 `projects -l` 查看系统上缺省的 `/etc/project` 文件。**

```
# projects -l
system
        projid : 0
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
user.root
        projid : 1
        comment: ""
```

```
        users  : (none)
        groups : (none)
        attribs:
noproject
        projid : 2
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
default
        projid : 3
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
group.staff
        projid : 10
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
```

3 添加名为 *booksite* 的项目。将该项目指定给名为 *mark* 的用户，同时指定项目 ID 号 *4113*。

```
# projadd -U mark -p 4113 booksite
```

4 再次查看 **/etc/project** 文件。

```
# projects -l
system
        projid : 0
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
user.root
        projid : 1
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
noproject
        projid : 2
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
default
        projid : 3
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
group.staff
        projid : 10
        comment: ""
        users  : (none)
```

```
        groups : (none)
        attribs:
booksite
        projid : 4113
        comment: ""
        users  : mark
        groups : (none)
        attribs:
```

5 在注释字段中添加描述项目的注释。

```
# projmod -c 'Book Auction Project' booksite
```

6 查看 **/etc/project** 文件中的更改。

```
# projects -l
system
        projid : 0
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
user.root
        projid : 1
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
noproject
        projid : 2
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
default
        projid : 3
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
group.staff
        projid : 10
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
booksite
        projid : 4113
        comment: "Book Auction Project"
        users  : mark
        groups : (none)
        attribs:
```

另请参见 要将项目、任务和进程绑定到池，请参见第 162 页中的“设置池属性并绑定到池”。

▼ 如何从 /etc/project 文件中删除项目

此示例显示如何使用 projdel 命令删除项目。

1 **成为 root 用户或承担等效角色。**

2 **使用 projdel 命令删除 *booksite* 项目。**

```
# projdel booksite
```

3 **显示 /etc/project 文件。**

```
# projects -l
system
        projid : 0
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
user.root
        projid : 1
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
noproject
        projid : 2
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
default
        projid : 3
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
group.staff
        projid : 10
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
```

4 **以用户 *mark* 的身份登录并键入 projects 来查看指定给此用户的项目。**

```
# su - mark
# projects
default
```

如何验证 /etc/project 文件的内容

如果没有给出编辑选项，projmod 命令便会验证 project 文件的内容。

要验证 NIS 映射，请键入以下内容：

```
# ypcat project | projmod -f –
```

要检查 /etc/project 文件的语法，请键入以下内容：

```
# projmod -n
```

如何获取项目成员身份信息

使用带有 -p 标志的 id 命令显示发出调用的进程的当前项目成员身份。

```
$ id -p
uid=100(mark) gid=1(other) projid=3(default)
```

▼ 如何创建新任务

1. **以目标项目（本例中为 *booksite*）的成员身份登录。**
2. **使用带有 -v（详细）选项的 newtask 命令在 *booksite* 项目中创建新任务，以获取系统任务 ID。**

   ```
   machine% newtask -v -p booksite
   16
   ```

   通过执行 newtask，可以在指定项目中创建新任务并将用户的缺省 shell 置于此任务中。
3. **查看发出调用的进程的当前项目成员身份。**

   ```
   machine% id -p
   uid=100(mark) gid=1(other) projid=4113(booksite)
   ```

   现在该进程成为新项目的成员。

▼ 如何将正在运行的进程移至新任务

此示例显示如何将正在运行的进程与不同的任务和新项目进行关联。要执行该操作，必须是 root 用户、具有所需权限配置文件或者是进程的所有者并且是新项目的成员。

1. **成为 root 用户或承担等效角色。**

   注 – 如果您是进程所有者或新项目成员，则可以跳过此步骤。
2. **获取 *book_catalog* 进程的进程 ID。**

   ```
   # pgrep book_catalog
       8100
   ```

3 **将进程 *8100* 与 *booksite* 项目中的新任务 ID 进行关联。**

```
# newtask -v -p booksite -c 8100
    17
```

`-c` 选项指定 `newtask` 作用于现有的命名进程。

4 **确认任务到进程 ID 的映射。**

```
# pgrep -T 17
    8100
```

编辑和验证项目属性

您可以使用 `projadd` 和 `projmod` 项目数据库管理命令来编辑项目属性。

`-K` 选项指定属性替换列表。属性由分号 (`;`) 进行分隔。如果将 `-K` 选项和 `-a` 选项一起使用，则会添加属性或属性值。如果将 `-K` 选项和 `-r` 选项一起使用，则会删除属性或属性值。如果将 `-K` 选项与 `-s` 选项一起使用，则会替换属性或属性值。

▼ 如何将属性和属性值添加到项目

可以使用带有 `-a` 和 `-K` 选项的 `projmod` 命令将值添加到项目属性中。如果属性不存在，则会创建一个。

1 **成为 root 用户或承担等效角色。**

2 **在项目 `myproject` 中添加无属性值的资源控制属性 *task.max-lwps*。加入项目的任务只有系统属性值。**

```
# projmod -a -K task.max-lwps myproject
```

3 **然后可以为 `myproject` 项目中的 *task.max-lwps* 添加值。此值包含特权级别、阈值以及与达到阈值关联的操作。**

```
# projmod -a -K "task.max-lwps=(priv,100,deny)" myproject
```

4 **由于资源控制属性可以具有多个值，因此可以使用同一个选项将其他值添加到现有的值列表中。**

```
# projmod -a -K "task.max-lwps=(priv,1000,signal=KILL)" myproject
```

用逗号分隔多个值。现在 `task.max-lwps` 条目应为：

```
task.max-lwps=(priv,100,deny),(priv,1000,signal=KILL)
```

▼ 如何从项目中删除属性值

此过程使用以下值：

```
task.max-lwps=(priv,100,deny),(priv,1000,signal=KILL)
```

1. **成为 root 用户或承担等效角色。**
2. **要删除 *myproject* 项目中 `task.max-lwps` 资源控制的属性值，请使用带有 `-r` 和 `-K` 选项的 `projmod` 命令。**

   ```
   # projmod -r -K "task.max-lwps=(priv,100,deny)" myproject
   ```

   如果 `task.max-lwps` 具有多个值，例如：

   ```
   task.max-lwps=(priv,100,deny),(priv,1000,signal=KILL)
   ```

   则会删除第一个匹配的值。结果将为：

   ```
   task.max-lwps=(priv,1000,signal=KILL)
   ```

▼ 如何从项目中删除资源控制属性

要删除 *myproject* 项目中的 `task.max-lwps` 资源控制属性，请使用带有 `-r` 和 `-K` 选项的 `projmod` 命令。

1. **成为 root 用户或承担等效角色。**
2. **从项目 *myproject* 中删除属性 `task.max-lwps` 及其所有值：**

   ```
   # projmod -r -K task.max-lwps myproject
   ```

▼ 如何替换项目的属性和属性值

要替换项目 *myproject* 中属性 `task.max-lwps` 的值，请使用带有 `-s` 和 `-K` 选项的 `projmod` 命令。如果属性不存在，则会创建一个。

1. **成为 root 用户或承担等效角色。**
2. **使用所示的新值替换当前的 `task.max-lwps` 值：**

   ```
   # projmod -s -K "task.max-lwps=(priv,100,none),(priv,120,deny)" myproject
   ```

   结果为：

   ```
   task.max-lwps=(priv,100,none),(priv,120,deny)
   ```

▼ 如何删除资源控制属性的现有值

1. 成为 root 用户或承担等效角色。
2. 要从项目 *myproject* 中删除 **task.max-lwps** 的当前值，请键入：

```
# projmod -s -K task.max-lwps myproject
```

◆ ◆ ◆ 第 4 章

扩展记帐（概述）

通过使用第 2 章，项目和任务（概述）中介绍的项目和任务功能对工作负荷进行标记和分隔，可以监视每个工作负荷的资源占用情况。您可以使用**扩展记帐**子系统捕获一组有关进程和任务的详细资源占用情况的统计信息。

本章包含以下主题：

要开始使用扩展记帐，请跳至第 64 页中的“如何为流、进程、任务和网络组件激活扩展记帐”。

扩展记帐介绍

扩展记帐子系统记录执行工作的项目的资源使用情况。您还可以将扩展记帐与《在 Oracle Solaris 11.1 中管理 IP 服务质量》中的第 5 章“使用流记帐和统计信息收集功能（任务）”中所介绍的 Internet 协议服务质量 (Internet Protocol Quality of Service, IPQoS) 流记帐模块结合使用，以捕获系统上的网络流信息。

在应用资源管理机制之前，必须首先能够识别各种工作负荷对系统的资源占用需求。Oracle Solaris 操作系统中的扩展记帐功能提供了记录以下内容的系统和网络资源占用情况的灵活方式：

- 任务。
- 进程。
- IPQoS `flowacct` 模块提供的选择器。有关更多信息，请参见 `ipqos`(7IPP)。
- 网络管理。请参见`dladm`(1M) 和 `flowadm`(1M)。

与可实时度量系统使用情况的联机监视工具不同，通过扩展记帐，可检查历史使用情况。然后，可以对将来工作负荷的容量要求进行评估。

有了扩展记帐数据，便可以开发或购买用于资源费用分摊、工作负荷监视或容量规划的软件。

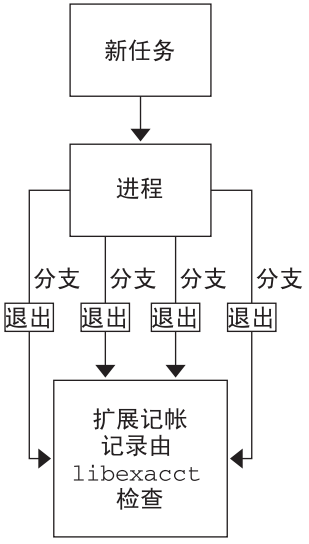
扩展记帐的工作原理

Oracle Solaris 操作系统中的扩展记帐功能使用一种版本化的可扩展文件格式来包含记帐数据。使用附带的库 `libexacct`（请参见 `libexacct(3LIB)`）中提供的 API，可以访问或创建采用此数据格式的文件。然后，可以在启用了扩展记帐的任何平台上分析这些文件，并且可以使用其数据进行容量规划和费用分摊。

如果扩展记帐处于活动状态，则会收集可由 `libexacct` API 检查的统计信息。使用 `libexacct` 可以向前或向后检查 `exacct` 文件。API 支持由 `libexacct` 生成的第三方文件以及由内核创建的文件。使用 `libexacct` 的实用摘录和报告语言 (Practical Extraction and Report Language, Perl) 接口，可以开发定制报告和摘录脚本。请参见第 59 页中的 “`libexacct` 的 Perl 接口”。

例如，如果启用了扩展记帐，则任务会跟踪其成员进程的总体资源使用情况。任务完成时会编写任务记帐记录，还会编写有关正在运行的进程和任务的临时记录。有关任务的更多信息，请参见第 2 章，项目和任务（概述）。

图 4–1 在激活了扩展记帐情况下的任务跟踪



可扩展的格式

扩展记帐格式的可扩展性要大大强于传统的系统记帐软件格式。扩展记帐允许在不同发行版的系统中添加和删除记帐度量标准，即使在系统操作过程中也是如此。

注 – 系统上的扩展记帐软件和传统系统记帐软件可以同时处于活动状态。

exacct 记录和格式

用于创建 exacct 记录的例程具有两个用途。

- 允许创建第三方 exacct 文件。
- 允许使用 putacct 系统调用（请参见 getacct(2)）创建嵌入到内核记帐文件中的标记记录。

注 – putacct 系统调用也可通过 Perl 接口使用。

此格式允许捕获不同形式的记帐记录，而不要求每次更改都是显式的版本更改。使用记帐数据且编写准确的应用程序必须忽略它们不了解的记录。

libexacct 库可转换和生成格式为 exacct 的文件。此库是 exacct 格式文件支持的**唯一**接口。

注 – getacct、putacct 和 wracct 系统调用不适用于流。配置 IPQoS 流记帐之后，内核便会创建流记录并将其写入文件。

在安装了区域的 Oracle Solaris 系统上使用扩展记帐

当扩展记帐子系统在全局区域中运行时，它会收集和报告整个系统（包括非全局区域）的信息。全局管理员或通过 zonecfg 实用程序获得了相应授权的用户也可以按区域确定资源的占用情况。有关更多信息，请参见第 327 页中的 “安装了区域的系统上的扩展记帐”。

扩展记帐配置

/var/adm/exacct 目录是放置扩展记帐数据的标准位置。您可以使用 acctadm 命令为进程和任务记帐数据文件指定其他位置。有关更多信息，请参见 acctadm(1M)。

启动及持久启用扩展记帐

acctadm(1M) 中介绍的 acctadm 命令通过 smf(5) 中介绍的 Oracle Solaris 服务管理工具 (Service Management Facility, SMF) 服务启动扩展记帐。

扩展记帐配置存储在 SMF 系统信息库中。该配置在引导时由一个服务实例恢复，每种记帐类型都对应一个服务实例。每个扩展记帐类型都由一个单独的 SMF 服务实例表示：

svc:/system/extended-accounting:flow
  流记帐

svc:/system/extended-accounting:process
  进程记帐

svc:/system/extended-accounting:task
  任务记帐

svc:/system/extended-accounting:net
  网络记帐

使用 acctadm(1M) 启用扩展记帐会导致启用对应的服务实例（如果其当前未启用），因此扩展记帐配置将在下次引导时恢复。同样，如果该配置导致对某个服务禁用记帐，将禁用服务实例。实例根据需要由 acctadm 启用或禁用。

要为资源持久激活扩展记帐，请运行：

```
# acctadm -e resource_list
```

resource_list 是资源或资源组的逗号分隔列表。

记录

acctadm 命令向 /var/adm/exacct 中的现有文件附加新记录。

用于扩展记帐的命令

命令参考	说明
acctadm(1M)	修改扩展记帐功能的各种属性，停止和启动扩展记帐，并用于针对进程、任务、流和网络选择要跟踪的记帐属性。
wracct(1M)	针对活动的进程和任务编写扩展记帐记录。
lastcomm(1)	显示以前调用的命令。lastcomm 既可以使用标准记帐进程数据，又可以使用扩展记帐进程数据。

有关与任务和项目相关联的命令的信息，请参见第 44 页中的 “命令和命令选项示例”。有关 IPQoS 流记帐的信息，请参见 ipqosconf(1M) 手册页和《在 Oracle Solaris 11.1 中管理 IP 服务质量》中的第 5 章 “使用流记帐和统计信息收集功能（任务）”。

libexacct 的 Perl 接口

通过 Perl 接口可以创建 Perl 脚本，该脚本可读取由 exacct 框架生成的记帐文件。您还可以创建编写 exacct 文件的 Perl 脚本。

此接口与底层 C API 在功能上是等效的。如果可能，通过底层 C API 获取的数据将显示为 Perl 数据类型。通过该接口可以更方便地访问数据，不再需要执行缓冲区压缩和解压缩操作。此外，所有内存管理均由 Perl 库执行。

各种与项目、任务和 exacct 相关的功能可分为多个组。每个功能组都位于单独的 Perl 模块中。每个模块都以 Oracle Solaris 标准的 Sun::Solaris:: Perl 软件包前缀开头。Perl exacct 库提供的所有类均位于 Sun::Solaris::Exacct 模块中。

底层 libexacct(3LIB) 库提供针对 exacct 格式文件、目录标记和 exacct 对象的操作。exacct 对象分为两种类型：

- 项，是指单一的数据值（标量）
- 组，是指项的列表

下表概述了每个模块。

模块（不应包含空格）	说明	更多信息
Sun::Solaris::Project	此模块提供了访问以下项目操作函数的功能：getprojid(2)、endprojent(3PROJECT)、fgetprojent(3PROJECT)、getdefaultproj(3PROJECT)、getprojbyid(3PROJECT)、getprojbyname(3PROJECT)、getprojent(3PROJECT)、getprojidbyname(3PROJECT)、inproj(3PROJECT)、project_walk(3PROJECT)、setproject(3PROJECT) 和 setprojent(3PROJECT)。	Project(3PERL)
Sun::Solaris::Task	此模块提供了访问以下任务操作函数的功能：gettaskid(2) 和 settaskid(2)。	Task(3PERL)
Sun::Solaris::Exacct	此模块是顶层 exacct 模块。此模块提供了访问与 exacct 相关的系统调用 getacct(2)、putacct(2) 和 wracct(2) 的功能。此模块还提供了访问 libexacct(3LIB) 库函数 ea_error(3EXACCT) 的功能。此模块同时也提供了所有 exacct EO_*、EW_*、EXR_*、P_* 和 TASK_* 宏的常量。	Exacct(3PERL)
Sun::Solaris::Exacct:: Catalog	此模块提供了面向对象的方法，以访问 exacct 目录标记中的位字段。此模块还提供了访问 EXC_*、EXD_* 和 EXD_* 宏常量的权限。	Exacct::Catalog(3PERL)
Sun::Solaris::Exacct:: File	此模块提供了面向对象的方法，以访问 libexacct 记帐文件函数 ea_open(3EXACCT)、ea_close(3EXACCT)、ea_get_creator(3EXACCT)、ea_get_hostname(3EXACCT)、ea_next_object(3EXACCT)、ea_previous_object(3EXACCT) 和 ea_write_object(3EXACCT)。	Exacct::File(3PERL)
Sun::Solaris::Exacct:: Object	此模块提供了面向对象的方法，以访问单个 exacct 记帐文件对象。exacct 对象表示为被指定隶属于相应 Sun::Solaris::Exacct::Object 子类的不透明参考。此模块分为项和组两种对象类型。在此级别上提供了访问 ea_match_object_catalog(3EXACCT) 和 ea_attach_to_object(3EXACCT) 函数的方法。	Exacct::Object(3PERL)

模块（不应包含空格）	说明	更多信息
Sun::Solaris::Exacct::Object::Item	此模块提供了面向对象的方法，以访问单个 exacct 记帐文件项。此类型的对象从 Sun::Solaris::Exacct::Object 中继承。	Exacct::Object::Item(3PERL)
Sun::Solaris::Exacct::Object::Group	此模块提供了面向对象的方法，以访问单个 exacct 记帐文件组。此类型的对象从 Sun::Solaris::Exacct::Object 中继承。这些对象提供了对 ea_attach_to_group(3EXACCT) 函数的访问。组中包含的各项表示为 Perl 数组。	Exacct::Object::Group(3PERL)
Sun::Solaris::Kstat	此模块提供了 kstat 功能的与 Perl 关联的散列接口。/bin/kstat 提供了此模块的使用示例，此示例采用 Perl 编写。	Kstat(3PERL)

有关说明如何使用上表中介绍的模块的示例，请参见第 66 页中的 “使用 libexacct 的 Perl 接口”。

◆ ◆ ◆ 第 5 章

管理扩展记帐（任务）

本章介绍如何管理扩展记帐子系统。

有关扩展记帐子系统的概述，请参见第 4 章，扩展记帐（概述）。

管理扩展记帐功能（任务列表）

任务	说明	参考
激活扩展记帐功能。	使用扩展记帐监视系统上运行的每个项目的资源消耗情况。可以使用**扩展记帐**子系统捕获任务、进程和流的历史数据。	第 64 页中的“如何为流、进程、任务和网络组件激活扩展记帐”
显示扩展记帐状态。	确定扩展记帐功能的状态。	第 64 页中的“如何显示扩展记帐状态”
查看可用的记帐资源。	查看系统上的可用记帐资源。	第 65 页中的“如何查看可用的记帐资源”
取消激活流、进程、任务和网络记帐实例。	禁用扩展记帐功能。	第 66 页中的“如何取消激活进程、任务、流和网络管理记帐”
将 Perl 接口用于扩展记帐功能。	使用 Perl 接口开发定制报告脚本和提取脚本。	第 66 页中的“使用 `libexacct` 的 Perl 接口”

使用扩展记帐功能

如果用户拥有所要管理的记帐类型的相应权限配置文件，则可以管理扩展记帐（启动记帐、停止记帐和更改记帐配置参数）：

- 扩展记帐流管理
- 进程管理
- 任务管理
- 网络管理

▼ 如何为流、进程、任务和网络组件激活扩展记帐

要为任务、进程、流和网络组件激活扩展记帐功能，请使用 `acctadm` 命令。`acctadm` 的可选的最终参数表示此命令是应该针对扩展记帐功能的流记帐组件、进程记帐组件、系统任务记帐组件还是网络记帐组件执行。

注 – 角色包含授权和具有特权的命令。有关如何创建角色以及通过 Oracle Solaris 的基于角色的访问控制 (role-based access control, RBAC) 功能将角色分配给用户的信息，请参见《Oracle Solaris 11.1 管理：安全服务》中的第 III 部分, “角色、权限配置文件和特权”。

---

1. **成为 root 用户或承担等效角色。**
2. **激活进程的扩展记帐。**

   ```
   # acctadm -e extended -f /var/adm/exacct/proc process
   ```

3. **激活任务的扩展记帐。**

   ```
   # acctadm -e extended,mstate -f /var/adm/exacct/task task
   ```

4. **激活流的扩展记帐。**

   ```
   # acctadm -e extended -f /var/adm/exacct/flow flow
   ```

5. **激活网络的扩展记帐。**

   ```
   # acctadm -e extended -f /var/adm/exacct/net net
   ```

   对 `dladm` 和 `flowadm` 命令所管理的链接和流运行 `acctadm`。

另请参见 有关更多信息，请参见 `acctadm(1M)`。

如何显示扩展记帐状态

键入不带参数的 `acctadm` 可以显示扩展记帐功能的当前状态。

```
machine% acctadm
                Task accounting: active
           Task accounting file: /var/adm/exacct/task
         Tracked task resources: extended
       Untracked task resources: none
             Process accounting: active
        Process accounting file: /var/adm/exacct/proc
      Tracked process resources: extended
    Untracked process resources: host
                Flow accounting: active
           Flow accounting file: /var/adm/exacct/flow
         Tracked flow resources: extended
       Untracked flow resources: none
```

在前一示例中，系统任务记帐在扩展模式和 mstate 模式下激活。进程记帐和流记帐在扩展模式下激活。

注 – 在扩展记帐的上下文中，微状态 (mstate) 是指与微状态进程转换关联的扩展数据，可从进程使用情况文件（请参见 proc(4)）中获取此数据。与基本记录或扩展记录相比，此数据可提供有关进程活动的更多详细信息。

如何查看可用的记帐资源

可用的资源随系统和平台的不同而有所不同。使用带有 -r 选项的 acctadm 命令可以查看系统上的可用记帐资源组。

```
machine% acctadm -r
process:
extended pid,uid,gid,cpu,time,command,tty,projid,taskid,ancpid,wait-status,zone,flag,
memory,mstate     displays as one line
basic    pid,uid,gid,cpu,time,command,tty,flag
task:
extended taskid,projid,cpu,time,host,mstate,anctaskid,zone
basic    taskid,projid,cpu,time
flow:
extended
saddr,daddr,sport,dport,proto,dsfield,nbytes,npkts,action,ctime,lseen,projid,uid
basic    saddr,daddr,sport,dport,proto,nbytes,npkts,action
net:
  extended name,devname,edest,vlan_tpid,vlan_tci,sap,cpuid, \
  priority,bwlimit,curtime,ibytes,obytes,ipkts,opks,ierrpkts \
  oerrpkts,saddr,daddr,sport,dport,protocol,dsfield
  basic    name,devname,edest,vlan_tpid,vlan_tci,sap,cpuid, \
  priority,bwlimit,curtime,ibytes,obytes,ipkts,opks,ierrpkts \
  oerrpkts
```

▼ 如何取消激活进程、任务、流和网络管理记帐

要取消激活进程记帐、任务记帐、流记帐和网络记帐，请使用带有 x 选项的 -acctadm 命令分别禁用每个记帐。

1. **成为 root 用户或承担等效角色。**
2. **禁用进程记帐。**

   ```
   # acctadm -x process
   ```

3. **禁用任务记帐。**

   ```
   # acctadm -x task
   ```

4. **禁用流记帐。**

   ```
   # acctadm -x flow
   ```

5. **禁用网络管理记帐。**

   ```
   # acctadm -x net
   ```

6. **检验是否已禁用任务记帐、进程记帐、流记帐和网络记帐。**

   ```
       # acctadm
               Task accounting: inactive
          Task accounting file: none
        Tracked task resources: none
      Untracked task resources: extended
            Process accounting: inactive
       Process accounting file: none
     Tracked process resources: none
   Untracked process resources: extended
               Flow accounting: inactive
          Flow accounting file: none
        Tracked flow resources: none
      Untracked flow resources: extended
               Net accounting: inactive
          Net accounting file: none
        Tracked Net resources: none
      Untracked Net resources: extended
   ```

使用 libexacct 的 Perl 接口

如何递归列显 exacct 对象的内容

使用以下代码可以递归列显 exacct 对象的内容。请注意，此功能作为 Sun::Solaris::Exacct::Object::dump() 函数由库提供。此功能还可以通过 ea_dump_object() 公用函数提供。

```
sub dump_object
     {
             my ($obj, $indent) = @_;
             my $istr = '  ' x $indent;

             #
             # Retrieve the catalog tag.  Because we are
             # doing this in an array context, the
             # catalog tag will be returned as a (type, catalog, id)
             # triplet, where each member of the triplet will behave as
             # an integer or a string, depending on context.
             # If instead this next line provided a scalar context, e.g.
             #    my $cat  = $obj->catalog()->value();
             # then $cat would be set to the integer value of the
             # catalog tag.
             #
             my @cat = $obj->catalog()->value();

             #
             # If the object is a plain item
             #
             if ($obj->type() == &EO_ITEM) {
                     #
                     # Note: The '%s' formats provide s string context, so
                     # the components of the catalog tag will be displayed
                     # as the symbolic values. If we changed the '%s'
                     # formats to '%d', the numeric value of the components
                     # would be displayed.
                     #
                     printf("%sITEM\n%s  Catalog = %s|%s|%s\n",
                        $istr, $istr, @cat);
                     $indent++;

                     #
                     # Retrieve the value of the item.  If the item contains
                     # in turn a nested exacct object (i.e., an item or
                     # group),then the value method will return a reference
                     # to the appropriate sort of perl object
                     # (Exacct::Object::Item or Exacct::Object::Group).
                     # We could of course figure out that the item contained
                     # a nested item orgroup by examining the catalog tag in
                     # @cat and looking for a type of EXT_EXACCT_OBJECT or
                     # EXT_GROUP.
                     #
                     my $val = $obj->value();
                     if (ref($val)) {
                             # If it is a nested object, recurse to dump it.
                             dump_object($val, $indent);
                     } else {
                             # Otherwise it is just a 'plain' value, so
                             # display it.
                             printf("%s  Value = %s\n", $istr, $val);
                     }

             #
             # Otherwise we know we are dealing with a group.  Groups
             # represent contents as a perl list or array (depending on
             # context), so we can process the contents of the group
             # with a 'foreach' loop, which provides a list context.
```

```
                # In a list context the value method returns the content
                # of the group as a perl list, which is the quickest
                # mechanism, but doesn't allow the group to be modified.
                # If we wanted to modify the contents of the group we could
                # do so like this:
                #    my $grp = $obj->value();   # Returns an array reference
                #    $grp->[0] = $newitem;
                # but accessing the group elements this way is much slower.
                #
                } else {
                        printf("%sGROUP\n%s  Catalog = %s|%s|%s\n",
                            $istr, $istr, @cat);
                        $indent++;
                        # 'foreach' provides a list context.
                        foreach my $val ($obj->value()) {
                                dump_object($val, $indent);
                        }
                        printf("%sENDGROUP\n", $istr);
                }
        }
```

如何创建新的组记录并将其写入文件

使用以下脚本可以创建新的组记录并将其写入名为 /tmp/exacct 的文件。

```
#!/usr/bin/perl

use strict;
use warnings;
use Sun::Solaris::Exacct qw(:EXACCT_ALL);
# Prototype list of catalog tags and values.
     my @items = (
             [ &EXT_STRING | &EXC_DEFAULT | &EXD_CREATOR      => "me"       ],
             [ &EXT_UINT32 | &EXC_DEFAULT | &EXD_PROC_PID     => $$         ],
             [ &EXT_UINT32 | &EXC_DEFAULT | &EXD_PROC_UID     => $<         ],
             [ &EXT_UINT32 | &EXC_DEFAULT | &EXD_PROC_GID     => $(         ],
             [ &EXT_STRING | &EXC_DEFAULT | &EXD_PROC_COMMAND => "/bin/rec" ],
     );

     # Create a new group catalog object.
     my $cat = ea_new_catalog(&EXT_GROUP | &EXC_DEFAULT | &EXD_NONE)

     # Create a new Group object and retrieve its data array.
     my $group = ea_new_group($cat);
     my $ary = $group->value();

     # Push the new Items onto the Group array.
     foreach my $v (@items) {
             push(@$ary, ea_new_item(ea_new_catalog($v->[0]), $v->[1]));
     }

     # Open the exacct file, write the record & close.
     my $f = ea_new_file('/tmp/exacct', &O_RDWR | &O_CREAT | &O_TRUNC)
        || die("create /tmp/exacct failed: ", ea_error_str(), "\n");
     $f->write($group);
     $f = undef;
```

如何列显 exacct 文件的内容

使用以下 Perl 脚本可以列显 exacct 文件的内容。

```
#!/usr/bin/perl

     use strict;
     use warnings;
     use Sun::Solaris::Exacct qw(:EXACCT_ALL);

     die("Usage is dumpexacct <exacct file>\n") unless (@ARGV == 1);

     # Open the exacct file and display the header information.
     my $ef = ea_new_file($ARGV[0], &O_RDONLY) || die(error_str());
     printf("Creator:  %s\n", $ef->creator());
     printf("Hostname: %s\n\n", $ef->hostname());

     # Dump the file contents
     while (my $obj = $ef->get()) {
             ea_dump_object($obj);
     }

     # Report any errors
     if (ea_error() != EXR_OK && ea_error() != EXR_EOF)  {
             printf("\nERROR: %s\n", ea_error_str());
             exit(1);
     }
     exit(0);
```

Sun::Solaris::Exacct::Object->dump() 的输出示例

以下是对在第 68 页中的 “如何创建新的组记录并将其写入文件”中创建的文件运行 `Sun::Solaris::Exacct::Object->dump()` 时生成的输出示例。

```
Creator:  root
Hostname: localhost
GROUP
       Catalog = EXT_GROUP|EXC_DEFAULT|EXD_NONE
       ITEM
         Catalog = EXT_STRING|EXC_DEFAULT|EXD_CREATOR
         Value = me
       ITEM
         Catalog = EXT_UINT32|EXC_DEFAULT|EXD_PROC_PID
         Value = 845523
       ITEM
         Catalog = EXT_UINT32|EXC_DEFAULT|EXD_PROC_UID
         Value = 37845
       ITEM
         Catalog = EXT_UINT32|EXC_DEFAULT|EXD_PROC_GID
         Value = 10
       ITEM
         Catalog = EXT_STRING|EXC_DEFAULT|EXD_PROC_COMMAND
         Value = /bin/rec
ENDGROUP
```

◆ ◆ ◆ 第 6 章

6 资源控制（概述）

按照第 4 章，扩展记帐（概述）中所述确定系统上工作负荷的资源消耗情况之后，便可对资源的使用情况设定限制。这些限制可防止工作负荷过度消耗资源。**资源控制**功能是用于此用途的约束机制。

本章包含以下主题：

有关如何管理资源控制的信息，请参见第 7 章，管理资源控制（任务）。

资源控制概念

在 Oracle Solaris 操作系统中，每进程资源限制的概念已扩展到第 2 章，项目和任务（概述）中所述的任务和项目实体。这些增强功能由资源控制 (resource control, rctl) 功能提供。此外，通过 `/etc/system` 可调参数设置的分配现在可以自动配置，也可以借助资源控制机制来配置。

资源控制由前缀 `zone`、`project`、`task` 或 `process` 标识。可以查看系统范围的资源控制。可以在正在运行的系统上更新资源控制值。

有关此发行版中提供的标准资源控制的列表，请参见第 73 页中的“可用的资源控制”。有关可用的区域范围的资源控制的信息，请参见第 214 页中的“资源类型属性”。

资源限制和资源控制

UNIX 系统一直以来都提供资源限制功能 (*rlimit*)。使用 rlimit 功能，管理员可以对进程可占用的资源设置一个或多个数值限制。这些限制包括每个进程使用的 CPU 时间、每进程核心文件大小以及每个进程的最大堆大小。**堆大小**是指为进程数据段分配的临时内存量。

资源控制功能提供了用于资源限制功能的兼容性接口。使用资源限制的现有应用程序将继续运行，不会更改。这些应用程序的观察方法，与修改之后可利用资源控制功能的应用程序的观察方法相同。

进程间通信和资源控制

使用几种进程间通信 (interprocess communication, IPC) 之一，进程可以相互通信。使用 IPC，可以在进程之间传输和同步信息。资源控制功能提供了可定义内核的 IPC 功能行为的资源控制。这些资源控制将替换 `/etc/system` 可调参数。

在本 Oracle Solaris 系统中的 `/etc/system` 文件内，可能包含用于初始化缺省资源控制值的已过时的参数。但是，不推荐使用过时参数。

要查看哪些 IPC 对象在使用项目资源，请使用带有 `-J` 选项的 `ipcs` 命令。要查看示例显示，请参见第 93 页中的 “如何使用 ipcs”。有关 `ipcs` 命令的更多信息，请参见 `ipcs(1)`。

有关 Oracle Solaris 系统调优的信息，请参见《Oracle Solaris 11.1 可调参数参考手册》。

资源控制约束机制

资源控制提供了一种系统资源约束机制，可以防止进程、任务、项目和区域占用指定的系统资源量。此机制通过防止占用过多的资源，可使系统更易于管理。

约束机制可用于支持容量规划过程。有一种偶尔会用到的约束，它可以提供有关应用程序资源需求的信息，而不必拒绝为应用程序分配的资源。

项目属性机制

资源控制还可以作为资源管理功能的简单属性机制。例如，可用于公平份额调度器 (fair share scheduler, FSS) 调度类中项目的 CPU 份额数由 `project.cpu-shares` 资源控制定义。由于此控制为项目指定了固定的份额数，因此，与超过控制有关的各项操作不相关联。在此上下文中，将 `project.cpu-shares` 控制的当前值视为指定项目的属性。

另一类型的项目属性用于控制附加到项目的进程集合对物理内存资源的消耗。这些属性具有前缀 rcap，例如 rcap.max-rss。与资源控制类似，此类型的属性也在 project 数据库中配置。但是，资源控制由内核同步执行，而资源上限则由资源上限执行守护进程 rcapd 在用户级别上异步执行。有关 rcapd 的信息，请参见第 10 章，使用资源上限设置守护进程控制物理内存（概述）和 rcapd (1M)。

project.pool 属性用于指定项目的池绑定。有关资源池的更多信息，请参见第 12 章，资源池（概述）。

配置资源控制和属性

通过 project 数据库配置资源控制功能。请参见第 2 章，项目和任务（概述）。资源控制和其他属性在 project 数据库条目的最终字段中设置。与每个资源控制关联的值都括在括号中，并显示为用逗号分隔的纯文本。括号中的值构成一条“操作子句”。每条操作子句都包含一个特权级别、一个阈值以及一个与特定阈值关联的操作。每个资源控制可以有多条操作子句，这些子句也用逗号分隔。以下条目定义了项目实体的按任务轻量进程限制和按进程最多 CPU 时间限制。当进程运行 1 小时之后，process.max-cpu-time 将会向此进程发送 SIGTERM；如果此进程持续运行的总时间达到 1 小时 1 分钟，则会向此进程发送 SIGKILL。请参见表 6–3。

```
development:101:Developers:::task.max-lwps=(privileged,10,deny);
  process.max-cpu-time=(basic,3600,signal=TERM),(priv,3660,signal=KILL)
    typed as one line
```

注 – 在启用了区域的系统上，使用稍有不同的格式在区域配置中指定整个区域范围的资源控制。有关更多信息，请参见第 210 页中的 “区域配置数据”。

使用 rctladm 命令，可以对**全局范围**的资源控制功能进行运行时询问和修改。使用 prctl 命令，可以对**本地范围**的资源控制功能进行运行时询问和修改。

有关更多信息，请参见第 79 页中的 “针对资源控制值的全局和本地操作”、rctladm(1M) 和 prctl(1)。

注 – 在安装了区域的系统上，不能在非全局区域中使用 rctladm 来修改设置。您可以在非全局区域中使用 rctladm 来查看每个资源控制的全局日志状态。

可用的资源控制

下表列出了此发行版中可用的标准资源控制。

该表介绍了每个控制所约束的资源，还列出了 project 数据库使用的该资源的缺省单位。缺省单位有两种类型：

- 数量代表有限数量。
- 索引代表最大有效标识符。

因此，`project.cpu-shares` 指定了项目有资格享有的份额数。`process.max-file-descriptor` 指定了可由 open(2) 系统调用指定给进程的最高文件编号。

表 6–1 标准项目、任务和进程资源控制

控制名称	说明	缺省单位
`project.cpu-cap`	项目可以占用的 CPU 资源量的绝对限制。值 `100` 表示将一个 CPU 的 100% 用作 `project.cpu-cap` 设置。值 125 表示 125%，因为在使用 CPU 上限时，100% 对应于系统中的一个 CPU。	数量（CPU 数目）
`project.cpu-shares`	授予此项目的 CPU 份额数，用于公平份额调度器（请参见 FSS(7)）。	数量（份额）
`project.max-crypto-memory`	`libpkcs11` 用于加速硬件加密的内核内存总量。内核缓冲区分配以及与会话相关的结构分配都按照此资源控制执行。	大小（字节）
`project.max-locked-memory`	允许的锁定物理内存总量。 如果将 `priv_proc_lock_memory` 指定给用户，请考虑同时设置此资源控制，以防止该用户锁定所有内存。 注意，该资源控制取代了已删除的 `project.max-device-locked-memory`。	大小（字节）
`project.max-msg-ids`	此项目允许的最大消息队列 ID 数。	数量（消息队列 ID）
`project.max-port-ids`	允许的最大事件端口数。	数量（事件端口数）
project.max-processes	此项目可同时使用的最大进程表槽数。 注意，由于常规进程和僵进程都使用进程表槽，因此 `max-processes` 控制可以防止僵进程用尽进程表。根据定义，由于僵进程没有任何 LWP（轻量级进程），`max-lwps` 控制无法防止这种可能性。	数量（进程表槽数）
`project.max-sem-ids`	此项目允许的最大信号 ID 数。	数量（信号量 ID）
`project.max-shm-ids`	此项目允许的最大共享内存 ID 数。	数量（共享内存 ID）

表 6–1 标准项目、任务和进程资源控制 （续）

控制名称	说明	缺省单位
project.max-shm-memory	此项目允许的 System V 共享内存总量。	大小（字节）
project.max-lwps	此项目可同时使用的最大 LWP 数。	数量 (LWP)
project.max-tasks	此项目中允许的最大任务数。	数量（任务数）
project.max-contracts	此项目中允许的最大合同数。	数量（合同）
task.max-cpu-time	此任务进程可用的最多 CPU 时间。	时间（秒）
task.max-lwps	此任务的进程可同时使用的最大 LWP 数。	数量 (LWP)
task.max-processes	此任务的进程可同时使用的最大进程表槽数。	数量（进程表槽数）
process.max-cpu-time	此进程可用的最长 CPU 时间。	时间（秒）
process.max-file-descriptor	此进程可用的最大文件描述符索引。	索引（最大文件描述符）
process.max-file-size	此进程可写入的最大文件偏移。	大小（字节）
process.max-core-size	此进程创建的最大核心文件大小。	大小（字节）
process.max-data-size	此进程可用的最大堆栈缓冲池内存。	大小（字节）
process.max-stack-size	此进程可用的最大堆栈缓冲池内存段。	大小（字节）
process.max-address-space	此进程可用的最大地址空间量，即段大小的总和。	大小（字节）
process.max-port-events	每事件端口允许的最大事件数。	数量（事件数）
process.max-sem-nsems	每信号集允许的最大信息数。	数量（每集合中的信号数）
process.max-sem-ops	每 semop 调用允许的最大信号操作数（在 semget() 时间从资源控制复制的值）。	数量（操作数）
process.max-msg-qbytes	消息队列中消息的最大字节数（在 msgget() 时间从资源控制复制的值）。	大小（字节）
process.max-msg-messages	消息队列中的最大消息数（在 msgget() 时间从资源控制复制的值）。	数量（消息数）

您可以在未设置或更改任何资源控制的系统上显示资源控制的缺省值。此类系统在 `/etc/system` 或 `project` 数据库中不包含任何非缺省条目。要显示值，请使用 `prctl` 命令。

区域范围的资源控制

区域范围的资源控制可限制区域内所有进程实体总的资源使用情况。也可以使用全局属性名称来设置区域范围的资源控制，如第 204 页中的 “设置区域范围的资源控制”和第 232 页中的 “如何配置区域”中所述。

表 6–2 区域资源控制

控制名称	说明	缺省单位
`zone.cpu-cap`	非全局区域可以占用的 CPU 资源量的绝对限制。 值 `100` 表示将一个 CPU 的 100% 用作 `project.cpu-cap` 设置。值 `125` 表示 125%，因为在使用 CPU 上限时，100% 对应于系统中的一个 CPU。	数量（CPU 数目）
`zone.cpu-shares`	此区域的公平份额调度器 (fair share scheduler, FSS) CPU 份额数	数量（份额）
`zone.max-lofi`	可以由区域创建的 `lofi` 设备的最大数量。 该值限制每个区域对次要节点名称空间的使用。	数量（`lofi` 设备的数量）
`zone.max-locked-memory`	区域可用的锁定物理内存的总量 在将 `priv_proc_lock_memory` 指定给区域时，请考虑同时设置此资源控制，以防止该区域锁定所有内存。	大小（字节）
`zone.max-lwps`	此区域可同时使用的最大 LWP 数	数量 (LWP)
`zone.max-msg-ids`	此区域允许的最大消息队列 ID 数	数量（消息队列 ID）
zone.max-processes	此区域可同时使用的最大进程表槽数。 由于常规进程和僵进程都使用进程表槽，因此 `max-processes` 控制可以防止僵进程用尽进程表。根据定义，由于僵进程没有任何 LWP（轻量级进程），`max-lwps` 控制无法防止这种可能性。	数量（进程表槽数）

表 6–2 区域资源控制 （续）

控制名称	说明	缺省单位
zone.max-sem-ids	此区域允许的最大信号量 ID 数	数量（信号量 ID）
zone.max-shm-ids	此区域允许的最大共享内存 ID 数	数量（共享内存 ID）
zone.max-shm-memory	此区域允许的系统 V 共享内存总量	大小（字节）
zone.max-swap	可用于此区域的用户进程地址空间映射和 tmpfs 挂载的交换空间总量	大小（字节）

有关配置区域范围的资源控制的信息，请参见第 214 页中的 “资源类型属性”和第 232 页中的 “如何配置区域”。

请注意，可将区域范围的资源控制应用于全局区域。有关其他信息，请参见第 356 页中的 “在安装了区域的 Oracle Solaris 系统上使用公平份额调度器”。

单位支持

所有资源控制均定义了标识资源控制类型的全局标志。系统使用这些标志将基本类型信息传递给应用程序（如 prctl 命令）。应用程序使用此信息确定以下内容：

- 适用于每个资源控制的单位字符串
- 解释标度值时要使用的正确标度

以下全局标志均可用：

全局标志	资源控制类型字符串	修饰符	标度
RCTL_GLOBAL_BYTES	bytes	B	1
		KB	2^{10}
		MB	2^{20}
		GB	2^{30}
		TB	2^{40}
		PB	2^{50}
		EB	2^{60}

全局标志	资源控制类型字符串	修饰符	标度
RCTL_GLOBAL_SECONDS	seconds	s	1
		Ks	10^3
		Ms	10^6
		Gs	10^9
		Ts	10^{12}
		Ps	10^{15}
		Es	10^{18}
RCTL_GLOBAL_COUNT	count	none	1
		K	10^3
		M	10^6
		G	10^9
		T	10^{12}
		P	10^{15}
		E	10^{18}

标度值可用于资源控制。以下示例显示了标度阈值：

```
task.max-lwps=(priv,1K,deny)
```

注 – 单位修饰符由 `prctl`、`projadd` 和 `projmod` 命令接受。您不能在 `project` 数据库本身中使用单位修饰符。

资源控制值和特权级别

资源控制的阈值设立了一个执行点，在此点可能会触发本地操作或者发生全局操作（如日志记录）。

资源控制的每个阈值都必须与某个特权级别相关联。特权级别必须为以下三种类型之一。

- 基本，此类型的权限级别可由调用过程的所有者修改
- 特权，此类型的特权级别仅可由特权 (root) 调用者修改
- 系统，此类型的权限级别在操作系统实例的持续时间内固定不变

每个资源控制都保证有一个由系统或资源提供器定义的系统值。系统值表示操作系统的当前实现可以提供的资源量。

可以定义任意数量的特权值，但仅允许定义一个基本值。缺省情况下，将为没有指定特权值时执行的操作指定基本特权。

资源控制值的特权级别在资源控制块（如 RCTL_BASIC、RCTL_PRIVILEGED 或 RCTL_SYSTEM）的特权字段中定义。有关更多信息，请参见 setrctl(2)。您可以使用 prctl 命令来修改与基本级别和特权级别关联的值。

针对资源控制值的全局和本地操作

针对资源控制值可执行两种类别的操作： 全局操作和本地操作。

针对资源控制值的全局操作

全局操作应用于系统中每个资源控制的资源控制值。您可以使用 rctladm(1M) 手册页中所述的 rctladm 命令来执行以下操作：

- 显示活动系统资源控制的全局状态
- 设置全局日志操作

您可以对资源控制禁用或启用全局日志操作。通过指定严重性级别，您可以将 syslog 操作设置为特定的级别 syslog=*level*。*level* 的可能设置如下：

- debug
- info
- notice
- warning
- err
- crit
- alert
- emerg

缺省情况下，没有资源控制违规的全局日志。级别 n/a 指示无法对其配置全局操作的资源控制。

针对资源控制值的本地操作

本地操作对试图超过控制值的进程执行。对于为资源控制设定的每个阈值，您都可以关联一个或多个操作。有三种类型的本地操作： none、deny 和 signal=。这三种操作按以下方式使用：

none　对于请求数量大于阈值的资源请求不执行任何操作。在不影响应用程序进度的情况下监视资源的使用情况时，此操作非常有用。虽然超过阈值的进程不会受到影响，但是您还可以启用在超过资源控制时显示的全局消息。

deny　您可以拒绝请求数量大于阈值的资源请求。例如，如果新的进程超过控制值，则带有操作 deny 的 task.max-lwps 资源控制会导致 fork 系统调用失败。请参见 fork(2) 手册页。

`signal=` 您可以在超过资源控制时启用全局信号消息操作。当超过阈值时，会向进程发送信号。如果进程占用了其他资源，则不会发送其他信号。表 6–3 中列出了可用的信号。

并非所有的操作都可应用于每个资源控制。例如，某个进程的 CPU 份额数不能超过为其所属的项目指定的 CPU 份额数。因此，不允许对 `project.cpu-shares` 资源控制执行拒绝操作。

由于存在实现限制，因此，每个控制的全局属性可以限制可对阈值设置的可用操作的范围。（请参见 `rctladm(1M)` 手册页。）下表列出了可用信号操作。有关信号的其他信息，请参见 `signal(3HEAD)` 手册页。

表 6–3 可用于资源控制值的信号

信号	说明	附注
SIGABRT	终止进程。	
SIGHUP	发送挂起信号。当载波在断开的线路上停止时出现。发送给控制终端的进程组的信号。	
SIGTERM	终止进程。由软件发送的终止信号。	
SIGKILL	终止进程并中止程序。	
SIGSTOP	停止进程。作业控制信号。	
SIGXRES	超过了资源控制限制。由资源控制功能生成。	
SIGXFSZ	终止进程。超过了文件大小限制。	仅可用于具有 RCTL_GLOBAL_FILE_SIZE 属性的资源控制 (`process.max-file-size`)。有关更多信息，请参见 `rctlblk_set_value(3C)`。
SIGXCPU	终止进程。超过了 CPU 时间限制。	仅可用于具有 RCTL_GLOBAL_CPUTIME 属性的资源控制 (`process.max-cpu-time`)。有关更多信息，请参见 `rctlblk_set_value(3C)`。

资源控制标志和属性

系统的每个资源控制都有一组特定的关联属性。这组属性定义为一组标志，这些标志与此资源的所有受控实例关联。不能修改全局标志，但是可以使用 `rctladm` 或 `getrctl` 系统调用检索这些标志。

本地标志可为特定进程或进程集合中资源控制的特定阈值定义缺省行为和配置。一个阈值的本地标志不会影响同一资源控制的其他已定义阈值的行为。但是，全局标志会影响与特定控制关联的每个值的行为。可以在本地标志对应的全局标志提供的约束内，使用 `prctl` 命令或 `setrctl` 系统调用对本地标志进行修改。请参见 `setrctl(2)`。

有关本地标志、全局标志及其定义的完整列表，请参见 `rctlblk_set_value(3C)`。

要确定在达到特定资源控制的阈值时的系统行为，请使用 `rctladm` 显示此资源控制的全局标志。例如，要显示 `process.max-cpu-time` 的值，请键入以下内容：

```
$ rctladm process.max-cpu-time
    process.max-cpu-time  syslog=off  [ lowerable no-deny cpu-time inf seconds ]
```

全局标志表示以下内容。

`lowerable` 不需要超级用户特权来减小此控制的特权值。

`no-deny` 即使当超过阈值时，也从不拒绝对资源的访问。

`cpu-time` SIGXCPU 可用于在到达此资源的阈值时发送。

`seconds` 资源控制的时间值。

`no-basic` 不能设置特权类型为 `basic` 的资源控制值。只允许有特权的资源控制值。

`no-signal` 不能对资源控制值设置本地信号操作。

`no-syslog` 不能为此资源控制设置全局 `syslog` 消息操作。

`deny` 超出阈值时总是拒绝资源请求。

`count` 资源控制的计数（整数）值。

`bytes` 资源控制大小的单位。

使用 `prctl` 命令可以显示资源控制的本地值和操作。

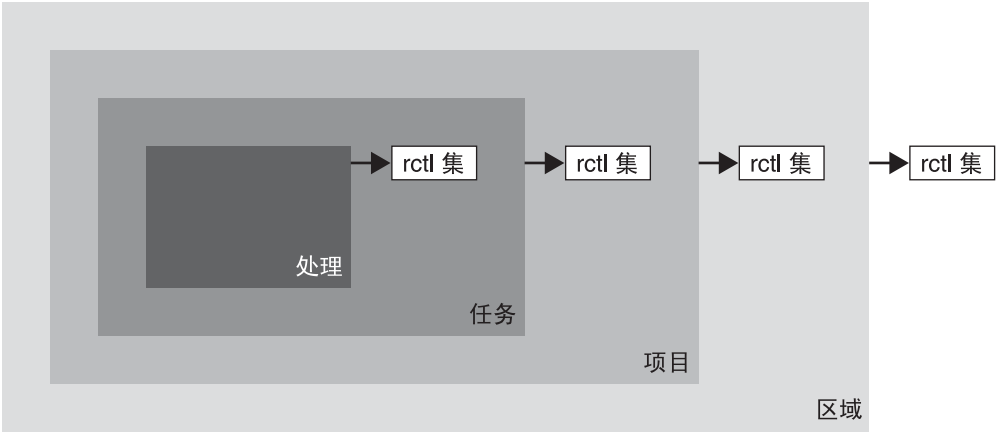
```
$ prctl -n process.max-cpu-time $$
    process 353939: -ksh
    NAME    PRIVILEGE    VALUE    FLAG   ACTION              RECIPIENT
 process.max-cpu-time
         privileged   18.4Es    inf   signal=XCPU                -
         system       18.4Es    inf   none
```

为两个阈值都设置了 max (RCTL_LOCAL_MAXIMAL) 标志，并且为此资源控制定义了 inf (RCTL_GLOBAL_INFINITE) 标志。inf 值可以是无穷大，但从不会达到。因此，如同配置的那样，两个阈值都表示从不会超过的无穷大值。

资源控制执行

一个资源可以存在多个资源控制。进程模型中的每个内嵌项目级别均可存在资源控制。如果同一资源的不同容器级别上的资源控制都处于活动状态，则首先执行最小容器的控制。因此，如果同时遇到 process.max-cpu-time 和 task.max-cpu-time 这两个控制，则先对前者执行操作。

图 6–1 进程集合、容器关系及其资源控制集



全局监视资源控制事件

通常，进程的资源消耗情况是未知的。要获取更多信息，请尝试执行全局资源控制操作，通过 rctladm 命令可实现这些操作。使用 rctladm 可以对资源控制设置 syslog 操作。然后，如果此资源控制管理的任意实体达到阈值，则会在已配置的日志级别上记录系统消息。有关更多信息，请参见第 7 章，管理资源控制（任务）和 rctladm(1M) 手册页。

应用资源控制

在登录或者调用 newtask、su 或项目识别的其他启动程序 at、 batch 或 cron 时，可以为项目指定表 6–1 中列出的每个资源控制。每个启动的命令都会在发出调用的用户的缺省项目的单独任务中启动。有关更多信息，请参见 login(1)、newtask(1)、at(1)、cron(1M) 和 su(1M) 手册页。

对 project 数据库中条目的更新（无论是对 /etc/project 文件还是对网络名称服务中此数据库表示的内容）不会应用于当前活动的项目。更新在新任务通过登录或 newtask 加入项目时应用。

在正在运行的系统上临时更新资源控制值

在 project 数据库中更改的值仅对项目中启动的新任务有效。但是，您可以使用 rctladm 命令和 prctl 命令在正在运行的系统上更新资源控制。

更新日志状态

rctladm 命令会影响系统范围内每个资源控制的全局日志状态。此命令可用于在超过控制时查看 syslog 日志的全局状态并设置此日志的级别。

更新资源控制

使用 prctl 命令，可以按进程、按任务或按项目查看资源控制值和操作，并临时更改资源控制值和操作。项目、任务或进程的 ID 作为输入提供，并且此命令在定义了控制的级别上针对资源控制运行。

对值和操作所做的修改会立即生效。但是，这些修改仅应用于当前的进程、任务或项目。更改不会在 project 数据库中记录。如果重新启动系统，则修改会丢失。必须在 project 数据库中对资源控制进行永久性更改。

所有可在 project 数据库中修改的资源控制设置也可使用 prctl 命令进行修改。可以添加或删除基本值和特权值，还可以修改其操作。缺省情况下，基本类型可用于所有设置的操作，但是具有 root 用户特权的进程和用户还可以修改特权资源控制。不能更改系统资源控制。

用于资源控制的命令

下表显示了用于资源控制的命令。

命令参考	说明
ipcs(1)	可用于查看使用项目资源的 IPC 对象
prctl(1)	可用于对资源控制功能在本地范围进行运行时询问和修改
rctladm(1M)	可用于对资源控制功能在全局范围进行运行时询问和修改

resource_controls(5) 手册页介绍了通过项目数据库提供的资源控制，其中包括单位和标度因数。

◆ ◆ ◆ 第 7 章

管理资源控制（任务）

本章介绍如何管理资源控制功能。

有关资源控制功能的概述，请参见第 6 章，资源控制（概述）。

管理资源控制（任务列表）

任务	说明	参考
设置资源控制。	为 /etc/project 文件中的项目设置资源控制。	第 86 页中的 “设置资源控制”
获取或修改本地范围的活动进程、任务或项目的资源控制值。	对与系统上的活动进程、任务或项目关联的资源控制进行运行时询问和修改。	第 88 页中的 “使用 prctl 命令”
在正在运行的系统上，查看或更新资源控制的全局状态。	查看整个系统范围内每个资源控制的全局日志状态。还在超过控制时设置 syslog 日志的级别。	第 92 页中的 “使用 rctladm”
报告活动的进程间通信 (interprocess communication, IPC) 功能的状态。	显示有关活动的进程间通信 (interprocess communication, IPC) 功能的信息。查看哪些 IPC 对象正在使用项目资源。	第 93 页中的 “使用 ipcs”
确定是否为 Web 服务器分配了足够的 CPU 容量。	设置对资源控制执行的全局操作。通过此操作，可以接收任何所设资源控制值太低的实体的通知。	第 93 页中的 “如何确定是否为 Web 服务器分配了足够的 CPU 容量”

设置资源控制

▼ 如何为项目中的每个任务设置最大 LWP 数

此过程将名为 x-files 的项目添加到 /etc/project 文件，并为在此项目中创建的任务设置最大 LWP 数。

1. **成为 root 用户或承担等效角色。**

2. **使用带有 K 选项的 -projadd 命令创建名为 x-files 的项目。将在此项目中创建的每个任务的最大 LWP 数设置为 3。**

```
# projadd -K 'task.max-lwps=(privileged,3,deny)' x-files
```

3. **使用以下方法之一查看 /etc/project 文件中的条目：**

   - 键入：

```
# projects -l
system
        projid : 0
        comment: ""
        users  : (none)
        groups : (none)
        attribs:
.
.
.
x-files
        projid : 100
        comment: ""
        users  : (none)
        groups : (none)
        attribs: task.max-lwps=(privileged,3,deny)
```

   - 键入：

```
# cat /etc/project
system:0:System:::
.
.
.
x-files:100::::task.max-lwps=(privileged,3,deny)
```

示例 7–1 会话样例

执行完此过程中的步骤后，如果 root 用户在项目 x-files 中创建新任务（通过 newtask 加入项目），则用户无法在运行此任务时创建三个以上的 LWP。以下带有注释的会话样例显示了这一原则。

```
# newtask -p x-files csh
```

```
# prctl -n task.max-lwps $$
process: 111107: csh
NAME    PRIVILEGE    VALUE    FLAG   ACTION            RECIPIENT
task.max-lwps
        usage            3
        privileged       3       -   deny                      -
        system       2.15G     max   deny                      -
# id -p
uid=0(root) gid=1(other) projid=100(x-files)

# ps -o project,taskid -p $$
 PROJECT TASKID
 x-files    73

# csh        /* creates second LWP */

# csh        /* creates third LWP */

# csh        /* cannot create more LWPs */
Vfork failed
#
```

▼ 如何对一个项目设置多个控制

/etc/project 文件可以包含每个项目的多个资源控制设置，还可包含每个控制的多个阈值。阈值在操作子句中定义，这些子句使用逗号分隔多个值。

1 **成为 root 用户或承担等效角色。**

2 **使用带有 -s 和 -K 选项的 projmod 命令对项目 x-files 设置资源控制：**

```
# projmod -s -K 'task.max-lwps=(basic,10,none),(privileged,500,deny);
process.max-file-descriptor=(basic,128,deny)' x-files      one line in file
```

将设置以下控制：

- 针对每个任务的最大 LWP 数不采取任何操作的 basic 控制。
- 针对每个任务的最大 LWP 数的特权 deny 控制。此控制会使所有超过最大值的 LWP 创建都失败，如前一示例第 86 页中的 “如何为项目中的每个任务设置最大 LWP 数”所示。
- 在 basic 级别对每个进程的最大文件描述符数的限制，它会强制任何超过最大数量的 open 调用均失败。

3 **使用以下方法之一，查看文件中的条目：**

- 键入：

```
# projects -l
.
.
.
x-files
```

```
projid : 100
comment: ""
users  : (none)
groups : (none)
attribs: process.max-file-descriptor=(basic,128,deny)
         task.max-lwps=(basic,10,none),(privileged,500,deny)      one line in file
```

- 键入：

```
# cat /etc/project
.
.
.
x-files:100::::process.max-file-descriptor=(basic,128,deny);
task.max-lwps=(basic,10,none),(privileged,500,deny)      one line in file
```

使用 prctl 命令

使用 prctl 命令，可以对与系统上的活动进程、任务或项目关联的资源控制进行运行时询问和修改。有关更多信息，请参见 prctl(1) 手册页。

▼ 如何使用 prctl 命令显示缺省资源控制值

必须在未设置或更改任何资源控制的系统上使用此过程。/etc/system 文件或 project 数据库中只能有非缺省条目。

- **在任意进程（如正在运行的当前 shell）中使用 prctl 命令。**

```
# prctl $$
process: 3320: bash
NAME    PRIVILEGE       VALUE    FLAG   ACTION                       RECIPIENT
process.max-port-events
        privileged      65.5K       -   deny                                 -
        system          2.15G     max   deny                                 -
process.max-msg-messages
        privileged      8.19K       -   deny                                 -
        system          4.29G     max   deny                                 -
process.max-msg-qbytes
        privileged      64.0KB      -   deny                                 -
        system          16.0EB    max   deny                                 -
process.max-sem-ops
        privileged        512       -   deny                                 -
        system          2.15G     max   deny                                 -
process.max-sem-nsems
        privileged        512       -   deny                                 -
        system          32.8K     max   deny                                 -
process.max-address-space
        privileged      16.0EB    max   deny                                 -
        system          16.0EB    max   deny                                 -
process.max-file-descriptor
        basic             256       -   deny                              3320
        privileged      65.5K       -   deny                                 -
```

```
        system          2.15G     max   deny                                 -
process.max-core-size
        privileged      8.00EB    max   deny                                 -
        system          8.00EB    max   deny                                 -
process.max-stack-size
        basic           10.0MB      -   deny                               3320
        privileged      32.0TB      -   deny                                 -
        system          32.0TB    max   deny                                 -
process.max-data-size
        privileged      16.0EB    max   deny                                 -
        system          16.0EB    max   deny                                 -
process.max-file-size
        privileged      8.00EB    max   deny,signal=XFSZ                     -
        system          8.00EB    max   deny                                 -
process.max-cpu-time
        privileged      18.4Es    inf   signal=XCPU                          -
        system          18.4Es    inf   none                                 -
task.max-cpu-time
        usage               0s
        system          18.4Es    inf   none                                 -
task.max-processes
        usage               2
        system          2.15G     max   deny                                 -
task.max-lwps
        usage               3
        system          2.15G     max   deny                                 -
project.max-contracts
        privileged      10.0K       -   deny                                 -
        system          2.15G     max   deny                                 -
project.max-locked-memory
        usage              0B
        system          16.0EB    max   deny                                 -
project.max-port-ids
        privileged      8.19K       -   deny                                 -
        system          65.5K     max   deny                                 -
project.max-shm-memory
        privileged       510MB      -   deny                                 -
        system          16.0EB    max   deny                                 -
project.max-shm-ids
        privileged        128       -   deny                                 -
        system          16.8M     max   deny                                 -
project.max-msg-ids
        privileged        128       -   deny                                 -
        system          16.8M     max   deny                                 -
project.max-sem-ids
        privileged        128       -   deny                                 -
        system          16.8M     max   deny                                 -
project.max-crypto-memory
        usage              0B
        privileged       510MB      -   deny                                 -
        system          16.0EB    max   deny                                 -
project.max-tasks
        usage               2
        system          2.15G     max   deny                                 -
project.max-processes
        usage               4
        system          2.15G     max   deny                                 -
project.max-lwps
        usage              11
```

```
        system          2.15G     max   deny                           -
project.cpu-cap
        usage              0
        system          4.29G     inf   deny                           -
project.cpu-shares
        usage              1
        privileged         1       -   none                           -
        system          65.5K     max   none                           -
zone.max-lofi
        usage              0
        system          18.4E     max   deny                           -
zone.max-swap
        usage           180MB
        system          16.0EB    max   deny                           -
zone.max-locked-memory
        usage              0B
        system          16.0EB    max   deny                           -
zone.max-shm-memory
        system          16.0EB    max   deny                           -
zone.max-shm-ids
        system          16.8M     max   deny                           -
zone.max-sem-ids
        system          16.8M     max   deny                           -
zone.max-msg-ids
        system          16.8M     max   deny                           -
zone.max-processes
        usage             73
        system          2.15G     max   deny                           -
zone.max-lwps
        usage            384
        system          2.15G     max   deny                           -
zone.cpu-cap
        usage              0
        system          4.29G     inf   deny                           -
zone.cpu-shares
        usage              1
        privileged         1       -   none                           -
        system          65.5K     max   none
```

▼ 如何使用 prctl 命令显示给定资源控制的信息

- 显示正在运行的当前 shell 的最大文件描述符。

```
# prctl -n process.max-file-descriptor $$
process: 110453: -sh
NAME    PRIVILEGE       VALUE    FLAG   ACTION       RECIPIENT
process.max-file-descriptor
        basic             256       -   deny            11731
        privileged      65.5K       -   deny                -
        system          2.15G     max   deny
```

▼ 如何使用 prctl 临时更改值

此示例过程使用 prctl 命令临时添加一个新的特权值，以便拒绝在每个 x-files 项目中使用三个以上的 LWP。可将此结果与第 86 页中的 “如何为项目中的每个任务设置最大 LWP 数”中的结果进行对比。

1. **成为 root 用户或承担等效角色。**
2. **使用 newtask 加入 x-files 项目。**

```
# newtask -p x-files
```

3. **使用带有 -p 选项的 id 命令检验是否已加入正确的项目。**

```
# id -p
uid=0(root) gid=1(other) projid=101(x-files)
```

4. **为 project.max-lwps 添加一个新的特权值，将 LWP 数限制为三个。**

```
# prctl -n project.max-lwps -t privileged -v 3 -e deny -i project x-files
```

5. **验证结果。**

```
# prctl -n project.max-lwps -i project x-files
process: 111108: csh
NAME    PRIVILEGE    VALUE    FLAG   ACTION            RECIPIENT
project.max-lwps
        usage          203
        privileged    1000       -   deny                    -
        system       2.15G     max   deny                    -
```

▼ 如何使用 prctl 降低资源控制值

1. **成为 root 用户或承担等效角色。**
2. **使用带有 -r 选项的 prctl 命令更改 process.max-file-descriptor 资源控制的最低值。**

```
# prctl -n process.max-file-descriptor -r -v 128 $$
```

▼ 如何使用 prctl 显示、替换和检验项目的控制值

1. **成为 root 用户或承担等效角色。**
2. **显示项目 *group.staff* 中 project.cpu-shares 的值。**

```
# prctl -n project.cpu-shares -i project group.staff
project: 2: group.staff
NAME    PRIVILEGE       VALUE    FLAG   ACTION     RECIPIENT
project.cpu-shares
```

```
        usage              1
        privileged         1       -   none                 -
        system          65.5K     max   none
```

3 **将当前 project.cpu-shares 值 1 替换为值 10。**

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

4 **显示项目 group.staff 中 project.cpu-shares 的值。**

```
# prctl -n project.cpu-shares -i project group.staff
project: 2: group.staff
NAME    PRIVILEGE       VALUE    FLAG   ACTION     RECIPIENT
project.cpu-shares
        usage              1
        privileged         1       -   none                                    -
        system          65.5K     max   none
```

使用 rctladm

如何使用 rctladm

使用 rctladm 命令可以对资源控制功能的全局状态进行运行时询问和修改。有关更多信息，请参见 rctladm(1M) 手册页。

例如，您可以使用带有 -e 选项的 rctladm 来启用资源控制的全局 syslog 属性。当超过控制时，便会在指定的 syslog 级别记录通知。要启用 process.max-file-descriptor 的全局 syslog 属性，请键入以下命令：

```
# rctladm -e syslog process.max-file-descriptor
```

在不使用参数的情况下，rctladm 命令将显示每个资源控制的全局标志，包括全局类型标志。

```
# rctladm
process.max-port-events     syslog=off  [ deny count ]
process.max-msg-messages    syslog=off  [ deny count ]
process.max-msg-qbytes      syslog=off  [ deny bytes ]
process.max-sem-ops         syslog=off  [ deny count ]
process.max-sem-nsems       syslog=off  [ deny count ]
process.max-address-space   syslog=off  [ lowerable deny no-signal bytes ]
process.max-file-descriptor syslog=off  [ lowerable deny count ]
process.max-core-size       syslog=off  [ lowerable deny no-signal bytes ]
process.max-stack-size      syslog=off  [ lowerable deny no-signal bytes ]
.
.
.
```

使用 ipcs

如何使用 ipcs

使用 ipcs 实用程序可以显示有关活动的进程间通信 (interprocess communication, IPC) 功能的信息。有关更多信息，请参见 ipcs(1) 手册页。

您可以使用带有 -J 选项的 ipcs 来查看分配 IPC 对象所遵循的项目限制。

```
# ipcs -J
    IPC status from <running system> as of Wed Mar 26 18:53:15 PDT 2003
T         ID      KEY        MODE       OWNER    GROUP    PROJECT
Message Queues:
Shared Memory:
m       3600      0       --rw-rw-rw-   uname    staff    x-files
m        201      0       --rw-rw-rw-   uname    staff    x-files
m       1802      0       --rw-rw-rw-   uname    staff    x-files
m        503      0       --rw-rw-rw-   uname    staff    x-files
m        304      0       --rw-rw-rw-   uname    staff    x-files
m        605      0       --rw-rw-rw-   uname    staff    x-files
m          6      0       --rw-rw-rw-   uname    staff    x-files
m        107      0       --rw-rw-rw-   uname    staff    x-files
Semaphores:
s          0      0       --rw-rw-rw-   uname    staff    x-files
```

容量警告

通过对资源控制执行全局操作，可以接收任何实体因资源控制值设置太低而失败的通知。

例如，假设您要确定 Web 服务器是否拥有处理一般工作负荷所需的 CPU。您可以分析 sar 数据以了解空闲的 CPU 时间和平均负荷值。您也可以检查扩展记帐数据以确定针对 Web 服务器进程同时运行的进程数。

但是，比较简单的方法是将 Web 服务器置于任务中。然后，可以使用 syslog 设置全局操作，以便在任务超过对应于计算机容量的预定 LWP 数时通知您。

有关更多信息，请参见 sar(1) 手册页。

▼ 如何确定是否为 Web 服务器分配了足够的 CPU 容量

1 **使用 prctl 命令在包含 httpd 进程的任务设置特权（root 用户拥有）资源控制。将每个任务的 LWP 总数限制为 40，并禁用所有的本地操作。**

```
# prctl -n task.max-lwps -v 40 -t privileged -d all ‘pgrep httpd‘
```

2 **对 task.max-lwps 资源控制启用系统日志全局操作。**

```
# rctladm -e syslog task.max-lwps
```

3 **查看工作负荷是否导致资源控制失败。**

如果是，将看到 /var/adm/messages，例如：

```
Jan  8 10:15:15 testmachine unix: [ID 859581 kern.notice]
NOTICE: privileged rctl task.max-lwps exceeded by task 19
```

◆ ◆ ◆ 第 8 章

公平份额调度器（概述）

对工作负荷数据进行分析可以指明特定工作负荷或工作负荷组是否在独占 CPU 资源。如果这些工作负荷没有违反 CPU 资源使用约束，则可以修改系统上 CPU 时间的分配策略。使用本章所述的公平份额调度类，您可以基于份额而不是分时 (timesharing, TS) 调度类的优先级方案来分配 CPU 时间。

本章包含以下主题：

要开始使用公平份额调度器，请参见第 9 章，管理公平份额调度器（任务）。

调度程序介绍

操作系统的基本工作是仲裁哪些进程可以访问系统资源。进程调度程序，也称为分发程序，是控制为进程分配 CPU 的内核部分。调度程序支持调度类的概念。每个类都定义了调度策略，用于调度类中的进程。Oracle Solaris 操作系统中的缺省调度程序（即 TS 调度程序）尝试为每个进程提供相对均等的访问可用 CPU 的权限。但是，您可能要指定为特定进程提供的资源多于为其他进程提供的资源。

可以使用**公平份额调度器** (fair share scheduler, FSS)，根据工作负荷的重要性控制可用 CPU 资源在工作负荷之间的分配。这种重要性通过您为每个工作负荷指定的 CPU 资源**份额**来表示。

您为每个项目指定 CPU 份额，以控制该项目访问 CPU 资源的权利。FSS 保证为各项目公平地分配 CPU 资源，这种公平分配基于已分配的份额，而与附加到项目的进程数无关。FSS 通过将某个项目与其他项目比较后，减少此项目对 CPU 的大量使用的权利，同时增加少量使用的权利来达到公平。

FSS 由一个内核调度类模块以及类的特定版本的 `dispadmin(1M)` 和 `priocntl(1)` 命令组成。FSS 使用的项目份额通过 `project(4)` 数据库中的 `project.cpu-shares` 属性指定。

注 – 如果您要在安装了区域的 Oracle Solaris 系统上使用 `project.cpu-shares` 资源控制，请参见第 210 页中的 “区域配置数据”、第 326 页中的 “在非全局区域中使用的资源控制”和第 356 页中的 “在安装了区域的 Oracle Solaris 系统上使用公平份额调度器”。

CPU 份额定义

术语“份额”用于定义系统 CPU 资源中分配给某一项目的部分。如果您为某个项目指定的 CPU 份额数多于为其他项目指定的份额数，则此项目将从公平份额调度器中接收更多的 CPU 资源。

CPU 份额并不等同于 CPU 资源的百分比。份额用于定义工作负荷相对于其他工作负荷的相对重要性。为项目指定 CPU 份额时，主要的关注对象并不是项目具有的份额数，更重要的是要知道此项目与其他项目相比具有多少份额。您还必须考虑有多少其他项目与此项目争用 CPU 资源。

注 – 零份额项目中的进程始终以最低的系统优先级 (0) 运行。这些进程仅在非零份额项目不使用 CPU 资源时运行。

CPU 份额和进程状态

在 Oracle Solaris 系统中，项目工作负荷通常由多个进程组成。从公平份额调度器的角度来看，每个项目工作负荷可以处于**空闲**或**活动**状态。如果某个项目的所有进程都没有使用 CPU 资源，则将此项目视为空闲项目。这通常表示此类进程处于**休眠**状态（等待 I/O 完成）或已停止。如果某个项目中至少有一个进程正在使用 CPU 资源，则将此项目视为活动项目。在计算为项目指定多少 CPU 资源时，将使用所有活动项目的份额总数。

活动项目增多时，为每个项目分配的 CPU 将减少，但是不同项目之间的分配比例并没有更改。

CPU 份额与使用率

份额分配并不等同于使用率。如果将 50% 的 CPU 资源分配给某个项目，它可能平均只使用 20% 的 CPU 资源。此外，仅当与其他项目争用资源时，份额才会限制对 CPU 的使用。如果某个项目在系统上单独运行，则无论为此项目分配多么低的资源百分比，它也始终能使用 100% 的处理能力。可用的 CPU 周期永远不会浪费，它们会分布在项目之间。

为处于忙碌状态的工作负荷分配少量份额可能会降低其性能。但是，只要系统没有过载，就不会阻止工作负荷完成其工作。

CPU 份额示例

假设您的系统具有两个 CPU，并且运行两个并行的计算密集型 (CPU-bound) 工作负荷，分别称为 *A* 和 *B*。每个工作负荷都正在作为单独的项目运行。已对这些项目进行了配置，从而为项目 *A* 指定了 S_A 个份额，为项目 *B* 指定了 S_B 个份额。

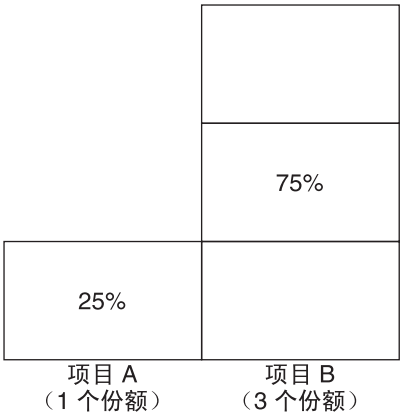
在传统的 TS 调度程序下，会为系统上正在运行的各个工作负荷平均地提供相同的 CPU 资源量。每个工作负荷将使用 50% 的系统容量。

如果在 FSS 调度程序的控制之下运行，并且 $S_A=S_B$，也会为这些项目提供大致等量的 CPU 资源。但是，如果为项目提供了不同的份额数，则它们的 CPU 资源分配量也就不同。

以下三个示例说明了份额在不同的配置中如何起作用。这些示例显示在可用资源能够满足或无法满足需求的情况下，从使用情况的角度来说，份额仅在算术意义上是精确的。

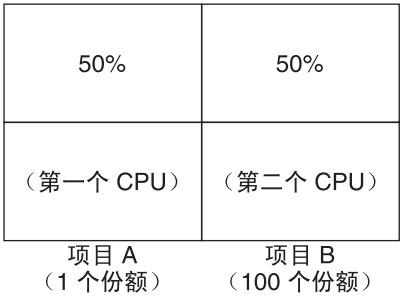
示例 1：每个项目中有两个计算密集型 (CPU-bound) 进程

如果 *A* 和 *B* 各具有两个计算密集型 (CPU-bound) 进程，并且 $S_A = 1$，$S_B = 3$，那么份额总数为 1 + 3 = 4。在该配置中，如果 CPU 请求充足，则分别向项目 *A* 和 *B* 分配 25% 和 75% 的 CPU 资源。



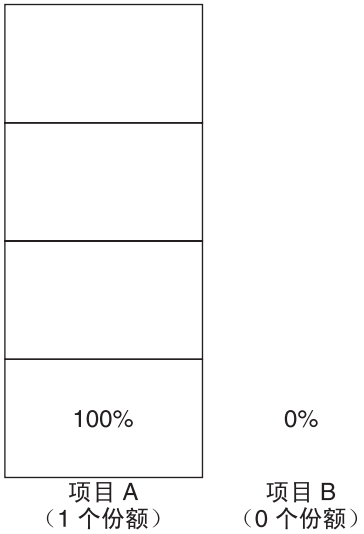
示例 2：项目之间没有争用

如果 *A* 和 *B* 各自仅有一个计算密集型 (CPU-bound) 进程，并且 $S_A = 1$，$S_B = 100$，那么份额的总数为 101。因为每个项目只有一个运行的进程，所以每个项目都不能使用一个以上的 CPU。由于在此配置中项目之间没有争用 CPU 资源，因此，为项目 *A* 和 *B* 各分配了全部 CPU 资源的 50%。在此配置中，CPU 份额值无关紧要。即使为两个项目都指定了零份额，项目的资源分配量也相同 (50/50)。



示例 3：一个项目无法运行

如果 *A* 和 *B* 各有两个计算密集型 (CPU-bound) 进程，并且为项目 *A* 提供 1 个份额，为项目 *B* 提供 0 个份额，则不会为项目 *B* 分配任何 CPU 资源，而为项目 *A* 分配所有 CPU 资源。*B* 中的进程始终以系统优先级 0 运行，因此它们永远不能运行，这是因为项目 *A* 中的进程始终具有较高的优先级。



FSS 设置

项目和用户

项目是指 FSS 调度程序中的工作负荷容器。为项目指定的用户组被视为单个可控制块。请注意，您可以为单个用户创建具有自身份额数的项目。

用户可以是多个指定了不同份额数的项目的成员。通过将进程从一个项目移动到另一个项目，可以为进程指定不同的 CPU 资源量。

有关 `project(4)` 数据库和名称服务的更多信息，请参见第 37 页中的 “project 数据库”。

CPU 份额配置

CPU 份额配置作为 `project` 数据库的一个属性由名称服务来管理。

当通过 `setproject(3PROJECT)` 库函数创建与项目关联的第一个任务（或进程）时，会将在 `project` 数据库中定义为资源控制 `project.cpu-shares` 的 CPU 份额数传递给内核。尚未定义 `project.cpu-shares` 资源控制的项目将被指定一个份额。

在以下示例中，`/etc/project` 文件中的这一条目将项目 *x-files* 的份额数设置为 *5*：

```
x-files:100::::project.cpu-shares=(privileged,5,none)
```

如果在进程运行时改变了分配给数据库中某个项目的 CPU 份额数，则此时将不会修改此项目的份额数。为使更改生效，必须重新启动项目。

如果您要临时更改为项目指定的份额数而不在 project 数据库中改变此项目的属性，请使用 prctl 命令。例如，要在与项目 *x-files* 关联的进程运行时将此项目的 project.cpu-shares 资源控制值更改为 *3*，请键入以下命令：

```
# prctl -r -n project.cpu-shares -v 3 -i project x-files
```

有关更多信息，请参见 prctl(1) 手册页。

-r　　替换命名资源控制的当前值。

-n *name*　　指定资源控制的名称。

-v *val*　　指定资源控制的值。

-i *idtype*　　指定下一个参数的 ID 类型。

x-files　　指定更改的对象。在此实例中，对象为项目 *x-files*。

项目 ID 为 0 的项目 system 中包括所有由引导时初始化脚本启动的系统守护进程。可以将 system 视为具有无限多个份额的项目。这意味着，无论为其他项目提供多少份额，始终先调度 system。如果您不希望 system 项目具有无限的份额，则可以在 project 数据库中为此项目指定一个份额数。

如前所述，属于零份额项目的进程的系统优先级始终为 0。具有一个或多个份额的项目以 1 或更高的优先级运行。这样，仅当 CPU 资源可用（即非零份额项目没有请求 CPU 资源）时，才会调度零份额项目。

可以为一个项目指定的最大份额数为 65535。

FSS 和处理器集

FSS 可以与处理器集一起使用，与单独使用处理器集相比，这样可以更精细地控制 CPU 资源在运行于每个处理器集中的项目之间的分配。FSS 调度程序将处理器集视为完全独立的分区，每个处理器集都单独控制 CPU 的分配。

运行于一个处理器集中的项目的 CPU 分配不会受到运行于另一个处理器集中的项目的 CPU 份额或活动的影响，因为这两个项目没有争用相同的资源。仅当项目在相同的处理器集中运行时，它们才会相互争用资源。

分配给项目的份额数是整个系统范围的份额数。无论项目在哪个处理器集中运行，此项目的每一部分都具有等量份额。

如果使用处理器集，则会针对每个处理器集中运行的活动项目来计算项目的 CPU 分配。

在不同处理器集中运行的项目分区可能具有不同的 CPU 分配。处理器集中每个项目分区的 CPU 分配仅依赖于在同一处理器集中运行的其他项目的分配。

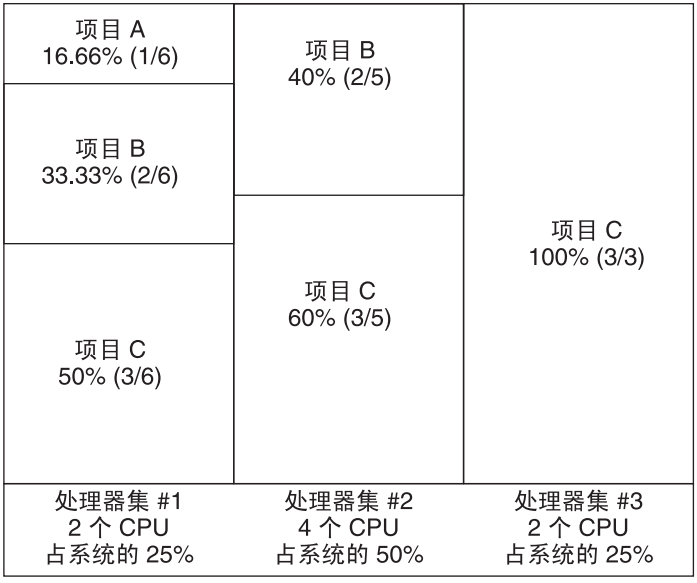
在处理器集边界内运行的应用程序的性能和可用性不会受到新处理器集引入的影响。应用程序也不会受到对其他处理器集中运行的项目的份额分配所做更改的影响。

空处理器集（无处理器的集合）或者没有绑定进程的处理器集不会对 FSS 调度程序行为产生任何影响。

FSS 和处理器集示例

假设配有八个 CPU 的服务器正在项目 *A*、*B* 和 *C* 中运行若干个计算密集型 (CPU-bound) 应用程序。项目 *A* 分配有一个份额，项目 *B* 分配有两个份额，项目 *C* 分配有三个份额。

项目 *A* 只在处理器集 1 上运行。项目 *B* 在处理器集 1 和 2 上运行。项目 *C* 在处理器集 1、2 和 3 上运行。假设每个项目都有足够的进程来利用所有可用的 CPU 资源。这样，每个处理器集中始终存在对 CPU 资源的争用。



下表显示了此类系统上系统范围内总的项目 CPU 分配。

项目	分配
项目 A	$4\% = (1/6 \text{ X } 2/8)_{pset1}$

项目	分配
项目 B	$28\% = (2/6 \text{ X } 2/8)_{pset1} + (2/5 * 4/8)_{pset2}$
项目 C	$67\% = (3/6 \text{ X } 2/8)_{pset1} + (3/5 \text{ X } 4/8)_{pset2} + (3/3 \text{ X } 2/8)_{pset3}$

这些百分比并没有与为项目提供的相应 CPU 份额量相匹配。但是，在每个处理器集中，每个项目的 CPU 分配率与各自的份额成比例。

在**没有**处理器集的同一系统上，CPU 资源的分发将有所不同，如下表所示。

项目	分配
项目 A	16.66% = (1/6)
项目 B	33.33% = (2/6)
项目 C	50% = (3/6)

将 FSS 与其他调度类组合

缺省情况下，FSS 调度类与分时 (timesharing, TS) 调度类、交互式 (interactive, IA) 调度类和固定优先级 (fixed priority, FX) 调度类使用相同的优先级范围（0 到 59）。因此，您应该避免这些调度类中的进程共享**同一**处理器集。FSS、TS、IA 和 FX 类中的混合进程可能会引起意外的调度行为。

使用处理器集时，您可以将 TS、IA、FX 和 FSS 纳入一个系统中。但是，在每个处理器集中运行的所有进程都必须属于**一个**调度类，这样它们就不会争用相同的 CPU。特别是，FX 调度程序不应与 FSS 调度类一起使用，除非使用处理器集。此操作防止 FX 类中的应用程序使用过高的优先级运行以至 FSS 类中的应用程序不能运行。

您可以将 TS 和 IA 类中的进程纳入同一处理器集中，也可纳入同一无处理器集的系统中。

Oracle Solaris 系统还为拥有超级用户特权的用户提供了实时 (Real-Time, RT) 调度程序。缺省情况下，RT 调度类使用与 FSS 不同的系统优先级范围（通常从 100 到 159）。由于 RT 和 FSS 使用**不相交**或不重叠的优先级范围，因此，FSS 可以与 RT 调度类共存于同一处理器集中。但是，FSS 调度类不对运行于 RT 类中的进程进行任何控制。

例如，在具有四个处理器的系统上，如果单线程 RT 进程具有 CPU 限制，则此进程便可占用整个处理器。如果系统也运行 FSS，则一般用户进程便会争用 RT 进程未使用的其余三个 CPU。请注意，RT 进程可能不会持续占用 CPU。当 RT 进程空闲时，FSS 便会使用所有四个处理器。

您可以键入以下命令来查看处理器集在哪些调度类中运行，并确保将每个处理器集配置为运行 TS、IA、FX 或 FSS 进程。

```
$ ps -ef -o pset,class | grep -v CLS | sort | uniq
1 FSS
1 SYS
2 TS
2 RT
3 FX
```

设置系统的调度类

要为系统设置缺省调度类，请参见第 107 页中的 “如何将 FSS 设置为缺省调度程序类”、第 194 页中的 “调度类”和 dispadmin(1M)。要将正在运行的进程移至其他调度类，请参见第 107 页中的 “配置 FSS”和 priocntl(1)。

安装了区域的系统上的调度类

非全局区域使用系统的缺省调度类。如果使用新的缺省调度类设置更新了系统，则在引导或重新引导后非全局区域会获取新的设置。

在此情况下，使用 FSS 的首选方法是通过 dispadmin 命令将 FSS 设置为系统缺省调度类。这样，所有区域都将从获取系统 CPU 资源的公平份额中受益。有关使用区域时的调度类的更多信息，请参见第 194 页中的 “调度类”。

有关在不更改缺省调度类和不重新引导的情况下将正在运行的进程移至其他调度类的信息，请参见表 25–5 和 priocntl(1) 手册页。

用于 FSS 的命令

下表中所示的命令提供了公平份额调度器的主要管理接口。

命令参考	说明
priocntl(1)	显示或设置指定进程的调度参数，将正在运行的进程移至其他调度类。
ps(1)	列出有关正在运行的进程的信息，识别运行处理器集所用的调度类。
dispadmin(1M)	列出系统的可用调度程序。设置系统的缺省调度程序。还用于检查和调整 FSS 调度程序的时间量程值。
FSS(7)	介绍公平份额调度器 (fair share scheduler, FSS)。

◆ ◆ ◆ 第 9 章

管理公平份额调度器（任务）

本章介绍如何使用公平份额调度器 (fair share scheduler, FSS)。

有关 FSS 的概述，请参见第 8 章，公平份额调度器（概述）。有关使用区域时调度类的信息，请参见第 194 页中的 “调度类”。

管理公平份额调度器（任务列表）

任务	说明	参考
监视 CPU 使用情况。	监视项目以及处理器集中项目的 CPU 使用情况。	第 106 页中的 “监视 FSS”
设置缺省调度程序类。	将 FSS 等调度程序设置为系统的缺省调度程序。	第 107 页中的 “如何将 FSS 设置为缺省调度程序类”
将正在运行的进程从一个调度程序类移至其他调度类（如 FSS 类）。	在不更改缺省调度类和不重新引导的情况下，将进程从一个调度类手动移至另一个调度类。	第 108 页中的 “如何将进程从 TS 类手动移至 FSS 类”
将所有正在运行的进程从所有调度类移至其他调度类（如 FSS 类）。	在不更改缺省调度类和不重新引导的情况下，将所有调度类中的进程手动移至另一个调度类。	第 108 页中的 “如何将进程从所有用户类手动移至 FSS 类”
将项目的进程移至其他调度类（如 FSS 类）。	将项目的进程从当前调度类手动移至其他调度类。	第 108 页中的 “如何将项目的进程手动移至 FSS 类”
检查和调整 FSS 参数。	调整调度程序的时间量程值。**时间量程**是指线程在必须放弃处理器之前可以运行的时间。	第 109 页中的 “如何调整调度程序参数”

监视 FSS

您可以使用 prstat(1M) 手册页中所述的 prstat 命令来监视活动项目的 CPU 使用情况。

您可以使用任务的扩展记帐数据来获取每个项目在较长一段时间内占用的 CPU 资源量的统计信息。有关更多信息，请参见第 4 章，扩展记帐（概述）。

▼ 如何按项目监视系统的 CPU 使用情况

- **要监视系统上运行的项目的 CPU 使用情况，请使用带有 -J 选项的 prstat 命令。**

```
# prstat -J
   PID USERNAME  SIZE   RSS STATE  PRI NICE      TIME  CPU PROCESS/NLWP
  5107 root     4556K 3268K cpu0    59    0   0:00:00 0.0% prstat/1
  4570 root       83M   47M sleep   59    0   0:00:25 0.0% java/13
  5105 bobbyc   3280K 2364K sleep   59    0   0:00:00 0.0% su/1
  5106 root     3328K 2580K sleep   59    0   0:00:00 0.0% bash/1
     5 root       0K    0K sleep   99  -20   0:00:14 0.0% zpool-rpool/138
   333 daemon   7196K 2896K sleep   59    0   0:00:07 0.0% rcapd/1
    51 netcfg   4436K 3460K sleep   59    0   0:00:01 0.0% netcfgd/5
  2685 root     3328K 2664K sleep   59    0   0:00:00 0.0% bash/1
   101 netadm   4164K 2824K sleep   59    0   0:00:01 0.0% ipmgmtd/6
   139 root     6940K 3016K sleep   59    0   0:00:00 0.0% syseventd/18
  5082 bobbyc   2236K 1700K sleep   59    0   0:00:00 0.0% csh/1
    45 root       15M 7360K sleep   59    0   0:00:01 0.0% dlmgmtd/7
    12 root       23M   22M sleep   59    0   0:00:45 0.0% svc.configd/22
    10 root       15M   13M sleep   59    0   0:00:05 0.0% svc.startd/19
   337 netadm   6768K 5620K sleep   59    0   0:00:01 0.0% nwamd/9
PROJID    NPROC  SWAP   RSS MEMORY      TIME  CPU PROJECT
     1        6   25M   18M   0.9%   0:00:00 0.0% user.root
     0       73  479M  284M    14%   0:02:31 0.0% system
     3        4   28M   24M   1.1%   0:00:26 0.0% default
    10        2   14M 7288K   0.3%   0:00:00 0.0% group.staff

Total: 85 processes, 553 lwps, load averages: 0.00, 0.00, 0.00
```

▼ 如何按处理器集中的项目监视 CPU 使用情况

- **要监视处理器集列表中项目的 CPU 使用情况，请键入：**

```
% prstat -J -C pset-list
```

其中，*pset-list* 是用逗号分隔的处理器集 ID 的列表。

配置 FSS

用于 Oracle Solaris 系统中的其他调度类的命令也可用于 FSS。您可以设置调度程序类，配置调度程序的可调参数，以及配置单个进程的属性。

请注意，可以使用 `svcadm restart` 重新启动调度程序服务。有关更多信息，请参见 `svcadm(1M)`。

列出系统中的调度程序类

要显示系统中的调度程序类，使用带有 `-l` 选项的 `dispadmin` 命令。

```
$ dispadmin -l
CONFIGURED CLASSES
==================

SYS     (System Class)
TS      (Time Sharing)
SDC     (System Duty-Cycle Class)
FSS     (Fair Share)
FX      (Fixed Priority)
IA      (Interactive)
```

▼ 如何将 FSS 设置为缺省调度程序类

FSS 必须是系统上的缺省调度程序才能使 CPU 份额分配生效。

使用 `priocntl` 和 `dispadmin` 命令的组合确保 FSS 既可立即设置为缺省调度程序，也可在重新引导之后设置为缺省调度程序。

1. **成为 root 用户或承担等效角色。**

2. **将系统的缺省调度程序设置为 FSS。**

   ```
   # dispadmin -d FSS
   ```

   此更改将在下次重新引导时生效。重新引导之后，系统上的每个进程都在 FSS 调度类中运行。

3. **在不重新引导的情况下，使此配置立即生效。**

   ```
   # priocntl -s -c FSS -i all
   ```

▼ 如何将进程从 TS 类手动移至 FSS 类

您可以在不更改缺省调度类和不重新引导的情况下，将进程从一个调度类手动移至另一个调度类。此过程显示了如何将进程从 TS 调度类手动移至 FSS 调度类。

1. **成为 root 用户或承担等效角色。**
2. **将 `init` 进程 (pid 1) 移至 FSS 调度类。**

   ```
   # priocntl -s -c FSS -i pid 1
   ```

3. **将所有进程从 TS 调度类移至 FSS 调度类。**

   ```
   # priocntl -s -c FSS -i class TS
   ```

注 – 重新引导之后，所有进程将再次在 TS 调度类中运行。

▼ 如何将进程从所有用户类手动移至 FSS 类

您可以使用 TS 之外的缺省类。例如，您的系统可能正在运行缺省情况下使用 IA 类的窗口环境。您可以在不更改缺省调度类和不重新引导的情况下，将所有进程手动移至 FSS 调度类。

1. **成为 root 用户或承担等效角色。**
2. **将 `init` 进程 (pid 1) 移至 FSS 调度类。**

   ```
   # priocntl -s -c FSS -i pid 1
   ```

3. **将所有进程从当前调度类移至 FSS 调度类。**

   ```
   # priocntl -s -c FSS -i all
   ```

注 – 重新引导之后，所有进程将再次在缺省调度类中运行。

▼ 如何将项目的进程手动移至 FSS 类

您可以将项目的进程从当前调度类手动移至 FSS 调度类。

1. **成为 root 用户或承担等效角色。**
2. **将使用项目 ID *10* 运行的进程移至 FSS 调度类。**

   ```
   # priocntl -s -c FSS -i projid 10
   ```

   重新引导之后，项目的进程将再次在缺省调度类中运行。

如何调整调度程序参数

当系统正在运行时，您可以使用 dispadmin 命令来显示或更改进程调度程序参数。例如，您可以使用 dispadmin 来检查和调整 FSS 调度程序的时间量程值。**时间量程**是指线程在必须放弃处理器之前可以运行的时间。

要在系统正在运行时显示 FSS 调度程序的当前时间量程，请键入：

```
$ dispadmin -c FSS -g
#
# Fair Share Scheduler Configuration
#
RES=1000
#
# Time Quantum
#
QUANTUM=110
```

使用 -g 选项时，您还可以使用 -r 选项来指定列显时间量程值所用的精度。如果未指定精度，则缺省情况下时间量程值将以毫秒显示。

```
$ dispadmin -c FSS -g -r 100
#
# Fair Share Scheduler Configuration
#
RES=100
#
# Time Quantum
#
QUANTUM=11
```

要为 FSS 调度类设置调度参数，请使用 dispadmin -s。*file* 中的值必须采用由 -g 选项输出的格式。这些值会覆盖内核中的当前值。键入以下命令：

```
$ dispadmin -c FSS -s file
```

◆ ◆ ◆ 第 10 章

使用资源上限设置守护进程控制物理内存（概述）

使用资源上限设置守护进程 rcapd，您可以调节已定义资源上限的项目中运行的进程所占用的物理内存。如果在系统中运行区域，则可以从全局区域中使用 rcapd 来控制非全局区域中物理内存的占用情况。请参见第 17 章，规划和配置非全局区域（任务）。

本章包含以下主题：

有关使用 rcapd 实用程序的过程，请参见第 11 章，管理资源上限设置守护进程（任务）。

资源上限设置守护进程介绍

资源**上限**是对资源（如物理内存）占用设定的上界。支持按项目设置物理内存上限。

资源上限设置守护进程及其关联的实用程序提供了物理内存资源上限执行和管理机制。

资源上限与资源控制一样，可以使用 project 数据库中项目条目的属性进行定义。但是，资源控制由内核同步执行，而资源上限由资源上限设置守护进程在用户级别上异步执行。在异步执行过程中，守护进程所用的抽样间隔会导致轻微的延迟。

有关 rcapd 的信息，请参见 rcapd(1M) 手册页。有关项目和 project 数据库的信息，请参见第 2 章，项目和任务（概述）和 project(4) 手册页。有关资源控制的信息，请参见第 6 章，资源控制（概述）。

资源上限设置工作原理

守护进程重复对具有物理内存上限的项目的资源利用率进行抽样。它所使用的抽样间隔由管理员指定。有关其他信息，请参见第 116 页中的 “确定抽样间隔”。当系统的物理内存使用率超过上限执行的阈值并且满足其他条件时，守护进程便会采取措施将具有内存上限的项目的资源使用率降到等于或低于上限的水平。

虚拟内存系统将物理内存分为多个段，这些段称为页面。在 Oracle Solaris 内存管理子系统中，页面是物理内存的基本单元。在将数据从文件读入内存时，虚拟内存系统一次读入文件的一页，或者说对文件执行**页入**操作。为了减少资源占用，守护进程可以对不常用的页面执行**页出**操作，即将其重新放置到交换设备中，该设备是位于物理内存以外的区域。

守护进程通过调整项目工作负荷驻留集相对其工作集的大小来管理物理内存。驻留集是驻留在物理内存中的一组页面。工作集是指处理工作负荷过程中实际使用的一组页面。工作集会随着时间的推移发生变化，具体取决于进程的运行模式以及正在处理的数据类型。理想的情况是，每个工作负荷可以访问的物理内存都足以使其工作集一直驻留在物理内存中。但是，工作集还可以使用辅助磁盘存储器来容纳物理内存之外的存储器。

在给定时间只能运行一个 `rcapd` 实例。

限制项目物理内存使用情况的属性

要定义项目的物理内存资源上限，请通过为 `project` 数据库条目添加以下属性来设定驻留集大小 (resident set size, RSS) 上限：

`rcap.max-rss`　　项目中的进程可用的物理内存总量（字节）。

例如，`/etc/project` 文件中的以下行将项目 `db` 的 RSS 上限设置为 10 GB。

```
db:100::db,root::rcap.max-rss=10737418240
```

注 – 系统可以将指定的上限值舍入为页面大小。

还可以使用 `projmod` 命令在 `/etc/project` 文件中设置 `rcap.max-rss` 属性。

有关详细信息，请参见“设置驻留集大小上限”。

rcapd 配置

您可以使用 rcapadm 命令配置资源上限设置守护进程。可以执行以下操作：

- 设置上限执行的阈值
- 设置 rcapd 执行操作的间隔
- 启用或禁用资源上限设置
- 显示已配置的资源上限设置守护进程的当前状态

要配置守护进程，必须是 root 用户或者具有所需的管理权限。

可以根据配置间隔（请参见第 115 页中的 “rcapd 操作间隔”）或者在需要时通过发送 SIGHUP（请参见 kill(1) 手册页），将配置更改并入 rcapd。

如果使用时不带参数，rcapadm 将显示资源上限设置守护进程（如果已配置）的当前状态。

以下各小节将讨论上限执行、上限值以及 rcapd 操作间隔。

在安装有区域的系统上使用资源上限设置守护进程

可以在配置区域时通过设置 capped-memory 资源，来控制该区域的驻留集大小 (resident set size, RSS) 的使用情况。有关详细信息，请参见第 194 页中的 “物理内存控制和 capped-memory 资源”。要使用 capped-memory 资源，必须在全局区域中安装 resource-cap 软件包。可以在区域（包括全局区域）**内**运行 rcapd，以便对该区域中的项目执行内存上限。

您可以为指定区域可占用的最大内存量设置一个临时上限（该值可持续到下次重新引导）。请参见第 123 页中的 “如何为区域指定临时资源上限”。

如果要在某个区域中使用 rcapd 来控制已定义资源上限的项目中运行的进程所占用的物理内存，则必须在这些区域中配置该守护进程。

为位于不同的区域中的应用程序选择内存上限时，通常不必考虑这些应用程序驻留在不同的区域中。但每区域服务则例外。每区域服务会占用内存。在确定系统的物理内存量和内存上限时，必须考虑此内存占用情况。

内存上限执行阈值

内存上限执行阈值是系统中触发上限执行的物理内存使用百分比。当系统超过此使用率时，便会执行上限。应用程序和内核使用的物理内存包括在此百分比中。此使用百分比确定执行内存上限的方式。

在执行上限时，会对项目工作负荷中的内存执行页出操作。

- 可以对内存执行页出操作，以减小给定工作负荷超过其上限的内存部分的大小。
- 可以对内存执行页出操作，以减小超过系统内存上限执行阈值的所用物理内存部分的大小。

某个工作负荷最多可以使用大小等于其上限的物理内存。只要系统内存使用率低于内存上限执行阈值，工作负荷便可使用更多的内存。

要设置上限执行值，请参见第 121 页中的 “如何设置内存上限执行阈值”。

确定上限值

如果项目上限设置得太低，就没有足够的内存来保证工作负荷在正常情况下有效地执行。由于工作负荷需要更多内存而产生的分页操作会对系统性能造成负面影响。

上限设置得太高的项目可能会在超过其上限值之前占用可用物理内存。在这种情况下，物理内存由内核而不是 rcapd 进行有效管理。

在确定项目的上限时，应考虑到以下因素。

对 I/O 系统的影响

守护进程可以尝试在抽样使用率超过项目上限时降低项目工作负荷的物理内存使用情况。在上限执行过程中，将使用交换设备和包含工作负荷映射的文件的其他设备。交换设备的性能是确定经常超过其上限的工作负荷的性能的重要因素。执行工作负荷类似于在具有等同于工作负荷上限的物理内存量的计算机上运行该工作负荷。

对 CPU 使用率的影响

守护进程的 CPU 使用率随着它已设置上限的项目工作负荷中的进程数和工作负荷的地址空间大小而变化。

守护进程的少部分 CPU 时间用在对每个工作负荷使用情况进行的抽样上。向工作负荷中添加进程会增加对使用率进行抽样所用的时间。

守护进程的另一部分 CPU 时间用在超过上限时执行上限上。所用的时间与涉及的虚拟内存量成比例。所用的 CPU 时间会根据工作负荷的地址空间总大小的相应更改而延长或缩短。此信息在 rcapstat 输出的 vm 列中显示。有关更多信息，请参见第 116 页中的 “使用 rcapstat 监视资源利用率”和 rcapstat(1) 手册页。

有关共享内存的报告

rcapd 守护进程报告与其他进程共享的内存页的 RSS，或在与合理的准确估算同样的进程中的多次映射。如果不同项目中的进程共享同一内存，那么将为共享该内存的所有项目为 RSS 总数计算该内存。

估算可用于广泛使用共享内存的工作负荷（如数据库）。对于这些数据库工作负荷，您也可以使用 prstat 命令的 -J 或 -Z 选项的输出对项目的常规使用进行抽样，以便确定适当的初始上限值。有关更多信息，请参见 prstat(1M) 手册页。

rcapd 操作间隔

您可以调整 rcapd 所执行的定期操作的间隔。

所有间隔都以秒为单位指定。下表介绍了 rcapd 操作及其缺省间隔值。

操作	缺省间隔值（秒）	说明
scan	15	对加入或保留项目工作负荷的进程进行扫描的间隔秒数。最小值为 1 秒。
sample	5	对驻留集大小和后续上限执行进行抽样的间隔秒数。最小值为 1 秒。
report	5	对分页统计信息进行更新的间隔秒数。如果设置为 0，则不更新统计信息，并且 rcapstat 的输出也不是最新的。
config	60	重新配置的间隔秒数。在重新配置事件中，rcapadm 读取配置文件以获得更新，并扫描 project 数据库以查找新的或已修改的项目上限。向 rcapd 发送 SIGHUP 会立即执行重新配置。

要调整间隔，请参见第 121 页中的 “如何设置操作间隔”。

确定 rcapd 扫描间隔

扫描间隔控制 rcapd 查找新进程的频率。在运行有多个进程的系统上，完全扫描这些进程列表会花费较多时间，因此，最好可以延长间隔，以便缩短所用的总 CPU 时间。但是，扫描间隔也代表进程为了纳入具有上限的工作负荷而必须存在的最短时间。如果工作负荷运行多个短期进程，则在延长扫描间隔的情况下，rcapd 无法将进程纳入工作负荷。

确定抽样间隔

使用 rcapadm 配置的抽样间隔是指，在对工作负荷使用情况进行抽样和执行上限（如果超过该上限）这两个操作之间 rcapd 等待的最短时间。如果缩短此间隔，在多数情况下，rcapd 会更频繁地执行上限，从而可能会因换页导致 I/O 增加。但是，较短的抽样间隔也可以减小特定工作负荷的物理内存使用情况突然增加而给其他工作负荷带来的影响。抽样之间的窗口（其中，工作负荷可能不受限制地占用内存并且可能从其他具有上限的工作负荷中获取内存）会缩小。

如果为 rcapstat 指定的抽样间隔小于使用 rcapadm 为 rcapd 指定的间隔，则某些间隔的输出可能为零。发生这种情况是因为 rcapd 更新统计信息的间隔大于使用 rcapadm 指定的间隔。使用 rcapadm 指定的间隔与 rcapstat 所用的抽样间隔无关。

使用 rcapstat 监视资源利用率

使用 rcapstat 可以监视具有上限的项目的资源利用率。要查看 rcapstat 报告示例，请参见第 123 页中的 “使用 rcapstat 生成报告”。

您可以为报告设置抽样间隔并指定重复统计信息的次数。

interval 按秒指定抽样间隔。缺省间隔为 5 秒。

count 指定重复统计信息的次数。缺省情况下，rcapstat 会一直报告统计信息，直至收到终止信号或出现 rcapd 进程。

rcapstat 发布的第一个报告中的分页统计信息显示自启动守护进程以来执行的活动。后续报告反映自发布最后一个报告以来执行的活动。

下表定义 rcapstat 报告中的列标题。

rcapstat 列标题	说明
id	具有上限的项目的 ID。
project	项目名称。
nproc	项目中的进程数。
vm	项目中的进程所用的总虚拟内存大小（包括所有映射的文件和设备），以千字节 (K)、兆字节 (M) 或千兆字节 (G) 为单位。
rss	项目中进程的总驻留集大小 (resident set size, RSS) 的估算量，以千字节 (K)、兆字节 (M) 或千兆字节 (G) 为单位，没有考虑共享的页面。

rcapstat 列标题	说明
cap	为项目定义的 RSS 上限。有关如何指定内存上限的信息，请参见第 112 页中的 “限制项目物理内存使用情况的属性”或 rcapd(1M) 手册页。
at	自上次 rcapstat 抽样以来，rcapd 尝试对其执行页出操作的内存总量。
avgat	自上次 rcapstat 抽样以来，rcapd 在所出现的每个抽样周期中尝试对其执行页出操作的平均内存量。使用 rcapadm 可以设置 rcapd 对集合 RSS 进行抽样的速率。请参见第 115 页中的 “rcapd 操作间隔”。
pg	自上次 rcapstat 抽样以来，rcapd 成功对其执行页出操作的内存总量。
avgpg	自上次 rcapstat 抽样以来，rcapd 在所出现的每个抽样周期中成功对其执行页出操作的平均内存量估算值。使用 rcapadm 可以设置 rcapd 对进程 RSS 大小进行抽样的速率。请参见第 115 页中的 “rcapd 操作间隔”。

用于 rcapd 的命令

命令参考	说明
rcapstat(1)	监视具有上限的项目的资源利用率。
rcapadm(1M)	配置资源上限设置守护进程，显示已配置的资源上限设置守护进程的当前状态，以及启用或禁用资源上限设置。还可用于设置临时内存上限。
rcapd(1M)	资源上限设置守护进程。

◆ ◆ ◆ 第 11 章

11 管理资源上限设置守护进程（任务）

本章介绍配置和使用资源上限设置守护进程 rcapd 的过程。

有关 rcapd 的概述，请参见第 10 章，使用资源上限设置守护进程控制物理内存（概述）。

设置驻留集大小上限

将 rcap.max-rss 属性添加到 project 数据库条目中，可以为项目定义物理内存资源驻留集大小 (Resident Set Size, RSS) 上限。

▼ 如何为项目添加 rcap.max-rss 属性

1. **成为 root 用户或承担等效角色。**
2. **将该属性添加到 /etc/project 文件中：**

   rcap.max-rss=*value*

示例 11–1 RSS 项目上限

/etc/project 文件中的以下行将项目 db 的 RSS 上限设置为 10 GB。

```
db:100::db,root::rcap.max-rss=10737418240
```

注意，系统可以将指定的上限值舍入为页面大小。

▼ 如何使用 projmod 命令为项目添加 rcap.max-rss 属性

1. 成为 root 用户或承担等效角色。
2. 在本例中，在 /etc/project 文件中将项目 db 的 rcap.max-rss 属性设置为 10 GB。

```
# projmod -a -K rcap.max-rss=10GB db
```

然后，/etc/project 文件将包含以下行：

```
db:100::db,root::rcap.max-rss=10737418240
```

配置和使用资源上限设置守护进程（任务列表）

任务	说明	参考
设置内存上限执行阈值。	配置一个将在可用于进程的物理内存很低时执行的上限。	第 121 页中的“如何设置内存上限执行阈值”
设置操作间隔。	间隔应用于由资源上限设置守护进程执行的定期操作。	第 121 页中的“如何设置操作间隔”
启用资源上限设置。	在系统上激活资源上限设置。	第 122 页中的“如何启用资源上限设置”
禁用资源上限设置。	在系统上取消激活资源上限设置。	第 122 页中的“如何禁用资源上限设置”
报告上限和项目信息。	查看用于生成报告的示例命令。	第 123 页中的“报告上限和项目信息”
监视项目的驻留集大小。	生成有关项目驻留集大小的报告。	第 124 页中的“监视项目的 RSS”
确定项目的工作集大小。	生成有关项目工作集大小的报告。	第 124 页中的“确定项目的工作集大小”
报告内存使用率和内存上限。	针对每个间隔在报告结尾列显一行有关内存使用率和上限执行的信息。	第 125 页中的“报告内存使用率和内存上限执行阈值”

使用 rcapadm 管理资源上限设置守护进程

本节介绍了使用 rcapadm 配置资源上限设置守护进程的过程。有关更多信息，请参见第 113 页中的 “rcapd 配置”和 rcapadm(1M) 手册页。此外，还介绍了使用 rcapadm 为区域指定临时资源上限的过程。

如果使用时不带参数，rcapadm 将显示资源上限设置守护进程（如果已配置）的当前状态。

▼ 如何设置内存上限执行阈值

可以对上限进行配置，以便在可用于进程的物理内存很低时执行。有关更多信息，请参见第 113 页中的 “内存上限执行阈值”。

最小（和缺省）值为 0，这意味着将始终执行内存上限。要设置不同的最小值，请遵照以下过程执行操作。

1. **成为 root 用户或承担等效角色。**
2. **使用 rcapadm 的 -c 选项为内存上限执行设置不同的物理内存使用率值。**

   ```
   # rcapadm -c percent
   ```

   percent 的范围为 0 至 100。值越高，限制就越小。较高的值表示在系统的内存使用率超过此阈值之前，可以在不执行上限的情况下执行具有上限的项目的工作负荷。

另请参见 要显示当前物理内存使用率和上限执行阈值，请参见第 125 页中的 “报告内存使用率和内存上限执行阈值”。

▼ 如何设置操作间隔

第 115 页中的 “rcapd 操作间隔”介绍了有关由 rcapd 执行的定期操作的间隔的信息。要使用 rcapadm 设置操作间隔，请遵照以下过程执行操作。

1. **成为 root 用户或承担等效角色。**
2. **使用 -i 选项设置间隔值。**

   ```
   # rcapadm -i interval=value,...,interval=value
   ```

注 – 所有间隔值都以秒为单位指定。

▼ 如何启用资源上限设置

可以通过三种方法在系统上启用资源上限设置。启用资源上限设置还可以使用缺省值设置 /etc/rcap.conf 文件。

1 **成为 root 用户或承担等效角色。**

2 **通过以下方法之一启用资源上限设置守护进程：**

- 使用 svcadm 命令启用资源上限设置。

  ```
  # svcadm enable rcap
  ```

- 启用资源上限设置守护进程，以使其现在启动并且也在每次引导系统时启动：

  ```
  # rcapadm -E
  ```

- 如果不是现在启用资源上限设置守护进程，而是在引导时启用它，则还应指定 -n 选项：

  ```
  # rcapadm -n -E
  ```

▼ 如何禁用资源上限设置

可以通过三种方法在系统上禁用资源上限设置。

1 **成为 root 用户或承担等效角色。**

2 **通过以下方法之一禁用资源上限设置守护进程：**

- **使用 svcadm 命令禁用资源上限设置。**

  ```
  # svcadm disable rcap
  ```

- **要禁用资源上限设置守护进程，以使其现在停止并且不会在引导系统时启动，请键入：**

  ```
  # rcapadm -D
  ```

- **要在不停止资源上限设置守护进程的情况下禁用它，还应指定 -n 选项：**

  ```
  # rcapadm -n -D
  ```

提示 – 安全禁用资源上限设置守护进程

使用 rcapadm -D 可以安全禁用 rcapd。如果中止该守护进程（请参见 kill(1) 手册页），则进程可能处于停止状态，并且需要手动重新启动。要使进程恢复运行，请使用 prun 命令。有关更多信息，请参见 prun(1) 手册页。

▼ 如何为区域指定临时资源上限

此过程用于分配指定区域可占用的最大内存量。此值只会持续到下次重新引导。要设置持久性上限，请使用 zonecfg 命令。

1 **成为 root 用户或承担等效角色。**

2 **为区域 my-zone 设置 512 MB 的最大内存值。**

```
# rcapadm -z testzone -m 512M
```

使用 rcapstat 生成报告

使用 rcapstat 可报告资源上限设置统计信息。第 116 页中的 “使用 rcapstat 监视资源利用率”说明了如何使用 rcapstat 命令生成报告。此节还介绍了报告中的列标题。rcapstat(1) 手册页也包含此信息。

以下各小节通过示例说明如何生成用于特定用途的报告。

报告上限和项目信息

在此示例中，为与两个用户相关联的两个项目定义了上限。user1 的上限为 50 MB，user2 的上限为 10 MB。

以下命令以 5 秒为抽样间隔生成 5 个报告。

```
user1machine% rcapstat 5 5
    id project  nproc     vm    rss   cap    at avgat    pg avgpg
112270   user1     24   123M    35M   50M   50M    0K 3312K    0K
 78194   user2      1  2368K  1856K   10M    0K    0K    0K    0K
    id project  nproc     vm    rss   cap    at avgat    pg avgpg
112270   user1     24   123M    35M   50M    0K    0K    0K    0K
 78194   user2      1  2368K  1856K   10M    0K    0K    0K    0K
    id project  nproc     vm    rss   cap    at avgat    pg avgpg
112270   user1     24   123M    35M   50M    0K    0K    0K    0K
 78194   user2      1  2368K  1928K   10M    0K    0K    0K    0K
    id project  nproc     vm    rss   cap    at avgat    pg avgpg
112270   user1     24   123M    35M   50M    0K    0K    0K    0K
 78194   user2      1  2368K  1928K   10M    0K    0K    0K    0K
    id project  nproc     vm    rss   cap    at avgat    pg avgpg
112270   user1     24   123M    35M   50M    0K    0K    0K    0K
 78194   user2      1  2368K  1928K   10M    0K    0K    0K    0K
```

输出的前三行构成了第一个报告，此报告包含自启动 rcapd 以来两个项目的上限和项目信息以及换页统计信息。对于 user1，at 和 pg 列中的数字大于零，对于 user2，这两列中的数字等于零，这表示在守护进程的历史记录中，有时 user1 超过其上限，但 user2 却没有。

后续各报告没有显示任何重要的活动。

监视项目的 RSS

以下示例使用项目 user1，此项目的 RSS 超过其 RSS 上限。

以下命令以 5 秒为抽样间隔生成 5 个报告。

```
user1machine% rcapstat 5 5

    id project  nproc    vm   rss   cap    at avgat     pg  avgpg
376565   user1      3 6249M 6144M 6144M  690M  220M  5528K  2764K
376565   user1      3 6249M 6144M 6144M    0M  131M  4912K  1637K
376565   user1      3 6249M 6171M 6144M   27M  147M  6048K  2016K
376565   user1      3 6249M 6146M 6144M 4872M  174M  4368K  1456K
376565   user1      3 6249M 6156M 6144M   12M  161M  3376K  1125K
```

user1 项目具有三个积极使用物理内存的进程。pg 列中的正值表示 rcapd 在尝试通过降低项目进程的物理内存使用率来满足上限要求时，始终对内存执行页出操作。但是，rcapd 无法成功保持 RSS 低于上限值。从不断变化却并未真正减小的 rss 值可以看出这一点。只要从内存中调出页面，工作负荷便会再次使用内存，于是 RSS 值将会再次回升。这意味着项目的所有驻留内存都在被使用，并且工作集大小 (*working set size, WSS*) 大于上限。因此，将会强制 rcapd 对某些工作集执行页出操作以满足上限要求。在这种情况下，系统将继续频繁出现缺页，并大量使用关联的 I/O，直到发生以下情况之一：

- WSS 变小。
- 上限增加。
- 应用程序更改其内存访问模式。

在这种情况下，缩短抽样间隔可能会减小 RSS 值和上限值之间的差异，因为缩短抽样间隔会使rcapd 更频繁地对工作负荷进行抽样并执行上限。

注 – 必须创建新的页面或者系统必须在交换设备的某页面中进行复制时，便会出现缺页。

确定项目的工作集大小

以下示例是前一示例的延续，它使用相同的项目。

前一示例显示 user1 项目使用的物理内存超过其上限所允许的内存量。此示例显示了项目工作负荷需要的内存量。

```
user1machine% rcapstat 5 5
    id project  nproc    vm   rss   cap    at avgat     pg  avgpg
376565   user1      3 6249M 6144M 6144M  690M    0K   689M     0K
376565   user1      3 6249M 6144M 6144M    0K    0K     0K     0K
376565   user1      3 6249M 6171M 6144M   27M    0K    27M     0K
376565   user1      3 6249M 6146M 6144M 4872K    0K  4816K     0K
```

```
376565   user1      3 6249M 6156M 6144M   12M    0K    12M     0K
376565   user1      3 6249M 6150M 6144M 5848K    0K  5816K     0K
376565   user1      3 6249M 6155M 6144M   11M    0K    11M     0K
376565   user1      3 6249M 6150M   10G   32K    0K    32K     0K
376565   user1      3 6249M 6214M   10G    0K    0K     0K     0K
376565   user1      3 6249M 6247M   10G    0K    0K     0K     0K
376565   user1      3 6249M 6247M   10G    0K    0K     0K     0K
376565   user1      3 6249M 6247M   10G    0K    0K     0K     0K
376565   user1      3 6249M 6247M   10G    0K    0K     0K     0K
376565   user1      3 6249M 6247M   10G    0K    0K     0K     0K
376565   user1      3 6249M 6247M   10G    0K    0K     0K     0K
```

在循环的中途，user1 项目的上限从 6 GB 增大到 10 GB。此增长会停止上限执行并允许驻留集大小增长（仅受计算机中的其他进程和内存量的限制）。rss 列可能会保持不变，以反映项目工作集大小 (working set size, WSS)，在此示例中大小为 6247 M。这是允许项目进程在不会连续出现缺页的情况下运行的最小上限值。

当 user1 的上限为 6 GB 时，在每隔 5 秒的抽样间隔内，由于 rcapd 会对某些工作负荷内存执行页出操作，因此 RSS 将减小，而 I/O 将增加。页出操作完成后不久，需要这些页面的工作负荷会在继续运行时再对这些页面执行页入操作。此循环会重复进行，直到在将近此示例的中途，上限增加到 10 GB。之后，RSS 保持在 6.1 GB。由于此时工作负荷的 RSS 低于上限，因此不再发生换页，还会停止与换页关联的 I/O。因此，项目需要 6.1 GB 来执行查看此项目时正在进行的工作。

另请参见 vmstat(1M) 和 iostat(1M) 手册页。

报告内存使用率和内存上限执行阈值

您可以使用 rcapstat 的 -g 选项报告以下内容：

- 以系统上安装的物理内存的百分比表示的当前物理内存使用率
- 由 rcapadm 设置的系统内存上限执行阈值

可使用 -g 选项针对每个间隔在报告结尾列显一行有关内存使用率和上限执行的信息。

```
# rcapstat -g
    id project   nproc    vm   rss   cap    at avgat   pg  avgpg
376565    rcap       0    0K    0K   10G    0K    0K   0K     0K
physical memory utilization: 55%   cap enforcement threshold: 0%
    id project   nproc    vm   rss   cap    at avgat   pg  avgpg
376565    rcap       0    0K    0K   10G    0K    0K   0K     0K
physical memory utilization: 55%   cap enforcement threshold: 0%
```

◆ ◆ ◆ 第 12 章

12 资源池（概述）

本章讨论以下技术：

- 资源池，用于对计算机资源进行分区
- 动态资源池 (dynamic resource pool, DRP)，可动态调整每个资源池的资源分配，以实现所建立的系统目标

资源池和动态资源池是 Oracle Solaris 服务管理工具 (Service Management Facility, SMF) 中的服务。其中，每项服务都是单独启用的。

本章包含以下主题：

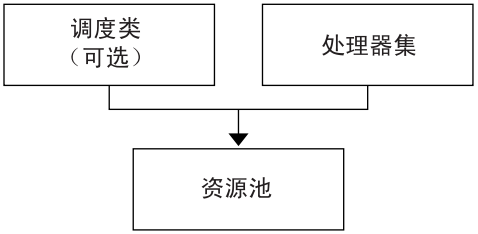
有关使用此功能的过程，请参见第 13 章，创建和管理资源池（任务）。

资源池介绍

通过**资源池**可以分散工作负荷，以便工作负荷占用的特定资源不会重叠。在具有混合工作负荷的系统上，这种资源预留有助于获得可预测的性能。

资源池提供了一种持久性配置机制，可配置处理器集 (pset)，还可选择性指定调度类。

图 12–1 资源池框架



可以将池视为系统上可用的各种资源集的特定绑定。您可以创建表示各种可能的资源组合的池：

```
pool1: pset_default
pool2: pset1
pool3: pset1, pool.scheduler="FSS"
```

通过对多个分区进行分组，池可以提供一个与已标记的工作负荷关联的句柄。`/etc/project` 文件中的每个项目条目都可以有一个与其关联的池，该池使用 `project.pool` 属性指定。

启用池时，**缺省池**和**缺省处理器集**构成了基本配置。可以创建其他用户定义的池和处理器集并将它们添加到配置中。一个 CPU 只能属于一个处理器集。可以销毁用户定义的池和处理器集，不能销毁缺省池和缺省处理器集。

缺省池的 `pool.default` 属性设置为 `true`。缺省处理器集的 `pset.default` 属性设置为 `true`。因此，即使更改了缺省池和缺省处理器集的名称，仍可以识别它们。

用户定义的池机制主要用于 CPU 超过四个的大型计算机。但是，小型计算机仍可以利用此功能。在小型计算机上，您可以创建共享非关键资源分区的池。池仅按关键资源进行分隔。

动态资源池介绍

动态资源池提供了一种机制，可动态调整每个池的资源分配，以便响应系统事件和应用程序负荷的变化。DRP 简化了管理员需要做出的决策并减少了决策数。调整是自动进行的，目的是确保始终达到管理员指定的系统性能目标。对配置所做的更改将会记录。这些功能主要通过资源控制器 `poold` 来实施，它是一种系统守护进程，需要进行动态资源分配时此进程应始终处于活动状态。`poold` 会定期检查系统负荷，并确定是否需要人为介入，以使系统在资源利用方面保持最佳性能。`poold` 配置保存在 `libpool` 配置中。有关 `poold` 的更多信息，请参见 `poold(1M)` 手册页。

关于启用和禁用资源池和动态资源池

要启用和禁用资源池和动态资源池，请参见第 148 页中的 “启用和禁用池功能”。

区域中使用的资源池

除了将区域与系统中已配置的资源池建立关联外，还可以使用 `zonecfg` 命令来创建一个临时池，以在该区域运行时生效。有关更多信息，请参见第 193 页中的 “`dedicated-cpu` 资源”。

在启用了区域的系统上，可以将非全局区域与一个资源池关联，虽然不需要将池专门指定给特定区域。此外，您不能使用全局区域中的 `poolbind` 命令将非全局区域中的单个进程绑定到其他池。要将非全局区域与池关联，请参见第 231 页中的 “配置、检验并提交区域”。

请注意，如果您为池设置了调度类并将该池与非全局区域关联，则缺省情况下，此区域会使用此调度类。

如果使用动态资源池，则 `poold` 执行实例的范围限制为全局区域。

在非全局区域中运行的 `poolstat` 实用程序仅显示与该区域关联的池的相关信息。在非全局区域中运行的不带参数的 `pooladm` 命令仅显示与此区域关联的池的相关信息。

有关资源池命令的信息，请参见第 145 页中的 “用于资源池功能的命令”。

何时使用池

资源池提供了一种通用机制，可应用于许多管理方案。

批处理计算服务器	使用池功能可以将一个服务器分为两个池。一个池由分时用户用于登录会话和交互式工作，另一个池用于通过批处理系统提交的作业。
应用程序或数据库服务器	根据交互式应用程序的要求对用于这些应用程序的资源进行分区。
分期启用应用程序	确定用户期望。 您最初可能将计算机部署为仅运行计算机最终应提供的服务的一部分。如果在计算机联机时未建立基于预留的资源管理机制，则用户可能会遇到问题。 例如，公平份额调度器会优化 CPU 使用率。仅运行一个应用程序时，计算机的响应速度可能会很快，但这仅是误导性的现象。如果装入多个应用程序，则用户将不会看到如此快的响应速度。通过为每个应用程序使用单独的池，您可以对可用于每个应用程序的 CPU 数设定一个上限，然后再部署所有的应用程序。
复杂分时服务器	对支持大量用户的服务器进行分区。对服务器进行分区提供了一种隔离机制，可使每个用户的响应更具可预测性。 通过将用户分为绑定到各个池的不同组，并使用公平份额调度 (fair share scheduling, FSS) 功能，您可以调整 CPU 分配以优先满足具有较高优先级的用户组。可以基于用户角色、记帐费用分摊等进行这种指定。
周期性改变的工作负荷	使用资源池适应变换的需求。 您的站点在工作负荷需求方面可能会出现长周期性（例如每月、每季度或每年）的可预测变化。如果您的站点出现这些变化，则可以通过从 `cron` 作业中调用 `pooladm` 在多个池配置之间进行切换。（请参见第 131 页中的 “资源池框架”。）
实时应用程序	使用 RT 调度程序和指定的处理器资源创建实时池。
系统使用率	执行建立的系统目标。 使用自动执行池守护进程这一功能识别可用的资源，然后监视工作负荷以检测何时不能再满足指定的目标。守护进程可以执行更正操作（如有可能），或者可以将情况记录下来。

资源池框架

/etc/pooladm.conf 配置文件说明了静态池配置。静态配置表示管理员根据资源池功能配置系统的方法。可以指定备用文件名。

当使用服务管理工具 (service management facility, SMF) 或 pooladm -e 命令启用资源池框架时，如果 /etc/pooladm.conf 文件存在，则将该文件中包含的配置应用到系统中。

内核包含有关资源池框架中资源部署的信息。这称为动态配置，它表示特定系统在某个时刻的资源池功能。可以使用 pooladm 命令查看动态配置。请注意，池和资源集的属性显示顺序可以改变。可按以下方法对动态配置进行修改：

- 间接方法，通过应用静态配置文件
- 直接方法，使用带有 -d 选项的 poolcfg 命令

可以存在多个静态池配置文件，在不同时间进行激活。您可以通过从 cron 作业中调用 pooladm 在多个池配置之间进行切换。有关 cron 实用程序的更多信息，请参见 cron(1M) 手册页。

缺省情况下，资源池框架不处于活动状态。必须启用资源池才能创建或修改动态配置。即使禁用了资源池框架，仍可以使用 poolcfg 或 libpool 命令处理静态配置文件。如果池功能不处于活动状态，则无法创建静态配置文件。有关配置文件的更多信息，请参见第 133 页中的 “创建池配置”。

以下手册页中描述了用于资源池和 poold 系统守护进程的命令：

- pooladm(1M)
- poolbind(1M)
- poolcfg(1M)
- poold(1M)
- poolstat(1M)
- libpool(3LIB)

/etc/pooladm.conf 内容

所有资源池配置（包括动态配置）都可以包含以下元素。

system	影响系统整体行为的属性
pool	资源池定义
pset	处理器集定义
cpu	处理器定义

可以处理所有这些元素的属性，以更改资源池框架的状态和行为。例如，池属性 `pool.importance` 表示指定池的相对重要性。此属性用于可能的资源争用解决方案。有关更多信息，请参见 `libpool(3LIB)`。

池属性

池功能支持可用于池、资源或组件的已命名的类型化属性。管理员可以存储各种池元素的其他属性。可以使用与项目属性类似的名称空间属性。

例如，以下注释表示指定的 pset 与特定的 `Datatree` 数据库关联。

```
Datatree,pset.dbname=warehouse
```

有关属性类型的其他信息，请参见第 138 页中的 “poold 属性”。

注 – 许多特殊属性将保留供内部使用，不能进行设置或删除。有关更多信息，请参见 `libpool(3LIB)` 手册页。

在系统上实现池

通过下列方法之一可以在系统上实现用户定义的池。

- 引导 Oracle Solaris 软件时，`init` 脚本会检查 `/etc/pooladm.conf` 文件是否存在。如果找到此文件，并且启用了这些池，则会调用 `pooladm` 以使此配置成为活动池配置。系统将创建动态配置以反映 `/etc/pooladm.conf` 中请求的组织，并相应地对计算机的资源进行分区。
- 当 Oracle Solaris 系统运行时，既可以在不存在池配置时激活一个池配置，也可以使用 `pooladm` 命令修改池配置。缺省情况下，对 `/etc/pooladm.conf` 执行 `pooladm` 命令。但是，您可以选择指定备用位置和文件名，并使用此文件更新池配置。

有关启用和禁用资源池的信息，请参见第 148 页中的 “启用和禁用池功能”。如果正在使用用户定义的池或资源，则不能禁用池功能。

要配置资源池，必须具有 root 特权或拥有所需的权限配置文件。

`poold` 资源控制器使用动态资源池功能来启动。

project.pool 属性

可以将 project.pool 属性添加到 /etc/project 文件中的项目条目，以便将单个池与该条目相关联。针对项目启动的新工作将绑定到相应的池。有关更多信息，请参见第 2 章，项目和任务（概述）。

例如，您可以使用 projmod 命令为 /etc/project 文件中的项目 *sales* 设置 project.pool 属性：

```
# projmod -a -K project.pool=mypool sales
```

SPARC: 动态重新配置操作和资源池

通过动态重新配置 (Dynamic Reconfiguration, DR)，可以在系统运行的同时重新配置硬件。DR 操作可以增大、减小对指定资源类型的影响，或者对其没有任何影响。由于 DR 会影响可用的资源量，因此，这些操作中必须包括池功能。启动 DR 操作之后，池框架便会执行操作以验证配置。

如果 DR 操作可以继续而不会导致当前池配置变为无效，则会更新专用配置文件。无效配置是指可用资源无法支持的配置。

如果 DR 操作导致池配置无效，则操作会失败，并且系统会通过向消息日志发送消息来通知您。如果您要强制完成配置，则必须使用 DR 强制选项。池配置然后会修改以符合新的资源配置。有关 DR 进程和强制选项的信息，请参见 Sun 硬件的动态重新配置用户指南。

如果使用动态资源池，请注意当 poold 守护进程处于活动状态时，分区可能不受该守护进程控制。有关更多信息，请参见第 142 页中的 “识别资源不足”。

创建池配置

配置文件中包含要在系统上创建的池的说明。此文件描述了可以处理的元素。

- system
- pool
- pset
- cpu

有关要处理的元素的更多信息，请参见 poolcfg(1M)。

启用池之后，您可以通过两种方法创建结构化的 /etc/pooladm.conf 文件。

- 可以使用带有 -s 选项的 pooladm 命令搜索当前系统上的资源，并将结果放入配置文件。

  此方法为首选方法。系统上所有可以通过池功能处理的活动资源和组件都将被记录。这些资源包括现有的处理器集配置。然后您可以修改配置以重命名处理器集或创建其他池（如有必要）。
- 可以使用带有 -c 选项以及 discover 或 create system *name* 子命令的 poolcfg 命令创建新的池配置。

  保留这些选项是为了向下兼容早期发行版。

使用 poolcfg 或 libpool 可以修改 /etc/pooladm.conf 文件。请勿直接编辑此文件。

直接处理动态配置

可以使用带有 -d 选项的 poolcfg 命令直接在动态配置中处理 CPU 资源类型。可以使用两种方法传送资源。

- 您可以发出常规请求，以便在处理器集之间传送任何已识别的可用资源。
- 您可以将具有特定 ID 的资源传送到目标集。请注意，更改资源配置时或重新引导系统之后，可以更改与资源关联的系统 ID。

有关示例，请参见第 160 页中的 “传送资源”。

如果正在使用 DRP，注意资源传输可能通过 poold 触发操作。有关更多信息，请参见第 134 页中的 “poold 概述”。

poold 概述

池资源控制器 poold 使用系统目标和可查看的统计信息，来保持您指定的系统性能目标。当需要动态分配资源时，此系统守护进程应始终处于活动状态。

poold 资源控制器先识别可用资源，再监视工作负荷，以确定不再满足系统使用率目标的时刻。然后，poold 根据目标考虑其他配置，并采取补救措施。如有可能，会重新配置资源以满足目标。如果无法执行此操作，则守护进程会记录不能再实现用户指定的目标。重新配置之后，守护进程恢复监视工作负荷目标。

poold 将维护它可以检查的决策历史记录。决策历史记录用于避免再次使用以前未带来任何改进的配置。

请注意，如果更改工作负荷目标或修改可用于系统的资源，还可以异步触发重新配置。

管理动态资源池

DRP 服务由服务管理工具 (service management facility, SMF) 管理，其服务标识符为 `svc:/system/pools/dynamic`。

可以使用 `svcadm` 命令对此服务执行管理操作，如启用、禁用或请求重新启动。可以使用 `svcs` 命令查询服务状态。有关更多信息，请参见 `svcs(1)` 和 `svcadm(1M)` 手册页。

SMF 接口是控制 DRP 的首选方法，但对于向后兼容性，还可使用以下方法。

- 如果不需要动态分配资源，则可以使用 `SIGQUIT` 或 `SIGTERM` 信号停止 `poold`。这两种信号都可以正常终止 `poold`。
- `poold` 会自动检测资源或池配置中的更改，但是，您也可以使用 `SIGHUP` 信号强制进行重新配置。

配置约束和目标

更改配置时，`poold` 会针对您提供的指示执行操作。可以将这些指示指定为一系列约束和目标。`poold` 根据您指定的内容，来确定其他可能配置相对于现有配置的相对值。然后，`poold` 更改当前配置的资源指定，以生成新的候选配置。

配置约束

约束通过排除某些可能会对配置进行的潜在更改来影响可能配置的范围。在 `libpool` 配置中指定的以下约束均可用。

- 最小和最大 CPU 分配量
- 无法从集中移动的固定组件
- 池的重要性系数

有关池属性的更多信息，请参见 `libpool(3LIB)` 手册页和第 132 页中的 “池属性”。

有关使用说明，请参见第 157 页中的 “如何设置配置约束”。

pset.min 和 pset.max 属性约束

这两个属性用于限制可以为处理器集分配的最小和最大处理器数。有关这些属性的更多详细信息，请参见表 12–1。

在遵守这些约束的情况下，可以将资源分区的资源分配给同一 Oracle Solaris 实例中的其他资源分区。通过将资源绑定到与资源集关联的池，可获取对此资源的访问权限。绑定可以在登录时执行，也可以由拥有 `PRIV_SYS_RES_CONFIG` 特权的管理员手动执行。

cpu.pinned 属性约束

cpu-pinned 属性指明，DRP 不应从特定 CPU 所在的处理器集中移动该 CPU。您可以设置此 libpool 属性，以最大化在处理器集中执行的特定应用程序的高速缓存利用率。

有关此属性的更多详细信息，请参见表 12–1。

pool.importance 属性约束

pool.importance 属性描述了池的相对重要性，该重要性由管理员定义。

配置目标

目标的指定方式与约束类似。表 12–1 中记录了完整的一组目标。

有两种类别的目标。

与工作负荷有关　与工作负荷有关的目标是指将随系统上运行的工作负荷的性质而变化的目标。utilization 目标便是一个示例。资源集的使用率数字将随此集中的活动工作负荷的性质而变化。

与工作负荷无关　与工作负荷无关的目标是指不会随系统上运行的工作负荷的性质而变化的目标。CPU locality 目标便是一个示例。资源集邻近性的评估标准不随此集中的活动工作负荷的性质而变化。

您可以定义三种类型的目标。

名称	有效元素	运算符	值
wt-load	system	N/A	N/A
locality	pset	N/A	loose | tight | none
utilization	pset	< > ~	0–100%

目标存储在 libpool 配置内的属性字符串中。这些属性名如下所示：

- system.poold.objectives
- pset.poold.objectives

目标的语法如下：

- objectives = objective [; objective]*
- objective = [n:] keyword [op] [value]

所有目标都有可选的重要性前缀。重要性用作目标的乘数，因此可增加它对目标函数评估的影响程度。范围从 0 到 INT64_MAX (9223372036854775807)。如果未指定，则缺省的重要性值为 1。

某些元素类型支持多种目标类型。pset 便是一个示例。您可以为这些元素指定多种目标类型，还可以针对单个 pset 元素指定多个使用率目标。

有关使用情况的示例，请参见第 157 页中的 “如何定义配置目标”。

wt-load 目标

wt-load 目标优先考虑资源分配与资源使用率匹配的配置。当此目标处于活动状态时，将为使用多个资源的资源集提供更多资源。wt-load 表示**加权负载**。

使用此目标的前提是：满足使用最小和最大值属性建立的约束，并希望守护进程在遵守这些约束的情况下自由处理资源。

locality 目标

locality 目标会影响由地址组 (lgroup) 数据度量的邻近性对选定配置的影响。邻近性的另一个定义是延迟。lgroup 描述了 CPU 资源和内存资源。Oracle Solaris 系统使用 lgroup 以时间为度量值来确定资源之间的距离。有关地址组摘要的更多信息，请参见《Programming Interfaces Guide》中的“Locality Groups Overview”。

此目标可采用以下三个值之一：

tight　如果设置，则优先考虑最大化资源邻近性的配置。

loose　如果设置，则优先考虑最小化资源邻近性的配置。

none　如果设置，则优先考虑配置时不受资源邻近性的影响。这是 locality 目标的缺省值。

通常，locality 目标应设置为 tight。但是，为了最大化内存带宽或最小化 DR 操作对资源集的影响，可以将此目标设置为 loose，也可以使其保留缺省设置 none。

utilization 目标

utilization 目标优先考虑将资源分配给未满足指定使用率目标的分区的配置。

此目标使用运算符和值来指定。运算符如下：

<　“小于”运算符表明指定的值为最大目标值。

>　“大于”运算符表明指定的值为最小目标值。

~　“约等于”运算符表明指定的值是可在一定程度上上下浮动的目标值。

对于每种运算符类型，pset 只能设置一个 utilization 目标。

- 如果设置了 ~ 运算符，则不能设置 < 和 > 运算符。

- 如果设置了 < 和 > 运算符，则不能设置 ~ 运算符。请注意，< 运算符和 > 运算符的设置不能互相冲突。

您可以同时设置 < 和 > 运算符来创建一个范围。要验证值以确保它们不重叠。

配置目标示例

在以下示例中，`poold` 将为 pset 评估这些目标：

- `utilization` 应保持在 30% 到 80% 之间。
- 应将处理器集的 `locality` 最大化。
- 目标应采用缺省重要性 `1`。

示例 12–1 `poold` 目标示例

```
pset.poold.objectives "utilization > 30; utilization < 80; locality tight"
```

有关其他使用情况的示例，请参见第 157 页中的 “如何定义配置目标”。

poold 属性

有四种类别的属性：

- 配置
- 约束
- 目标
- 目标参数

表 12–1 定义的属性名

属性名	类型	类别	说明
`system.poold.log-level`	字符串	配置	日志级别
`system.poold.log-location`	字符串	配置	日志位置
`system.poold.monitor-interval`	uint64	配置	监视抽样间隔
`system.poold.history-file`	字符串	配置	决策历史记录的位置
`pset.max`	uint64	约束	此处理器集的最大 CPU 数
`pset.min`	uint64	约束	此处理器集的最小 CPU 数
`cpu.pinned`	布尔型	约束	固定到此处理器集的 CPU

表 12–1 定义的属性名 （续）

属性名	类型	类别	说明
system.poold.objectives	字符串	目标	遵循 poold 的目标表达式语法的格式化字符串
pset.poold.objectives	字符串	目标	遵循 poold 的表达式语法的格式化字符串
pool.importance	int64	目标参数	用户指定的重要性

可以配置的 poold 功能

您可以对守护进程行为的以下方面进行配置。

- 监视间隔
- 日志级别
- 日志位置

这些选项在池配置中指定。您也可以通过调用 poold，从命令行控制日志级别。

poold 监视间隔

使用属性名 system.poold.monitor-interval 可以指定以毫秒为单位的值。

poold 日志信息

通过日志可提供三种类别的信息。日志中标识了这些类：

- 配置
- 监视
- 优化

使用属性名 system.poold.log-level 可以指定日志参数。如果未指定此属性，则缺省的日志级别为 NOTICE。参数级别具有层次结构。设置 DEBUG 的日志级别会让 poold 记录所有定义的消息。INFO 级别为多数管理员提供了有用的信息平衡。

您可以使用带有 -l 选项的 poold 命令以及参数在命令行中指定生成的日志信息级别。

以下参数为可用参数：

- ALERT
- CRIT
- ERR
- WARNING

- NOTICE
- INFO
- DEBUG

参数级别直接映射到其 syslog 对等项上。有关使用 syslog 的更多信息，请参见第 141 页中的 “日志位置”。

有关如何配置 poold 日志的更多信息，请参见第 159 页中的 “如何设置 poold 日志级别”。

配置信息日志

可以生成以下类型的消息：

ALERT　访问 libpool 配置时出现的问题，或者是 libpool 功能的其他一些基本、无法预测的故障。它会导致守护进程退出，需要管理员立即关注。

CRIT　由于无法预测的故障产生的问题。它会导致守护进程退出，需要管理员立即关注。

ERR　用于控制操作的用户指定参数出现的问题，如资源集的相互冲突且无法解决的使用率目标。需要管理性介入来更正目标。poold 尝试通过忽略相冲突的目标来采取补救措施，但有些错误会导致守护进程退出。

WARNING　与配置参数的设置相关的警告，即使从技术角度来说是正确的，但可能不适合指定的执行环境。例如将所有 CPU 资源标记为固定，这意味着 poold 不能在处理器集之间移动 CPU 资源。

DEBUG　包含进行配置调试时所需详细信息的消息。通常情况下，管理员不使用此信息。

监视信息日志

可以生成以下类型的消息：

CRIT　由于无法预测的监视故障产生的问题。它会导致守护进程退出，需要管理员立即关注。

ERR　由于无法预测的监视错误产生的问题。可请管理员来干预和更正。

NOTICE　有关资源控制区转换的消息。

INFO　有关资源使用率统计信息的消息。

DEBUG　包含进行监视调试时所需详细信息的消息。通常情况下，管理员不使用此信息。

优化信息日志

可以生成以下类型的消息：

WARNING　可显示有关做出最佳决策的问题的消息。例如可能包括受最小值和最大值或固定的组件数严格约束的资源集。

可显示与执行最佳分配时由于无法预测的限制而产生的问题相关的消息。例如从包含绑定资源使用者的处理器集上移除最后一个处理器。

NOTICE　可显示有关可用配置或由于会覆盖决策历史记录而未能实现的配置的消息。

INFO　可显示有关可考虑的备用配置的消息。

DEBUG　包含进行优化调试时所需详细信息的消息。通常情况下，管理员不使用此信息。

日志位置

system.poold.log-location 属性用于指定 poold 记录的输出的位置。您可以为 poold 输出指定 SYSLOG 的位置（请参见 syslog(3C)）。

如果未指定此属性，则 poold 记录的输出的缺省位置为 /var/log/pool/poold。

当从命令行调用 poold 时，不使用此属性。日志条目将写入发出调用的终端上的 stderr。

使用 logadm 管理日志

如果 poold 处于活动状态，则 logadm.conf 文件将包含管理缺省文件 /var/log/pool/poold 的条目。此条目为：

```
/var/log/pool/poold -N -s 512k
```

请参见 logadm(1M) 和 logadm.conf(4) 手册页。

动态资源分配如何工作

本节介绍了 poold 用来动态分配资源的进程和因素。

关于可用资源

可用资源即为可在 poold 进程的范围内使用的所有资源。控制的范围最多为一个 Oracle Solaris 实例。

在启用区域的系统上，poold 执行实例的范围限制为全局区域。

确定可用资源

资源池包含可供应用程序使用的所有系统资源。

对一个单独执行的 Oracle Solaris 实例来说，必须将单一类型的资源（如 CPU）分配到单个分区上。对于每种资源类型，可以有一个或多个分区。每个分区包含一个唯一的资源集。

例如，装有四个 CPU 和两个处理器集的计算机可以具有以下设置：

pset 0： 0 1

pset 1： 2 3

其中，冒号后的 0、1、2 和 3 表示 CPU ID。请注意，这两个处理器集包含了所有四个 CPU。

同样的计算机不能具有以下设置：

pset 0： 0 1

pset 1： 1 2 3

不能使用这种设置，因为 CPU 1 一次只能出现在一个 pset 中。

不能从资源所属分区以外的任何分区来访问资源。

要搜索可用资源，poold 需要询问活动池的配置来查找分区。所有分区内的所有资源的总和决定所控制的每种资源类型的可用资源总量。

此资源量是 poold 操作过程中使用的基本数字。但是，对此数字存在一些约束，限制了 poold 在进行分配时的灵活性。有关可用约束的信息，请参见第 135 页中的 “配置约束”。

识别资源不足

poold 的控制范围定义为 poold 对其有效分区和管理具有主要责任的可用资源集。但是，其他可在此控制范围内处理资源的机制仍会影响配置。如果在 poold 处于活动状态时某个分区不受控制，则 poold 会尝试通过对可用资源的审慎操作来恢复控制。如果 poold 在其范围内无法找到其他资源，则守护进程将记录有关资源不足的信息。

确定资源利用率

poold 通常情况下会使用最多的时间在其控制范围内观察资源的使用情况。执行这种监视是为了验证是否满足了与工作负荷有关的目标。

例如，对于处理器集来说，在此集中的所有处理器都会进行度量。资源利用率显示了在抽样间隔内资源被使用的时间比例。资源利用率显示为 0 到 100 的百分比。

识别控制违规

第 135 页中的 “配置约束和目标”中所述的指令用于检测系统即将出现的进而无法满足其目标的故障。这些目标与工作负荷直接相关。

未满足用户配置目标的分区即为控制违规。控制违规的两种类型为同步违规和异步违规。

- 目标的同步违规由守护进程在监视工作负荷的过程中进行检测。
- 目标的异步违规的出现与守护进程执行的监视操作无关。

以下事件将导致异步目标违规：

- 向控制范围中添加资源或从中删除资源。
- 重新配置控制范围。
- 重新启动 poold 资源控制器。

假定与工作负荷无关的目标的影响在目标函数的评估期间保持不变。与工作负荷无关的目标仅在其中一个异步违规触发重新评估时才会再次评估。

确定适当的补救措施

当资源控制器确定某个资源使用者的资源不足时，第一反应就是增加资源以改善性能。

此时将检查并评估在控制范围的配置中指定的满足目标的备用配置。

由于针对响应监视了变化的资源并评估了每个资源分区，因此，此进程会随着时间不断完善。可参阅决策历史记录，以避免再次使用过去在获取目标函数方面未带来任何改进的配置。其他信息（如进程名称和数量）用于进一步评估历史数据的实用性。

如果守护进程不能进行更正操作，则会记录此情况。有关更多信息，请参见第 139 页中的 “poold 日志信息”。

使用 poolstat 监视池功能和资源利用率

poolstat 实用程序用于在系统上启用池的情况下监视资源利用率。此实用程序会重复检查系统上所有活动的池，并基于选定的输出模式来报告统计信息。通过 poolstat 统计信息，您可以确定哪些资源分区过度使用。您可以分析这些统计信息，做出有关在系统处于资源压力下时资源重新分配的决策。

poolstat 实用程序包括可用于检查特定池并报告资源集特定的统计信息的选项。

如果您在系统上实现区域并且在非全局区域中使用 poolstat，则会显示有关与此区域的池关联的资源的信息。

有关 poolstat 实用程序的更多信息，请参见 poolstat(1M) 手册页。有关 poolstat 任务和使用情况的信息，请参见第 164 页中的 “使用 poolstat 报告与池相关的资源统计信息”。

poolstat 输出

在缺省输出格式下，poolstat 会输出一个标题行，然后为每个池显示一行信息。池信息行以池 ID 和池名称开头，后接一列连接到池上的处理器集的统计数据。附加在多个池上的资源集将多次显示，一次显示一个池的资源集。

列标题如下：

id　池 ID。

pool　池名。

rid　资源集 ID。

rset　资源集名。

type　资源集类型。

min　资源集大小的最小值。

max　资源集大小的最大值。

size　当前资源集大小。

used　当前资源集使用量的度量。

此使用量的计算方法为资源集的利用率百分比乘以资源集大小。如果资源集在上次抽样间隔期间已重新配置，则可能不报告该值。未报告的值以连字符 (-) 的形式出现。

load　资源集上的负荷的完全表示。

有关此属性的更多信息，请参见 libpool(3LIB) 手册页。

您可以在 poolstat 输出中指定以下内容：

- 列的顺序
- 显示的标题

调整 poolstat 操作间隔

您可以定制 poolstat 执行的操作。您可以设置报告的抽样间隔并指定统计信息重复的次数。

interval 调整 poolstat 执行的定期操作的间隔。所有间隔都以秒为单位指定。

count 指定统计信息重复的次数。缺省情况下，poolstat 仅报告一次统计信息。

如果未指定 *interval* 和 *count*，则报告一次统计信息。如果指定了 *interval* 而未指定 *count*，则会无限次地报告统计信息。

用于资源池功能的命令

下表中介绍的命令提供了池功能的主要管理接口。有关在启用了区域的系统上使用这些命令的信息，请参见第 129 页中的 “区域中使用的资源池”。

手册页参考	说明
pooladm(1M)	在系统上启用或禁用池功能。激活特定配置或删除当前配置，并将关联的资源返回到其缺省状态。如果在不带选项的情况下运行，则 pooladm 会显示当前的动态池配置。
poolbind(1M)	启用手动绑定功能，将项目、任务和进程绑定到资源池中。
poolcfg(1M)	提供对池和集的配置操作。使用此工具创建的配置通过使用 pooladm 在目标主机上进行实例化。 如果带 c 选项的 -info 子命令参数运行，则 poolcfg 会显示有关 /etc/pooladm.conf 中的静态配置的信息。如果添加了一个文件名参数，则此命令将显示有关命名文件中包含的静态配置的信息。例如，poolcfg -c info /tmp/newconfig 会显示有关 /tmp/newconfig 文件中包含的静态配置的信息。
poold(1M)	池系统守护进程。此守护进程使用系统目标和可查看的统计信息来达到管理员指定的系统性能目标。如果在未满足目标的情况下无法进行更正操作，则 poold 将记录此情况。
poolstat(1M)	显示与池相关的资源统计信息。简化性能分析并为系统管理员提供资源分区和重新分区任务方面的支持信息。提供了一些选项来检查指定的池并报告资源集特定的统计信息。

库 API 由 libpool 提供（请参见 libpool(3LIB) 手册页）。程序可使用库来处理池配置。

◆ ◆ ◆ 第 13 章

13 创建和管理资源池（任务）

本章介绍如何设置和管理系统上的资源池。

有关资源池的背景信息，请参见第 12 章，资源池（概述）。

管理资源池（任务列表）

任务	说明	参考
启用或禁用资源池。	激活或禁用系统上的资源池。	第 148 页中的“启用和禁用池功能”
启用或禁用动态资源池。	激活或禁用系统上的动态资源池功能。	第 148 页中的“启用和禁用池功能”
创建静态资源池配置。	创建与当前动态配置相匹配的静态配置文件。有关更多信息，请参见第 131 页中的“资源池框架”。	第 152 页中的“如何创建静态配置”
修改资源池配置。	修改系统上的池配置（例如通过创建其他池）。	第 153 页中的“如何修改配置”
将资源池与调度类关联。	将池与调度类关联，以便所有绑定到该池的进程都使用指定的调度程序。	第 155 页中的“如何将池与调度类关联”
设置配置约束和定义配置目标。	为 poold 指定目标以考虑何时执行更正操作。有关配置目标的更多信息，请参见第 134 页中的“poold 概述”。	第 157 页中的“如何设置配置约束”和第 157 页中的“如何定义配置目标”
设置日志级别。	指定 poold 生成的日志信息的级别。	第 159 页中的“如何设置 poold 日志级别”

任务	说明	参考
通过 poolcfg 命令使用文本文件。	poolcfg 命令可以从文本文件提取输入。	第 160 页中的“如何通过 poolcfg 使用命令文件”
在内核中传送资源。	在内核中传送资源。例如，将具有特定 ID 的资源传送到目标集。	第 160 页中的“传送资源”
激活池配置。	激活缺省配置文件中的配置。	第 161 页中的“如何激活池配置”
在提交池配置之前验证此配置。	验证池配置，以测试验证时将发生的情况。	第 161 页中的“如何在提交配置之前验证配置”
删除系统中的池配置。	将所有关联的资源（如处理器集）返回到其缺省状态。	第 161 页中的“如何删除池配置”
将进程绑定到池。	手动将系统上运行的进程与资源池关联。	第 162 页中的“如何将进程绑定到池”
将任务或项目绑定到池。	将任务或项目与资源池关联。	第 163 页中的“如何将任务或项目绑定到池”
将新进程绑定到资源池。	要将项目中的新进程自动绑定到指定的池，请向 project 数据库中的每个条目添加一个属性。	第 163 页中的“如何设置项目的 project.pool 属性”
使用 project 属性将进程绑定到其他池。	修改已启动的新进程的池绑定。	第 163 页中的“如何使用 project 属性将进程绑定到其他池”
使用 poolstat 实用程序生成报告。	在指定的间隔生成多个报告。	第 164 页中的“按特定间隔生成多个报告”
报告资源集统计信息。	使用 poolstat 实用程序报告 pset 资源集的统计信息。	第 164 页中的“报告资源集统计信息”

启用和禁用池功能

您可以使用在 svcadm(1M) 手册页中所述的 svcadm 命令在系统中启用和禁用资源池和动态资源池服务。

您还可以使用 pooladm(1M) 手册页中所述的 pooladm 命令执行以下任务：

- 启用池功能以对池进行处理
- 禁用池功能以便不能对池进行处理

注 – 在升级系统时，如果启用了资源池框架，而且 /etc/pooladm.conf 文件存在，则池服务将被启用，该文件中包含的配置将应用到系统中。

▼ 如何使用 svcadm 启用资源池服务

1. 成为 root 用户或承担等效角色。
2. 启用资源池服务。

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

▼ 如何使用 svcadm 禁用资源池服务

1. 成为 root 用户或承担等效角色。
2. 禁用资源池服务。

```
# svcadm disable system/pools:default
```

▼ 如何使用 svcadm 启用动态资源池服务

1. 成为 root 用户或承担等效角色。
2. 启用动态资源池服务。

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

示例 13–1 动态资源池服务对资源池服务的依赖性

本示例表明，如果要运行 DRP，则必须首先启用资源池。

资源池和动态资源池之间存在相关性。DRP 现在是资源池的一项相关服务。DRP 可以独立于资源池单独启用和禁用。

以下显示表明，当前已禁用了资源池和动态资源池：

```
# svcs "*pool*"
STATE          STIME    FMRI
disabled       2011     svc:/system/pools:default
disabled       2011     svc:/system/pools/dynamic:default
```

启用动态资源池：

```
# svcadm enable svc:/system/pools/dynamic:default
# svcs -a | grep pool
STATE          STIME    FMRI
disabled       2011     svc:/system/pools:default
offline        2011     svc:/system/pools/dynamic:default
```

请注意，DRP 服务仍处于脱机状态。

可使用 svcs 命令的 -x 选项确定 DRP 服务处于脱机状态的原因：

```
# svcs -x "*pool*"
svc:/system/pools:default (resource pools framework)
 State: disabled since Sat Feb 12 02:36:15 2011
Reason: Disabled by an administrator.
   See: http://support.oracle.com/msg/SMF-8000-05
   See: libpool(3LIB)
   See: pooladm(1M)
   See: poolbind(1M)
   See: poolcfg(1M)
   See: poolstat(1M)
Impact: This service is not running.

svc:/system/pools/dynamic:default (dynamic resource pools)
 State: disabled since Sat Feb 12 02:36:16 2011
Reason: Disabled by an administrator.
   See: http://support.oracle.com/msg/SMF-8000-05
   See: poold(1M)
Impact: This service is not running.
```

启用资源池服务，以便 DRP 服务可以运行：

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

在使用 svcs "*pool* 命令时，系统将显示如下内容：

```
# svcs "*pool*"
STATE          STIME    FMRI
online         2011     svc:/system/pools/dynamic:default
online         2011     svc:/system/pools:default
```

示例 13–2 资源池服务禁用时对动态资源池的影响

如果这两种服务都联机，并且您禁用了资源池服务：

```
# svcadm disable svc:/system/pools:default
```

在使用 svcs "*pool* 命令时，系统将显示如下内容：

```
# svcs "*pool*"
STATE          STIME    FMRI
disabled       2011     svc:/system/pools:default
online         2011     svc:/system/pools/dynamic:default
```

但最终，DRP 服务最终将转入 offline，原因是资源池服务已被禁用：

```
# svcs "*pool*"
STATE          STIME    FMRI
disabled       2011     svc:/system/pools:default
offline        2011     svc:/system/pools/dynamic:default
```

确定 DRP 服务处于脱机状态的原因：

```
# svcs -x "*pool*"
svc:/system/pools:default (resource pools framework)
 State: disabled since Sat Feb 12 02:36:15 2011
Reason: Disabled by an administrator.
   See: http://support.oracle.com/msg/SMF-8000-05
   See: libpool(3LIB)
   See: pooladm(1M)
   See: poolbind(1M)
   See: poolcfg(1M)
   See: poolstat(1M)
Impact: 1 dependent service is not running.  (Use -v for list.)

svc:/system/pools/dynamic:default (dynamic resource pools)
 State: offline since Sat Feb 12 02:36:15 2011
Reason: Service svc:/system/pools:default is disabled.
   See: http://support.oracle.com/msg/SMF-8000-GE
   See: poold(1M)
   See: /var/svc/log/system-pools-dynamic:default.log
Impact: This service is not running.
```

必须启动资源池，DRP 才能工作。例如，可以使用带有 `-e` 选项的 `pooladm` 命令来启动资源池：

```
# pooladm -e
```

然后，`svcs "*pool*"` 命令显示以下内容：

```
# svcs "*pool*"
STATE          STIME    FMRI
online         2011     svc:/system/pools:default
online         2011     svc:/system/pools/dynamic:default
```

▼ 如何使用 `svcadm` 禁用动态资源池服务

1. **成为 root 用户或承担等效角色。**
2. **禁用动态资源池服务。**

   ```
   # svcadm disable system/pools/dynamic:default
   ```

▼ 如何使用 `pooladm` 启用资源池

1. **成为 root 用户或承担等效角色。**
2. **启用池功能。**

   ```
   # pooladm -e
   ```

▼ 如何使用 pooladm 禁用资源池

1 **成为 root 用户或承担等效角色。**

2 **禁用池功能。**

```
# pooladm -d
```

配置池

▼ 如何创建静态配置

对 -/usr/sbin/pooladm 使用 s 选项可以创建与当前动态配置相匹配的静态配置文件，从而在每次重新引导时都能保留更改。如果没有指定其他文件名，则使用缺省位置 /etc/pooladm.conf。

使用带有 -c 选项的 pooladm 命令提交配置。然后，使用带有 -s 选项的 pooladm 命令更新静态配置，以便与动态配置的状态相匹配。

注 – 创建与动态配置相匹配的新配置时，应优先使用较晚的功能 pooladm -s，再考虑使用较早的功能 poolcfg -c discover。

开始之前 在系统上启用池。

1 **成为 root 用户或承担等效角色。**

2 **更新静态配置文件，以便与当前动态配置相匹配。**

```
# pooladm -s
```

3 **查看可读形式的配置文件的内容。**

请注意，配置中包含系统创建的缺省元素。

```
# poolcfg -c info
system tester
        string  system.comment
        int     system.version 1
        boolean system.bind-default true
        int     system.poold.pid 177916

        pool pool_default
                int     pool.sys_id 0
                boolean pool.active true
                boolean pool.default true
                int     pool.importance 1
```

```
                string  pool.comment
                pset    pset_default

        pset pset_default
                int     pset.sys_id -1
                boolean pset.default true
                uint    pset.min 1
                uint    pset.max 65536
                string  pset.units population
                uint    pset.load 10
                uint    pset.size 4
                string  pset.comment
                boolean testnullchanged true

                cpu
                        int     cpu.sys_id 3
                        string  cpu.comment
                        string  cpu.status on-line

                cpu
                        int     cpu.sys_id 2
                        string  cpu.comment
                        string  cpu.status on-line

                cpu
                        int     cpu.sys_id 1
                        string  cpu.comment
                        string  cpu.status on-line

                cpu
                        int     cpu.sys_id 0
                        string  cpu.comment
                        string  cpu.status on-line
```

4 **提交 /etc/pooladm.conf 中的配置。**

```
# pooladm -c
```

5 **（可选的）要将动态配置复制到名为 /tmp/backup 的静态配置文件，请键入以下命令：**

```
# pooladm -s /tmp/backup
```

▼ 如何修改配置

要增强配置，请创建名为 pset_batch 的处理器集以及名为 pool_batch 的池。然后，使用关联连接池和处理器集。

请注意，必须用引号将包含空格的子命令参数括起来。

1 **成为 root 用户或承担等效角色。**

2 **创建处理器集 pset_batch。**

```
# poolcfg -c 'create pset pset_batch (uint pset.min = 2; uint pset.max = 10)'
```

3 创建池 **pool_batch**。

```
# poolcfg -c 'create pool pool_batch'
```

4 使用关联连接池和处理器集。

```
# poolcfg -c 'associate pool pool_batch (pset pset_batch)'
```

5 显示已编辑的配置。

```
# poolcfg -c info
system tester
        string  system.comment kernel state
        int     system.version 1
        boolean system.bind-default true
        int     system.poold.pid 177916

        pool pool_default
                int     pool.sys_id 0
                boolean pool.active true
                boolean pool.default true
                int     pool.importance 1
                string  pool.comment
                pset    pset_default

        pset pset_default
                int     pset.sys_id -1
                boolean pset.default true
                uint    pset.min 1
                uint    pset.max 65536
                string  pset.units population
                uint    pset.load 10
                uint    pset.size 4
                string  pset.comment
                boolean testnullchanged true

                cpu
                        int     cpu.sys_id 3
                        string  cpu.comment
                        string  cpu.status on-line

                cpu
                        int     cpu.sys_id 2
                        string  cpu.comment
                        string  cpu.status on-line

                cpu
                        int     cpu.sys_id 1
                        string  cpu.comment
                        string  cpu.status on-line

                cpu
                        int     cpu.sys_id 0
                        string  cpu.comment
                        string  cpu.status on-line

        pool pool_batch
                boolean pool.default false
                boolean pool.active true
```

```
                int pool.importance 1
                string pool.comment
                pset pset_batch

        pset pset_batch
                int pset.sys_id -2
                string pset.units population
                boolean pset.default true
                uint pset.max 10
                uint pset.min 2
                string pset.comment
                boolean pset.escapable false
                uint pset.load 0
                uint pset.size 0

                cpu
                        int     cpu.sys_id 5
                        string  cpu.comment
                        string  cpu.status on-line

                cpu
                        int     cpu.sys_id 4
                        string  cpu.comment
                        string  cpu.status on-line
```

6 **提交 /etc/pooladm.conf 中的配置。**

```
# pooladm -c
```

7 **（可选的）要将动态配置复制到名为 /tmp/backup 的静态配置文件，请键入以下命令：**

```
# pooladm -s /tmp/backup
```

▼ 如何将池与调度类关联

您可以将池与调度类关联，以便所有绑定到该池的进程都可以使用此调度程序。为此，请将 pool.scheduler 属性设置为调度程序的名称。以下示例将池 pool_batch 与公平份额调度器 (fair share scheduler, FSS) 关联。

1 **成为 root 用户或承担等效角色。**

2 **修改池 pool_batch 以便与 FSS 关联。**

```
# poolcfg -c 'modify pool pool_batch (string pool.scheduler="FSS")'
```

3 **显示已编辑的配置。**

```
# poolcfg -c info
system tester
        string  system.comment
        int     system.version 1
        boolean system.bind-default true
        int     system.poold.pid 177916

        pool pool_default
```

```
        int     pool.sys_id 0
        boolean pool.active true
        boolean pool.default true
        int     pool.importance 1
        string  pool.comment
        pset    pset_default

pset pset_default
        int     pset.sys_id -1
        boolean pset.default true
        uint    pset.min 1
        uint    pset.max 65536
        string  pset.units population
        uint    pset.load 10
        uint    pset.size 4
        string  pset.comment
        boolean testnullchanged true

        cpu
                int     cpu.sys_id 3
                string  cpu.comment
                string  cpu.status on-line

        cpu
                int     cpu.sys_id 2
                string  cpu.comment
                string  cpu.status on-line

        cpu
                int     cpu.sys_id 1
                string  cpu.comment
                string  cpu.status on-line

        cpu
                int     cpu.sys_id 0
                string  cpu.comment
                string  cpu.status on-line

pool pool_batch
        boolean pool.default false
        boolean pool.active true
        int pool.importance 1
        string pool.comment
        string pool.scheduler FSS
        pset batch

pset pset_batch
        int pset.sys_id -2
        string pset.units population
        boolean pset.default true
        uint pset.max 10
        uint pset.min 2
        string pset.comment
        boolean pset.escapable false
        uint pset.load 0
        uint pset.size 0

        cpu
                int     cpu.sys_id 5
```

```
                string  cpu.comment
                string  cpu.status on-line

        cpu
                int     cpu.sys_id 4
                string  cpu.comment
                string  cpu.status on-line
```

4 **提交 `/etc/pooladm.conf` 中的配置：**

```
# pooladm -c
```

5 **（可选的）要将动态配置复制到名为 `/tmp/backup` 的静态配置文件，请键入以下命令：**

```
# pooladm -s /tmp/backup
```

▼ 如何设置配置约束

约束通过删除一些可能会对配置进行的潜在更改来影响可能配置的范围。此过程显示如何设置 `cpu.pinned` 属性。

在以下示例中，`cpuid` 是一个整数。

1 **成为 root 用户或承担等效角色。**

2 **修改静态或动态配置中的 `cpu.pinned` 属性：**

- **修改引导时（静态）配置：**

  ```
  # poolcfg -c 'modify cpu <cpuid> (boolean cpu.pinned = true)'
  ```

- **不修改引导时配置而修改运行（动态）配置：**

  ```
  # poolcfg -dc 'modify cpu <cpuid> (boolean cpu.pinned = true)'
  ```

▼ 如何定义配置目标

您可以为 `poold` 指定目标，以考虑何时执行更正操作。

在以下过程中，将设置 `wt-load` 目标，以便 `poold` 尝试将资源分配与资源利用率相匹配。禁用 `locality` 目标有助于实现此配置目标。

1 **成为 root 用户或承担等效角色。**

2 **修改系统 `tester` 以优先考虑 `wt-load` 目标。**

```
# poolcfg -c 'modify system tester (string system.poold.objectives="wt-load")'
```

3 **禁用缺省处理器集的 `locality` 目标。**

```
# poolcfg -c 'modify pset pset_default (string pset.poold.objectives="locality none")'    one line
```

4 禁用 **pset_batch** 处理器集的 **locality** 目标。

```
# poolcfg -c 'modify pset pset_batch (string pset.poold.objectives="locality none")'
```
one line

5 显示已编辑的配置。

```
# poolcfg -c info
system tester
        string  system.comment
        int     system.version 1
        boolean system.bind-default true
        int     system.poold.pid 177916
        string  system.poold.objectives wt-load

        pool pool_default
                int     pool.sys_id 0
                boolean pool.active true
                boolean pool.default true
                int     pool.importance 1
                string  pool.comment
                pset    pset_default

        pset pset_default
                int     pset.sys_id -1
                boolean pset.default true
                uint    pset.min 1
                uint    pset.max 65536
                string  pset.units population
                uint    pset.load 10
                uint    pset.size 4
                string  pset.comment
                boolean testnullchanged true
                string  pset.poold.objectives locality none

                cpu
                        int     cpu.sys_id 3
                        string  cpu.comment
                        string  cpu.status on-line

                cpu
                        int     cpu.sys_id 2
                        string  cpu.comment
                        string  cpu.status on-line

                cpu
                        int     cpu.sys_id 1
                        string  cpu.comment
                        string  cpu.status on-line

                cpu
                        int     cpu.sys_id 0
                        string  cpu.comment
                        string  cpu.status on-line

        pool pool_batch
                boolean pool.default false
                boolean pool.active true
                int pool.importance 1
                string pool.comment
```

```
                string pool.scheduler FSS
                pset batch

        pset pset_batch
                int pset.sys_id -2
                string pset.units population
                boolean pset.default true
                uint pset.max 10
                uint pset.min 2
                string pset.comment
                boolean pset.escapable false
                uint pset.load 0
                uint pset.size 0
                string  pset.poold.objectives locality none

                cpu
                        int     cpu.sys_id 5
                        string  cpu.comment
                        string  cpu.status on-line

                cpu
                        int     cpu.sys_id 4
                        string  cpu.comment
                        string  cpu.status on-line
```

6 **提交 /etc/pooladm.conf 中的配置。**

```
# pooladm -c
```

7 **（可选的）要将动态配置复制到名为 /tmp/backup 的静态配置文件，请键入以下命令：**

```
# pooladm -s /tmp/backup
```

▼ 如何设置 poold 日志级别

要指定 poold 生成的日志信息的级别，请在 poold 配置中设置 system.poold.log-level 属性。poold 配置保存在 libpool 配置中。有关更多信息，请参见第 139 页中的 “poold 日志信息”以及 poolcfg(1M) 和 libpool(3LIB) 手册页。

您还可以在命令行中使用 poold 命令，以指定 poold 生成的日志信息的级别。

1 **成为 root 用户或承担等效角色。**

2 **使用带有 -l 选项以及参数（如 INFO）的 poold 命令设置日志级别。**

```
# /usr/lib/pool/poold -l INFO
```

有关可用参数的信息，请参见第 139 页中的 “poold 日志信息”。缺省日志级别为 NOTICE。

▼ 如何通过 poolcfg 使用命令文件

带有 -f 选项的 poolcfg 命令可以从包含 -c 选项的 poolcfg 子命令参数的文本文件提取输入。此方法适用于要执行一组操作的情况。当处理多个命令时，仅在所有命令都成功的情况下才会更新配置。对于庞大或复杂的配置，此技术比调用每个子命令更有用。

请注意，在命令文件中，# 字符用作注释标记，表示其后面的内容为注释。

1. **创建输入文件 poolcmds.txt。**

```
$ cat > poolcmds.txt
create system tester
create pset pset_batch (uint pset.min = 2; uint pset.max = 10)
create pool pool_batch
associate pool pool_batch (pset pset_batch)
```

2. **成为 root 用户或承担等效角色。**

3. **执行命令：**

```
# /usr/sbin/poolcfg -f poolcmds.txt
```

传送资源

使用 poolcfg（带有 -d 选项）的 -c 选项的 transfer 子命令参数可以在内核中传送资源。-d 选项指定此命令直接对内核执行操作，而不从文件提取输入。

以下过程将两个 CPU 从内核中的处理器集 pset1 移动到处理器集 pset2。

▼ 如何在处理器集之间移动 CPU

1. **成为 root 用户或承担等效角色。**

2. **将两个 CPU 从 pset1 移动到 pset2。**

   可以按任意顺序使用 from 和 to 子句。每个命令只支持一个 to 和 from 子句。

```
# poolcfg -dc 'transfer 2 from pset pset1 to pset2'
```

示例 13–3 在处理器集之间移动 CPU 的替换方法

如果要传送资源类型的特定已知 ID，请提供其他语法。例如，以下命令为 pset_large 处理器集指定 ID 分别为 0 和 2 的两个 CPU：

```
# poolcfg -dc 'transfer to pset pset_large (cpu 0; cpu 2)'
```

更多信息 **疑难解答**

如果由于没有足够的资源可满足请求或者无法找到指定的 ID 而使传送失败，则系统将显示一条错误消息。

激活和删除池配置

使用 `pooladm` 命令可以激活特定的池配置或删除当前活动的池配置。有关此命令的更多信息，请参见 `pooladm(1M)` 手册页。

▼ 如何激活池配置

要激活缺省配置文件 `/etc/pooladm.conf` 中的配置，请调用带有 `-c` 选项（提交配置）的 `pooladm`。

1. **成为 root 用户或承担等效角色。**
2. **提交 `/etc/pooladm.conf` 中的配置。**

   ```
   # pooladm -c
   ```

3. **（可选的）将动态配置复制到静态配置文件，例如 `/tmp/backup`。**

   ```
   # pooladm -s /tmp/backup
   ```

▼ 如何在提交配置之前验证配置

您可以使用 `-n` 选项和 `-c` 选项来测试验证时将发生的情况。配置实际上将不会提交。

以下命令尝试验证 `/home/admin/newconfig` 中包含的配置。所有遇到的错误情况都将显示，但是不会修改配置本身。

1. **成为 root 用户或承担等效角色。**
2. **在提交配置之前测试此配置的有效性。**

   ```
   # pooladm -n -c /home/admin/newconfig
   ```

▼ 如何删除池配置

要删除当前的活动配置并使所有的关联资源（如处理器集）都恢复为缺省状态，请使用表示“删除配置”的 `-x` 选项。

1. **成为 root 用户或承担等效角色。**

2 **删除当前活动配置。**

```
# pooladm -x
```

-pooladm 的 x 选项可从动态配置中删除所有用户定义的元素。所有资源将恢复到其缺省状态，并且所有池绑定将替换为与缺省池的绑定。

更多信息 在处理器集中混合调度类

您可以在同一处理器集中安全地混合 TS 和 IA 类中的进程。在一个处理器集中混合其他调度类可能会导致不可预测的结果。如果使用 pooladm -x 导致在一个处理器集中出现混合调度类，请使用 priocntl 命令将运行的进程移至其他调度类。请参见第 108 页中的 “如何将进程从 TS 类手动移至 FSS 类”。另请参见 priocntl(1) 手册页。

设置池属性并绑定到池

可以设置 project.pool 属性，以便将资源池与项目关联。

可通过两种方法将正在运行的进程绑定到池：

- 可以使用 poolbind(1M) 手册页中所述的 poolbind 命令将特定进程绑定到已命名的资源池。
- 可以使用 project 数据库中的 project.pool 属性来标识通过 newtask 命令启动的新登录会话或任务的池绑定。请参见 newtask(1)、projmod(1M) 和 project(4) 手册页。

▼ 如何将进程绑定到池

以下过程使用带有 -p 选项的 poolbind 将进程（在此例中为当前 shell）手动绑定到名为 ohare 的池。

1 **成为 root 用户或承担等效角色。**

2 **将进程手动绑定到池：**

```
# poolbind -p ohare $$
```

3 **使用带有 -q 选项的 poolbind 验证进程的池绑定。**

```
$ poolbind -q $$
155509 ohare
```

系统将显示进程 ID 和池绑定。

▼ 如何将任务或项目绑定到池

要将任务或项目绑定到池，请使用带有 -i 选项的 poolbind 命令。以下示例将 airmiles 项目中的所有进程绑定到 laguardia 池。

1 **成为 root 用户或承担等效角色。**

2 **将 airmiles 项目中的所有进程绑定到 laguardia 池。**

```
# poolbind -i project -p laguardia airmiles
```

▼ 如何设置项目的 project.pool 属性

您可以设置 project.pool 属性，以将项目的进程绑定到资源池。

1 **成为 root 用户或承担等效角色。**

2 **将 project.pool 属性添加到 project 数据库中的每个条目。**

```
# projmod -a -K project.pool=poolname  project
```

▼ 如何使用 project 属性将进程绑定到其他池

假设配置中具有两个名为 studio 和 backstage 的池。/etc/project 文件具有以下内容：

```
user.paul:1024::::project.pool=studio
user.george:1024::::project.pool=studio
user.ringo:1024::::project.pool=backstage
passes:1027::paul::project.pool=backstage
```

使用此配置，可以在缺省情况下将用户 paul 启动的进程绑定到 studio 池。

用户 paul 可以为其启动的进程修改池绑定。paul 也可以使用 newtask，通过在 passes 项目中启动来将工作绑定到 backstage 池。

1 **在 passes 项目中启动进程。**

```
$ newtask -l -p passes
```

2 **使用带有 -q 选项的 poolbind 命令验证进程的池绑定。还可使用双美元符号 ($$) 将父级 shell 的进程号传递给该命令。**

```
$ poolbind -q $$
6384  pool backstage
```

系统将显示进程 ID 和池绑定。

使用 poolstat 报告与池相关的资源统计信息

poolstat 命令用于显示与池相关的资源的统计信息。有关更多信息，请参见第 144 页中的 “使用 poolstat 监视池功能和资源利用率”和 poolstat(1M) 手册页。

以下各小节通过示例说明如何生成用于特定用途的报告。

显示缺省的 poolstat 输出

键入不带参数的 poolstat 将针对每个池输出一个标题行和一行信息。信息行将显示池 ID、池的名称以及连接到池的处理器集的资源统计信息。

```
machine% poolstat
                           pset
       id pool           size used load
        0 pool_default      4  3.6  6.2
        1 pool_sales        4  3.3  8.4
```

按特定间隔生成多个报告

以下命令按 5 秒的抽样间隔生成 3 个报告。

```
machine% poolstat 5 3
                            pset
 id pool                  size used load
 46 pool_sales               2  1.2  8.3
  0 pool_default             2  0.4  5.2
                            pset
 id pool                  size used load
 46 pool_sales               2  1.4  8.4
  0 pool_default             2  1.9  2.0
                            pset
 id pool                  size used load
 46 pool_sales               2  1.1  8.0
  0 pool_default             2  0.3  5.0
```

报告资源集统计信息

以下示例使用带有 -r 选项的 poolstat 命令报告处理器集资源集的统计信息。请注意，资源集 pset_default 连接到多个池，因此此处理器集将针对每个池成员关系列出一次。

```
machine% poolstat -r pset
      id pool          type rid rset          min  max size used load
       0 pool_default  pset  -1 pset_default    1  65K    2  1.2  8.3
       6 pool_sales    pset   1 pset_sales      1  65K    2  1.2  8.3
       2 pool_other    pset  -1 pset_default    1  10K    2  0.4  5.2
```

◆ ◆ ◆ 第 14 章

资源管理配置示例

本章概述了资源管理框架，并介绍虚拟的服务器整合项目。

本章包含以下主题：

要整合的配置

在此示例中，要将五个应用程序整合到单个系统中。目标应用程序具有不同的资源需求、用户群和体系结构。当前，每个应用程序都位于目的在于满足应用程序要求的专用服务器上。下表介绍了各个应用程序及其特征。

应用程序说明	特征
应用服务器	CPU 超过 2 个时，可伸缩性会降低
应用服务器的数据库实例	超负荷的事务处理
测试和开发环境中的应用服务器	基于 GUI，并且执行未经测试的代码
事务处理服务器	主要顾虑是响应时间
独立数据库实例	处理大量事务并为多个时区提供服务

整合配置

下面的配置用来将应用程序整合到启用了资源池和动态资源池设备的单个系统中。

- 应用服务器具有一个双 CPU 处理器集。
- 将应用服务器的数据库实例和独立数据库实例整合到一个至少具有四个 CPU 的处理器集中。保证为独立数据库实例留出 75% 的资源。
- 测试和开发应用服务器需要 IA 调度类，以确保 UI 的响应。对内存强加限制，以减轻错误代码造成的影响。
- 将事务处理服务器指定给一个至少具有两个 CPU 的专用处理器集，以最大程度地缩短响应时间。

此配置适用于执行和占用每个资源集中的处理器时钟周期的已知应用程序。因此，可以建立约束，以便将处理器资源转移到需要资源的集中。

- `wt-load` 目标设置为允许高利用率资源集比低利用率资源集获得更多的资源分配。
- `locality` 目标设置为 `tight`，这用于最大化处理器的邻近性。

此外还应用了其他约束，以防止利用率超过任何资源集的 80%。此约束确保应用程序可以访问所需的资源。此外，对于事务处理器集，保持利用率低于 80% 的目标的重要性是指定的任何其他目标的两倍。这种重要性将在配置中定义。

创建配置

编辑 `/etc/project` 数据库文件。添加条目以实现所需的资源控制并将用户映射到资源池，然后查看此文件。

```
# cat /etc/project
.
.
.
user.app_server:2001:Production Application Server:::project.pool=appserver_pool
user.app_db:2002:App Server DB:::project.pool=db_pool;project.cpu-shares=(privileged,1,deny)
development:2003:Test and development::staff:project.pool=dev_pool;
process.max-address-space=(privileged,536870912,deny)     keep with previous line
user.tp_engine:2004:Transaction Engine:::project.pool=tp_pool
user.geo_db:2005:EDI DB:::project.pool=db_pool;project.cpu-shares=(privileged,3,deny)
.
.
.
```

注 – 开发小组必须执行开发项目中的任务，因为对此项目的访问基于用户的组 ID (group ID, GID)。

创建名为 `pool.host` 的输入文件，此文件将用于配置所需的资源池。查看此文件。

```
# cat pool.host
create system host
create pset dev_pset (uint pset.min = 0; uint pset.max = 2)
create pset tp_pset (uint pset.min = 2; uint pset.max=8)
create pset db_pset (uint pset.min = 4; uint pset.max = 6)
create pset app_pset (uint pset.min = 1; uint pset.max = 2)
create pool dev_pool (string pool.scheduler="IA")
create pool appserver_pool (string pool.scheduler="TS")
create pool db_pool (string pool.scheduler="FSS")
create pool tp_pool (string pool.scheduler="TS")
associate pool dev_pool (pset dev_pset)
associate pool appserver_pool (pset app_pset)
associate pool db_pool (pset db_pset)
associate pool tp_pool (pset tp_pset)
modify system tester (string system.poold.objectives="wt-load")
modify pset dev_pset (string pset.poold.objectives="locality tight; utilization < 80")
modify pset tp_pset (string pset.poold.objectives="locality tight; 2: utilization < 80")
modify pset db_pset (string pset.poold.objectives="locality tight;utilization < 80")
modify pset app_pset (string pset.poold.objectives="locality tight; utilization < 80")
```

使用 `pool.host` 输入文件更新配置。

```
# poolcfg -f pool.host
```

使配置处于活动状态。

```
# pooladm -c
```

现在框架可在系统上正常运行。

启用 DRP。

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

查看配置

要查看框架配置（此配置还包含由系统创建的缺省元素），请键入：

```
# pooladm
system host
        string  system.comment
        int     system.version 1
        boolean system.bind-default true
        int     system.poold.pid 177916
        string  system.poold.objectives wt-load

        pool dev_pool
                int     pool.sys_id 125
                boolean pool.default false
                boolean pool.active true
                int     pool.importance 1
                string  pool.comment
                string  pool.scheduler IA
```

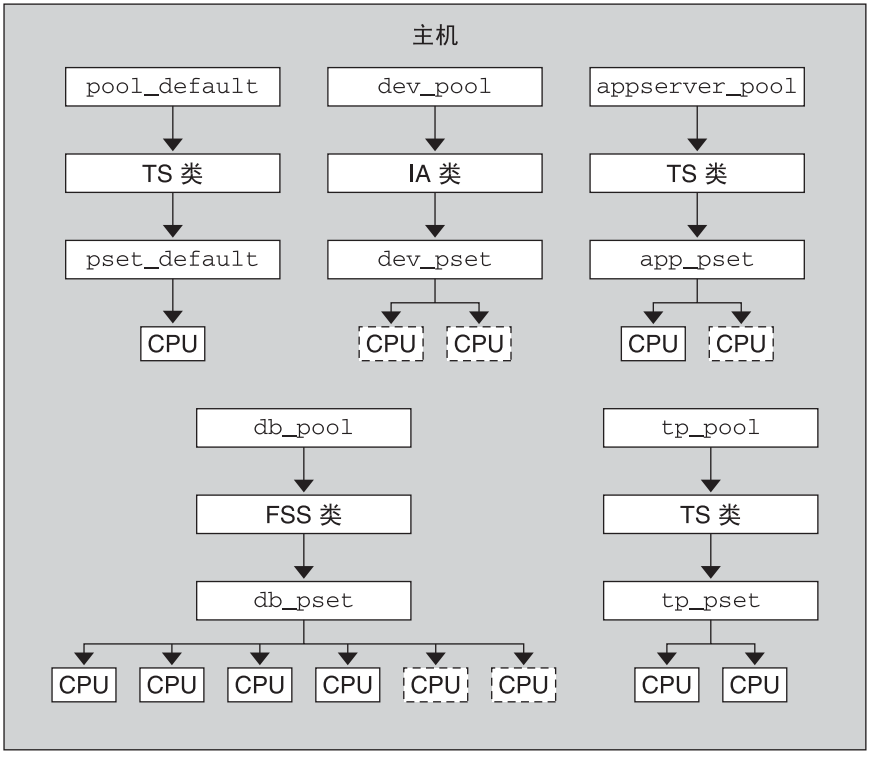
```
        pset    dev_pset

pool appserver_pool
        int     pool.sys_id 124
        boolean pool.default false
        boolean pool.active true
        int     pool.importance 1
        string  pool.comment
        string  pool.scheduler TS
        pset    app_pset

pool db_pool
        int     pool.sys_id 123
        boolean pool.default false
        boolean pool.active true
        int     pool.importance 1
        string  pool.comment
        string  pool.scheduler FSS
        pset    db_pset

pool tp_pool
        int     pool.sys_id 122
        boolean pool.default false
        boolean pool.active true
        int     pool.importance 1
        string  pool.comment
        string  pool.scheduler TS
        pset    tp_pset

pool pool_default
        int     pool.sys_id 0
        boolean pool.default true
        boolean pool.active true
        int     pool.importance 1
        string  pool.comment
        string  pool.scheduler TS
        pset    pset_default

pset dev_pset
        int     pset.sys_id 4
        string  pset.units population
        boolean pset.default false
        uint    pset.min 0
        uint    pset.max 2
        string  pset.comment
        boolean pset.escapable false
        uint    pset.load 0
        uint    pset.size 0
        string  pset.poold.objectives locality tight; utilization < 80

pset tp_pset
        int     pset.sys_id 3
        string  pset.units population
        boolean pset.default false
        uint    pset.min 2
        uint    pset.max 8
        string  pset.comment
        boolean pset.escapable false
        uint    pset.load 0
```

```
                uint    pset.size 0
                string  pset.poold.objectives locality tight; 2: utilization < 80

                cpu
                        int     cpu.sys_id 1
                        string  cpu.comment
                        string  cpu.status on-line

                cpu
                        int     cpu.sys_id 2
                        string  cpu.comment
                        string  cpu.status on-line

        pset db_pset
                int     pset.sys_id 2
                string  pset.units population
                boolean pset.default false
                uint    pset.min 4
                uint    pset.max 6
                string  pset.comment
                boolean pset.escapable false
                uint    pset.load 0
                uint    pset.size 0
                string  pset.poold.objectives locality tight; utilization < 80

                cpu
                        int     cpu.sys_id 3
                        string  cpu.comment
                        string  cpu.status on-line

                cpu
                        int     cpu.sys_id 4
                        string  cpu.comment
                        string  cpu.status on-line

                cpu
                        int     cpu.sys_id 5
                        string  cpu.comment
                        string  cpu.status on-line

                cpu
                        int     cpu.sys_id 6
                        string  cpu.comment
                        string  cpu.status on-line
        pset app_pset
                int     pset.sys_id 1
                string  pset.units population
                boolean pset.default false
                uint    pset.min 1
                uint    pset.max 2
                string  pset.comment
                boolean pset.escapable false
                uint    pset.load 0
                uint    pset.size 0
                string  pset.poold.objectives locality tight; utilization < 80
                cpu
                        int     cpu.sys_id 7
                        string  cpu.comment
                        string  cpu.status on-line
```

```
pset pset_default
        int     pset.sys_id -1
        string  pset.units population
        boolean pset.default true
        uint    pset.min 1
        uint    pset.max 4294967295
        string  pset.comment
        boolean pset.escapable false
        uint    pset.load 0
        uint    pset.size 0

        cpu
                int     cpu.sys_id 0
                string  cpu.comment
                string  cpu.status on-line
```

下面是框架的图形表示。

图 14–1 服务器整合配置



注 – 在池 `db_pool` 中，保证为独立数据库实例留出 75% 的 CPU 资源。

第 2 部分

Oracle Solaris Zones

此部分介绍 Oracle Solaris Zones 软件分区技术，该技术提供了一种虚拟化操作系统服务以创建运行应用程序的隔离环境的方法。这种隔离可阻止在一个区域中运行的进程监视或影响在其他区域中运行的进程。

◆ ◆ ◆ 第 15 章

15 Oracle Solaris Zones 介绍

Oracle Solaris 操作系统中的 Oracle Solaris Zones 功能提供了一个可在其中运行系统上应用程序的隔离环境。

本章概述了区域。

此外，还介绍了以下常规区域主题：

如果您可以开始在系统上创建区域，请跳至第 16 章，非全局区域配置（概述）。

注 – 有关 Oracle Solaris 10 Zones 的信息，请参见第 3 部分。

有关在 Oracle Solaris Trusted Extensions 系统上使用区域的信息，请参见《Trusted Extensions 配置和管理》中的第 13 章 “在 Trusted Extensions 中管理区域”。

区域概述

Oracle Solaris Zones 分区技术用于虚拟化操作系统服务，提供安全的隔离环境以便运行应用程序。非全局区域，也称为**区域**，是在 Oracle Solaris 操作系统的一个单独实例中创建的一个虚拟化的操作系统环境。操作系统实例称为全局区域。

虚拟化的目标是从管理各个数据中心组件转变为管理资源池。成功的服务器虚拟可提高服务器利用率，改善服务器资产利用效率。服务器虚拟对于维护单独系统隔离的成功服务器整合项目也非常重要。

由于将多个主机和服务整合到单台计算机上的需求而产生了虚拟化。虚拟可通过共享硬件、基础结构和管理降低成本。益处具体如下：

- 增加了硬件利用率
- 大大提高了资源分配的灵活性
- 降低了功率要求
- 缩减了管理成本
- 降低了总体拥有成本
- 系统上各应用程序之间可设置管理和资源界限

创建区域时，便创建了一个应用程序执行环境，其中的进程与系统的其余部分相隔离。这种隔离阻止了在一个区域中运行的进程监视或影响在其他区域中运行的进程。对于正在运行的进程，即使具有 root 凭证也不能查看或影响其他区域中的活动。使用 Oracle Solaris Zones，您可以维护每台服务器一个应用程序的部署模式，同时共享硬件资源。

区域还提供了一个抽象层，用于分隔应用程序和部署这些应用程序的计算机的物理属性。这些属性的示例包括物理设备路径。

可以在任何运行 Oracle Solaris 10 或更高 Oracle Solaris 发行版的计算机上使用区域。系统上区域数量的上限为 8192。单个系统上可有效托管的区域数量由所有区域中运行的应用程序软件的总资源需求和系统的大小确定。

这些概念在第 17 章，规划和配置非全局区域（任务）中介绍。

关于此版本的 Oracle Solaris Zones

本节概述了 Oracle Solaris Zones 的新增功能及对其所做的更改，包括自 Oracle Solaris 10 发行版以来所做的改进。

此发行版中的缺省非全局区域为 `solaris`，本指南及 `solaris(5)` 手册页中都有相关介绍。

Oracle Solaris 11.1 发行版已定义为受支持平台的所有体系结构均支持在单个全局区域中运行的非全局区域。

要检验 Oracle Solaris 发行版本和计算机体系结构，请键入：

```
#uname -r -m
```

`solaris` 区域使用标记区域框架（如 `brands(5)` 手册页中所述）运行与全局区域安装了相同软件的区域。使用 `solaris` 非全局区域时，系统软件必须始终与全局区域保持同步。区域中的系统软件包使用映像包管理系统 (Image Packaging System, IPS) 进行管理。IPS 是 Oracle Solaris 11 发行版中的包管理系统，`solaris` 区域使用这种模式。

在 Oracle Solaris 11 Express 发行版中创建的缺省 `ipkg` 区域将映射为 `solaris` 区域。请参见第 179 页中的 “关于将 ipkg 区域转换为 solaris 区域”。

在自动安装 (Automated Install, AI) 清单中指定的每个非全局区域将在客户机安装过程中进行安装和配置。非全局区域是在安装全局区域后首次重新引导时安装并配置的。当系统第一次引导时，区域自组装 (self-assembly) SMF 服务 `svc:/system/zones-install:default` 会配置并安装全局区域 AI 清单中定义的每个非全局区域。有关更多信息，请参见《安装 Oracle Solaris 11.1 系统》。也可以在已安装的 Oracle Solaris 系统上手动配置并安装区域。

对于软件包更新，应该通过使用 `--proxy` 选项在映像中设置持久性代理。如果未使用持久性映像代理配置，则可以设置 `http_proxy` 和 `https_proxy` 环境变量。

可以将区域配置为并行更新而不是串行更新。并行更新可大幅缩短更新系统上的所有区域所需的时间。

缺省情况下，使用专用 IP 类型创建区域。如果没有指定联网配置，可以通过 `anet` 资源将 VNIC 自动包含在区域配置中。有关更多信息，请参见第 197 页中的 “区域网络接口”。

共享存储上的区域具有 `zonecfg rootzpool` 资源。将区域封装到专用 zpool。共享存储上的区域访问和管理用于区域的共享存储资源。

用于指定基于 InfiniBand 的 IP (IP over InfiniBand, IPoIB) 数据链路的两个新属性可用于 `zonecfg anet` 资源。`solaris` 和 `solaris10` 标记区域均支持 IPoIB。

专用 IP 和共享 IP 非全局区域都支持可靠数据报套接字 (Reliable Datagram Socket, RDS) IPC 协议。

已扩展 `fsstat` 实用程序以支持区域。`fsstat` 实用程序提供每区域统计信息和聚合统计信息。

`solaris` 区域可以是 NFS 服务器，如第 314 页中的 “在区域内运行 NFS 服务器”中所述。

试运行（也称为预运行）`zoneadm attach -n` 提供了 `zonecfg` 验证，但不执行软件包内容验证。

所有以文件为参数的 `zoneadm` 选项都需要使用绝对路径。

Oracle Solaris 10 Zones 可在 Oracle Solaris 11 上提供 Oracle Solaris 10 环境。您可以将 Oracle Solaris 10 系统或区域迁移到 Oracle Solaris 11 系统上的 `solaris10` 区域。

`zonep2vchk` 工具可识别可能会影响将 Oracle Solaris 11 系统或 Oracle Solaris 10 系统迁移到运行 Oracle Solaris 11 发行版系统上的某个区域的问题，包括联网问题。在迁移开始之前，先在源系统上执行 `zonep2vchk` 工具。此工具还会输出 `zonecfg` 脚本以便在目标系统上使用。此脚本将创建一个与源系统配置相匹配的区域。有关更多信息，请参见第 22 章，关于区域迁移和 zonep2vchk 工具。

应注意 `solaris` 区域与 Oracle Solaris 10 发行版中的 `native` 区域之间存在的以下差异：

- Oracle Solaris 11 系统中创建 `solaris` 标记，而非 `native` 标记，后者是 Oracle Solaris 10 系统中的缺省标记。

- `solaris` 区域仅为完全根类型。

  Oracle Solaris 10 上提供的本机区域的稀疏根类型使用 SVR4 软件包管理系统，但 IPS 不使用这一框架。提供类似于稀疏根类型的只读根区域配置。
- 在本发行版的区域中，与软件管理相关的功能与 Oracle Solaris 10 发行版之间存在以下方面的区别：
  - IPS 与 SVR4 包管理。
  - 安装、分离、附加和物理转换到虚拟功能。
  - 非全局区域根目录是一个 ZFS 数据集。

    全局区域中安装的软件包不再安装到所有当前区域和未来的区域中。总体而言，对于 IPS 和 SVR4 包管理，全局区域的软件包内容不再指定每个区域的软件包内容。
- 非全局区域使用引导环境。区域与 `beadm` 集成在一起，它是用于管理 ZFS 引导环境 (Boot Environment, BE) 的用户界面命令。

  区域支持 `beadm` 命令，以便用于 `pkg` 更新，就像在全局区域中一样。`beadm` 命令可以删除与区域相关联的任何非活动区域 BE。请参见 `beadm(1M)` 手册页。
- 安装区域时，所有已启用的 IPS 软件包系统信息库都必须可访问。有关更多信息，请参见第 259 页中的 “如何安装已配置的区域”。
- 区域软件以最小化形式启动。必须添加区域所需的所有附加软件包。有关更多信息，请参见 solaris 发布者 (`http://pkg.oracle.com/solaris/release/`)。

区域可以使用 Oracle Solaris 11.1 产品和功能，如下所示：

- Oracle Solaris ZFS 加密
- 网络虚拟化和 QoS
- CIFS 和 NFS

不能在非全局区域中配置以下功能：

- 共享 IP 区域中的 DHCP 地址指定
- `ndmpd`
- SMB 服务器
- SSL 代理服务器
- 通过 `zpool` 命令管理 ZFS 池

只读 solaris 非全局区域

不可编辑的区域是根目录为只读的区域。可通过设置 `file-mac-profile` 属性配置只读区域。提供了多种配置。只读区域根目录扩展了安全运行时界限。

使用 `zonecfg add dataset` 指定附加数据集的区域仍可对这些数据集进行完全控制。使用 `zonecfg add fs` 指定附加文件系统的区域可对这些文件系统进行完全控制，除非文件系统设置为只读。

有关更多信息，请参见第 27 章，配置和管理不可编辑的区域。

关于将 ipkg 区域转换为 solaris 区域

为支持 Oracle Solaris 11 Express 发行版客户，配置为 `ipkg` 区域的任何区域将被转换为 `solaris` 区域，并在 `pkg` 更新或 `zoneadm attach` 时向 Oracle Solaris 11.1 报告为 `solaris`。`ipkg` 名称将映射为 `solaris` 名称（如果在配置区域时使用）。支持导入从 Oracle Solaris 11 Express 主机导出的 `zonecfg` 文件。

对于 Oracle Solaris 11.1 系统上的缺省区域，`zonecfg info` 或 `zoneadm list -v` 等命令的输出显示 `solaris` 标记。

关于标记区域

缺省情况下，系统上的非全局区域运行与全局区域相同的操作系统软件。Oracle Solaris 操作系统中的标记区域 (branded zone, BrandZ) 功能是 Oracle Solaris Zones 的简单扩展。BrandZ 框架用于创建所含的操作环境与全局区域不同的非全局标记区域。在 Oracle Solaris 操作系统上使用标记区域来运行应用程序。BrandZ 框架通过多种方式扩展了 Oracle Solaris Zones 基础结构。这些扩展可能比较复杂（例如，提供在区域内运行不同操作系统环境的功能），也可能比较简单（例如，增强基础区域命令以便提供新功能）。例如，Oracle Solaris 10 Zones 是一个非全局标记区域，可以模拟 Oracle Solaris 10 操作系统。即使与全局区域共享相同操作系统的缺省区域也要配有**标记**。

标记定义了可在区域中安装的操作环境并确定系统在该区域内的行为方式，以便在该区域中安装的软件可以正常运行。此外，区域的标记可用于在应用程序启动时识别正确的应用程序类型。所有标记区域管理都通过扩展标准区域结构来执行。所有区域的大多数管理步骤都相同。

标记文档中介绍了此配置中包含的缺省资源，例如，定义的文件系统和特权。

BrandZ 通过以下方式来扩展区域工具：

- 配置区域时，使用 `zonecfg` 命令来设置区域的标记类型。
- 使用 `zoneadm` 命令来报告区域的标记类型并管理区域。

尽管您可以在已启用标签的 Oracle Solaris Trusted Extensions 系统上配置和安装标记区域，但是您不能在此系统配置中引导标记区域，**除非**所引导的标记是已认证系统配置中的**有标签的**标记。

可以在**已配置**状态下更改区域标记。一旦**安装了**标记区域，就不能更改或删除标记。

注意 – 如果您打算将现有 Oracle Solaris 10 系统迁移到运行 Oracle Solaris 11 发行版的系统上的某个 `solaris10` 标记区域，您必须先将所有现有区域迁移到目标系统。由于区域并不嵌套，因此系统迁移过程将使任何现有区域变得不可用。有关更多信息，请参见第 3 部分。

在标记区域中运行的进程

标记区域在内核中提供了一组插入点，这些插入点只应用于在标记区域中执行的进程。

- 这些点位于 `syscall` 路径、进程装入路径和线程创建路径之类的路径中。
- 在其中每个点处，标记可以选择补充或替换标准 Oracle Solaris 行为。

标记还能为 `librtld_db` 提供插件库。通过插件库，Oracle Solaris 工具（如 `mdb(1)` 中介绍的调试器和 `dtrace(1M)` 中介绍的 DTrace）可以访问在标记区域内运行的进程的符号信息。

请注意，区域不支持静态链接的二进制文件。

本发行版中可用的非全局区域

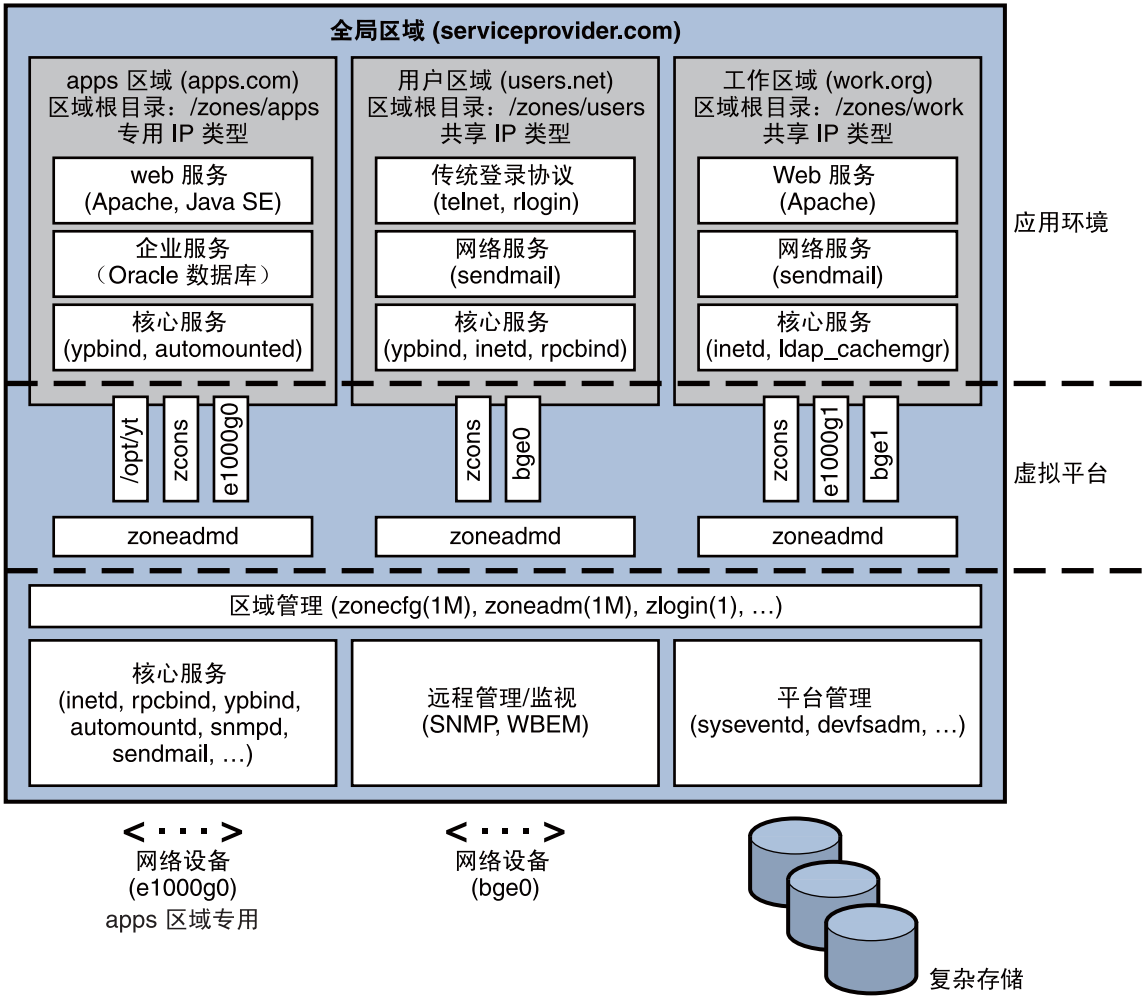
除了缺省的 Oracle Solaris Zone 之外，本发行版中还包括 Oracle Solaris 10 Zones（`solaris10` 标记区域）产品。有关更多信息，请参见第 3 部分。

何时使用区域

对于将多个应用程序整合在一个服务器中的环境而言，使用区域是明智之举。管理大量计算机所带来的成本和复杂性促使在更大、更具伸缩性的服务器上整合多个应用程序。

下图显示了具有三个区域的系统。在整合环境样例中，`apps`、`users` 和 `work` 这几个区域运行的工作负荷都与其他区域的工作负荷无关。此示例说明：为了符合整合要求，不同版本的同一应用程序可以在不同区域中运行，而不会造成负面影响。每个区域都可提供一组定制的服务。

图 15–1 区域服务器整合示例



使用区域，可以更有效地利用系统上的资源。使用动态资源重新分配，可以根据需要将未使用的资源转移到其他区域。故障和安全隔离意味着运行欠佳的应用程序不需要一个未充分利用的专用系统。使用区域，可以将这些应用程序与其他应用程序进行整合。

使用区域，可以在维护整体系统安全的同时委托某些管理功能。

区域如何工作

可以将一个非全局区域想象为一个盒子。一个或多个应用程序可在这个盒子中运行，而不与系统的其余部分交互。区域使用灵活、软件定义的边界将各软件应用程序或服务分隔开来。然后，便可分别管理在 Oracle Solaris 操作系统的同一实例中运行的应用程序。因此，为了符合配置要求，不同版本的同一应用程序可以在不同区域中运行。

指定给某区域的进程可以处理、监视指定给同一区域的其他进程，并可直接与这些进程进行通信。进程不能对指定给系统中其他区域的进程执行这些功能，也不能对未指定给区域的进程执行这些功能。指定给不同区域的进程只能通过网络 API 进行通信。

IP 联网可按两种不同的方式进行配置，具体用哪种方式取决于该区域是具有其自己的专用 IP 实例还是将 IP 层配置和状态与全局区域共享。专用 IP 为缺省类型。有关区域中 IP 类型的更多信息，请参见第 197 页中的 “区域网络接口”。有关配置信息，请参见第 232 页中的 “如何配置区域”。

每个 Oracle Solaris 系统都包含一个**全局区域**。全局区域具有双重功能。全局区域既是系统的缺省区域，也是用于在整个系统中实施管理控制的区域。如果**全局管理员**或具有区域安全配置文件的用户没有创建任何**非全局**区域（简称为区域），则所有进程将在全局区域中运行。

只能从全局区域配置、安装、管理或卸载非全局区域。只有全局区域才可从系统硬件进行引导。只能在全局区域中进行系统基础结构（如物理设备）的管理、共享 IP 区域中的路由或动态重新配置 (dynamic reconfiguration, DR)。全局区域中运行的具有适当特权的进程可以访问与其他区域关联的对象。

全局区域中的非特权进程可以执行非全局区域中不允许特权进程执行的操作。例如，全局区域中的用户可以查看有关系统中每个进程的信息。如果此功能会使站点出现问题，则可以限制对全局区域进行访问。

包括全局区域在内的每个区域都会被指定一个区域名称。全局区域始终命名为 `global`。每个区域还具有唯一的数字标识符，这是引导区域时由系统指定的。全局区域始终映射到 ID `0`。区域名称和数字 ID 在第 207 页中的 “使用 zonecfg 命令”中介绍。

每个区域还具有节点名称，此名称完全独立于区域名称。节点名称由区域管理员指定。有关更多信息，请参见第 314 页中的 “非全局区域节点名称”。

每个区域都具有一个与全局区域根目录相对的根目录路径。有关更多信息，请参见第 207 页中的 “使用 zonecfg 命令”。

缺省情况下，非全局区域的调度类设置为系统的调度类。有关在区域中设置调度类的方法讨论，请参见第 194 页中的 “调度类”。

区域摘要（按功能）

下表总结了全局区域和非全局区域的特征。

区域类型	特征
全局	■ 由系统指定 ID `0` ■ 提供正在系统上运行的可引导的 Oracle Solaris 内核的单个实例 ■ 包含 Oracle Solaris 系统软件包的完整安装 ■ 可以包含其他软件包或未通过软件包安装的其他软件、目录、文件以及其他数据 ■ 提供一个完整一致的产品数据库，该数据库包含安装在全局区域中的所有软件组件的有关信息 ■ 仅存放特定于全局区域的配置信息，如全局区域主机名和文件系统表 ■ 是识别所有设备和所有文件系统的唯一区域 ■ 是识别非全局区域存在和配置的唯一区域 ■ 是可以从中配置、安装、管理或卸载非全局区域的唯一区域
非全局	■ 引导区域时由系统指定区域 ID ■ 共享从全局区域引导的 Oracle Solaris 内核下的操作 ■ 包含完整 Oracle Solaris 操作系统软件包中已安装的一部分 ■ 可以包含其他已安装的软件包 ■ 可以包含在非全局区域上创建的，未通过软件包安装的其他软件、目录、文件以及其他数据 ■ 具有一个完整一致的产品数据库，该数据库包含安装在区域中的所有软件组件的相关信息 ■ 不识别其他任何区域的存在 ■ 无法安装、管理或卸载其他区域，包括其本身 ■ 仅具有特定于非全局区域的配置信息，例如非全局区域主机名和文件系统表 ■ 可以具有自己的时区设置

如何管理非全局区域

全局管理员具有超级用户特权或等效的管理权限。当全局管理员登录到全局区域时，可以将系统作为一个整体进行监视和控制。

区域管理员可以管理非全局区域。全局管理员可向区域管理员指定所需的授权，如第 193 页中的 “`admin` 资源”中所述。区域管理员的特权仅限于特定的非全局区域。

如何创建非全局区域

您可以在自动安装 (Automated Install, AI) 客户机的安装过程中指定非全局区域的配置和安装。有关更多信息，请参见《安装 Oracle Solaris 11.1 系统》。

为在 Oracle Solaris 系统上创建区域，全局管理员将通过为区域的虚拟平台和应用程序环境指定各种参数，使用 `zonecfg` 命令来配置区域。然后，全局管理员安装区域，使用区域管理命令 `zoneadm` 将软件包中的软件安装到为区域建立的文件系统分层结构。使用 `zoneadm` 命令引导区域。然后，全局管理员或授权用户可以使用 `zlogin` 命令登录到已安装的区域。如果使用基于角色的访问控制 (role-based access control, RBAC)，则区域管理员必须具备 `solaris.zone.manage/`*zonename* 授权。

有关区域配置的信息，请参见第 16 章，非全局区域配置（概述）。有关区域安装的信息，请参见第 18 章，关于安装、关闭、停止、卸载和克隆非全局区域（概述）。有关区域登录的信息，请参见第 20 章，非全局区域登录（概述）。

非全局区域状态模型

非全局区域可以处于以下七种状态之一：

已配置
: 区域配置已完成并提交到稳定存储器。但是，那些必须在初始引导之后指定的区域应用程序环境元素还不存在。

未完成
: 在安装或卸载操作期间，`zoneadm` 将目标区域的状态设置为未完成。成功完成操作之后，便将状态设置为正确的状态。

  可以使用 `zoneadm` 的 `mark` 子命令将被损坏的已安装区域标记为未完成。处于未完成状态下的区域如 `zoneadm list -iv` 的输出所示。

不可用
: 指示区域已安装，但无法验证、就绪、引导、附加或移动。出现以下情况时区域会进入不可用状态：

  - 区域的存储不可用而 `svc:/system/zones:default` 已开始，如在系统引导期间
  - 当区域的存储不可用时
  - 当成功提取归档文件之后基于归档文件的安装失败时
  - 当区域的软件与全局区域的软件不兼容时，如在不正确的 `-F`（强制）附加之后

已安装
: 已在系统上实例化区域配置。使用 `zoneadm` 命令检验是否可以在指定的 Oracle Solaris 系统上成功使用配置。软件包安装在区域的根路径下。在此状态下，区域没有关联的虚拟平台。

就绪
: 已建立区域的虚拟平台。已由内核创建 `zsched` 进程，已设置网络接口且可用于该区域，已挂载文件系统，并且已配置设备。系统会指定唯一的区域 ID。在此阶段，没有启动与区域关联的进程。

正在运行　　正在运行与区域应用程序环境关联的用户进程。创建了与应用程序环境关联的第一个用户进程 (`init`) 之后，区域便会立即进入正在运行状态。

正在关闭和关闭　　这两种状态是停止区域时出现的过渡状态。但是，因某种原因无法关闭的区域将会在这两种状态下停止。

第 19 章，安装、引导、关闭、停止、卸载和克隆非全局区域（任务）和 `zoneadm`(1M) 手册页介绍了如何使用 `zoneadm` 命令在这些状态之间进行转换。

表 15–1　影响区域状态的命令

当前区域状态	适用的命令

已配置	`zonecfg -z` *zonename* `verify`
	`zonecfg -z` *zonename* `commit`
	`zonecfg -z` *zonename* `delete`
	`zoneadm -z` *zonename* `attach`
	`zoneadm -z` *zonename* `verify`
	`zoneadm -z` *zonename* `install`
	`zoneadm -z` *zonename* `clone`
	`zoneadm - z` *zonename* `mark` *incomplete*
	`zoneadm -z` *zonename* `mark` *unavailable*
	您还可以使用 `zonecfg` 重命名处于已配置或已安装状态的区域。

未完成	`zoneadm -z` *zonename* `uninstall`
不可用	`zoneadm -z` *zonename* `uninstall` 可从指定的系统中卸载区域。 `zoneadm -z` *zonename* `attach` `zonecfg -z` *zonename* 可用于更改 `zonepath` 和其他任何处于已安装状态时可以进行更改的属性和资源。

表 15–1 影响区域状态的命令 （续）

当前区域状态	适用的命令

已安装	`zoneadm -z` *zonename* `ready`（可选的）
	`zoneadm -z` *zonename* `boot`
	`zoneadm -z` *zonename* `uninstall` 可从系统中卸载指定区域的配置。
	`zoneadm -z` *zonename* `move` *path*
	`zoneadm -z` *zonename* `detach`
	`zonecfg -z` *zonename* 可用于添加或删除 `attr`、`bootargs`、`capped-memory`、`dataset`、`capped-cpu`、`dedicated-cpu`、`device`、`fs`、`ip-type`、`limitpriv`、`net`、`rctl` 或 `scheduling-class` 属性。您还可以重命名处于已安装状态的区域。
	`zoneadm - z` *zonename* `mark` *incomplete*
	`zoneadm -z` *zonename* `mark` *unavailable*

就绪	`zoneadm -z` *zonename* `boot` `zoneadm halt` 加上系统重新引导可使区域从就绪状态恢复为已安装状态。 `zonecfg -z` *zonename* 可用于添加或删除 `attr`、`bootargs`、`capped-memory`、`dataset`、`capped-cpu`、`dedicated-cpu`、`device`、`fs`、`ip-type`、 `limitpriv`、`net`、`rctl` 或 `scheduling-class` 属性。

正在运行	`zlogin` *options* `zonename`
	`zoneadm -z` *zonename* `reboot`
	`zoneadm -z` *zonename* `halt` 可使就绪区域恢复为已安装状态。
	`zoneadm halt` 加上系统重新引导可使区域从正在运行状态恢复为已安装状态。
	`zoneadm - z shutdown` 可干净地关闭区域。
	`zonecfg -z` *zonename* 可用于添加或删除 `attr`、`bootargs`、`capped-memory`、`dataset`、`capped-cpu`、`dedicated-cpu`、`device`、`fs`、 `ip-type`、`limitpriv`、`anet`、`net`、`rctl` 或 `scheduling-class` 属性。不能更改 `zonepath` 资源。

注 – 通过 `zonecfg` 更改的参数不会影响正在运行的区域。必须重新引导区域才能使更改生效。

非全局区域特征

区域提供的隔离几乎可细化到您所需的任何程度。区域不需要专用的 CPU、物理设备或部分物理内存。可以在单个域或系统中运行的多个区域之间复用这些资源，也可借助操作系统中可用的资源管理功能为每个区域分别分配这些资源。

每个区域都可提供一组定制的服务。要执行基本进程隔离，一个进程只能看到同一区域中的各个进程，或向这些进程发送信号。区域间的基本通信是通过每个区域的 IP 网络连接来完成的。在某个区域中运行的应用程序看不到其他区域的网络流量。即使各个软件包的流使用同一物理接口，也会维护这种隔离。

每个区域都在文件系统分层结构中拥有一个位置。因为每个区域都只限于文件系统分层结构中的一个子树，所以在某一特定区域中运行的工作负荷不能访问在其他区域中运行的另一个工作负荷的盘上数据。

命名服务使用的文件驻留在区域本身的根文件系统视图中。因此，不同区域的命名服务之间相互分离并可单独配置。

将资源管理功能用于非全局区域

如果您使用资源管理功能，则应当使此功能可以完全控制区域范围。通过指定上述控制范围，可以创建更完整的虚拟机模型，可对其中的名称空间访问、安全隔离和资源使用情况进行完全控制。

对于将各种资源管理功能用于区域的任何特殊要求，将在本手册中介绍这些功能的各章节中介绍。

与区域相关的 SMF 服务

全局区域中与区域相关的 SMF 服务包括以下内容：

`svc:/system/zones:default`
  启动具备 `autoboot=true` 的每个区域。

`svc:/system/zones-install:default`
  如果需要，首次引导时执行区域安装。

`svc:/application/pkg/zones-proxyd:default`
  包管理系统使用该服务提供对系统信息库的区域访问。

`svc:/application/pkg/system-repository:default`
  高速缓存代理服务器，该服务器用于高速缓存区域安装和其他 `pkg` 操作期间使用的 `pkg` 数据和元数据。请参见 `pkg(1)` 和 `pkg(5)` 手册页。

`svc:/system/zones-monitoring:default`
  控制 `zonestatd`。

svc:/application/pkg/zones-proxy-client:default 区域代理客户机 SMF 服务仅在非全局区域运行。包管理系统使用该服务提供对系统信息库的区域访问。

监视非全局区域

有关如何报告当前正在运行的区域的 CPU、内存和资源控制使用情况，请参见第 343 页中的 “在非全局区域中使用 zonestat 实用程序”。zonestat 实用程序还可报告专用 IP 区域中的网络带宽使用情况。专用 IP 区域具有其自己的 IP 相关状态以及一个或多个专用数据链路。

可以使用 fsstat 实用程序为非全局区域报告文件操作统计信息。请参见 fsstat(1M) 手册页和第 313 页中的 “使用 fsstat 实用程序监视非全局区域”。

非全局区域提供的功能

非全局区域可提供以下功能：

安全性　一旦将进程放入全局区域之外的区域，此进程或其后续子进程便不能更改区域。

可以在区域中运行网络服务。通过在区域中运行网络服务，可限制出现安全违规时可能引起的损坏。如果入侵者成功利用了区域中运行的软件中的安全缺陷，则此入侵者只能在此区域中执行一部分可能的操作。区域中可用的特权是整个系统中可用特权的一部分。

隔离　使用区域，可以在同一计算机上部署多个应用程序，即使这些应用程序运行在不同的信任域中，需要独占访问全局资源或者全局配置出现问题也是如此。应用程序还无法监视或拦截其他应用程序的网络流量、文件系统数据或进程活动。

网络隔离　区域在缺省情况下配置为专用 IP 类型。在 IP 层，这些区域与全局区域隔离，并且相互隔离。出于运行和安全方面的原因而采用了这种隔离。可通过区域来合并必须使用其自己的 LAN 或 VLAN 在不同子网上通信的应用程序。每个区域还可以定义其自己的 IP 层安全规则。

虚拟化　区域提供了一个虚拟环境，此环境可以在应用程序中隐藏详细信息（例如物理设备、系统的主 IP 地址以及主机名）。可以在不同的物理计算机上维护同一应用程序环境。通过虚拟环境，可以单独管理每个区域。区域管理员在非全局区域中执行的操作不会影响系统的其余部分。

粒度　区域提供的隔离几乎可细化到任何程度。有关更多信息，请参见第 187 页中的 “非全局区域特征”。

环境　区域不更改应用程序的执行环境，但为实现安全和隔离目标而必须更改的情况除外。区域不显示应用程序必须连接的新 API 或 ABI。相反，区域提

供具有某些限制的标准 Oracle Solaris 接口和应用程序环境。这些限制主要影响尝试执行特权操作的应用程序。

无论是否配置其他区域，全局区域中的应用程序始终会运行而无需修改。

在系统上设置区域（任务列表）

下表简要介绍了首次在系统上设置区域所涉及的任务。

任务	说明	参考
标识您要在区域中运行的应用程序。	查看正在系统上运行的应用程序： ■ 确定对您的业务目标至关重要的应用程序。 ■ 评估正在运行的应用程序对系统的要求。	如有必要，请参阅您的业务目标和系统文档。
确定要配置的区域数。	评估： ■ 您要在区域中运行的应用程序的性能要求。 ■ 要安装的每个区域是否有 1 GB 的磁盘空间。所需的空间量取决于要在区域内安装的软件，应相应进行调整。使用 ZFS 压缩将减少所需的磁盘空间量。 请注意，在非全局区域安装以及后续软件包安装和更新期间，需要占用一些临时空间。1 GB 的磁盘空间要求考虑了这一点。	请参见第 227 页中的 “评估当前的系统设置”。
确定您的区域是使用资源池还是使用指定的 CPU 来分区计算机资源。	如果您还要在系统上使用资源管理功能，则需要使资源管理范围能够覆盖这些区域。请在配置区域之前配置资源池。 请注意，通过使用 `zonecfg` 属性可以向区域中快速添加区域范围的资源控制和池功能。	请参见第 232 页中的 “如何配置区域”和第 13 章，创建和管理资源池（任务）。

任务	说明	参考
执行预配置任务。	确定区域名称和区域路径。 确定区域的任何其他要求，如是否在共享存储中托管区域。 缺省情况下，使用 anet 资源创建专用 IP 类型的非全局区域。anet 资源会自动为非全局区域创建虚拟 NIC (virtual NIC, VNIC)。或者，您可以使用 net 资源将区域配置为共享 IP 区域或专用 IP 区域。确定每个区域的必需文件系统和设备。确定用于区域的调度类。确定在标准缺省特权集不充足的情况下，应为区域内的进程设置的特权集。请注意，有些 zonecfg 设置会自动添加特权。例如，ip-type=exclusive 会自动添加配置和管理网络栈所需的多个特权。	有关区域名称和路径、IP 类型、IP 地址、文件系统、设备、调度类以及特权的信息，请参见第 16 章，非全局区域配置（概述）和第 227 页中的 “评估当前的系统设置”。有关缺省特权和可在非全局区域中配置的特权的列表，请参见第 327 页中的 “非全局区域中的特权”。有关 IP 功能的信息，请参见第 320 页中的 “共享 IP 非全局区域中的联网”和第 322 页中的 “专用 IP 非全局区域中的联网”。
开发配置。	配置非全局区域。	请参见第 231 页中的 “配置、检验并提交区域”和 zonecfg(1M) 手册页。
以全局管理员或具备相应授权的用户身份检验并安装已配置的区域。	必须在登录之前检验和安装区域。 在安装时创建并配置区域的初始内部配置。	请参见第 18 章，关于安装、关闭、停止、卸载和克隆非全局区域（概述）、第 19 章，安装、引导、关闭、停止、卸载和克隆非全局区域（任务）、sysconfig(1M) 和《安装 Oracle Solaris 11.1 系统》中的第 6 章 “取消配置或重新配置 Oracle Solaris 实例”。
以全局管理员或被授予相应授权的用户身份引导非全局区域。	引导每个区域以将区域置于运行状态。	请参见第 18 章，关于安装、关闭、停止、卸载和克隆非全局区域（概述）和第 19 章，安装、引导、关闭、停止、卸载和克隆非全局区域（任务）。
为生产使用准备新区域。	创建用户帐户，添加其他软件，并定制区域配置。	请参阅用于设置新安装的计算机的文档。本指南包含适用于已安装区域的系统的特殊注意事项。

◆ ◆ ◆ 第 16 章

16 非全局区域配置（概述）

本章介绍非全局区域配置。

本章中的主题包括以下内容：

了解区域配置之后，请转至第 17 章，规划和配置非全局区域（任务）以配置要在系统上安装的非全局区域。

关于区域中的资源

可以在区域中控制的资源包括：

- 资源池或指定的 CPU，用于对计算机资源进行分区。
- 资源控制，提供了一种系统资源约束机制。
- 调度类，可让您根据资源的重要性在区域中控制可用 CPU 资源的分配。这种重要性通过您为每个区域指定的 CPU 资源份额来表示。

在区域管理中使用权限配置文件和角色

有关权限配置文件和角色的信息，请参见《Oracle Solaris 11 安全准则》中的“Oracle Solaris 安全技术”。

安装前配置过程

在系统上安装非全局区域并使用它之前，必须先配置该区域。

`zonecfg` 命令用于创建配置，并确定指定的资源和属性是否在虚拟系统上有效。`zonecfg` 对给定配置执行的检查将检验以下内容：

- 确保已指定区域路径.
- 确保已为每个资源指定所有必需的属性。
- 确保配置没有冲突。例如，如果您有一个 `anet` 资源，则区域为专用 IP 类型，不能为共享 IP 区域。此外，如果已有别名的数据集与设备之间有潜在冲突，则 `zonecfg` 命令会发出一个警告。

有关 `zonecfg` 命令的更多信息，请参见 `zonecfg(1M)` 手册页。

区域组件

本节讨论可以配置的必需区域组件和可选的区域组件。只有区域名称和区域路径是必需的。第 210 页中的 “区域配置数据”中还提供了附加信息。

区域名称和路径

必须为区域选择名称和路径。区域必须位于 ZFS 数据集中。在安装或附加区域时，将自动创建 ZFS 数据集。如果无法创建 ZFS 数据集，也无法安装或附加区域。请注意，区域路径的父目录也必须是一个数据集。

区域自动引导

`autoboot` 属性设置决定了是否在引导全局区域时自动引导该区域。区域服务，`svc:/system/zones:default` 也必须启用。

只读根区域的 file-mac-profile 属性

在 `solaris` 区域中，`file-mac-profile` 用于配置只读根区域。

有关更多信息，请参见第 27 章，配置和管理不可编辑的区域。

admin 资源

admin 设置允许您设置区域管理授权。定义授权的首选方法是通过 zonecfg 命令。

user　指定用户名。

auths　指定用户名授权。

solaris.zone.login	如果使用了 RBAC，则交互式登录需要 solaris.zone.login/ *zonename* 授权。将在区域中进行口令验证。
solaris.zone.manage	如果使用 RBAC，则对于非交互式登录或者要跳过口令验证，需要 solaris.zone.manage/ *zonename* 授权。
solaris.zone.clonefrom	如果使用 RBAC，则用于生成其他区域副本的子命令需要 solaris.zone.clonefrom/ *source_zone* 授权。

dedicated-cpu 资源

dedicated-cpu 资源可指定在非全局区域运行时应将系统处理器的某个子集专用于该非全局区域。在引导区域时，系统将动态创建一个临时池，以便在区域运行时使用。

根据 zonecfg 的指定，池设置将在迁移期间进行传播。

dedicated-cpu 资源可为 ncpus 以及 importance（可选的）设置限制。

ncpus　指定 CPU 数目或指定一个范围（如 2–4 个 CPU）。如果指定一个范围（因为需要动态资源池行为），则还应执行以下操作：

- 设置 importance 属性。
- 启用 poold 服务。有关说明，请参见第 149 页中的 “如何使用 svcadm 启用动态资源池服务”。

importance　如果使用 CPU 范围来获取动态行为，还要设置 importance 属性。importance 属性是**可选的**属性，用来定义池的相对重要性。仅当为 ncpus 指定了范围并且使用由 poold 管理的动态资源池时，才需要此属性。如果 poold 未运行，则会忽略 importance。如果 poold 正在运行并且未设置 importance，那么 importance 将缺省设置为 1。有关更多信息，请参见第 136 页中的 “pool.importance 属性约束”。

注 – capped-cpu 资源与 dedicated-cpu 资源不兼容。cpu-shares rctl 与 dedicated-cpu 资源不兼容。

---

capped-cpu 资源

capped-cpu 资源对某一项目或区域可占用的 CPU 资源量设立绝对的细粒度限制。在与处理器集结合使用时，CPU 上限将限制某一处理器集内的 CPU 使用。capped-cpu 资源有一个 ncpus 属性，该属性是一个正小数，小数点右侧有两位。该属性与 CPU 的单位相对应。此资源不接受范围值，但接受小数。指定 ncpus 时，值为 1 表示某个 CPU 的 100%。值为 1.25 表示 125%，因为 100% 对应于系统中的一个 CPU。

注 – capped-cpu 资源与 dedicated-cpu 资源不兼容。

调度类

可以使用**公平份额调度器** (fair share scheduler, FSS)，根据区域的重要性控制可用 CPU 资源在区域之间的分配。这种重要性通过您为每个区域指定的 CPU 资源**份额**来表示。即使您没有使用 FSS 来管理区域之间的 CPU 资源分配，您也可以将区域的调度类设置为使用 FSS，以便您可为区域中的项目设置份额。

在显式设置 cpu-shares 属性时，公平份额调度器 (fair share scheduler, FSS) 将用作该区域的调度类。但是，在此情况下使用 FSS 的首选方法是通过 dispadmin 命令将 FSS 设置为系统缺省的调度类。这样，所有区域都将从获取系统 CPU 资源的公平份额中受益。如果未为区域设置 cpu-shares，区域将使用系统缺省的调度类。以下操作可为区域设置调度类：

- 可以使用 zonecfg 中的 scheduling-class 属性为区域设置调度类。
- 可以通过资源池功能为区域设置调度类。如果区域与 pool.scheduler 属性设置为有效调度类的池相关联，则缺省情况下区域中运行的进程会以该调度类运行。请参见第 128 页中的 “资源池介绍”和第 155 页中的 “如何将池与调度类关联”。
- 如果设置了 cpu-shares rctl，但未通过其他操作将 FSS 设置为区域的调度类，则 zoneadmd 将在区域引导时将调度类设置为 FSS。
- 如果未通过其他任何操作设置调度类，区域将继承系统的缺省调度类。

请注意，您可以使用 priocntl(1) 手册页中所述的 priocntl，在不更改缺省调度类和不重新引导的情况下将正在运行的进程移至其他调度类。

物理内存控制和 capped-memory 资源

capped-memory 资源可为 physical、swap 和 locked 内存设置限制。每个限制均是可选的，但至少要设置一个限制。要使用 capped-memory 资源，必须在全局区域中安装 resource-cap 软件包。

- 如果计划在全局区域中使用 rcapd 为区域设置内存上限，请确定此资源的值。rcapd 将 capped-memory 资源的 physical 属性用作区域的 max-rss 值。
- capped-memory 资源的 swap 属性是用于设置 zone.max-swap 资源控制的首选方法。
- capped-memory 资源的 locked 属性是用于设置 zone.max-locked-memory 资源控制的首选方法。

注 – 应用程序通常不会锁定大量内存，但是如果已知道区域的应用程序会锁定内存，则您可能会决定设置锁定内存。如果区域信任是一个需要关注的问题，还可以考虑将锁定内存上限设为系统物理内存的百分之十，或区域物理内存上限的百分之十。

有关更多信息，请参见第 10 章，使用资源上限设置守护进程控制物理内存（概述）、第 11 章，管理资源上限设置守护进程（任务）和第 232 页中的 “如何配置区域”。要临时为区域设置资源上限，请参见第 123 页中的 “如何为区域指定临时资源上限”。

rootzpool 资源

zonecfg 实用程序中的可选 rootzpool 资源用于为区域安装创建专用 ZFS zpool。区域的根 ZFS zpool 可以位于由一个或多个统一资源标识符 (Universal Resource Identifier, URI) 定义的共享存储设备上。必需的 storage 属性标识包含区域的根 zfs 文件系统的存储对象 URI。对于一个给定区域，只能定义一个 rootzpool。引导区域时将自动为区域配置存储。

在执行区域安装或附加操作期间，将自动创建或导入相应的 zpool。卸载或分离区域时，将执行以下操作：

- 自动取消配置存储资源。
- 自动导出或销毁相应的 zpool。

要在区域安装中重用预先创建的 zpool，必须从系统中导出该 zpool。

区域框架支持以下 URI 类型：

- dev

  本地设备路径 URI

  格式：

  dev:*local-path-under-*/dev
  dev://*absolute-path-with-dev*
  dev:*absolute-path-with-dev*

  示例：

```
dev:dsk/c7t0d0s0
dev:///dev/dsk/c7t0d0s0
dev:/dev/dsk/c7t0d0s0
```

- `lu`（逻辑单元）

  光纤通道 (Fibre Channel, FC) 和串行连接 SCSI (Serial Attached SCSI, SAS)

  格式：

  ```
  lu:luname.naa.ID
  lu:initiator.naa.ID,target.naa.ID,luname.naa.ID
  ```

  示例：

```
lu:luname.naa.5000c5000288fa25
lu:initiator.naa.2100001d38089fb0,target.naa.2100001d38089fb0,luname.naa.5000c5000288fa25
```

- `iscsi`

  iSCSI URI

  格式：

  ```
  iscsi:///luname.naaID
  iscsi://host[:port]/luname.naa.ID
  ```

  示例：

  ```
  iscsi:///luname.naa.600144f03d70c80000004ea57da10001
  iscsi://[::1]/luname.naa.600144f03d70c80000004ea57da10001
  iscsi://127.0.0.1/luname.naa.600144f03d70c80000004ea57da10001
  iscsi://127.0.0.1:3620/luname.naa.600144f03d70c80000004ea57da10001
  iscsi://hostname:3620/luname.naa.600144f03d70c80000004ea57da10001
  ```

`suriadm` 工具用于根据存储 URI 管理共享目标文件。有关 ID、名称地址机构 (Name Address Authority, NAA) 和获取现有存储对象 URI 的信息，请参见 `suriadm(1M)` 和 `suri(5)` 手册页。

系统将根据与 rootzpool 关联的区域为新建或导入的 `rootzpool` 命名。指定的名称采用 *zonename*`_rpool` 形式。

可使用以下命令在 `rootzpool` 资源范围内管理 `storage` 属性：

- `add storage` *URI string*
- `remove storage` *URI string*

自动添加 zpool 资源

通过在 `zonecfg` 实用程序中配置可选的 `zpool` 资源，可以将 `zpool` 委托给非全局区域。引导区域时将自动为区域配置 `zpool`。

在执行区域安装或附加操作期间，将自动创建或导入相应的 zpool。

卸载或分离区域时，将执行以下操作：

- 自动取消配置存储资源。
- 自动导出或销毁相应的 zpool。

必需的 `storage` 属性标识与该资源关联的存储对象 URI。

可使用以下设置在 `zpool` 资源范围内管理 `storage` 属性：

- `add storage` *URI string*
- `remove storage` *URI string*

`name` 属性是 `zpool` 资源的必需属性。该属性用在委托给区域的 zpool 的名称中。ZFS 文件系统的 `name` 组件不能包含正斜杠 (`/`).

为新建或导入的 `zpool` 指定的名称采用 *zonename_name* 形式。该名称是在非全局区域内可见的 zpool 名称。

注 – 存储对象包含预先存在的分区、zpool 或 UFS 文件系统时，区域安装会失败。有关更多信息，请参见第 259 页中的 “如何安装已配置的区域”中的步骤 4。

区域网络接口

引导区域时，将在其中自动设置并放置通过 `zonecfg` 实用程序配置的用于提供网络连接的区域网络接口。

Internet 协议 (Internet Protocol, IP) 层可接受和传送网络包。该层包括 IP 路由、地址解析协议 (Address Resolution Protocol, ARP)、Internet 协议安全体系结构 (Internet Protocol Security Architecture, IPsec) 和 IP 过滤器。

可用于非全局区域的 IP 类型有两种：共享 IP 和专用 IP。专用 IP 是缺省 IP 类型。共享 IP 区域与全局区域共享网络接口。要使用共享 IP 区域，必须通过 `ipadm` 实用程序完成全局区域中的配置。专用 IP 区域必须有专用的网络接口。如果使用 `anet` 资源配置专用 IP 区域，将自动创建一个专用 VNIC 并将其分配给该区域。通过使用自动的 `anet` 资源，不必在全局区域中创建和配置数据链路并将数据链路指定给非全局区域。使用 `anet` 资源可完成以下任务：

- 允许全局区域管理员为指定给非全局区域的数据链路选择特定名称
- 允许多个区域使用同名的数据链路

为了实现向后兼容性，可以向非全局区域指定预配置的数据链路。

有关每种类型中 IP 功能的信息，请参见第 320 页中的 “共享 IP 非全局区域中的联网”和第 322 页中的 “专用 IP 非全局区域中的联网”。

注 – 链路保护在《Oracle Solaris Administration: Network Interfaces and Network Virtualization》中的第 20 章 “Using Link Protection in Virtualized Environments”中进行了介绍，可用在运行区域的系统上。此功能在全局区域中配置。

关于数据链路

数据链路是 OSI 协议栈的第 2 层接口，在系统中表示为 STREAMS DLPI (v2) 接口。此类接口可以在 TCP/IP 等协议栈下检测到。数据链路也称为物理接口，例如，网络接口卡 (Network Interface Card, NIC)。数据链路是使用 `zonecfg` (1M) 配置的 `physical` 属性。`physical` 属性可以为 VNIC，具体请参见《Oracle Solaris Administration: Network Interfaces and Network Virtualization》中的第 III 部分, “Network Virtualization and Resource Management”。

数据链路示例有物理接口（如 `e1000g0` 和 `bge1`）、NIC（如 `bge3`）、集合（如 `aggr1`、`aggr2`）或有 VLAN 标记的接口（如 `e1000g123000` 和 `bge234003`）（分别作为 `e1000g0` 和 `bge3` 上的 VLAN 123 和 VLAN 234）。

有关使用基于 InfiniBand 的 IP (IP over InfiniBand, IPoIB) 的信息，请参见第 214 页中的 “资源类型属性”中关于 `anet` 的说明。

共享 IP 非全局区域

共享 IP 区域使用全局区域中的现有 IP 接口。区域必须有一个或多个专用 IP 地址。共享 IP 区域与全局区域共享 IP 层配置和状态。如果以下两个条件同时成立，则区域应该使用共享 IP 实例：

- 非全局区域使用全局区域使用的相同数据链路，而不管全局区域和非全局区域是否在同一子网中。
- 您不想使用专用 IP 区域提供的其他功能。

使用 `zonecfg` 命令的 `net` 资源为共享 IP 区域指定一个或多个 IP 地址。数据链路名称也必须在全局区域中配置。

在 `zonecfg net` 资源中，必须设置 `address` 和 `physical` 属性。`defrouter` 属性为可选的。

要在全局区域中使用共享 IP 类型联网配置，必须使用 `ipadm`，而不是自动网络配置。要确定 `ipadm` 是否正在进行联网配置，请运行以下命令。显示的响应必须为 `DefaultFixed`。

```
# svcprop -p netcfg/active_ncp svc:/network/physical:default
DefaultFixed
```

指定给共享 IP 区域的 IP 地址与逻辑网络接口相关联。

可以在全局区域中使用 `ipadm` 命令来在运行的区域中指定或删除逻辑接口。

要添加接口，请使用以下命令：

```
global# ipadm set-addrprop -p zone=my-zone net0/addr1
```

要删除接口，请使用以下命令之一：

```
global# ipadm set-addrprop -p zone=global net0/addr
```

或者：

```
global# ipadm reset-addrprop -p zone net0/addr1
```

有关更多信息，请参见第 321 页中的“共享 IP 网络接口”。

专用 IP 非全局区域

专用 IP 是非全局区域的缺省联网配置。

专用 IP 区域具有其自己的 IP 相关状态以及一个或多个专用数据链路。

可以在专用 IP 区域中使用以下功能：

- DHCPv4 和 IPv6 无状态地址自动配置
- IP 过滤器，包括网络地址转换 (network address translation, NAT) 功能
- IP 网络多路径 (IP Network Multipathing, IPMP)
- IP 路由
- `ipadm`，用于设置 TCP/UDP/SCTP 和 IP/ARP 级别可调参数
- IP 安全 (IP security, IPsec) 和 Internet 密钥交换 (Internet Key Exchange, IKE)，可自动提供用于 IPsec 安全关联的验证加密材料

有两种配置专用 IP 区域的方式：

- 使用 `zonecfg` 实用程序的 `anet` 资源在引导区域时自动为区域创建临时 VNIC，然后在区域停止时删除它。
- 在全局区域中预先配置数据链路，然后使用 `zonecfg` 实用程序的 `net` 资源将其指定给专用 IP 区域。数据链路使用 `net` 资源的 `physical` 属性指定。`physical` 属性可以为 VNIC，具体请参见《Oracle Solaris Administration: Network Interfaces and Network Virtualization》中的第 III 部分, “Network Virtualization and Resource Management”。没有设置 `net` 资源的 `address` 属性。

缺省情况下，专用 IP 区域可以在关联接口上配置和使用任何 IP 地址。（可选的）可以使用 `allowed-address` 属性指定逗号分隔的 IP 地址列表。专用 IP 区域不能使用 `allowed-address` 列表中没有的 IP 地址。此外，将在引导区域时自动为专用 IP 区域永久配置 `allowed-address` 列表中的所有地址。如果不需要此接口配置，则必须将 `configure-allowed-address` 属性设置为 `false`。缺省值为 `true`。

注意，通过指定的数据链路，可使用 snoop 命令。

可以将 dladm 命令与 show-linkprop 子命令一起使用，以显示正在运行的专用 IP 区域的数据链路分配。可以将 dladm 命令与 set-linkprop 子命令一起使用，以将其他数据链路指定给正在运行的区域。有关用法示例，请参见第 354 页中的 “在独占 IP 非全局区域中管理数据链路”。

在已指定自己的数据链路集的正在运行的专用 IP 区域中，可以使用 ipadm 命令来配置 IP，包括添加或删除逻辑接口的能力。通过使用 sysconfig(1M) 手册页中所述的 sysconfig 接口，可以按全局区域的设置方式对区域中的 IP 配置进行设置。

专用 IP 区域的 IP 配置仅可在全局区域中使用 zlogin 命令进行查看。

```
global# zlogin zone1 ipadm show-addr
ADDROBJ           TYPE     STATE        ADDR
lo0/v4            static   ok           127.0.0.1/8
nge0/_b           dhcp     ok           10.134.62.47/24
lo0/v6            static   ok           ::1/128
nge0/_a           addrconf ok           fe80::2e0:81ff:fe5d:c630/10
```

非全局区域对可靠数据报套接字的支持

专用 IP 和共享 IP 非全局区域都支持可靠数据报套接字 (Reliable Datagram Socket, RDS) IPC 协议。RDSv3 驱动程序作为 SMF 服务 rds 启用。缺省情况下，安装后禁用此服务。通过区域管理员授予的相应授权，可以在给定非全局区域中启用该服务。在执行 zlogin 后，可以在要运行 rds 的每个区域中启用 rds。

示例 16–1 如何在非全局区域中启用 rds 服务

1. 要在专用 IP 或共享 IP 区域中启用 RDSv3 服务，请执行 zlogin 并执行 svcadm enable 命令：

   ```
   # svcadm enable rds
   ```

2. 验证 rds 是否已启用：

   ```
   # svcs rds
       STATE          STIME    FMRI
       online         22:50:53 svc:/system/rds:default
   ```

有关更多信息，请参见 svcadm(1M) 手册页。

共享 IP 非全局区域和专用 IP 非全局区域之间的安全差异

在共享 IP 区域中，此区域中的应用程序（包括超级用户）不能发送带有源 IP 地址的包，只能发送通过 zonecfg 实用程序指定给该区域的包。此类型的区域不能发送和接收任意数据链路（第 2 层）包。

但是，对于专用 IP 区域，zonecfg 会将指定数据链路的一切权限都授予该区域。因此，在专用 IP 区域中，超级用户或具有所需权限配置文件的用户可以如同在全局区域中一样，在这些数据链路上发送欺骗性包。可以通过设置 allowed-address 属性来禁用 IP 地址欺骗。对于 anet 资源，可以通过设置 link-protection 属性来启用其他保护（例如，mac-nospoof 和 dhcp-nospoof）。

同时使用共享 IP 和专用 IP 非全局区域

共享 IP 区域总是与全局区域共享 IP 层，而专用 IP 区域总是有其自己的 IP 层实例。共享 IP 区域和专用 IP 区域都可在同一计算机中使用。

在区域中挂载的文件系统

缺省情况下，每个区域都有一个委托给该区域的 ZFS 数据集。这一缺省委托数据集模拟缺省全局区域的数据集布局。名为 .../rpool/ROOT 的数据集包含引导环境。不应直接处理该数据集。rpool 数据集必须存在，缺省情况下挂载在 ... /rpool 下。.../rpool/export 和 .../rpool/export/home 数据集挂载在 /export 和 /export/home 下。这些非全局区域数据集与对应的全局区域数据集具有相同的用途，并可以按相同的方式进行管理。区域管理员可以在 .../rpool、.../rpool/export 和 ... /rpool/export/home 数据集中创建附加数据集。

不应使用 zfs(1M) 手册页中描述的 zfs 命令创建、删除或重命名以区域的 rpool/ROOT 文件系统开始的分层结构中的文件系统。zfs 命令可用于设置 canmount、mountpoint、sharesmb、zoned、com.oracle.*:*、com.sun:* 和 org.opensolaris.*.* 以外的属性。

通常，在区域中挂载的文件系统包括：

- 初始化虚拟平台时挂载的文件系统集合
- 在应用程序环境本身中挂载的文件系统集合

例如，这些集合可以包括以下文件系统：

- 具有 mountpoint（none 或 legacy 除外）且 canmount 属性值为 yes 的 ZFS 文件系统。
- 在区域的 /etc/vfstab 文件中指定的文件系统。
- AutoFS 挂载和 AutoFS 触发的挂载。使用 sharectl(1M) 中介绍的 sharectl 设置 autofs 属性。
- 区域管理员明确执行的挂载

  正在运行的区域内的文件系统挂载权限也可通过 zonecfg fs-allowed 属性进行定义。此属性不适用于通过使用 zonecfg add fs 或 add dataset 资源挂载到区域中的文件系统。缺省情况下，区域内只允许挂载区域的缺省委托数据集中的文件系统、hsfs 文件系统和网络文件系统（如 NFS）。

注意 – 将对在应用程序环境中执行的非缺省挂载设定特定限制。这些限制可防止区域管理员拒绝为系统的其余部分提供服务，或者对其他区域产生不良影响。

在区域中挂载特定的文件系统时存在安全限制。其他文件系统在区域中挂载时会显示出特殊行为。有关更多信息，请参见第 314 页中的 “文件系统和非全局区域”。

有关数据集的更多信息，请参见 `datasets(5)` 手册页。有关 BE 的更多信息，请参见《创建和管理 Oracle Solaris 11.1 引导环境》

文件系统挂载和更新

不支持会隐藏区域系统映像中的任何文件、符号链接或目录的文件系统挂载方式，如 `pkg(5)` 手册页中所述。例如，如果安装的软件包都没有在 `/usr/local` 中提供内容，则允许在 `/usr/local` 下挂载文件系统。但是，如果有任何软件包（包括传统 SVR4 软件包）在以 `/usr/local` 开头的路径下提供了文件、目录或符号链接，则不支持在 `/usr/local` 下挂载文件系统。支持在 `/mnt` 下临时挂载文件系统。

由于文件系统在区域中的挂载顺序，如果 `/export` 来自区域的 `rpool/export` 数据集或其他委托数据集，则不能用 `fs` 资源在 `/export/filesys` 下挂载文件系统。

区域中的主机 ID

您可以为非全局区域设置 `hostid` 属性，该属性与全局区域的 `hostid` 属性不同。例如，如果机器迁移到其他系统上的区域，将执行此操作。区域内的现有应用程序可能取决于原始 `hostid`。有关更多信息，请参见第 210 页中的 “资源类型和属性”。

非全局区域中的 /dev 文件系统

`zonecfg` 命令使用与规则匹配的系统来指定应在特定区域中出现的设备。与其中一个规则匹配的设备包括在区域的 `/dev` 文件系统中。有关更多信息，请参见第 232 页中的 “如何配置区域”。

非全局区域中的可删除 lofi 设备

可在非全局区域中配置可移除的回送文件 `lofi` 设备，其工作方式与 CD-ROM 设备类似。可以更改设备映射到的文件并创建以只读模式使用相同文件的多个 `lofi` 设备。该类型的 `lofi` 设备是使用带 `-r` 选项的 `lofiadm` 命令创建的。创建时不一定要指定文件名。在可移除 `lofi` 设备的生命周期内，可以将文件与空设备关联，或取消文件与非空

设备的关联。一个文件可以同时与多个可移除 lofi 设备安全关联。可移除 lofi 设备是只读的。不能对已经映射到普通的读写 lofi 设备或可移除 lofi 设备的文件进行重新映射。潜在 lofi 设备的数量受 zone.max-lofi 资源控制的限制，可以在全局区域中使用 zonecfg(1M) 设置该资源控制。

可移除 lofi 设备在创建后是只读的。如果对可移除 lofi 设备执行任何写操作，lofi 驱动程序将返回错误。

lofiadm 命令还可用于列出可移除 lofi 设备。

示例 16–2 创建带关联文件的可移除 lofi 设备

```
# lofiadm -r /path/to/file
/dev/lofi/1
```

示例 16–3 创建一个空的可移除 lofi 设备

```
# lofiadm -r
/dev/lofi/2
```

示例 16–4 将文件插入可移除 lofi 设备

```
# lofiadm -r /path/to/file /dev/lofi/1
/dev/lofi/1
```

有关更多信息，请参见 lofiadm(1M)、zonecfg(1M) 和 lofi(7D) 手册页。另请参见第 76 页中的 “区域范围的资源控制”。

非全局区域中的磁盘格式支持

可通过 zonecfg 工具启用磁盘分区和使用 uscsi 命令。有关示例，请参见第 214 页中的 “资源类型属性”中的 device。有关 uscsi 命令的更多信息，请参见 uscsi(7I)。

- 仅 solaris 区域支持委托。
- 磁盘必须使用通过使用带 -D 选项的 prtconf 命令显示的 sd 目标。请参见 prtconf(1M)。

可配置的特权

引导区域时，配置中包括**安全**特权的缺省集合。这些特权被视为安全特权，因为它们可以阻止区域中的特权进程影响系统中其他非全局区域或全局区域中的进程。您可使用 zonecfg 命令执行以下操作：

- 将特权添加至缺省特权集，需要了解此类更改可能允许一个区域中的进程通过控制全局资源来影响其他区域中的进程。

- 从缺省特权集中删除特权，需要了解此类更改可能会阻止某些进程正常运行（如果这些进程要求具有特定特权才能运行的话）。

注 – 目前，有些特权不能从区域的缺省特权集中删除，还有一些特权不能添加到缺省特权集中。

有关更多信息，请参见第 327 页中的 “非全局区域中的特权”、第 232 页中的 “如何配置区域”和 `privileges(5)`。

资源池关联

如果按第 13 章，创建和管理资源池（任务）所述在系统中配置了资源池，则可在配置区域时使用 `pool` 属性将该区域与其中一个资源池相关联。

如果未配置资源池，还可使用 `dedicated-cpu` 资源来指定在某个非全局区域运行时将系统处理器的某个子集专用于该非全局区域。系统将动态创建一个临时池，以便在区域运行时使用。根据 `zonecfg` 的指定，池配置将在迁移期间进行传播。

注 – 使用通过 `pool` 属性设置的永久池的区域配置与通过 `dedicated-cpu` 资源配置的临时池不兼容。只能设置这两个属性中的其中一个。

设置区域范围的资源控制

全局管理员或具有相应授权的用户可以为区域设置区域范围的特权资源控制。区域范围的资源控制可限制区域内所有进程实体总的资源使用情况。

使用 `zonecfg` 命令同时为全局区域和非全局区域指定这些限制。请参见第 232 页中的 “如何配置区域”。

设置区域范围的资源控制的首选简单方法是使用属性名称或资源（如 `capped-cpu`），而不使用 `rctl` 资源（如 `cpu-cap`）。

`zone.cpu-cap` 资源控制可以对某个区域可占用的 CPU 资源量设置绝对限制。值 `100` 表示将一个 CPU 的 100% 用作设置。值 `125` 表示 125%，因为在使用 CPU 上限时，100% 对应于系统中的一个 CPU。

注 – 设置 `capped-cpu` 资源时，可以使用小数来表示单位。该值对应于 `zone.cpu-cap` 资源控制，但设置减小 100 倍。设置为 `1` 等效于资源控制设置 `100`。

zone.cpu-shares 资源控制可以对区域的公平份额调度器 (fair share scheduler, FSS) CPU 份额数设置限制。CPU 份额首先分配给区域，然后在区域内的项目之间进一步分配，如 project.cpu-shares 项中所述。有关更多信息，请参见第 356 页中的 “在安装了区域的 Oracle Solaris 系统上使用公平份额调度器”。此控制的全局属性名称是 cpu-shares。

zone.max-locked-memory 资源控制可以限制某个区域可以使用的锁定物理内存量。可以使用 project.max-locked-memory 资源控制来控制如何在区域中的项目间分配锁定内存资源。有关更多信息，请参见表 6–1。

zone.max-lofi 资源控制可以限制某个区域可以创建的潜在 lofi 设备的数量。

zone.max-lwps 资源控制通过禁止一个区域中有过多 LWP 影响其他区域，来增强资源隔离功能。对此区域中项目的 LWP 资源的分配可使用 project.max-lwps 资源控制进行控制。有关更多信息，请参见表 6–1。此控制的全局属性名称是 max-lwps。

zone.max-processes 资源控制通过防止某个区域使用太多的进程表槽并给其他区域造成影响，来增强资源隔离。可以使用第 73 页中的 “可用的资源控制”中介绍的 project.max-processes 资源控制来设置区域内各项目间的进程表槽资源分配。此控制的全局属性名称是 max-processes。zone.max-processes 资源控制还包括 zone.max-lwps 资源控制。如果设置了 zone.max-processes，但未设置 zone.max-lwps，则在引导区域时 zone.max-lwps 将被隐式设置为 zone.max-processes 值的 10 倍。注意，由于常规进程和僵进程都使用进程表槽，因此 max-processes 控制可以防止僵进程用尽进程表。由于僵进程没有定义任何 LWP，因此 max-lwps 无法防止这种可能性。

zone.max-msg-ids、zone.max-sem-ids、zone.max-shm-ids 和 zone.max-shm-memory 资源控制可用于限制区域中的所有进程使用的 System V 资源。对区域中项目的 System V 资源的分配可使用这些资源控制的项目版本来进行控制。这些控制的全局属性名称是 max-msg-ids、max-sem-ids、max-shm-ids 和 max-shm-memory。

zone.max-swap 资源控制可限制区域中的用户进程地址空间映射和 tmpfs 挂载所占用的交换空间。prstat -Z 的输出将显示一个 SWAP 列。报告的交换是区域进程和 tmpfs 挂载所使用的总交换量。此值有助于监视每个区域预留的交换空间，可用于选择适当的 zone.max-swap 设置。

表 16–1 区域范围的资源控制

控制名称	全局属性名称	说明	缺省单位	所用值
zone.cpu-cap		此区域可用的 CPU 资源量的绝对限制	数量（CPU 数目），以百分比表示 注 – 设置 capped-cpu 资源时，可以使用小数来表示单位。	
zone.cpu-shares	cpu-shares	此区域的公平份额调度器 (fair share scheduler, FSS) CPU 份额数	数量（份额）	
zone.max-locked-memory		区域可用的锁定物理内存的总量 如果将 priv_proc_lock_memory 指定给某个区域，请考虑同时设置此资源控制，以防止该区域锁定所有内存。	大小（字节）	capped-memory 的 locked 属性
zone.max-lofi	max-lofi	可由区域创建的潜在 lofi 设备的数量限制	数量（lofi 设备的数量）	
zone.max-lwps	max-lwps	此区域可同时使用的最大 LWP 数	数量 (LWP)	
zone.max-msg-ids	max-msg-ids	此区域允许的最大消息队列 ID 数	数量（消息队列 ID）	
zone.max-processes	max-processes	此区域可同时使用的最大进程表槽数	数量（进程表槽数）	
zone.max-sem-ids	max-sem-ids	此区域允许的最大信号量 ID 数	数量（信号量 ID）	
zone.max-shm-ids	max-shm-ids	此区域允许的最大共享内存 ID 数	数量（共享内存 ID）	
zone.max-shm-memory	max-shm-memory	此区域允许的系统 V 共享内存总量	大小（字节）	

表 16–1 区域范围的资源控制 （续）

控制名称	全局属性名称	说明	缺省单位	所用值
zone.max-swap		可用于此区域的用户进程地址空间映射和 tmpfs 挂载的交换空间总量	大小（字节）	capped-memory 的 swap 属性

可以使用 prctl 命令为正运行的进程指定这些限制。第 356 页中的 “如何使用 prctl 命令在全局区域中设置 FSS 份额”中还提供了一个示例。通过 prctl 命令指定的限制不是持久的。在重新引导系统后，此限制将失效。

包含区域注释

您可以使用 attr 资源类型为区域添加注释。有关更多信息，请参见第 232 页中的 “如何配置区域”。

使用 zonecfg 命令

zonecfg 命令（在 zonecfg(1M) 手册页中介绍）用于配置非全局区域。

也可以使用 zonecfg 命令永久指定全局区域的资源管理设置。例如，可以使用此命令将全局区域配置为通过使用 dedicated-cpu 资源来使用专用 CPU。

zonecfg 命令可以在交互模式、命令行模式或命令文件模式下使用。可以使用此命令执行以下操作：

- 创建或删除（销毁）区域配置
- 将资源添加到特定配置
- 为添加到配置的资源设置属性
- 从特定配置中删除资源
- 查询或检验配置
- 提交到配置
- 恢复到先前配置
- 重命名区域
- 从 zonecfg 会话中退出

zonecfg 提示符的格式如下：

zonecfg:*zonename*>

当您配置特定的资源类型（例如文件系统）时，此资源类型也包含在提示符中：

zonecfg:*zonename*:fs>

有关更多信息，包括显示如何使用本章中所述的各种 zonecfg 组件的过程，请参见第 17 章，规划和配置非全局区域（任务）。

zonecfg 模式

范围的概念用于用户界面。范围可以是**全局**的，也可以是**资源特定**的。缺省范围为全局。

在全局范围内，add 子命令和 select 子命令用于选择特定资源。然后范围更改为此资源类型。

- 对于 add 子命令，end 或 cancel 子命令用于完成资源指定。
- 对于 select 子命令，end 或 cancel 子命令用于完成资源修改。

然后范围恢复为全局。

某些子命令（例如 add、remove 和 set）在每个范围中都有不同的语义。

zonecfg 交互模式

在交互模式中，支持以下子命令。有关用于这些子命令的语义和选项的详细信息，请参见 zonecfg(1M) 手册页。对于可能会导致破坏性操作或所做工作丢失的任何子命令，系统均要求用户在继续之前进行确认。您可以使用 -F（强制）选项，跳过此项确认操作。

help　列显一般帮助，或者显示有关给定资源的帮助。

```
zonecfg:my-zone:capped-cpu> help
```

create　开始为指定的新区域配置内存中的配置，以实现以下用途之一：

- 将 Oracle Solaris 缺省设置应用于新的配置。此方法为缺省方法。
- 与 -t *template* 选项一起使用时，用于创建与指定模板相同的配置。区域名称从模板名称更改为新区域名称。
- 与 -F 选项一起使用时，用于覆盖现有配置。
- 与 -b 选项一起使用时，用于创建其中未设置任何内容的空配置。

export　采用可以在命令文件中使用的格式，在标准输出或指定输出文件中列显配置。

add　在全局范围中，将指定的资源类型添加到配置。

在资源范围中，添加具有给定名称和给定值的属性。

有关更多信息，请参见第 232 页中的 “如何配置区域”和 zonecfg(1M) 手册页。

`set` 将给定属性名称设置为给定属性值。请注意，某些属性（例如 `zonepath`）为全局属性，而其他属性则为资源特定的属性。因此，此命令适用于全局范围和资源范围。

`select` 仅适用于全局范围。选择与给定属性名称-属性值对的修改条件相匹配的给定类型资源。将范围更改为此资源类型。您必须为要唯一标识的资源指定足够数量的属性名称-值对。

`clear` 清除可选的设置的值。不能清除必需设置。但可以通过指定新值来更改某些必需设置。

`remove` 在全局范围中，删除指定的资源类型。您必须为要唯一标识的资源类型指定足够数量的属性名称-值对。如果没有指定属性名称-值对，则会删除所有实例。当存在多个属性名称-值对时，如果未使用 `-F` 选项，则需要进行确认。

在资源范围中，从当前资源中删除指定的属性名称-属性值。

`end` 仅适用于资源范围。结束资源指定。

然后，`zonecfg` 命令将检验是否完全指定当前资源。

- 如果资源完全指定，则可以将其添加到内存中的配置，并且范围将恢复为全局。
- 如果未完全指定，则系统将显示一条描述需要执行何种操作的错误消息。

`cancel` 仅适用于资源范围。结束资源指定并将范围重置为全局。系统不会保留任何未完全指定的资源。

`delete` 销毁指定的配置。从内存和稳定存储器中删除配置。您必须将 `-F`（强制）选项与 `delete` 一起使用。

注意 – 此操作为即时操作。不需要提交，并且无法恢复已删除的区域。

`info` 显示有关当前配置或全局资源属性 `zonepath`、`autoboot` 和 `pool` 的信息。如果指定了资源类型，则仅显示有关此类型资源的信息。在资源范围中，此子命令仅应用于要添加或修改的资源。

`verify` 检验当前配置是否正确。确保所有资源都指定了所有必需的属性。检验任何 `rootzpool` 资源组及其属性的语法。不会检验任何用 URI 指定的存储的可访问性。

`commit` 将当前配置从内存提交到稳定存储器。在提交内存中的配置之前，可以使用 `revert` 子命令删除更改。必须提交配置以供 `zoneadm` 使用。完成 `zonecfg` 会话时，便会自动尝试此操作。由于仅可提交正确的配置，因此，提交操作将自动进行检验。

`revert` 将配置恢复到上次提交时的状态。

`exit` 退出 `zonecfg` 会话。您可以将 `-F`（强制）选项与 `exit` 一起使用。

如果需要，会自动尝试 `commit`。请注意，也可以使用 EOF 字符退出会话。

zonecfg 命令文件模式

在命令文件模式中，输入来自文件。可以使用第 208 页中的 “zonecfg 交互模式”中所述的 `export` 子命令生成此文件。可以在标准输出中列显配置，也可以使用 `-f` 选项指定输出文件。

区域配置数据

区域配置数据由两种类型的实体组成：资源和属性。每个资源都有一种类型，并且每个资源还可以有一个包含一个或多个属性的集合。属性具有名称和值。属性集取决于资源类型。

唯一必需的属性是 `zonename` 和 `zonepath`。

资源类型和属性

资源和属性类型如下所述：

`zonename` 区域的名称。以下规则适用于区域名称：

- 每个区域必须具有唯一的名称。
- 区域名称区分大小写。
- 区域名称必须以字母数字字符开头。

  名称可以包含字母数字字符、下划线 (_)、连字符 (-) 和句点 (.)。
- 名称不能超过 63 个字符。
- 名称 `global` 和所有以 `SYS` 开头的名称均保留，不能使用。

`zonepath` `zonepath` 属性指定将在其下安装区域的路径。每个区域都具有一个与全局区域根目录相对的根目录路径。安装时，需要全局区域目录以提供限定的可见性。区域路径必须由 `root` 拥有，并且模式为 `700`。如果区域路径不存在，将在安装期间自动创建该路径。如果权限不正确，将自动进行更正。

非全局区域的根路径低一个级别。区域的根目录与全局区域中的根目录 (/) 具有相同的所有权和权限。区域目录必须由 root 所有，并且模式为 755。此分层结构可防止全局区域中的非特权用户遍历非全局区域的文件系统。

区域必须位于 ZFS 数据集中。在安装或附加区域时，将自动创建 ZFS 数据集。如果无法创建 ZFS 数据集，也无法安装或附加区域。

路径	说明
/zones/my-zone	zonecfg zonepath
/zones/my-zone/root	区域的根目录

有关更多信息，请参见第 319 页中的 “遍历文件系统”。

注 – 通过使用 zoneadm 的 move 子命令指定一个完整的新 zonepath，可将区域移至同一系统上的其他位置。有关说明，请参见第 269 页中的 “移动非全局区域”。

---

autoboot

如果此属性设置为 true，则引导全局区域时会自动引导该区域。缺省设置为 false。请注意，如果禁用了区域服务 svc:/system/zones:default，则无论如何设置此属性，区域都不会自动引导。您可以使用 svcadm(1M) 手册页中所述的 svcadm 命令来启用区域服务：

```
global# svcadm enable zones
```

有关在 pkg 更新期间进行此设置的信息，请参见第 303 页中的 “区域包管理概述”。

bootargs

此属性用于为区域设置引导参数。除非被 reboot、zoneadm boot 或 zoneadm reboot 命令覆盖，否则将应用该引导参数。请参见第 254 页中的 “区域引导参数”。

limitpriv

此属性用于指定缺省特权集之外的特权掩码。请参见第 327 页中的 “非全局区域中的特权”。

通过指定特权名称可添加特权，特权名称中可包含或不包含前导 priv_。在特权名称前添加破折号 (-) 或感叹号 (!) 可以排除特权。特权值以逗号分隔，并放在引号 (") 内。

如 priv_str_to_set(3C) 中所述，特殊特权集 none、all 和 basic 对其标准定义进行了扩展。由于区域配置在全局区域内进行，因

此不能使用特殊特权集 zone。由于常见用法是通过添加或删除某些特权来更改缺省特权集，因此特殊特权集 default 将映射为缺省特权集。当 default 出现在 limitpriv 属性开头时，它将扩展为缺省权限集。

以下条目增加了使用 DTrace 程序的功能，该程序只要求区域中具有 dtrace_proc 和 dtrace_user 特权：

```
global# zonecfg -z userzone
zonecfg:userzone> set limitpriv="default,dtrace_proc,dtrace_user"
```

如果区域的特权集包含不允许的特权、缺少必需特权或包含未知特权，则检验、准备或引导该区域的尝试都将失败，并将显示错误消息。

scheduling-class

此属性可为区域设置调度类。有关其他信息和提示，请参见第 194 页中的 “调度类”。

ip-type

必须为所有非全局区域设置该属性。请参见第 199 页中的 “专用 IP 非全局区域”、第 198 页中的 “共享 IP 非全局区域”和第 232 页中的 “如何配置区域”。

dedicated-cpu

此资源可在某个区域运行时使系统处理器的某个子集专用于该区域。dedicated-cpu 资源可为 ncpus 以及 importance（可选的）提供限制。有关更多信息，请参见第 193 页中的 “dedicated-cpu 资源”。

capped-cpu

此资源对区域在运行时可占用的 CPU 资源量设置限制。capped-cpu 资源可为 ncpus 提供限制。有关更多信息，请参见第 194 页中的 “capped-cpu 资源”。

capped-memory

此资源可对为区域设置内存上限时使用的属性分组。capped-memory 资源可为 physical、swap 和 locked 内存提供限制。至少必须指定其中一个属性。要使用 capped-memory 资源，必须在全局区域中安装 service/resource-cap 软件包。

anet

anet资源会在区域引导时自动为专用 IP 区域创建临时 VNIC 接口，并在区域停止时删除该接口。

net

net 资源可将全局区域中的现有网络接口指定给非全局区域。网络接口资源是接口名称。当区域从已安装状态转换为就绪状态时，每个区域都可以具有可以设置的网络接口。

dataset

数据集是描述文件系统、卷或快照的通用术语。添加 ZFS 数据集资源可以将存储管理委托给非全局区域。如果委托数据集为文件系统，区域管理员可在该数据集内创建和销毁文件系统以及修改数据集的属性。区域管理员可创建快照、子文件系统和卷以及其后代的克隆。如果委托数据集为卷，则区域管理员可设置属性和创建快照。区域管理员无法影响尚未添加到区域的数据集，也无

法超过对指定给区域的数据集设置的任何顶层配额。将数据集委托给非全局区域后，将自动设置 zoned 属性。zoned 文件系统不能挂载在全局区域中，因为区域管理员可能需要将挂载点设置为不可接受的值。

可以按以下方式将 ZFS 数据集添加到区域中。

- 作为一个 lofs 挂载文件系统（在目标单独与全局区域共享空间时）
- 作为一个委托数据集

请参见《Oracle Solaris 11.1 管理：ZFS 文件系统》中的第 9 章 “Oracle Solaris ZFS 高级主题”、第 314 页中的 “文件系统和非全局区域”和 datasets(5) 手册页。

有关数据集问题的信息，另请参见第 28 章，各种 Oracle Solaris Zones 问题的故障排除。

fs

当区域从已安装状态转换为就绪状态时，每个区域都可以拥有已挂载的各种文件系统。文件系统资源指定文件系统挂载点的路径。有关在区域中使用文件系统的更多信息，请参见第 314 页中的 “文件系统和非全局区域”。

注 – 要在非全局区域中通过 fs 资源使用 UFS 文件系统，必须在安装后或通过 AI 清单脚本将 system/file-system/ufs 软件包安装到区域中。

不能使用 quota(1M) 中所述的 quota 命令来检索通过 fs 资源添加的 UFS 文件系统的配额信息。

---

fs-allowed

设置该属性后，区域管理员便能够挂载该类型的任何文件系统（不论是区域管理员创建的文件系统，还是使用 NFS 导入的文件系统），并可以管理该文件系统。正在运行的区域内的文件系统挂载权限也受 fs-allowed 属性限制。缺省情况下，区域内仅允许挂载 hsfs 文件系统和网络文件系统（如 NFS）。

该属性还可用于块设备或委托给区域的 ZVOL 设备。

fs-allowed 属性接受可以在区域中挂载的其他文件系统的逗号分隔列表，例如，ufs,pcfs。

```
zonecfg:my-zone> set fs-allowed=ufs,pcfs
```

该属性不会影响全局区域通过 add fs 或 add dataset 属性管理的区域挂载。

有关安全注意事项，请参见第 314 页中的 “文件系统和非全局区域”和第 324 页中的 “非全局区域中的设备使用”。

`device`

设备资源是与设备匹配的说明符。当区域从已安装状态转换为就绪状态时，每个区域都具有应配置的设备。

注 – 要在非全局区域中通过 `device` 资源使用 UFS 文件系统，必须在安装后或通过 AI 清单脚本将 `system/file-system/ufs` 软件包安装到区域中。

`pool`

此属性用于将区域与系统中的资源池相关联。多个区域可以共享一个池的资源。另请参见第 193 页中的 “`dedicated-cpu` 资源”。

`rctl`

`rctl` 资源用于区域范围的资源控制。当区域从已安装状态转换为就绪状态时，将启用这些控制。

有关更多信息，请参见第 204 页中的 “设置区域范围的资源控制”。

注 – 要使用 `zonefig` 的 *set* `global_property_name` 子命令而非 `rctl` 资源来配置区域范围的控制，请参见第 232 页中的 “如何配置区域”。

`attr`

此通用属性可用于用户注释或其他子系统。`attr` 的 `name` 属性必须以字母数字字符开头。`name` 属性可以包含字母数字字符、连字符 (`-`) 和句点 (`.`)。以 `zone.` 开头的属性名称将保留，以供系统使用。

资源类型属性

资源也有要配置的属性。以下属性与所示的资源类型关联。

`admin`

针对给定区域定义该用户的用户名和授权。

```
zonecfg:my-zone> add admin
zonecfg:my-zone:admin> set user=zadmin
zonecfg:my-zone:admin> set auths=login,manage
zonecfg:my-zone:admin> end
```

`auths` 属性可以采用以下值：

- `login` (`solaris.zone.login`)
- `manage` (`solaris.zone.manage`)
- `clone` (`solaris.zone.clonefrom`)

请注意，不能通过这些 auths 创建区域。该功能包含在区域安全性配置文件中。

rootzpool　　storage

标识为区域安装提供专用 ZFS zpool 的存储对象 URI。有关 URI 和 storage 允许值的信息，请参见第 195 页中的 “rootzpool 资源”。在区域安装期间，将自动创建 zpool，或导入预先创建的 zpool。指定 *my-zone*_rpool 名称。

```
zonecfg:my-zone> add rootzpool
zonecfg:my-zone:rootzpool> add storage dev:dsk/c4t1d0
zonecfg:my-zone:rootzpool> end
```

如果要创建镜像配置，可以添加其他 storage 属性：

```
add storage dev:dsk/c4t1d0
add storage dev:dsk/c4t3d0
```

一个区域只能配置一个 rootzpool 资源。

zpool　　storage、name

定义将 zpool 委托给区域的一个或多个存储对象 URI。有关 URI 和 storage 属性允许值的信息，请参见第 195 页中的 “rootzpool 资源”。zpool(1M) 手册页中定义了 name 属性的允许值。

在此示例中，将一个 zpool 存储资源委托给区域。在安装期间，将自动创建 zpool，或导入以前创建的 zpool。zpool 的名称为 my-zone_pool1。

```
zonecfg:my-zone> add zpool
zonecfg:my-zone:zpool> set name=pool1
zonecfg:my-zone:zpool> add storage dev:dsk/c4t2d0
zonecfg:my-zone:zpool> add storage dev:dsk/c4t4d0
zonecfg:my-zone:zpool> end
```

一个区域可以配置有一个或多个 zpool 资源。

dedicated-cpu　　ncpus、importance

指定 CPU 个数以及池的相对重要性（可选的）。以下示例指定了供区域 my-zone 使用的 CPU 范围，还设置了 importance。

```
zonecfg:my-zone> add dedicated-cpu
zonecfg:my-zone:dedicated-cpu> set ncpus=1-3
zonecfg:my-zone:dedicated-cpu> set importance=2
zonecfg:my-zone:dedicated-cpu> end
```

capped-cpu　　ncpus

指定 CPU 数目。以下示例指定了供区域 my-zone 使用的 CPU 的 CPU 上限为 3.5 个。

```
zonecfg:my-zone> add capped-cpu
zonecfg:my-zone:capped-cpu> set ncpus=3.5
zonecfg:my-zone:capped-cpu> end
```

capped-memory　　physical、swap、locked

为区域 my-zone 指定内存限制。每个限制均是可选的，但至少要设置一个限制。

```
zonecfg:my-zone> add capped-memory
zonecfg:my-zone:capped-memory> set physical=50m
zonecfg:my-zone:capped-memory> set swap=100m
zonecfg:my-zone:capped-memory> set locked=30m
zonecfg:my-zone:capped-memory> end
```

要使用 capped-memory 资源，必须在全局区域中安装 resource-cap 软件包。

fs　　dir、special、raw、type、options

fs 资源参数提供的值可确定如何以及在何处挂载文件系统。fs 参数定义如下：

dir　　为文件系统指定挂载点

special　　指定要从全局区域挂载的特殊块设备名称或目录

raw　　指定在挂载文件系统之前运行 fsck 所在的原始设备（不适用于 ZFS）

type　　指定文件系统类型

options　　指定挂载选项，这些选项类似于使用 mount 命令找到的挂载选项

以下示例中的行指定全局区域中名为 pool1/fs1 的数据集在所配置的区域中被挂载为 /shared/fs1。所使用的文件系统类型为 ZFS。

```
zonecfg:my-zone> add fs
zonecfg:my-zone:fs> set dir=/shared/fs1
zonecfg:my-zone:fs> set special=pool1/fs1
zonecfg:my-zone:fs> set type=zfs
zonecfg:my-zone:fs> end
```

有关参数的更多信息，请参见第 314 页中的 “-o nosuid 选项”、第 316 页中的 “安全限制和文件系统行为”、fsck(1M) 和 mount(1M) 手册页。另请注意，有关专用于特定文件系统的挂载选项的信息可以在 1M 手册页部分中找到。这些手册页名称的格式为 mount_*filesystem*。

注 – 不能使用 quota(1M) 中所述的 quota 命令来检索通过该资源添加的 UFS 文件系统的配额信息。

dataset name, alias

name

以下示例的几行代码指定数据集 *sales* 将在非全局区域中可见并在该区域中进行挂载，但在全局区域中不再可见。

```
zonecfg:my-zone> add dataset
zonecfg:my-zone> set name=tank/sales
zonecfg:my-zone> end
```

委托数据集可以具有一个下例中所示的非缺省别名。请注意，数据集别名不能含有正斜杠 (/)。

```
zonecfg:my-zone> add dataset
zonecfg:my-zone:dataset> set name=tank/sales
zonecfg:my-zone:dataset> set alias=data
zonecfg:my-zone:dataset> end
```

要恢复缺省别名，请使用 clear alias。

```
zonecfg:my-zone> clear alias
```

anet

linkname、lower-link 、allowed-address、auto-mac-address、configure-allowed-address、defrouter、linkmode (IPoIB)、mac-address（非 IPoIB）、mac-slot（非 IPoIB）、mac-prefix（非 IPoIB）、mtu、maxbw、pkey (IPoIB)、priority、vlan-id（非 IPoIB）、rxfanout、rxrings、txrings、link-protection、allowed-dhcp-cids

请勿在 zonecfg 中为 IPoIB 数据链路设置以下 anet 属性：

- mac-address
- mac-prefix
- mac-slot
- vlan-id

请勿在 zonecfg 中为非 IPoIB 数据链路设置以下 anet 属性。

- linkmode
- pkey

anet 资源在区域引导时创建自动 VNIC 接口或 IPoIB 接口，在区域停止时删除 VNIC 或 IPoIB 接口。资源属性通过 zonecfg 命令进行管理。有关可用属性的完整信息，请参见 zonecfg(1M) 手册页。

`lower-link`	为要创建的链路指定底层链路。如果设置为 `auto`，每次区域引导时，`zoneadmd` 守护进程都会自动选择用来创建 VNIC 的链路。 选择了自动创建 VNIC 的数据链路时，引导期间将跳过所有 IPoIB 链路。
`linkname`	为自动创建的 VNIC 或 IPoIB 接口指定一个名称。
`mac-address`（**不适用于 IPoIB**）	根据指定值或关键字设置 VNIC 的 MAC 地址。如果该值不是关键字，则将其解释为单点传送 MAC 地址。有关支持的关键字，请参见 `zonecfg(1M)` 手册页。如果选择随机 MAC 地址，则区域引导、区域分离和附加操作将保留生成的地址。
`pkey`（**仅限 IPoIB**）	设置用于创建 IPoIB 数据链路接口的分区密钥。此属性为强制属性。指定的 `pkey` 始终作为十六进制数处理，无论其是否有 `0x` 前缀。
`linkmode`（**仅限 IPoIB**）	设置数据链路接口的 `linkmode`。缺省值为 `cm`。有效值包括： `cm`（缺省值）　连接模式。此模式使用缺省 MTU（`65520` 字节），其支持的最大 MTU 为

		65535 字节。
	ud	不可靠的数据报模式。如果远程节点无法使用连接模式，则会自动改用不可靠的数据报模式。此模式使用缺省 MTU（2044 字节），其支持的最大 MTU 为 4092 字节。
allowed-address		为专用 IP 区域配置 IP 地址，同时限制专用 IP 区域可以使用的可配置 IP 地址集。要指定多个地址，请使用逗号分隔的 IP 地址列表。
defrouter		当非全局区域和全局区域驻留在单独的网络上时，可使用 defrouter 属性来设置缺省路由。 设置了 defrouter 属性的任何区域必须位于没有为全局区域配置的子网上。

zonecfg 命令使用 SYSdefault 模板创建区域时，如果没有设置其他 IP 资源，将在区域配置中自动包括具有以下属性的 anet 资源。将在物理以太网链路上自动创建 linkname，该名称将设置为第一个可用名称，格式为 net*N*, net*0*。要更改这些缺省值，请使用 zonecfg 命令。

缺省值通过物理以太网链路（如 nxge0）创建自动 VNIC，并向 VNIC 指定出厂 MAC 地址。可选的 lower-link 属性设置为要创建的自动 VNIC 的底层链路 (nxge0)。可以使用 zonecfg 命令指定链路名称、底层物理链路、MAC 地址、带宽限制等 VNIC 属性以及其他 VNIC 属性。请注意，还必须指定 ip-type=exclusive。

```
zonecfg:my-zone> set ip-type=exclusive
zonecfg:my-zone> add anet
zonecfg:my-zone:anet> set linkname=net0
zonecfg:my-zone:anet> set lower-link=auto
zonecfg:my-zone:anet> set mac-address=random
zonecfg:my-zone:anet> set link-protection=mac-nospoof
zonecfg:my-zone:anet> end
```

以下示例显示在物理链路 net5 上配置了 IPoIB 数据链路接口（使用 IB 分区密钥 0xffff）的区域：

```
zonecfg:my-zone> set ip-type=exclusive
zonecfg:my-zone:anet> add anet
zonecfg:my-zone:anet> set linkname=ib0
zonecfg:my-zone:anet> set lower-link=net5
zonecfg:my-zone:anet> set pkey=0xffff
zonecfg:my-zone:anet> end
```

有关属性的更多信息，请参见 zonecfg(1M) 手册页。有关链路属性的更多信息，请参见 dladm(1M) 手册页。

net	address、allowed-addressphysical、defrouter

注 – 对于共享 IP 区域，必须指定 IP 地址和物理设备这两项。或者设置缺省路由器。

对于专用 IP 区域，只需指定物理接口。

- `allowed-address` 属性可以限制专用 IP 区域可以使用的可配置 IP 地址集。
- 当非全局区域和全局区域驻留在单独的网络上时，可使用 `defrouter` 属性来设置缺省路由。
- 设置了 `defrouter` 属性的任何区域必须位于没有为全局区域配置的子网上。
- 来自具有缺省路由器的区域的通信将在回到目标区域之前先进入路由器中。

当共享 IP 区域位于不同的子网上时，请不要在全局区域中配置数据链路。

在以下共享 IP 区域示例中，物理接口 `nge0` 被添加到 IP 地址为 `192.168.0.1` 的区域中。要列出系统上的网络接口，请键入：

```
global# ipadm show-if -po ifname,class,active,persistent
lo0:loopback:yes:46--
nge0:ip:yes:----
```

除回送行以外的每一行输出将包含一个网络接口的名称。说明中包含 `loopback` 的行不适用于卡。`46` 持久性标志表示该接口已在全局区域中永久配置。`yes` 活动值表示当前已配置接口，`ip` 类值表示 `nge0` 是一个非回送接口。区域的缺省路由设置为 `10.0.0.1`。`defrouter` 属性的设置为可选的。请注意，`ip-type=shared` 为必需项。

```
zonecfg:my-zone> set ip-type=shared
zonecfg:my-zone> add net
zonecfg:my-zone:net> set physical=nge0
zonecfg:my-zone:net> set address=192.168.0.1
zonecfg:my-zone:net> set defrouter=10.0.0.1
zonecfg:my-zone:net> end
```

在以下专用 IP 区域的示例中，`bge32001` 链路用于物理接口，该接口是 `bge1` 上的 VLAN。要确定哪些数据链路可用，请使用命令 `dladm show-link`。`allowed-address` 属性可对区域可以使用的 IP 地址加以限制。`defrouter` 属性用于设置缺省路由。请注意，还必须指定 `ip-type=exclusive`。

```
zonecfg:my-zone> set ip-type=exclusive
zonecfg:my-zone> add net
```

```
zonecfg:myzone:net> set allowed-address=10.1.1.32/24
zonecfg:my-zone:net> set physical=bge32001
zonecfg:myzone:net> set defrouter=10.1.1.1
zonecfg:my-zone:net> end
```

在 add net 步骤中只会指定物理设备类型。physical 属性可以为 VNIC，具体请参见《Oracle Solaris Administration: Network Interfaces and Network Virtualization》中的第 III 部分, "Network Virtualization and Resource Management"。

注 – Oracle Solaris 操作系统支持所有以太网类型的接口，并且可以使用 dladm 命令来管理其数据链路。

---

device　　match、allow-partition、allow-raw-io

要匹配的设备名称可以是匹配模式或绝对路径。allow-partition 和 allow-raw-io 都可以设置为 true 或 false。缺省值是 /。allow-partition 可执行分区。allow-raw-io 可以执行 uscsi。有关这些资源的更多信息，请参见 zonecfg(1M)。

在以下示例中，区域配置中包括对磁盘设备的 uscsi 操作。

```
zonecfg:my-zone> add device
zonecfg:my-zone:device> set match=/dev/*dsk/cXtYdZ*
zonecfg:my-zone:device> set allow-raw-io=true
zonecfg:my-zone:device> end
```

使用 add device 将 Veritas 卷管理器设备委托给非全局区域。

注意 – 添加设备前，请参见第 324 页中的 "非全局区域中的设备使用"、第 325 页中的 "在非全局区域中运行应用程序"和第 327 页中的 "非全局区域中的特权"，以了解限制和有关安全的注意事项。

---

rctl　　name、value

以下是可用的区域范围的资源控制。

- zone.cpu-cap
- zone.cpu-shares（首选： cpu-shares ）
- zone.max-locked-memory
- zone.max-lofi
- zone.max-lwps（首选： max-lwps）
- zone.max-msg-ids（首选： max-msg-ids）

- zone.max-processes（首选：max-processes）
- zone.max-sem-ids（首选： max-sem-ids）
- zone.max-shm-ids（首选： max-shm-ids ）
- zone.max-shm-memory（首选： max-shm-memory）
- zone.max-swap

注意，设置区域范围资源控制的首选的、更简单的方法是使用属性名称，而不是使用 rctl 资源，如第 232 页中的 “如何配置区域”中所示。如果区域中的区域范围资源控制条目是使用 add rctl 配置的，则其格式与 project 数据库中的资源控制条目不同。在区域配置中，rctl 资源类型由三个名称/值对组成。名称分别是 priv、limit 和 action。每个名称都有一个简单值。

```
zonecfg:my-zone> add rctl
zonecfg:my-zone:rctl> set name=zone.cpu-shares
zonecfg:my-zone:rctl> add value (priv=privileged,limit=10,action=none)
zonecfg:my-zone:rctl> end

zonecfg:my-zone> add rctl
zonecfg:my-zone:rctl> set name=zone.max-lwps
zonecfg:my-zone:rctl> add value (priv=privileged,limit=100,action=deny)
zonecfg:my-zone:rctl> end
```

有关资源控制和属性的一般信息，请参见第 6 章，资源控制（概述）和第 326 页中的 “在非全局区域中使用的资源控制”。

attr　　name、type、value

在以下示例中，添加了有关区域的注释。

```
zonecfg:my-zone> add attr
zonecfg:my-zone:attr> set name=comment
zonecfg:my-zone:attr> set type=string
zonecfg:my-zone:attr> set value="Production zone"
zonecfg:my-zone:attr> end
```

可以使用 export 子命令在标准输出中列显区域配置。通过可以在命令文件中使用的格式保存配置。

区域配置示例

区域配置示例包括以下区域：

- 位于共享存储上的具有 rootzpool 资源的区域。操作系统软件存储在专用 zpool 中，所有非操作系统软件和数据存储在属于区域根数据集后代的数据集中。创建操作系统快照和克隆操作系统的系统操作也会对非操作系统软件和数据创建快照和进行克隆。

- 位于共享存储上的具有 rootzpool 资源和一个或多个 zpool 资源的区域。操作系统存储在 rootzpool 中，所有非操作系统软件和数据存储在其他 zpool 中。创建操作系统快照和克隆操作系统的系统操作不包括非操作系统软件和数据。
- 具有缺省的系统本地 zonepath 的区域。该区域在 zonepath 中存储操作系统。创建操作系统快照和克隆操作系统的系统操作可以包括非操作系统软件和数据。

Tecla 命令行编辑库

配置中提供了 Tecla 命令行编辑库，可与 zonecfg 命令一起使用。此库为命令行历史记录和编辑支持提供了一种机制。

有关更多信息，请参见 tecla(5) 手册页。

◆ ◆ ◆ 第 17 章

规划和配置非全局区域（任务）

本章介绍在系统上配置区域之前需要执行的操作，同时还介绍了如何在系统上配置区域、修改区域配置以及删除区域配置。

有关区域配置过程的介绍，请参见第 16 章，非全局区域配置（概述）。

有关 `solaris10` 标记区域配置的信息，请参见第 3 部分。

规划和配置非全局区域（任务列表）

在将系统设置为使用区域之前，必须先收集信息并决定如何配置区域。以下任务列表概括了如何规划和配置区域。

任务	说明	参考
规划区域策略。	■ 评估在系统上运行的应用程序，并确定需要在区域中运行的应用程序。 ■ 评估磁盘空间的可用性，以便可以保存区域内特有的文件。 ■ 如果您也使用资源管理功能，请确定如何使资源管理范围能够覆盖整个区域。 ■ 如果要使用资源池，请根据需要配置池。 ■ 决定是否应在共享存储上托管区域。	请参阅历史使用情况。另请参见第 227 页中的 “磁盘空间需求”和第 129 页中的 “区域中使用的资源池”。

任务	说明	参考
确定区域名称。	基于命名约定决定区域的名称。	请参见第 210 页中的 “区域配置数据”和第 228 页中的 “区域主机名”。
确定区域路径（必需）。	每个区域都具有一个与全局区域根目录相对的根目录路径。	请参见第 210 页中的 “区域配置数据”。
如果没有配置资源池，请评估是否需要进行 CPU 限制。请注意，根据 `zonecfg` 的指定，池设置将在迁移期间进行传播。	查看您的应用程序要求。	请参见第 193 页中的 “`dedicated-cpu` 资源”。
如果计划使用全局区域中的 `rcapd` 来为区域设置内存上限，请评估是否需要进行内存分配。	查看您的应用程序要求。	请参见第 10 章，使用资源上限设置守护进程控制物理内存（概述）、第 11 章，管理资源上限设置守护进程（任务）和第 194 页中的 “物理内存控制和 `capped-memory` 资源”。
将 FSS 设置为系统中的缺省调度程序。	为每个区域指定 CPU 份额，以控制区域访问 CPU 资源的权利。FSS 保证为各个区域公平地分配 CPU 资源，这种公平分配基于已分配的份额。	第 8 章，公平份额调度器（概述），第 194 页中的 “调度类”。
请注意，区域的缺省类型为专用 IP。	对于用 `anet` 资源配置的专用 IP 区域，系统在每次引导区域时会自动创建一个 VNIC。对于用 `net` 资源配置的专用 IP 区域，请确定要指定给该区域的数据链路。该区域需要独占访问一个或多个网络接口。该接口可以是 VNIC、单独的 LAN（如 `bge1`），也可以是单独的 VLAN（如 `bge2000`）。请参见《Oracle Solaris Administration: Network Interfaces and Network Virtualization》中的第 III 部分, “Network Virtualization and Resource Management”。 如果配置的是共享 IP 区域，请获得或配置该区域的 IP 地址。根据配置的不同，您必须为需要网络访问的每个非全局区域获取至少一个 IP 地址。	请参见第 228 页中的 “确定区域主机名和网络要求”、第 232 页中的 “如何配置区域”和《配置和管理 Oracle Solaris 11.1 网络》。

任务	说明	参考
确定要在区域内挂载的文件系统。	查看您的应用程序要求。	有关更多信息，请参见第 201 页中的 “在区域中挂载的文件系统”。
确定应使哪些网络接口可在区域中使用。	查看您的应用程序要求。	有关更多信息，请参见第 321 页中的 “共享 IP 网络接口”。
确定是否必须更改缺省的非全局区域权限集。	检查特权集：缺省特权集、可以添加和删除的特权，以及目前不能使用的特权。	请参见第 327 页中的 “非全局区域中的特权”。
确定每个区域中应该配置的设备。	查看您的应用程序要求。	有关应用程序的信息，请参阅相关文档。
配置区域。	使用 `zonecfg` 可以创建区域的配置。	请参见第 231 页中的 “配置、检验并提交区域”。
检验并提交已配置的区域。	确定指定的资源和属性是否在虚拟系统上有效。	请参见第 231 页中的 “配置、检验并提交区域”。

评估当前的系统设置

可以在任何运行 Oracle Solaris 10 或更新发行版的计算机上使用区域。以下主要的计算机注意事项与区域的使用相关联。

- 每个区域内运行的应用程序的性能要求。
- 保存每个区域内特有文件的磁盘空间的可用性。

磁盘空间需求

对区域可以使用的磁盘空间量没有任何限制。全局管理员或具有相应授权的用户负责设置空间限制。全局管理员必须确保本地或共享存储足以保存非全局区域的根文件系统。即使小型单处理器系统也可支持同时运行多个区域。

非全局区域中安装的软件包的性质会影响区域的空间要求。软件包的数量也是一个因素。

磁盘要求由当前安装在全局区域中的软件包以及已安装软件所使用的磁盘空间决定。

区域要求每个区域至少具有 150 MB 的空闲磁盘空间。但是，当在全局区域中安装了所有标准 Oracle Solaris 软件包时，所需的空闲磁盘空间通常在 500 MB 到 1 GB 之间。如果添加更多软件，此数字还会增加。

建议每个区域再增加 40 MB 的 RAM，如果计算机有足够的交换空间则不作此要求。

限制区域大小

对于具有由 ZFS 数据集支持的 zonepath 的区域，可以使用 ZFS 数据集配额来限制区域大小。能够访问 zonepath 数据集的管理员可以修改数据集的 quota 和 reservation 属性，以控制每个区域可以使用的最大磁盘空间数量。这些属性在 zfs(1M)手册页中介绍。

管理员也可以创建具有固定大小的 ZFS 卷，并将区域安装在此卷的数据集中。卷会限制其中所安装的区域的大小。

确定区域主机名和网络要求

您必须确定区域的主机名。

在专用 IP 区域内，可按照为全局区域配置地址的方式来配置地址。

对于要具备网络连接的共享 IP 区域，必须执行以下一项操作：

- 为区域指定一个 IPv4 地址
- 为区域手动配置并指定一个 IPv6 地址

有关专用 IP 和共享 IP 类型的更多信息，请参见第 197 页中的 “区域网络接口”。

区域主机名

如果您正在使用 NIS 或 DNS 名称服务，或 LDAP 目录服务，则主机信息存储在服务器上的数据库中，例如 *hosts.byname*。

如果将本地文件用于命名服务，则 hosts 数据库将保留在 /etc/inet/hosts 文件中。区域网络接口的主机名从 /etc/inet/hosts 中的本地 hosts 数据库解析而来。或者，对于共享 IP 区域，可以在配置区域时直接指定 IP 地址，从而不需要对任何主机名进行解析。有关更多信息，请参见 hosts(4) 和 nodename(4) 手册页。另请参见《配置和管理 Oracle Solaris 11.1 网络》中的第 7 章 “IPv4 参考信息”。

共享 IP 区域网络地址

需要网络连接的每个共享 IP 区域都有一个或多个专有 IP 地址。同时支持 IPv4 和 IPv6 地址。

IPv4 区域网络地址

如果您使用的是 IPv4，则获取地址并将该地址指定到区域。

也可以指定 IP 地址前缀的长度。该前缀的格式为 *address*/*prefix-length*，例如 `192.168.1.1/24`。因此，要使用的地址是 `192.168.1.1`，要使用的网络掩码是 `255.255.255.0`，或者是前 24 位为 1 的掩码。

对于共享 IP 区域，可以在配置区域时直接指定 IP 地址，从而不需要对任何主机名进行解析。

有关更多信息，请参见 `hosts(4)`、`netmasks(4)` 和 `nodename(4)`。

IPv6 区域网络地址

如果您使用的是 IPv6，则必须手动配置地址。通常情况下，必须至少配置以下两种地址类型：

Link-local address	链路本地地址的格式为 `fe80::`*64-bit interface ID*`/10`。`/10` 表明前缀长度为 10 位。
由子网上配置的全局前缀构成的地址	全局单点传送地址基于管理员为每个子网配置的 64 位前缀以及一个 64 位接口 ID。在配置为使用 IPv6 的同一子网上的任何系统上运行 `ipadm show-addr` 命令，可以获得该前缀。 64 位接口 ID 通常是从系统的 MAC 地址派生而来。为了便于区域使用，可使用如下方式从全局区域的 IPv4 地址中派生出唯一的备用地址： `16 bits of zero:upper 16 bits of IPv4 address:lower 16 bits of IPv4 address:a zone-unique number` 例如，如果全局区域的 IPv4 地址是 192.168.200.10，则对于使用 `1` 作为区域专有数字的非全局区域，适合的链路本地地址是 `fe80::c0a8:c80a:1/10`。如果在该子网中使用的全局前缀是 `2001:0db8:aabb:ccdd/64`，则同一非全局区域的唯一全局单点传送地址是 `2001:0db8:aabb:ccdd::c0a8:c80a:1/64`。请注意，在配置 IPv6 地址时，您必须指定前缀长度。

有关链路本地地址和全局单点传送地址的更多信息，请参见 `ipadm(1M)` 和 `inet6(7P)` 手册页。

专用 IP 区域网络地址

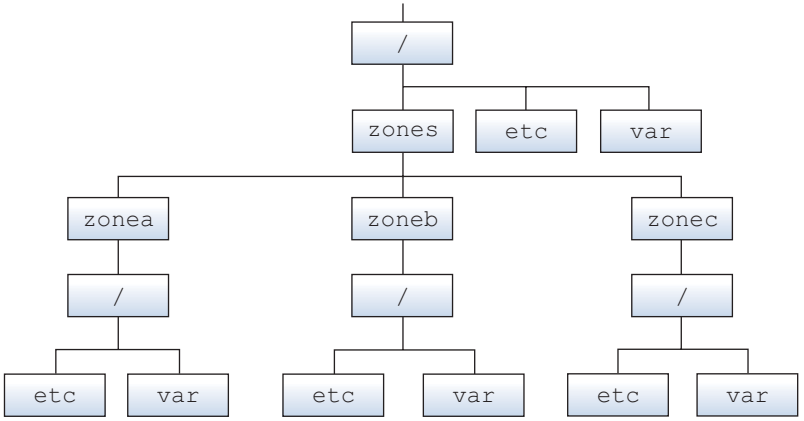
在独占 IP 区域内，可按照在全局区域中的方式来配置地址。请注意，可使用 DHCP 和 IPv6 无状态地址自动配置配置地址。

文件系统配置

在设置虚拟平台时，您可以指定一些要执行的挂载。使用回送虚拟文件系统 (loopback file system, LOFS) 回送挂载到区域的文件系统应使用 `nodevices` 选项挂载。有关 `nodevices` 选项的信息，请参见第 314 页中的 “文件系统和非全局区域”。

使用 LOFS，您可以创建一个新的虚拟文件系统，以便使用一个备用的路径名称访问文件。在非全局区域中，使用回送挂载可以使文件系统的分层结构看起来在区域根目录下是重复的。在该区域中，使用以区域的根目录开头的路径名，可以访问所有文件。LOFS 挂载将保留文件系统名称空间。

图 17–1 回送挂载的文件系统



有关更多信息，请参见 `lofs`(7S) 手册页。

创建、修订和删除非全局区域配置（任务列表）

任务	说明	参考
配置非全局区域。	使用 zonecfg 命令可以创建区域、检验并提交该配置。您也可以使用脚本在系统上配置和引导多个区域。 可以使用 zonecfg 命令来显示非全局区域的配置。	第 231 页中的 “配置、检验并提交区域”、第 237 页中的 “配置多个区域的脚本”
修改区域配置。	使用这些过程可修改区域配置中的资源类型，修改属性类型（如区域名称），或为区域添加专用设备。	第 242 页中的 “使用 zonecfg 命令修改区域配置”
恢复或删除区域配置。	使用 zonecfg 命令可以撤消对区域配置所做的资源设置，或删除区域配置。	第 245 页中的 “使用 zonecfg 命令恢复或删除区域配置”
删除区域配置。	使用带有 delete 子命令的 zonecfg 命令可以从系统中删除区域配置。	第 247 页中的 “如何删除区域配置”

配置、检验并提交区域

使用 zonecfg(1M) 手册页中所述的 zonecfg 命令可以执行以下操作。

- 创建区域配置
- 检验是否具备所需的全部信息
- 提交非全局区域配置

也可以使用 zonecfg 命令永久指定全局区域的资源管理设置。

当使用 zonecfg 实用程序配置区域时，您可以使用 revert 子命令来撤消资源设置。请参见第 245 页中的 “如何恢复区域配置”。

在系统上配置多个区域的脚本在第 237 页中的 “配置多个区域的脚本”中提供。

有关如何显示非全局区域配置，请参见第 242 页中的 “如何显示非全局区域的配置”。

▼ 如何配置区域

请注意，创建非全局区域的必需元素只有 zonename 和 zonepath 属性。其他资源和属性都是可选的。有些可选的资源还需要在备选项之间进行选择，例如决定使用 dedicated-cpu 资源还是 capped-cpu 资源。有关可用的 zonecfg 属性和资源的信息，请参见第 210 页中的 “区域配置数据”。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**

2. **用所选的区域名称来设置区域配置。**

   此示例过程中使用名称 my-zone。

   ```
   global# zonecfg -z my-zone
   ```

   如果是第一次配置该区域，则可以看到以下系统消息：

   ```
   my-zone: No such zone configured
   Use ’create’ to begin configuring a new zone.
   ```

3. **创建新的区域配置。**

   此过程使用缺省设置。

   ```
   zonecfg:my-zone> create
   create: Using system default template ’SYSdefault’
   ```

4. **设置区域路径，在此过程中为 /zones/my-zone。**

   ```
   zonecfg:my-zone> set zonepath=/zones/my-zone
   ```

   区域必须位于 ZFS 数据集中。在安装或附加区域时，将自动创建 ZFS 数据集。如果无法创建 ZFS 数据集，也无法安装或附加区域。请注意，如果区域路径有父目录，它必须是已挂载数据集的挂载点。

5. **设置自动引导值。**

   如果设置为 true，则在引导全局区域时将自动引导该区域。缺省值为 false。请注意，要自动引导区域，还必须启用区域服务 svc:/system/zones:default。缺省情况下将启用该服务。

   ```
   zonecfg:my-zone> set autoboot=true
   ```

6. **为区域设置持久引导参数。**

   ```
   zonecfg:my-zone> set bootargs="-m verbose"
   ```

7 **指定一个 CPU 专用于该区域。**

```
zonecfg:my-zone> add dedicated-cpu
```

a. **设置 CPU 数量。**

```
zonecfg:my-zone:dedicated-cpu> set ncpus=1-2
```

b. **（可选的）设置重要性。**

```
zonecfg:my-zone:dedicated-cpu> set importance=10
```

缺省值为 1。

c. **结束指定。**

```
zonecfg:my-zone:dedicated-cpu> end
```

8 **修改缺省特权集。**

```
zonecfg:my-zone> set limitpriv="default,sys_time"
```

该行添加了将系统时钟设置为缺省特权集的功能。

9 **将调度类设置为 FSS。**

```
zonecfg:my-zone> set scheduling-class=FSS
```

10 **添加内存上限。**

```
zonecfg:my-zone> add capped-memory
```

a. **设置内存上限。**

```
zonecfg:my-zone:capped-memory> set physical=1g
```

b. **设置交换内存上限。**

```
zonecfg:my-zone:capped-memory> set swap=2g
```

c. **设置锁定内存上限。**

```
zonecfg:my-zone:capped-memory> set locked=500m
```

d. **结束内存上限指定。**

```
zonecfg:my-zone:capped-memory> end
```

注 – 要使用 `capped-memory` 资源，全局区域中必须安装 `resource-cap` 软件包。

---

11 **添加文件系统。**

```
zonecfg:my-zone> add fs
```

a. **设置文件系统的挂载点，在此过程中为 `/usr/local`。**

```
zonecfg:my-zone:fs> set dir=/usr/local
```

b. 指定在区域中配置 /usr/local 之后，才能挂载全局区域中的 /opt/local。

```
zonecfg:my-zone:fs> set special=/opt/local
```

在非全局区域中，/usr/local 文件系统是可读写的。

c. 指定文件系统类型，在此过程中为 lofs。

```
zonecfg:my-zone:fs> set type=lofs
```

此类型指明了内核与文件系统的交互方式。

d. 结束文件系统指定。

```
zonecfg:my-zone:fs> end
```

可多次执行此步骤来添加多个文件系统。

12 如有必要，设置 hostid。

```
zonecfg:my-zone> set hostid=80f0c086
```

13 在存储池 *tank* 中添加一个名为 *sales* 的 ZFS 数据集。

```
zonecfg:my-zone> add dataset
```

a. 指定指向 ZFS 数据集 *sales* 的路径。

```
zonecfg:my-zone> set name=tank/sales
```

b. 结束数据集指定。

```
zonecfg:my-zone> end
```

区域管理员可在数据集中创建和销毁快照、文件系统和卷。区域管理员可以修改数据集的属性，并控制压缩和加密。

14 创建具有自动 VNIC 的专用 IP 区域。

```
zonecfg:my-zone> set ip-type=exclusive

zonecfg:my-zone> add anet
```

a. 为要创建的链接将 auto 指定为底层链接。

```
zonecfg:my-zone:anet> set lower-link=auto
```

每次引导区域时，zoneadmd 守护进程都会自动选择用来创建 VNIC 的链接。当选择数据链路时跳过 IPoIB 链路。

b. 结束指定。

```
zonecfg:my-zone:anet> end
```

15 添加设备。

```
zonecfg:my-zone> add device
```

a. 设置设备匹配，在此过程中为 /dev/sound/*。

```
zonecfg:my-zone:device> set match=/dev/sound/*
```

b. 结束设备指定。

```
zonecfg:my-zone:device> end
```

可多次执行此步骤来添加多个设备。

16 为 OFUV 组件（而不是 IB 诊断工具）添加开放光纤网络用户组件 (Open Fabric User Verb, OFUV) 设备。

```
zonecfg:my-zone> add device
```

a. 设置设备匹配，在此过程中为 infiniband/ofs/*。

```
zonecfg:my-zone:device> set match=infiniband/ofs/*
```

b. 结束设备指定。

```
zonecfg:my-zone:device> end
```

非全局区域不支持 IB 诊断工具。添加的设备可以与 OFUV 组件（例如谓词 (verb) 和 rdma_cm）一起使用。

可多次执行此步骤来添加多个设备。

17 为 OFUV 组件（而不是 IB 诊断工具）添加 OFUV 设备。

```
zonecfg:my-zone> add device
```

a. 设置设备匹配，在此过程中为 infiniband/hca/*。

```
zonecfg:my-zone:device> set match=infiniband/hca/*
```

b. 结束设备指定。

```
zonecfg:my-zone:device> end
```

非全局区域不支持 IB 诊断工具。添加的设备可以与 OFUV 组件（例如谓词 (verb) 和 rdma_cm）一起使用。

可多次执行此步骤来添加多个设备。

18 要允许使用 format 命令标记磁盘，应将整个磁盘/LUN 委托到区域，并设置 allow-partition 属性。

```
zonecfg:my-zone> add device
```

a. 设置设备匹配，在此过程中为 /dev/*dsk/c2t40d3*。

```
zonecfg:my-zone:device> set match=/dev/*dsk/c2t40d3*
```

b. 将 **allow-partition** 设置为 **true**。

```
zonecfg:my-zone:device> set allow-partition=true
```

c. 结束设备指定。

```
zonecfg:my-zone:device> end
```

可多次执行此步骤来添加多个设备。

19 要允许在磁盘中执行 **uscsi** 操作，应设置 **allow-raw-io** 属性。

```
zonecfg:my-zone> add device
```

a. 设置设备匹配，在此过程中为 **/dev/*dsk/c2t40d3***。

```
zonecfg:my-zone:device> set match=/dev/*dsk/c2t40d3*
```

b. 将 **allow-raw-io** 设置为 **true**。

```
zonecfg:my-zone:device> set allow-raw-io=true
```

c. 结束设备指定。

```
zonecfg:my-zone:device> end
```

注意 – 如果允许区域在磁盘上执行 uscsi 操作，也会允许区域访问与磁盘连接到同一总线的任何其他设备。因此，启用此功能可能会带来安全风险，让攻击者有机可乘去攻击全局区域或使用同一总线上资源的其他区域。请参见 uscsi(7I)。

可多次执行此步骤来添加多个设备。

20 使用属性名称添加区域范围的资源控制。

```
zonecfg:my-zone> set max-sem-ids=10485200
```

可多次执行此步骤来添加多个资源控制。

21 使用 **attr** 资源类型来添加注释。

```
zonecfg:my-zone> add attr
```

a. 将名称设置为 **comment**。

```
zonecfg:my-zone:attr> set name=comment
```

b. 将类型设置为 **string**。

```
zonecfg:my-zone:attr> set type=string
```

c. 将值设置为说明区域的注释。

```
zonecfg:my-zone:attr> set value="This is my work zone."
```

d. 结束 **attr** 资源类型指定。

```
zonecfg:my-zone:attr> end
```

22 **检验区域的配置。**

```
zonecfg:my-zone> verify
```

23 **提交区域的配置。**

```
zonecfg:my-zone> commit
```

24 **退出 zonecfg 命令。**

```
zonecfg:my-zone> exit
```

请注意，即使您没有在提示符下明确键入 `commit`，也会在键入 `exit` 或出现 EOF 时自动执行 `commit`。

更多信息 **在命令行中使用多个子命令**

提示 – `zonecfg` 命令还支持通过同一个 shell 调用多条子命令，这些子命令放在引号中并用分号进行分隔。

```
global# zonecfg -z my-zone "create ; set zonepath=/zones/my-zone"
```

对于共享 IP 区域，只能在 `zonecfg net` 资源中指定静态地址。不能使用命令行提供地址。

下一步执行的操作

请参见第 258 页中的 “安装和引导区域”来安装已提交的区域配置。

配置多个区域的脚本

可以使用此脚本在系统中配置和引导多个区域。所创建的区域缺省情况下是带有 `anet` 资源的独占 IP 区域。

在执行脚本之前，先通过运行 SCI 工具创建一个配置文件：

```
global# sysconfig create-profile -o sc_config.xml
```

此脚本采用以下参数：

- 要创建的区域个数
- *zonename* 前缀
- 可用作基目录的目录
- 新创建的配置文件的完整路径名

您必须是在全局区域中具有 root 特权的全局管理员或具有适当权限配置文件的用户才能执行此脚本。

```
#!/bin/ksh
#
# Copyright 2006-2012 Oracle Corporation. All rights reserved.
# Use is subject to license terms.
#

#
# This script serves as an example of how to instantiate several zones
# with no administrative interaction.  Run the script with no arguments to
# get a usage message.  The general flow of the script is:
#
#   1) Parse and check command line arguments
#   2) Configure all zones that are not yet configured
#   3) Install the first zone, if needed
#   4) Create the remaining zones as clones of the first zone
#
# Upon successful completion, the requested number of zones will be
# been installed and booted.
#

export PATH=/usr/bin:/usr/sbin

me=$(basename $0)
function fail_usage {
    print -u2 "Usage:
    $me <#-of-zones> <zonename-prefix> <basedir> <sysconfig.xml>

Generate sysconfig.xml with:
    sysconfig create-profile -o sysconfig.xml

When running sysconfig, choose \"Automatically\" or \"None\" for network
configuration. The value entered for \"Computer Name\" will ignored:
each zone's nodename will be set to match the zone name."

    exit 2
}

function log {
    print "$(date +%T) $@"
}

function error {
    print -u2 "$me: ERROR: $@"
}

function get_zone_state {
    zoneadm -z "$1" list -p 2>/dev/null | cut -d: -f3
}

#
# Parse and check arguments
#
(( $# != 4 )) && fail_usage

# If $1 is not a number nzones will be set to 0.
integer nzones=$1
if (( nzones < 1 )); then
    error "Invalid number of zones \"$1\""
    fail_usage
```

```
fi
# Be sure that zonename prefix is an allowable zone name and not too long.
prefix=$2
if [[ $prefix != @([a-zA-Z0-9])*([-_.a-zA-Z0-9]) || ${#prefix} > 62 ]]; then
    error "Invalid zonename prefix"
    fail_usage
fi
# Be sure that basedir is an absolute path.  zoneadm will create the directory
# if needed.
dir=$3
if [[ $dir != /* ]]; then
    error "Invalid basedir"
    fail_usage
fi
# Be sure the sysconfig profile is readable and ends in .xml
sysconfig=$4
if [[ ! -f $sysconfig || ! -r $sysconfig || $sysconfig != *.xml ]]; then
    error "sysconfig profile missing, unreadable, or not *.xml"
    fail_usage
fi

#
# Create a temporary directory for all temp files
#
export TMPDIR=$(mktemp -d /tmp/$me.XXXXXX)
if [[ -z $TMPDIR ]]; then
    error "Could not create temporary directory"
    exit 1
fi
trap 'rm -rf $TMPDIR' EXIT

#
# Configure all of the zones
#
for (( i=1; i <= nzones; i++ )); do
    zone=$prefix$i
    state=$(get_zone_state $zone)
    if [[ -n $state ]]; then
        log "Skipping configuration of $zone: already $state"
        continue
    fi

    log "Configuring $zone"
    zonecfg -z "$zone" "create; set zonepath=$dir/$zone"
    if (( $? != 0 )); then
        error "Configuration of $zone failed"
        exit 1
    fi
done

#
# Install the first zone, then boot it for long enough for SMF to be
# initialized.  This will make it so that the first boot of all the clones
# goes much more quickly.
#
zone=${prefix}1
state=$(get_zone_state $zone)
if [[ $state == configured ]]; then
    log "Installing $zone"
```

```
    # Customize the nodename in the sysconfig profile
    z_sysconfig=$TMPDIR/$zone.xml
    search="<propval type=\"astring\" name=\"nodename\" value=\".*\"/>"
    replace="<propval type=\"astring\" name=\"nodename\" value=\"$zone\"/>"
    sed "s|$search|$replace|" $sysconfig > $z_sysconfig

    zoneadm -z $zone install -c $z_sysconfig
    if (( $? != 0 )); then
        error "Installation of $zone failed."
        rm -f $z_sysconfig
        exit 1
    fi
    rm -f $z_sysconfig
elif [[ $state != installed ]]; then
    error "Zone $zone is currently in the $state state."
    error "It must be in the installed state to be cloned."
    exit 1
fi
# Boot the zone no further than single-user.  All we really want is for
# svc:/system/manifest-import:default to complete.
log "Booting $zone for SMF manifest import"
zoneadm -z $zone boot -s
if (( $? != 0 )); then
    error "Failed to boot zone $zone"
    exit 1
fi
# This zlogin will return when manifest-import completes
log "Waiting for SMF manifest import in $zone to complete"
state=
while [[ $state != online ]]; do
    printf "."
    sleep 1
    state=$(zlogin $zone svcs -Ho state \
        svc:/system/manifest-import:default 2>/dev/null)
done
printf "\n"
log "Halting $zone"
zoneadm -z $zone halt
if (( $? != 0 )); then
    error "failed to halt $zone"
    exit 1
fi
firstzone=$zone

#
# Clone and boot the remaining zones
#
for (( i=2; i <= $nzones; i++ )); do
    zone=$prefix$i

    # Be sure that it needs to be installed
    state=$(get_zone_state $zone)
    if [[ $state != configured ]]; then
        log "Skipping installation of $zone: current state is $state."
        continue
    fi

    log "Cloning $zone from $firstzone"
```

```
    # Customize the nodename in the sysconfig profile
    z_sysconfig=$TMPDIR/$zone.xml
    search='<propval type="astring" name="nodename" value=".*"/>'
    replace='<propval type="astring" name="nodename" value="'$zone'"/>'
    sed "s|$search|$replace|" $sysconfig > $z_sysconfig

    # Clone the zone
    zoneadm -z $zone clone -c $z_sysconfig $firstzone
    if (( $? != 0 )); then
        error "Clone of $firstzone to $zone failed"
        rm -f $z_sysconfig
        exit 1
    fi
    rm -f $z_sysconfig

    # Boot the zone
    log "Booting $zone"
    zoneadm -z $zone boot
    if (( $? != 0 )); then
        error "Boot of $zone failed"
        exit 1
    fi
done

#
# Boot the first zone now that clones are done
#
log "Booting $firstzone"
zoneadm -z $firstzone boot
if (( $? != 0 )); then
    error "Boot of $firstzone failed"
    exit 1
fi

log "Completed in $SECONDS seconds"
exit 0
```

脚本的输出：

```
$ ./buildzones
Usage:
        buildzones <#-of-zones> <zonename-prefix> <basedir> <sysconfig.xml>

Generate sysconfig.xml with:
        sysconfig create-profile -o sysconfig.xml

When running sysconfig, choose "Automatically" or "None" for network
configuration. The value entered for "Computer Name" will be ignored:
each zone's nodename will be set to match the zone name.

# ~user/scripts/buildzones 3 bz /tank/bz /var/tmp/sysconfig.xml
12:54:04 Configuring bz1
12:54:05 Configuring bz2
12:54:05 Configuring bz3
12:54:05 Installing bz1
A ZFS file system has been created for this zone.
Progress being logged to /var/log/zones/zoneadm.20110816T195407Z.bz1.install
```

```
      Image: Preparing at /tank/bz/bz1/root.

 Install Log: /system/volatile/install.24416/install_log
AI Manifest: /usr/share/auto_install/manifest/zone_default.xml
  SC Profile: /tmp/buildzones.F4ay4T/bz1.xml
    Zonename: bz1
Installation: Starting ....
```

▼ 如何显示非全局区域的配置

您必须是全局区域中的全局管理员或具有适当权限配置文件的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**
2. **显示区域配置。**

```
global# zonecfg -z zonename info
```

使用 zonecfg 命令修改区域配置

也可使用 zonecfg 命令执行以下操作：

- 修改区域配置中的资源类型
- 清除区域配置中的属性值
- 在区域中添加专用设备
- 修改区域的特权集
- 添加和删除存储

▼ 如何修改区域配置中的资源类型

可以选择一个资源类型并修改该资源的指定。

您必须是全局区域中的全局管理员或具有适当权限配置文件的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**
2. **选择要修改的区域，在此过程中为 my-zone。**

```
global# zonecfg -z my-zone
```

3. **选择要更改的资源类型，例如，资源控制。**

```
zonecfg:my-zone> select rctl name=zone.cpu-shares
```

4. **删除当前值。**

```
zonecfg:my-zone:rctl> remove value (priv=privileged,limit=20,action=none)
```

5 **添加新值。**

```
zonecfg:my-zone:rctl> add value (priv=privileged,limit=10,action=none)
```

6 **结束修改后的 rctl 的指定。**

```
zonecfg:my-zone:rctl> end
```

7 **提交区域的配置。**

```
zonecfg:my-zone> commit
```

8 **退出 zonecfg 命令。**

```
zonecfg:my-zone> exit
```

请注意，即使您没有在提示符下明确键入 commit，也会在键入 exit 或出现 EOF 时自动执行 commit。

由 zonecfg 提交的更改在下次引导区域时生效。

▼ 如何清除区域配置中的属性

使用此过程可以重置一个独立属性。

1 **成为 root 用户或承担等效角色。**

2 **选择要修改的区域，在此过程中为 my-zone。**

```
global# zonecfg -z my-zone
```

3 **清除要更改的属性，在此过程中为现有的池关联。**

```
zonecfg:my-zone> clear pool
```

4 **提交区域的配置。**

```
zonecfg:my-zone> commit
```

5 **退出 zonecfg 命令。**

```
zonecfg:my-zone> exit
```

请注意，即使您没有在提示符下明确键入 commit，也会在键入 exit 或出现 EOF 时自动执行 commit。

由 zonecfg 提交的更改在下次引导区域时生效。

▼ 如何重命名区域

可以使用此过程对处于已配置状态或已安装状态的区域进行重命名。

请注意，具有 rootzpool 或 zpool 资源的区域在已安装状态下不能重命名，因为 zonename 是现有 zpool 名称的一部分。要重命名这些区域，请参见此过程结尾处的“**重命名共享存储上的区域**”。

您必须是全局区域中的全局管理员或具有适当权限配置文件的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**

2. **选择要重命名的区域，在此过程中为 my-zone。**

   ```
   global# zonecfg -z my-zone
   ```

3. **例如，将区域名称更改为 newzone。**

   ```
   zonecfg:my-zone> set zonename=newzone
   ```

4. **提交更改。**

   ```
   zonecfg:newzone> commit
   ```

5. **退出 zonecfg 命令。**

   ```
   zonecfg:newzone> exit
   ```

   由 zonecfg 提交的更改在下次引导区域时生效。

更多信息 **重命名共享存储上的区域**

具有 rootzpool 或 zpool 资源的已安装状态下的区域不能重命名，因为 zonename 是现有 zpool 名称的一部分。要重命名已安装和具有联机 zpool 的共享存储上的区域，请执行以下步骤。my-zone 区域在此过程中重命名。

- 在此过程中分离区域 my-zone：

  ```
  # zoneadm -z my-zone detach
  ```

- 使用 zonecfg 命令重命名此区域。

  ```
  # zonecfg -z my-zone ; "set zonename=newname ; set zonepath=/store/newname"
  ```

- 使用 zoneadm attach 重新附加区域。

  ```
  # zoneadm -z newname attach
  ```

▼ 如何在区域中添加专用设备

以下过程说明如何在非全局区域配置中放置扫描设备。

您必须是全局区域中的全局管理员或具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**

2 **添加设备。**

```
zonecfg:my-zone> add device
```

3 **设置设备匹配，在此过程中为 /dev/scsi/scanner/c3t4*。**

```
zonecfg:my-zone:device> set match=/dev/scsi/scanner/c3t4*
```

4 **结束设备指定。**

```
zonecfg:my-zone:device> end
```

5 **退出 zonecfg 命令。**

```
zonecfg:my-zone> exit
```

▼ 如何在全局区域中设置 zone.cpu-shares

可以使用此过程在全局区域中永久设置共享。

您必须是全局区域中的全局管理员或在全局区域中具有适当权限配置文件的用户才能执行此过程。

1 **成为 root 用户或承担等效角色。**

2 **使用 zonecfg 命令。**

```
# zonecfg -z global
```

3 **为全局区域设置 5 个份额。**

```
zonecfg:global> set cpu-shares=5
```

4 **退出 zonecfg。**

```
zonecfg:global> exit
```

使用 zonecfg 命令恢复或删除区域配置

使用 zonecfg(1M) 中所述的 zonecfg 命令可以恢复或删除区域配置。

▼ 如何恢复区域配置

当使用 zonecfg 实用程序配置区域时，请使用 revert 子命令来撤消对区域配置执行的资源设置。

您必须是全局区域中的全局管理员或在全局区域中具有区域安全权限配置文件的用户才能执行此过程。

1. 成为 root 用户或承担等效角色。

2. 在配置名为 **tmp-zone** 的区域时，键入 **info** 查看您的配置：

```
zonecfg:tmp-zone> info
```

配置的 net 资源段显示如下：

```
.
.
.
fs:
        dir: /tmp
        special: swap
        type: tmpfs
net:
        address: 192.168.0.1
        physical: eri0
device
        match: /dev/pts/*
.
.
.
```

3. 删除网络地址：

```
zonecfg:tmp-zone> remove net address=192.168.0.1
```

4. 检验 **net** 条目是否已被删除。

```
zonecfg:tmp-zone> info

.
.
.
fs:
        dir: /tmp
        special: swap
        type: tmpfs
device
        match: /dev/pts/*
.
.
.
```

5. 键入 **revert**。

```
zonecfg:tmp-zone> revert
```

6. 对下面的问题回答是：

```
Are you sure you want to revert (y/[n])? y
```

7 检验网络地址是否再次出现：

```
zonecfg:tmp-zone> info

.
.
.
fs:
        dir: /tmp
        special: swap
        type: tmpfs
net:
        address: 192.168.0.1
        physical: eri0
device
        match: /dev/pts/*
.
.
.
```

▼ 如何删除区域配置

使用带有 delete 子命令的 zonecfg，可以从系统中删除区域配置。

您必须是全局管理员或在全局区域中具有安全权限配置文件的用户才能执行此过程。

1 成为 root 用户或承担等效角色。

2 使用以下两种方法之一来删除区域 a-zone 的配置：

- 使用 -F 选项强制执行操作：

  ```
  global# zonecfg -z a-zone delete -F
  ```

- 对系统提示回答是，从而以交互方式删除区域：

  ```
  global# zonecfg -z a-zone delete
  Are you sure you want to delete zone a-zone (y/[n])? y
  ```

◆ ◆ ◆ 第 18 章

18 关于安装、关闭、停止、卸载和克隆非全局区域（概述）

本章介绍如何在 Oracle Solaris 系统上安装区域。同时还介绍管理虚拟平台和应用程序环境的两个进程，zoneadmd 和 zsched。此外，还提供了有关停止、重新引导、克隆和卸载区域的信息。

本章包含以下主题：

有关如何克隆、安装和引导非全局区域，或者停止或卸载非全局区域，请参见第 19 章，安装、引导、关闭、停止、卸载和克隆非全局区域（任务）。

有关 solaris10 标记区域安装的信息，请参见第 33 章，安装 solaris10 标记区域。

区域安装和管理概念

zoneadm(1M) 手册页中所述的 zoneadm 命令是用于安装和管理非全局区域的主要工具。必须从全局区域运行使用 zoneadm 命令的操作。如果使用基于角色的访问控制 (role-based access control, RBAC)，则用于生成其他区域副本的子命令需具备 solaris.zone.clonefrom/*source_zone* 授权。

可以使用 zoneadm 命令执行以下任务：

- 检验区域
- 安装区域
- 附加区域
- 将某个已安装区域的状态更改为 "incomplete"（不完整）

- 引导区域，类似于引导常规的 Oracle Solaris 系统
- 显示有关正在运行的区域的信息
- 关闭区域
- 停止区域
- 重新引导区域
- 卸载区域
- 将区域从系统中某个位置重定位到同一系统的另一位置
- 根据同一系统中某个现有区域的配置置备新区域
- 使用 `zonecfg` 命令迁移区域

有关区域安装和检验的过程，请参见第 19 章，安装、引导、关闭、停止、卸载和克隆非全局区域（任务）和 `zoneadm`(1M) 手册页。有关 `zoneadm list` 命令支持的选项，另请参阅 `zoneadm`(1M) 手册页。有关区域配置的过程，请参见第 17 章，规划和配置非全局区域（任务）和 `zonecfg`(1M) 手册页。区域状态在第 184 页中的 “非全局区域状态模型”中介绍。

如果您打算为区域生成 Oracle Solaris 审计记录，请在安装非全局区域之前先阅读第 332 页中的 “在区域中使用 Oracle Solaris 审计”。

区域构建

本节适用于初始非全局区域构建，不适用于现有区域的克隆。

使用传递给 `zoneadm install -m` 命令的清单所指定的软件包安装区域。如果未提供任何清单，则缺省清单使用 `pkg:/group/system/solaris-small-server`。新区域具有缺省 `solaris` 配置和日志（SMF 系统信息库、`/etc`、`/var`），它们只能由传递给 `zoneadm install -s` 的配置文件进行修改，以及通过在任何 `zonecfg add net` 条目中指定的联网信息来修改。

系统信息库、区域的已配置发布者和与全局区域保持同步的软件包在第 24 章，关于安装了区域的 Oracle Solaris 11.1 系统上的自动安装和软件包中讨论。

区域的根文件系统所需的文件由系统安装在区域的根路径下。

已成功安装的区域可进行引导和初始登录。

安装区域时，不引用或复制以下数据：

- 未安装的软件包
- CD 和 DVD 上的数据
- 网络安装映像

此外，以下信息类型（可能在全局区域中存在）也不会复制到正在安装的区域：

- `/etc/passwd` 文件中的新用户或已更改的用户
- `/etc/group` 文件中的新组或已更改的组

- 联网服务（例如，DHCP 地址指定）的配置
- 联网服务（例如，sendmail）的定制
- 网络服务（例如命名服务）的配置
- 新的或已更改的 `crontab`、打印机和邮件文件
- 系统日志、消息和记帐文件

如果使用 Oracle Solaris Auditing，则可能需要对文件进行修改。有关更多信息，请参见第 332 页中的 “在区域中使用 Oracle Solaris 审计”。

当区域从已安装状态转换为就绪状态时，便会添加在配置文件中指定的资源。系统会指定唯一的区域 ID。将挂载文件系统，设置网络接口并配置设备。转换为就绪状态之后，虚拟平台便可开始运行用户进程。在就绪状态下，会启动 `zsched` 和 `zoneadmd` 进程来管理虚拟平台。

- `zsched` 是一个类似于 `sched` 的系统调度进程，用于跟踪与区域关联的内核资源。
- `zoneadmd` 是区域管理守护进程。

处于就绪状态的区域中不存在任何正在执行的用户进程。就绪区域与正在运行的区域之间的主要差异在于，正在运行的区域中至少有一个进程正在执行。有关更多信息，请参见 `init(1M)` 手册页。

如何安装区域

`solaris` 标记安装程序支持使用以下任一方法来安装区域：

- 缺省系统信息库，即 solaris 发布者 (`http://pkg.oracle.com/solaris/release/`)。
- 运行 Oracle Solaris 发行版或 `solaris` 非全局区域的已安装系统的映像。

  系统映像可以是 ZFS 发送流。其他支持的映像包括 `cpio(1)` 归档文件或 `pax(1)` `xustar` 归档文件。`cpio` 归档文件可以使用 `gzip` 或 `bzip2` 实用程序进行压缩。该映像也可以是到系统根目录树顶级的路径，或者是预存在的区域路径。

  要从系统或非全局区域映像安装区域，需要使用 `-a` 或 `-d` 选项。如有必要，将执行软件包更新。如果 `-a` 或 `-d` 选项均未使用，则从软件系统信息库安装区域。
- 区域 BE，使用 `zoneadm install -z` *zbe*。如有必要，将执行软件包更新。

安装选项如下表所示。有关命令行示例，请参见第 259 页中的 “如何安装已配置的区域”。

选项	说明
`-m` *manifest*	AI 清单是一个 XML 文件，定义了如何安装区域。指定文件参数时必须使用绝对路径。

选项	说明
-c *profile*| *dir*	提供要在配置期间应用的配置文件或配置文件目录。指定文件参数时必须使用绝对路径。如果应用配置文件，则以非交互方式执行配置步骤。如果未提供配置文件，则使用交互式系统配置工具来配置系统。所有配置文件必须具有 .xml 文件扩展名。如果您为 -c 提供了目录选项，则该目录中的所有配置文件必须是格式正确的有效配置文件。
-a *archive*	用于安装非全局区域的归档文件的路径。归档文件可以使用 gzip 或 bzip 进行压缩。-d 和 -a 选项不兼容。 使用 - a *archive* 选项时，如有必要，将执行软件包更新。如果希望将该区域重新附加到源主机，可以使用 zoneadm attach 子命令。
-d *path*	已安装的系统或非全局区域的根目录路径。如有必要，将执行软件包更新。如果 *path* 是连字符 (-)，则假定 zonepath 已经填充了系统映像。-d 和 -a 选项不兼容。
-p	在安装区域后保留系统标识。-p 和 -u 选项不兼容。
-s	无提示安装。-s 和 -v 选项不兼容。
-u	在安装区域后取消区域的配置并在引导区域时提示进行新的配置。-p 和 -u 选项不兼容。
-U	如有必要，将所有软件包更新为与安装在全局区域中的软件包兼容的最新版本。
-v	安装过程的详细输出。-s 和 -v 选项不兼容。
-x	使用 force-zpool-import 和 -x 选项可强制导入显示为正在使用的任何 zpool。 如果存储对象包含预先存在的分区、zpool 或 UFS 文件系统，则 install 会失败并显示错误消息。可以使用 zoneadm install 的 -x 选项继续安装和覆盖所有预先存在的数据。该选项和 zpool create -f 命令类似。 使用 force-zpool-create-all 和 -x 选项强制创建所有 zpool 资源。使用 force-zpool-create= *zpoolname* 将选项限制在特定 zpool 或 zpool 组。有关用法，请参见 zoneadm(1M) 手册页。

zoneadmd 守护进程

区域管理守护进程 zoneadmd 是管理区域虚拟平台的主要进程。此守护进程还负责管理区域引导和关闭。对于系统上的每个活动（就绪、正在运行或正在关闭）区域，都有一个 zoneadmd 进程在运行。

zoneadmd 守护进程将按照区域配置中指定的方式设置区域。此过程包括以下操作：

- 分配区域 ID 并启动 zsched 系统进程
- 设置区域范围的资源控制
- 准备区域配置中指定的区域设备
- 设置网络接口
- 挂载回送文件系统和常规文件系统
- 实例化和初始化区域控制台设备

除非 zoneadmd 守护进程已经运行，否则它会由 zoneadm 自动启动。因此，如果此守护进程因某种原因没有运行，则调用 zoneadm 来管理区域时将重新启动 zoneadmd。

zoneadmd 守护进程的手册页为 zoneadmd(1M)。

zsched 区域调度程序

活动区域是指处于就绪状态、正在运行状态或正在关闭状态的区域。每个活动区域都有一个关联的内核进程 zsched。代表区域执行操作的内核线程由 zsched 所拥有。通过 zsched 进程，区域子系统可跟踪每个区域的内核线程。

区域应用程序环境

zoneadm 命令用于创建区域应用程序环境。

区域的内部配置通过使用 sysconfig 接口来指定。内部配置指定要使用的命名服务、缺省语言环境 (locale) 和时区、区域的 root 用户口令，以及应用程序环境的其他方面。sysconfig 接口在《安装 Oracle Solaris 11.1 系统》中的第 6 章 “取消配置或重新配置 Oracle Solaris 实例”和 sysconfig(1M) 手册页中介绍。请注意，可以独立于全局设置来配置区域的缺省语言环境和时区。

关于关闭、停止、重新引导和卸载区域

本节概述了停止、重新引导、卸载和克隆区域的过程。

关闭区域

`zoneadm shutdown c` 命令用于干净地关闭某个区域。此操作等效于在区域中运行 `/usr/sbin/init 0`。如果还指定了 `-r` 选项，将重新引导区域。有关支持的引导选项，请参见第 254 页中的 “区域引导参数”。

当全局区域关闭时，`svc:/system/zones` 服务会使用 `zoneadm shutdown` 来干净地关闭区域。

`shutdown` 子命令会等到区域成功关闭。如果此操作未在合理时间内完成，则可以使用 `zoneadm halt` 来强制停止区域。请参见第 265 页中的 “如何停止区域”。

停止区域

`zoneadm halt` 命令用于终止正在区域中运行的所有进程并删除虚拟平台。然后，区域便恢复为已安装状态。将中止所有进程，取消设备配置，销毁网络接口，卸载文件系统，以及销毁内核数据结构。

`halt` 命令**不**在区域内运行任何关闭脚本。要关闭区域，请参见第 254 页中的 “关闭区域”。或者，您可以登录到区域并运行关闭。请参见第 287 页中的 “如何使用 `zlogin` 关闭区域”。

停止操作失败时，请参见第 364 页中的 “区域无法停止”。

重新引导区域

`zoneadm reboot` 命令用于重新引导区域。区域将停止，然后再次引导。重新引导区域之后，区域 ID 会更改。

区域引导参数

区域支持用于 `zoneadm boot` 和 `reboot` 命令的以下引导参数：

- `-i` *altinit*
- `-m` *smf_options*
- `-s`

以下定义适用：

`-i` *altinit*
: 选择一个备用可执行文件作为第一个进程。*altinit* 必须是可执行文件的有效路径。缺省的第一个进程在 `init(1M)` 中进行了介绍。

`-m` *smf_options*
: 控制 SMF 的引导行为。有两类选项：恢复选项和消息选项。消息选项可确定启动期间显示的消息类型和数量。服务选项可确定用于引导系统的服务。

  恢复选项包括：

  debug
  : 打印标准的每个服务的输出以及所有要记录的 `svc.startd` 消息。

  `milestone=`*milestone*
  : 引导至由给定里程碑定义的子图。合法里程碑包括 `none`、`single-user`、`multi-user`、`multi-user-server` 和 `all`。

  消息选项包括：

  `quiet`
  : 打印标准的每个服务的输出以及需要管理员介入的错误消息。

  `verbose`
  : 打印标准的每个服务的输出以及提供更多信息的消息。

`-s`
: 仅引导至里程碑 `svc:/milestone/single-user:default`。此里程碑相当于 `init` 级别 `s`。

有关用法示例，请参见第 263 页中的 “如何引导区域”和第 263 页中的 “如何在单用户模式下引导区域”。

有关 Oracle Solaris 服务管理工具 (service management facility, SMF) 和 `init` 的信息，请参见《Oracle Solaris 管理：常见任务》中的第 6 章 “管理服务（概述）”、`svc.startd(1M)` 和 `init(1M)`。

区域 autoboot 设置

要在引导全局区域时自动引导某个区域，请在区域配置中将 `autoboot` 资源属性设置为 `true`。缺省设置为 `false`。

请注意，要自动引导区域，还必须启用区域服务 `svc:/system/zones:default`。缺省情况下将启用该服务。

有关在 `pkg update` 期间设置 `autoboot` 的信息，请参见第 303 页中的 “区域包管理概述”。

卸载区域

zoneadm uninstall 命令用于卸载区域根文件系统下的所有文件。除非还使用了 -F（强制）选项，否则该命令会提示您确认此操作以继续执行。使用 uninstall 命令时应谨慎，因为此操作是无法恢复的。

关于克隆非全局区域

通过克隆可以复制系统上现有的已配置和已安装区域，从而在同一系统上快速置备新区域。请注意，对于在不同的区域中不能相同的组件，必须至少要为其重置属性和资源。因此，zonepath 必须总是变化的。此外，对于共享 IP 区域，任何网络资源中的 IP 地址必须不同。对于专用 IP 区域，任何网络资源的物理属性必须不同。特定于应用程序的配置通常必须在克隆中进行重新配置。例如，如果区域中存在数据库实例，并且要克隆该区域，则您可能需要在克隆中重新配置数据库实例，以便将其自身识别为不同的实例。

- 克隆区域是安装区域的一种比较快速的方法。
- 新区域将包括因定制源区域而进行的所有更改，如添加的软件包或进行的文件修改。

当源 zonepath 和目标 zonepath 都驻留在 ZFS 上并且位于同一个池中，zoneadm clone 命令会自动使用 ZFS 来克隆区域。使用 ZFS 克隆时，在数据被修改前并不实际复制数据。因此，初始克隆只需极少的时间。zoneadm 命令会捕获源 zonepath 的 ZFS 快照，并设置目标 zonepath。目标区域的 zonepath 用于命名 ZFS 克隆。

注 – 可以指定复制 ZFS zonepath 而不是进行 ZFS 克隆（尽管可按这种方式来克隆源）。

有关更多信息，请参见第 268 页中的 “在同一系统中克隆非全局区域”。

◆ ◆ ◆ 第 19 章

19 安装、引导、关闭、停止、卸载和克隆非全局区域（任务）

本章介绍如何安装和引导非全局区域，并提供了使用克隆在同一系统上安装区域的方法。此外，还介绍了与安装相关的其他任务（例如停止、重新引导和卸载区域）。将现有的非全局区域移动到同一计算机上的新位置。还阐述了将现有非全局区域移动到同一计算机上的新位置以及从系统中完全删除区域的过程。

有关区域安装和相关操作的常规信息，请参见第 18 章，关于安装、关闭、停止、卸载和克隆非全局区域（概述）。

有关 `solaris10` 标记区域安装和克隆的信息，请参见第 33 章，安装 `solaris10` 标记区域。

区域安装（任务列表）

任务	说明	参考
（可选的）在安装区域之前检验已配置的区域。	确保区域满足安装要求。如果您跳过此过程，则会在安装区域时自动执行检验。	第 258 页中的“（可选的）如何在安装已配置的区域之前检验该区域”
安装已配置的区域。	安装处于已配置状态的区域。	第 259 页中的“如何安装已配置的区域”
获取区域的通用唯一标识符 (universally unique identifier, UUID)。	在安装区域时指定的这个单独的标识符是标识区域的另一种方法。	第 261 页中的“如何获取已安装的非全局区域的 UUID”
（可选的）将已安装的区域转换为就绪状态。	如果您要引导区域并立即使用，则可以跳过此过程。	第 262 页中的“（可选的）如何将已安装区域转换为就绪状态”

任务	说明	参考
引导区域。	引导区域时会将此区域置于运行状态。既可以从就绪状态引导区域，也可以从已安装状态引导区域。	第 263 页中的 “如何引导区域”
在单用户模式下引导区域。	仅引导至里程碑 `svc:/milestone/single-user:default`。此里程碑相当于 `init` 级别 `s`。请参见 `init`(1M) 和 `svc.startd`(1M) 手册页。	第 263 页中的 “如何在单用户模式下引导区域”

安装和引导区域

使用 `zoneadm`(1M) 手册页中介绍的 `zoneadm` 命令可以执行非全局区域的安装任务。您必须是全局管理员或者具备相应授权的用户，才能执行区域安装。本章中的示例使用在第 231 页中的 “配置、检验并提交区域”中建立的区域名称和区域路径。

▼ （可选的）如何在安装已配置的区域之前检验该区域

可以在安装区域之前对其进行检验。需执行的检查之一是检查是否有足够的磁盘容量。如果您跳过此过程，则会在安装区域时自动执行检验。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1 成为 root 用户或承担等效角色。

2 使用 `-z` 选项、区域名称和 `verify` 子命令检验名为 `my-zone` 的已配置区域。

```
global# zoneadm -z my-zone verify
```

将显示以下有关区域路径检验的消息：

```
WARNING: /zones/my-zone does not exist, so it could not be verified.
    When 'zoneadm install' is run, 'install' will try to create
    /zones/my-zone, and 'verify' will be tried again,
    but the 'verify' may fail if:
    the parent directory of /zones/my-zone is group- or other-writable
    or

    /zones/my-zone overlaps with any other installed zones
    or
    /zones/my-zone is not a mountpoint for a zfs file system.
```

但是，如果显示错误消息并且无法检验区域，请执行消息中指定的更正操作，并再次尝试执行此命令。

如果未显示错误消息，则可以安装区域。

更多信息

验证共享存储上的区域

对于在共享存储中配置的区域，`zonecfg verify` 可确认对于处于已配置状态的区域，配置的 zpool 资源在系统上都不处于联机状态。

对于在共享存储中配置的区域，`zoneadm verify` 可确认对于处于已安装状态的区域，配置为 `zpool` 和 `rootzpool` 资源的所有 zpool 在系统上都处于联机状态。如果资源不可用，`verify` 将失败，并将显示关于失败的 zpool 的信息。

▼ 如何安装已配置的区域

此过程用于安装已配置的非全局区域。有关安装选项的信息，请参见第 251 页中的 “如何安装区域”。

区域必须驻留在自己的 ZFS 数据集中。仅支持 ZFS。安装区域时，`zoneadm install` 命令会自动为 `zonepath` 创建 ZFS 文件系统（数据集）。如果无法创建 ZFS 数据集，区域也无法安装。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**

2. **使用带 `install` 子命令的 `zoneadm` 命令安装已配置的区域 `my-zone` ，同时为 `zonepath` ZFS 自动创建 ZFS 数据集。请注意，区域路径的父目录也必须是一个数据集，否则将无法创建文件系统。**

   - **安装区域：**

     ```
     global# zoneadm -z my-zone install
     ```

   - **从系统信息库安装区域：**

     ```
     global# zoneadm -z my-zone install -m manifest -c [ profile | dir ]
     ```

   - **从映像安装区域：**

     ```
     global# zoneadm -z my-zone install -a archive -s -u
     ```

   - **从目录安装区域：**

     ```
     global# zoneadm -z my-zone install -d path -p -v
     ```

   系统将显示："a ZFS file system has been created for this zone"（已经为该区域创建了一个 ZFS 文件系统）。

   当区域的根文件系统所需的文件和目录安装在区域的根路径下时，您将看到各种消息。

3 **（可选的）如果显示错误消息并且无法安装区域，请键入以下命令来获取区域状态：**

```
global# zoneadm list -v
# zoneadm list -cvd
  ID NAME             STATUS      PATH                         BRAND    IP
   0 global           running     /                            solaris  shared
   - my-zone          configured  /zones/my-zone               solaris  excl
```

- 如果显示为已配置状态，请执行消息中指定的更正操作，并再次尝试执行 `zoneadm install` 命令。
- 如果显示为未完成状态，请首先执行以下命令：

  ```
  global# zoneadm -z my-zone uninstall
  ```

  然后执行消息中指定的更正操作，并再次尝试执行 `zoneadm install` 命令。

4 **（可选）如果存储对象包含预先存在的分区、zpool 或 UFS 文件系统，则 `install` 会失败并显示错误消息。**

源区域必须处于卸载状态，才能使用 `force` 子命令：

```
zoneadm -z my-zone uninstall
```

然后通过使用 `zoneadm install` 的 `-x` 选项，继续安装及覆盖所有预先存在的数据。

```
-x force-zpool-import
-x force-zpool-create=zpoolname
-x force-zpool-create=zpoolname1,zpoolname2,zpoolname3
-x force-zpool-create-all
```

该选项和 `zpool create -f` 命令类似。

`-x force-zpool-create=`*zpoolname* 可以使用一次或多次。

5 **当安装完成时，使用带有 `-i` 和 `-v` 选项的 `list` 子命令来列出已安装的区域并检验状态。**

```
global# zoneadm list -iv
```

将显示以下类似信息：

```
ID  NAME     STATUS       PATH                         BRAND      IP
 0  global   running      /                            solaris    shared
 -  my-zone  installed    /zones/my-zone               solaris    excl
```

故障排除 如果区域安装中断或失败，则此区域会处于未完成状态。请使用 `uninstall -F` 将此区域重置为已配置状态。

接下来的步骤 缺省情况下，此区域是使用最小网络配置安装的，该配置在《Oracle Solaris 管理：常见任务》中的第 7 章 “管理服务（任务）”中进行了介绍。在登录到该区域后，可以切换到开放式网络配置，或者启用或禁用个别服务。有关详细信息，请参见第 288 页中的 “启用服务”。

▼ 如何获取已安装的非全局区域的 UUID

安装区域时，会为其指定一个通用唯一标识符 (universally unique identifier, UUID)。通过将 `zoneadm` 与 `list` 子命令和 `-c -p` 选项一起使用，可以获取 UUID。UUID 是显示的第五个字段。

- **查看已安装区域的 UUID。**

  ```
  global# zoneadm list -cp
  ```

  将显示以下类似信息：

  ```
  0:global:running:/::solaris:shared:-:none
  6:my-zone:running:/zones/my-zone:61901255-35cf-40d6-d501-f37dc84eb504:solaris:excl:-:
  ```

示例 19–1 如何在命令中使用区域 UUID

```
global# zoneadm -z my-zone -u 61901255-35cf-40d6-d501-f37dc84eb504:solaris:excl list -v
```

如果 `-u` *uuid-match* 和 `-z` *zonename* 都存在，则先根据 UUID 执行匹配。如果找到具有指定 UUID 的区域，则使用该区域并忽略 `-z` 参数。如果找不到具有指定 UUID 的区域，则系统将按区域名称进行搜索。

更多信息 关于 UUID

可以卸载区域，然后以相同的名称重新安装，但内容不同。也可以对区域进行重命名，而不更改内容。由于以上原因，UUID 比区域名称更可靠。

另请参见 有关更多信息，请参见 `zoneadm(1M)` 和 `libuuid(3LIB)`。

▼ 如何将已安装的非全局区域标记为未完成

如果对系统的管理性更改导致区域不可用或不一致，则可以将已安装区域的状态更改为未完成。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**
2. **将区域 `testzone` 标记为未完成。**

   ```
   global# zoneadm -z testzone mark incomplete
   ```

3. **使用带有 `-i` 选项和 `-v` 选项的 `list` 子命令检验状态。**

   ```
   global# zoneadm list -iv
   ```

将显示以下类似信息：

```
ID  NAME     STATUS       PATH                           BRAND      IP
 0  global   running      /                              solaris    shared
 -  my-zone  installed    /zones/my-zone                 solaris    excl
 -  testzone incomplete   /zones/testzone                solaris    excl
```

更多信息 **将区域标记为未完成**

-R *root* 选项可以与 zoneadm 的 mark 和 list 子命令结合使用以指定备用引导环境。有关更多信息，请参见 zoneadm(1M)。

注 – 将区域标记为未完成的操作是无法恢复的。可对标记为未完成的区域执行的唯一操作是卸载该区域，使其返回已配置状态。请参见第 267 页中的 “如何卸载区域”。

▼ （可选的）如何将已安装区域转换为就绪状态

转换为就绪状态可使虚拟平台做好开始运行用户进程的准备。处于就绪状态的区域中没有执行任何用户进程。

如果您要引导区域并立即使用，则可以跳过此过程。引导区域时便会自动从就绪状态进行转换。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**

2. **使用带有 -z 选项、区域名称 my-zone 以及 ready 子命令的 zoneadm 命令将区域转换为就绪状态。**

   ```
   global# zoneadm -z my-zone ready
   ```

3. **在提示符下，使用带有 -v 选项的 zoneadm list 命令来检验状态。**

   ```
   global# zoneadm list -v
   ```

   将显示以下类似信息：

   ```
   ID  NAME     STATUS       PATH                           BRAND      IP
    0  global   running      /                              solaris    shared
    1  my-zone  ready        /zones/my-zone                 solaris    excl
   ```

   请注意，系统已指定唯一的区域 ID 1。

▼ 如何引导区域

引导区域时会将此区域置于运行状态。既可以从就绪状态引导区域，也可以从已安装状态引导区域。处于已安装状态的区域经透明引导，会从就绪状态转换为正在运行状态。允许登录到处于正在运行状态下的区域。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**
2. **使用带有 -z 选项、区域名称 my-zone 以及 boot 子命令的 zoneadm 命令引导区域。**

```
global# zoneadm -z my-zone boot
```

3. **当引导完成时，使用带有 -v 选项的 list 子命令来检验状态。**

```
global# zoneadm list -v
```

将显示以下类似信息：

```
ID  NAME     STATUS       PATH                           BRAND      IP
 0  global   running      /                              solaris    shared
 1  my-zone  running      /zones/my-zone                 solaris    excl
```

示例 19–2 为区域指定引导参数

使用 -m verbose 选项引导区域：

```
global# zoneadm -z my-zone boot -- -m verbose
```

使用 -m verbose 引导选项重新引导区域：

```
global# zoneadm -z my-zone reboot -- -m verbose
```

区域管理员使用 -m verbose 选项重新引导区域 *my-zone*：

```
my-zone# reboot -- -m verbose
```

▼ 如何在单用户模式下引导区域

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**
2. **在单用户模式下引导区域。**

```
global# zoneadm -z my-zone boot -- -s
```

下一步执行的操作

有关如何登录到区域并执行初始内部配置，请参见第 20 章，非全局区域登录（概述）和第 21 章，登录到非全局区域（任务）。

关闭、停止、重新引导、卸载、克隆和删除非全局区域（任务列表）

任务	说明	参考
关闭区域。	**关闭**过程用于通过运行关闭脚本干净地关闭区域。此外，还支持 `zlogin` 方法。有关更多信息，请参见第 287 页中的“如何使用 `zlogin` 关闭区域”。	第 265 页中的“如何停止区域”
停止区域。	停止过程用于删除区域的应用程序环境和虚拟平台。此过程可将区域从就绪状态返回到已安装状态。有关如何干净地关闭区域，请参见第 287 页中的“如何使用 `zlogin` 关闭区域”。	第 265 页中的“如何停止区域”
重新引导区域。	重新引导过程会停止区域，然后再次引导它。	第 266 页中的“如何重新引导区域”
卸载区域。	此过程可删除区域根文件系统中的所有文件。**使用此过程时应谨慎。**此操作是无法恢复的。	第 267 页中的“如何卸载区域”
根据同一系统中某个现有区域的配置置备新的非全局区域。	克隆区域是安装区域的另外一种更快速的方法。在安装新区域之前，仍然需要先对其进行配置。	第 268 页中的“在同一系统中克隆非全局区域”
从系统中删除非全局区域。	此过程将从系统中完全删除区域。	第 270 页中的“从系统中删除非全局区域”

关闭、停止、重新引导和卸载区域

▼ 如何关闭区域

关闭过程可干净地关闭区域。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**

2. **列出系统上正在运行的区域。**

   ```
   global# zoneadm list -v
   ```

   将显示以下类似信息：

   ```
   ID  NAME     STATUS       PATH                           BRAND      IP
    0  global   running      /                              solaris    shared
    1  my-zone  running      /zones/my-zone                 solaris    excl
   ```

3. **使用带有 `-z` 选项、区域名称（例如 `my-zone`）以及 `shutdown` 子命令的 `zoneadm` 命令关闭给定区域。**

   ```
   global# zoneadm -z my-zone shutdown
   ```

4. **并指定 `-r` 选项以重新引导区域。**

   ```
   global# zoneadm -z my-zone shutdown -r boot_options
   ```

   请参见示例 19–2。

5. **列出系统上正在运行的区域，以确认该区域是否已关闭。**

   ```
   global# zoneadm list -v
   ```

▼ 如何停止区域

停止过程用于删除区域的应用程序环境和虚拟平台。有关如何干净地关闭区域，请参见第 287 页中的 “如何使用 zlogin 关闭区域”。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**

2. **列出系统上正在运行的区域。**

   ```
   global# zoneadm list -v
   ```

   将显示以下类似信息：

   ```
   ID  NAME     STATUS       PATH                           BRAND      IP
    0  global   running      /                              solaris    shared
    1  my-zone  running      /zones/my-zone                 solaris    excl
   ```

3. **使用带有 `-z` 选项、区域名称（例如 `my-zone`）以及 `halt` 子命令的 `zoneadm` 命令停止给定区域。**

   ```
   global# zoneadm -z my-zone halt
   ```

4. **再次列出系统上的区域来检验是否已停止 `my-zone`。**

   ```
   global# zoneadm list -iv
   ```

将显示以下类似信息：

```
ID  NAME     STATUS       PATH                           BRAND      IP
 0  global   running      /                              solaris    shared
 -  my-zone  installed    /zones/my-zone                 solaris    excl
```

5 **如果您要重新启动区域，请引导它。**

```
global# zoneadm -z my-zone boot
```

故障排除 如果区域没有正常停止，请参见第 364 页中的 “区域无法停止”以获得疑难解答提示。

▼ 如何重新引导区域

您必须是全局管理员或在全局区域具有相应授权的用户才能执行此过程。另请参见第 264 页中的 “如何关闭区域”。

1 **成为 root 用户或承担等效角色。**

2 **列出系统上正在运行的区域。**

```
global# zoneadm list -v
```

将显示以下类似信息：

```
ID  NAME     STATUS       PATH                           BRAND      IP
 0  global   running      /                              solaris    shared
 1  my-zone  running      /zones/my-zone                 solaris    excl
```

3 **使用带有 `-z reboot` 选项的 `zoneadm` 命令来重新引导区域 `my-zone`。**

```
global# zoneadm -z my-zone reboot
```

4 **再次列出系统上的区域来检验是否已重新引导 `my-zone`。**

```
global# zoneadm list -v
```

将显示以下类似信息：

```
ID  NAME     STATUS       PATH                           BRAND      IP
 0  global   running      /                              solaris    shared
 2  my-zone  running      /zones/my-zone                 solaris    excl
```

提示 – 请注意，`my-zone` 的区域 ID 已更改。区域 ID 通常会在重新引导后更改。

▼ 如何卸载区域

注意 – 使用此过程时需慎重。删除区域根文件系统中的所有文件的操作是无法恢复的。

区域不能处于正在运行状态。`uninstall` 操作对于正在运行的区域无效。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**

2. **列出系统上的区域。**

   ```
   global# zoneadm list -v
   ```

   将显示以下类似信息：

   ```
   ID  NAME     STATUS       PATH                           BRAND      IP
    0  global   running      /                              solaris    shared
    -  my-zone  installed    /zones/my-zone                 solaris    excl
   ```

3. **使用带有 `-z uninstall` 选项的 `zoneadm` 命令来删除区域 `my-zone`。**

   您还可以使用 `-F` 选项强制执行操作。如果未指定此选项，则系统将提示进行确认。

   ```
   global# zoneadm -z my-zone uninstall -F
   ```

   请注意，针对 `zonepath` 卸载具有自己的 ZFS 文件系统的区域时，将销毁该 ZFS 文件系统。

4. **再次列出系统上的区域来检验是否不再列出 `my-zone`。**

   ```
   global# zoneadm list -iv
   ```

   将显示以下类似信息：

   ```
   ID  NAME     STATUS       PATH                           BRAND      IP
    0  global   running      /                              solaris    shared
   ```

故障排除 如果区域卸载中断，则此区域停留在未完成状态。请使用 `zoneadm uninstall` 命令将此区域重置为已配置状态。

如果 `zonepath` 未被删除，则这可能表示该区域安装在其他引导环境中。当引导环境中有具有给定 `zonepath` 的已安装区域时，不会删除 `zonepath` 和 `zonepath` 数据集中的各种数据集。有关引导环境的更多信息，请参见 `beadm(1M)`。

使用 `uninstall` 命令时应谨慎，因为此操作是无法恢复的。

在同一系统中克隆非全局区域

克隆用于从源 zonepath 向目标 zonepath 复制数据来在系统上置备新区域。

当源 zonepath 和目标 zonepath 都驻留在 ZFS 上并且位于同一个池中，zoneadm clone 命令会自动使用 ZFS 来克隆区域。但您可以指定，复制 ZFS zonepath 但不进行 ZFS 克隆。

▼ 如何克隆区域

在安装新区域之前，必须先对其进行配置。传递给 zoneadm create 子命令的参数是要克隆的区域名称。必须停止此源区域。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1 **成为 root 用户或承担等效角色。**

2 **停止要克隆的源区域，在此过程中该区域为 my-zone。**

```
global# zoneadm -z my-zone halt
```

3 **通过将源区域 my-zone 的配置导出到文件（例如 master），开始配置新区域。**

```
global# zonecfg -z my-zone export -f /zones/master
```

注 – 也可以通过使用第 232 页中的 “如何配置区域”中的过程而不是通过修改现有配置来创建新区域配置。如果使用此方法，请在创建区域后，直接跳到步骤 6。

---

4 **编辑文件 master。对于在不同的区域中不能相同的组件，请为其设置不同的属性和资源。例如，您必须设置新的 zonepath。对于共享 IP 区域，必须更改任何网络资源中的 IP 地址。对于专用 IP 区域，必须更改任何网络资源的物理属性。**

5 **通过使用文件 *master* 中的命令创建新区域 zone1。**

```
global# zonecfg -z zone1 -f /zones/master
```

6 **通过克隆 my-zone 安装新区域 zone1。**

```
global# zoneadm -z zone1 clone my-zone
```

系统将显示：

```
Cloning zonepath /zones/my-zone...
```

7 （可选）如果存储对象包含预先存在的分区、zpool 或 UFS 文件系统，则 clone 会失败并显示错误消息。

要继续操作和覆盖所有预先存在的数据，请使用 zoneadm clone 的相应 -x 选项。必须先卸载源区域，然后才能使用 force 子命令。

-x force-zpool-import
-x force-zpool-create=*zpoolname*
-x force-zpool-create=*zpoolname1*,*zpoolname2*,*zpoolname3*
-x force-zpool-create-all

该选项和 zpool create -f 命令类似。

-x force-zpool-create=*zpoolname* 选项可以多次使用。

请注意，必须先停止源区域，然后才能使用 -x force 选项。

8 列出系统上的区域。

```
ID  NAME     STATUS       PATH                           BRAND      IP
 0  global   running      /                              solaris    shared
 -  my-zone  installed    /zones/my-zone                 solaris    excl
 -  zone1    installed    /zones/zone1                   solaris    excl
```

示例 19–3 将系统配置文件应用到克隆区域

要包括配置文件：

```
# zoneadm -z zone1 clone -c /path/config.xml my-zone
```

请注意，必须提供配置文件的绝对路径。

移动非全局区域

此过程用于通过更改 zonepath 将区域移动到同一系统上的新位置。必须停止该区域。需要满足第 210 页中的 “资源类型和属性”中所述的标准 zonepath 条件。

该信息还适用于移动 solaris10 标记区域。有关 solaris10 标记区域的信息，请参见第 3 部分。

注 – 您不能移动存在于其他 BE 中的区域。您可以先删除这些 BE，或者通过克隆区域在新路径中创建新区域。

注 – 不能将包含 rootzpool 资源的共享存储上的区域移至系统上的其他位置。支持重命名 zonepath。

▼ 如何移动不在共享存储中的区域

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为超级用户，或者具有等效授权。**
2. **停止要移动的区域，在此过程中为 db-zone。**

```
global# zoneadm -z db-zone halt
```

3. **使用带有 move 子命令的 zoneadm 命令将区域移动到新 zonepath，即 /zones/db-zone。**

```
global# zoneadm -z db-zone move /zones/db-zone
```

4. **检验路径。**

```
ID  NAME     STATUS       PATH                           BRAND      IP
 0  global   running      /                              solaris    shared
 -  my-zone  installed    /zones/my-zone                 solaris    excl
 -  db-zone  installed    /zones/db-zone                 solaris    excl
```

从系统中删除非全局区域

本节中所述的过程会从系统中完全删除区域。

▼ 如何删除非全局区域

1. **可使用以下方法之一来关闭区域 my-zone。首选使用 zoneadm shutdown 方法。**

   - **使用 zoneadm：**

     ```
     global# zoneadm -z my-zone shutdown
     my-zone
     ```

   - **使用 zlogin：**

     ```
     global# zlogin my-zone shutdown
     my-zone
     ```

2. **删除 my-zone 的根文件系统。**

```
global# zoneadm -z my-zone uninstall -F
```

通常，不需要使用 -F 选项来强制操作。

3. **删除 my-zone 的配置。**

```
global# zonecfg -z my-zone delete -F
```

通常，不需要使用 -F 选项来强制操作。

4 **列出系统上的区域来检验是否不再列出 my-zone。**

```
global# zoneadm list -iv
```

将显示以下类似信息：

```
ID  NAME     STATUS       PATH                           BRAND      IP
 0  global   running      /                              solaris    shared
```

◆ ◆ ◆ 第 20 章

20 非全局区域登录（概述）

本章介绍如何从全局区域登录到区域。

本章包含以下主题：

有关过程和用法的信息，请参见第 21 章，登录到非全局区域（任务）。有关可用选项的完整列表，请参见 zlogin(1) 手册页。

zlogin 命令

如果使用基于角色的访问控制 (role-based access control, RBAC)，则对区域控制台的访问需具备 solaris.zone.manage/*zonename* 授权。斜线字符 (/) 后面的特定 *zonename* 后缀是可选的。如果省略区域名称，用户将有权进入任何区域。

除非使用 -C 选项连接到区域控制台，否则使用 zlogin 登录到区域会启动新任务。一个任务不能跨两个区域。

使用 zlogin 命令可以从全局区域登录到任何处于正在运行状态或就绪状态的区域。

注 – 只能使用带有 -C 选项的 zlogin 命令登录到不处于运行状态的区域。

如第 286 页中的 “如何使用非交互模式访问区域”中所述，可以通过提供要在区域内部运行的命令，在非交互模式下使用 zlogin 命令。但是，该命令或它所作用的所有文件都不能驻留在 NFS 上。如果命令的任意打开的文件或其地址空间的任意部分驻留在 NFS 上，则此命令将失败。地址空间包括可执行的命令本身以及命令的链接库。

只有在全局区域中操作的全局管理员或具备相应授权的用户可以使用 zlogin 命令。有关更多信息，请参见 zlogin(1) 手册页。

内部区域配置

系统配置数据可以是单个配置文件 (sc_profile.xml)，也可以是 SMF 配置文件的目录 (profiles)。不论是单个文件还是目录，都描述了在区域安装期间将传递给自动安装程序的区域系统配置数据。如果在安装区域期间没有指定任何 sc_profile.xml 文件或 profiles 目录，则 sysconfig 交互式工具将在第一次使用控制台 zlogin 命令时要求管理员提供该数据。

该发行版使用 SMF 来集中配置信息。

在安装期间会创建并配置一个 Oracle Solaris 实例。Oracle Solaris 实例在全局区域或非全局区域中定义为引导环境。您可以使用 sysconfig 实用程序来对 Oracle Solaris 实例执行配置任务，或者取消对 Oracle Solaris 实例的配置，然后再重新进行配置。可以使用 sysconfig 命令来创建 SMF 配置文件。

在全局区域或非全局区域中安装或创建 Oracle Solaris 实例后（需要系统配置），将自动进行系统配置。对于 zoneadm clone 操作（如果指定了用于保留系统标识的 -p 选项）或 attach 操作（如果没有指定 -c*profile.xml*sysconfig 文件选项），不需要系统配置。

您可以执行以下操作：

- 使用 sysconfig configure 命令重新配置（先取消配置然后再配置）该 Oracle Solaris 实例。
  - 使用 sysconfig configure 命令配置该 Oracle Solaris 实例，并在控制台上启动 SCI 工具。

    ```
    # sysconfig configure
    ```
  - 在全局区域或非全局区域中使用 sysconfig configure 命令来配置已取消配置的 Solaris 实例。

    ```
    # sysconfig configure -c sc_profile.xml
    ```

    如果使用此命令指定现有配置文件，将执行非交互式配置。如果您没有使用此命令指定现有配置文件，交互式系统配置 (System Configuration Interactive, SCI) 工具将运行。使用 SCI 工具，您可以为该 Oracle Solaris 实例提供特定的配置信息。
- 您可以使用 sysconfig create-profile 命令来创建新的系统配置文件。

sysconfig 接口在《安装 Oracle Solaris 11.1 系统》中的第 6 章 “取消配置或重新配置 Oracle Solaris 实例”和 sysconfig(1M) 手册页中介绍。

交互式系统配置工具

通过交互式系统配置 (System Configuration Interactive, SCI) 工具，您可以为新安装的 Oracle Solaris 11.1 实例指定配置参数。

不带 `-c` *profile.xml* 选项的 `sysconfig configure` 将取消对系统的配置，然后打开 SCI 工具来向管理员进行查询并将配置写入 `/etc/svc/profile/site/scit_profile.xml`。之后，该工具将使用此信息来配置系统。

`sysconfig create-profile` 向管理员查询，并在 `/system/volatile/scit_profile.xml` 中创建 SMF 配置文件。参数包括系统主机名、时区、用户和 root 用户帐户、名称服务。

要在该工具中进行导航：

- 使用每个屏幕底部列出的功能键在屏幕间移动并执行其他操作。如果您的键盘没有功能键，或者按键不响应，请按 Esc 键。屏幕底部的图例将会更改，以显示用于导航和其他功能的 Esc 键。
- 使用向上和向下方向键更改选择或在输入字段之间移动。

有关更多信息，请参见《安装 Oracle Solaris 11.1 系统》中的第 6 章 “取消配置或重新配置 Oracle Solaris 实例”和 `sysconfig(1M)` 手册页。

区域配置文件示例

具有自动配置的专用 IP 区域：

```
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="sysconfig">
  <service version="1" type="service" name="system/config-user">
    <instance enabled="true" name="default">
      <property_group type="application" name="root_account">
        <propval type="astring" name="login" value="root"/>
        <propval type="astring" name="password" value="$5$KeNRy1zU$lqzy9rIsNloUhfvJFIWmVewE75aB5/EBA77kY7EP6F0"/>
        <propval type="astring" name="type" value="role"/>
      </property_group>
      <property_group type="application" name="user_account">
        <propval type="astring" name="login" value="admin1"/>
        <propval type="astring" name="password" value="$5$/g353K5q$V8Koe/XuAeR/zpBvpLsgVIqPrvc.9z0hYFYoyoBkE37"/>
        <propval type="astring" name="type" value="normal"/>
        <propval type="astring" name="description" value="admin1"/>
        <propval type="count" name="gid" value="10"/>
        <propval type="astring" name="shell" value="/usr/bin/bash"/>
        <propval type="astring" name="roles" value="root"/>
        <propval type="astring" name="profiles" value="System Administrator"/>
        <propval type="astring" name="sudoers" value="ALL=(ALL) ALL"/>
      </property_group>
    </instance>
  </service>
  <service version="1" type="service" name="system/timezone">
```

```
    <instance enabled="true" name="default">
      <property_group type="application" name="timezone">
        <propval type="astring" name="localtime" value="UTC"/>
      </property_group>
    </instance>
  </service>
  <service version="1" type="service" name="system/environment">
    <instance enabled="true" name="init">
      <property_group type="application" name="environment">
        <propval type="astring" name="LC_ALL" value="C"/>
      </property_group>
    </instance>
  </service>
  <service version="1" type="service" name="system/identity">
    <instance enabled="true" name="node">
      <property_group type="application" name="config">
        <propval type="astring" name="nodename" value="my-zone"/>
      </property_group>
    </instance>
  </service>
  <service version="1" type="service" name="system/keymap">
    <instance enabled="true" name="default">
      <property_group type="system" name="keymap">
        <propval type="astring" name="layout" value="US-English"/>
      </property_group>
    </instance>
  </service>
  <service version="1" type="service" name="system/console-login">
    <instance enabled="true" name="default">
      <property_group type="application" name="ttymon">
        <propval type="astring" name="terminal_type" value="vt100"/>
      </property_group>
    </instance>
  </service>
  <service version="1" type="service" name="network/physical">
    <instance enabled="true" name="default">
      <property_group type="application" name="netcfg">
        <propval type="astring" name="active_ncp" value="Automatic"/>
      </property_group>
    </instance>
  </service>
</service_bundle>
```

具有使用 NIS 没有 DNS 的静态配置的专用 IP 区域：

```
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="sysconfig">
  <service version="1" type="service" name="system/config-user">
    <instance enabled="true" name="default">
      <property_group type="application" name="root_account">
        <propval type="astring" name="login" value="root"/>
        <propval type="astring" name="password" value="$5$m8OR3zqK$Ox5XGubRJdi4zj0JzNSmVJ3Ni4opDOGpxi2nK/GGzmC"/>
        <propval type="astring" name="type" value="normal"/>
      </property_group>
    </instance>
  </service>
  <service version="1" type="service" name="system/timezone">
    <instance enabled="true" name="default">
      <property_group type="application" name="timezone">
```

```
      <propval type="astring" name="localtime" value="UTC"/>
    </property_group>
  </instance>
</service>
<service version="1" type="service" name="system/environment">
  <instance enabled="true" name="init">
    <property_group type="application" name="environment">
      <propval type="astring" name="LC_ALL" value="C"/>
    </property_group>
  </instance>
</service>
<service version="1" type="service" name="system/identity">
  <instance enabled="true" name="node">
    <property_group type="application" name="config">
      <propval type="astring" name="nodename" value="my-zone"/>
    </property_group>
  </instance>
</service>
<service version="1" type="service" name="system/keymap">
  <instance enabled="true" name="default">
    <property_group type="system" name="keymap">
      <propval type="astring" name="layout" value="US-English"/>
    </property_group>
  </instance>
</service>
<service version="1" type="service" name="system/console-login">
  <instance enabled="true" name="default">
    <property_group type="application" name="ttymon">
      <propval type="astring" name="terminal_type" value="vt100"/>
    </property_group>
  </instance>
</service>
<service version="1" type="service" name="network/physical">
  <instance enabled="true" name="default">
    <property_group type="application" name="netcfg">
      <propval type="astring" name="active_ncp" value="DefaultFixed"/>
    </property_group>
  </instance>
</service>
<service version="1" type="service" name="network/install">
  <instance enabled="true" name="default">
    <property_group type="application" name="install_ipv4_interface">
      <propval type="astring" name="address_type" value="static"/>
      <propval type="net_address_v4" name="static_address" value="10.10.10.13/24"/>
      <propval type="astring" name="name" value="net0/v4"/>
      <propval type="net_address_v4" name="default_route" value="10.10.10.1"/>
    </property_group>
    <property_group type="application" name="install_ipv6_interface">
      <propval type="astring" name="stateful" value="yes"/>
      <propval type="astring" name="stateless" value="yes"/>
      <propval type="astring" name="address_type" value="addrconf"/>
      <propval type="astring" name="name" value="net0/v6"/>
    </property_group>
  </instance>
</service>
<service version="1" type="service" name="system/name-service/switch">
  <property_group type="application" name="config">
    <propval type="astring" name="default" value="files nis"/>
    <propval type="astring" name="printer" value="user files nis"/>
```

```
      <propval type="astring" name="netgroup" value="nis"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
  <service version="1" type="service" name="system/name-service/cache">
    <instance enabled="true" name="default"/>
  </service>
  <service version="1" type="service" name="network/dns/client">
    <instance enabled="false" name="default"/>
  </service>
  <service version="1" type="service" name="network/nis/domain">
    <property_group type="application" name="config">
      <propval type="hostname" name="domainname" value="example.net"/>
      <property type="host" name="ypservers">
        <host_list>
          <value_node value="192.168.224.11"/>
        </host_list>
      </property>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
  <service version="1" type="service" name="network/nis/client">
    <instance enabled="true" name="default"/>
  </service>
</service_bundle>
```

具有带 NIS 的动态配置的专用 IP 区域

```
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="sysconfig">
 <service version="1" type="service" name="system/config-user">
   <instance enabled="true" name="default">
     <property_group type="application" name="root_account">
       <propval type="astring" name="login" value="root"/>
       <propval type="astring" name="password" value="$5$Iq/.A.K9$RQyt6RqsAY8TgnuxL9i0/84QwgIQ/nqcK8QsTQdvMy/"/>
       <propval type="astring" name="type" value="normal"/>
     </property_group>
   </instance>
 </service>
 <service version="1" type="service" name="system/timezone">
   <instance enabled="true" name="default">
     <property_group type="application" name="timezone">
       <propval type="astring" name="localtime" value="UTC"/>
     </property_group>
   </instance>
 </service>
 <service version="1" type="service" name="system/environment">
   <instance enabled="true" name="init">
     <property_group type="application" name="environment">
       <propval type="astring" name="LC_ALL" value="C"/>
     </property_group>
   </instance>
 </service>
 <service version="1" type="service" name="system/identity">
   <instance enabled="true" name="node">
     <property_group type="application" name="config">
       <propval type="astring" name="nodename" value="my-zone"/>
     </property_group>
   </instance>
```

```
</service>
<service version="1" type="service" name="system/keymap">
  <instance enabled="true" name="default">
    <property_group type="system" name="keymap">
      <propval type="astring" name="layout" value="US-English"/>
    </property_group>
  </instance>
</service>
<service version="1" type="service" name="system/console-login">
  <instance enabled="true" name="default">
    <property_group type="application" name="ttymon">
      <propval type="astring" name="terminal_type" value="sun-color"/>
    </property_group>
  </instance>
</service>
<service version="1" type="service" name="system/name-service/switch">
  <property_group type="application" name="config">
    <propval type="astring" name="default" value="files nis"/>
    <propval type="astring" name="printer" value="user files nis"/>
    <propval type="astring" name="netgroup" value="nis"/>
  </property_group>
  <instance enabled="true" name="default"/>
</service>
<service version="1" type="service" name="system/name-service/cache">
  <instance enabled="true" name="default"/>
</service>
<service version="1" type="service" name="network/dns/client">
  <instance enabled="false" name="default"/>
</service>
<service version="1" type="service" name="network/nis/domain">
  <property_group type="application" name="config">
    <propval type="hostname" name="domainname" value="special.example.com"/>
    <property type="host" name="ypservers">
      <host_list>
        <value_node value="192.168.112.3"/>
      </host_list>
    </property>
  </property_group>
  <instance enabled="true" name="default"/>
</service>
<service version="1" type="service" name="network/nis/client">
  <instance enabled="true" name="default"/>
</service>
</service_bundle>
```

非全局区域登录方法

本节介绍登录区域的方法。

区域控制台登录

每个区域都维护一个虚拟控制台 `/dev/console`。在控制台上执行操作称为控制台模式。当某个区域处于已安装状态时，可以通过控制台登录到该区域。区域控制台非常

类似于系统上的串行控制台。即使重新引导区域，控制台的连接也仍然保持。有关如何区分控制台模式与登录会话（例如 `telnet`），请参见第 281 页中的 “远程登录”。

可以使用带有 `C` 选项和 `-zonename` 的 *zlogin* 命令来访问区域控制台。区域不必处于运行状态。

也可以使用 `-d` 选项。该选项指定，如果区域停止，则区域断开与控制台的连接。只能使用 `-C` 选项指定该选项。

区域内的进程可以打开并将消息写入控制台。如果 `zlogin -C` 进程退出，则其他进程便可访问控制台。

如果使用基于角色的访问控制 (role-based access control, RBAC)，则对区域控制台的访问需具备 `solaris.zone.manage/`*zonename* 授权。斜线字符 (`/`) 后面的特定 *zonename* 后缀是可选的。如果省略区域名称，用户将有权进入任何区域。

要在引导时打开交互式系统配置 (System Configuration Interactive, SCI) 工具，请键入以下内容：

```
root@test2:~# sysconfig configure -s
```

用户登录方法

要使用用户名登录到区域，请使用带有 `l` 选项、用户名以及 `-zonename` 的 *zlogin* 命令。例如，全局区域管理员可以通过为 `zlogin` 指定 `-l` 选项，以普通用户身份在非全局区域中登录：

```
global# zlogin -l user zonename
```

要以用户 `root` 身份登录，请使用不带选项的 `zlogin` 命令。

故障安全模式

如果出现登录问题，并且您无法使用 `zlogin` 命令或带有 `-C` 选项的 `zlogin` 命令访问区域，则可以选择另外一种方法。您可以使用带有 `-S`（安全）选项的 `zlogin` 命令来进入区域。仅当其他登录方式不成功时，才使用此模式来恢复损坏的区域。在这个最小环境中，可以诊断区域登录失败的原因。

远程登录

远程登录区域的能力取决于您确定的网络服务选择。如果需要，可以通过启用 `pkg:/service/network/legacy-remote-utilities` 服务添加通过 `rlogin` 和 `telnet` 登录。

有关登录命令的更多信息，请参见 `rlogin(1)`、`ssh(1)` 和 `telnet(1)`。

交互模式与非交互模式

`zlogin` 命令还提供了其他两种方法来访问区域以及在区域内部执行命令。这两种方法为交互模式和非交互模式。

交互模式

在交互模式下，会分配新的伪终端，以供在区域内使用。与允许独占访问控制台设备的控制台模式不同，在交互模式下，可以随时打开任意数量的 `zlogin` 会话。未提供要执行的命令时，便会激活交互模式。需要终端设备的程序（例如编辑器）在此模式下可正常运行。

如果使用 RBAC，则对于交互式登录，需具备区域的 `solaris.zone.login/`*zonename* 授权。将在区域中进行口令验证。

非交互模式

非交互模式用于运行可管理区域的 shell 脚本。非交互模式不会分配新的伪终端。当您提供了要在区域内部运行的命令时，便会启用非交互模式。

对于非交互式登录或要跳过口令验证，则需具备 `solaris.zone.manage/`*zonename* 授权。

◆ ◆ ◆ 第 21 章

21 登录到非全局区域（任务）

本章提供用于完成已安装区域的配置、从全局区域登录到某个区域以及关闭区域的过程。同时还介绍如何使用 zonename 命令来列显当前区域的名称。

有关区域登录过程的介绍，请参见第 20 章，非全局区域登录（概述）。

初始区域引导与区域登录过程（任务列表）

任务	说明	参考
执行内部配置或取消对区域的配置。	可以通过使用文本用户界面来执行交互式系统配置，也可以通过使用配置文件来执行非交互式系统配置。此外，还可以使用 sysconfig 实用程序来取消对 Solaris 实例的配置。	请参见《安装 Oracle Solaris 11.1 系统》中的第 6 章 “取消配置或重新配置 Oracle Solaris 实例”和 sysconfig(1M) 手册页。
登录到区域。	您可以使用交互模式分配伪终端或提供要在区域中运行的命令，通过控制台登录到区域。提供要运行的命令不会分配伪终端。当指向区域的连接被拒绝时，您还可以使用故障安全模式进行登录。	第 284 页中的 “登录到区域”
退出非全局区域。	从非全局区域断开。	第 286 页中的 “如何退出非全局区域”
关闭区域。	使用 shutdown 实用程序或脚本来关闭区域。	第 287 页中的 “如何使用 zlogin 关闭区域”
列显区域名称。	列显当前区域的区域名称。	第 288 页中的 “列显当前区域的名称”

登录到区域

使用 zlogin 命令，可以从全局区域登录到任何处于正在运行状态或就绪状态的区域。有关更多信息，请参见 zlogin(1) 手册页。

如以下过程中所述，您可以通过多种方法登录到区域。您还可以远程登录，如第 281 页中的 “远程登录”中所述。

▼ 如何创建配置文件

注意 – 请注意，必须提供所有必需的数据。如果您为配置文件提供缺失的数据，区域也将用缺失的数据进行配置。这种配置可能会阻止用户登录或运行网络。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**
2. **使用 sysconfig 工具创建配置文件。**
   - **对于专用 IP 区域**

     ```
     # sysconfig create-profile -o /path/sysconf.xml
     ```

   - **对于共享 IP 区域：**

```
# sysconfig create-profile -o /path/sysconf.xml -g location,identity,naming_services,users
```

3. **在区域安装、克隆或附加操作期间使用创建的配置文件。**

   ```
   # zoneadm -z my-zone install -c /path/sysconf.xml
   ```

   如果使用此配置文件，则在初始 zlogin 时，系统不会在控制台上启动交互式系统配置 (System Configuration Interactive, SCI) 工具。指定文件参数时必须使用绝对路径。

▼ 如何登录到区域控制台以执行内部区域配置

如果 config.xml 文件被传递给了 zoneadm clone、attach 或 install 命令，则使用此配置文件来配置系统。如果在 clone、attach 或 install 操作期间未提供 config.xml 文件，则在第一次引导区域时，将在控制台上启动 SCI 工具。

为避免错过配置信息初始提示，建议使用两个终端窗口，以使 zlogin 在第二个会话引导区域之前运行。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**

2. **使用带有 -C 选项和区域名称（例如 my-zone）的 zlogin 命令。**

   ```
   global# zlogin -C my-zone
   ```

3. **从其他终端窗口中引导区域。**

   ```
   global# zoneadm -z my-zone boot
   ```

   将在 zlogin 终端窗口中显示以下类似信息：

   ```
   [NOTICE: Zone booting up]
   ```

4. **响应一系列关于您新安装的区域的配置参数的问题。参数包括系统主机名、时区、用户和 root 用户帐户以及名称服务。缺省情况下，SCI 工具会在 /system/volatile/scit_profile.xml 中生成 SMF 配置文件。**

故障排除 如果没有出现初始 SCI 屏幕，您可以键入 Ctrl L 以刷新 SCI 屏幕。

▼ 如何登录到区域控制台

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**

2. **使用带有 -C 选项、-d 选项和区域名称（例如 my-zone）的 zlogin 命令。**

   ```
   global# zlogin -C -d my-zone
   ```

   如果尚未执行配置，使用带有 -C 选项的 zlogin 命令将启动 SCI 工具。

3. **当显示区域控制台时，以 root 身份登录，按回车键，并在提示时键入 root 用户口令。**

   ```
   my-zone console login: root
   Password:
   ```

▼ 如何使用交互模式访问区域

在交互模式下，会分配新的伪终端以在区域内部使用。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**

2. **从全局区域登录到某个区域，例如 my-zone。**

   ```
   global# zlogin my-zone
   ```

将显示以下类似信息：

```
[Connected to zone 'my-zone' pts/2]
Last login: Wed Jul  3 16:25:00 on console
```

3 **键入 exit 关闭连接。**

将显示以下类似消息：

```
[Connection to zone 'my-zone' pts/2 closed]
```

▼ 如何使用非交互模式访问区域

当用户提供要在区域内部运行的命令时，便会启用非交互模式。非交互模式不会分配新的伪终端。

请注意，命令或运行命令的所有文件都不能驻留在 NFS 上。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1 **成为 root 用户或承担等效角色。**

2 **从全局区域登录到 my-zone 区域并提供命令名称。**

在此使用命令 zonename。

```
global# zlogin my-zone zonename
```

将显示以下输出：

```
my-zone
```

▼ 如何退出非全局区域

- **要从非全局区域断开连接，请使用下列方法之一。**
  - **要退出区域非虚拟控制台：**

    ```
    zonename# exit
    ```

  - **要从区域虚拟控制台断开连接，请使用波浪号 (~) 字符和句点：**

    ```
    zonename# ~.
    ```

    将显示以下类似信息：

    ```
    [Connection to zone 'my-zone' pts/6 closed]
    ```

注 – ssh 的缺省转义序列也是 ~，这会导致 ssh 会话退出。如果使用 ssh 远程登录到服务器，则使用 ~~. 退出区域。

另请参见 有关 zlogin 命令选项的更多信息，请参见 zlogin(1) 手册页。

▼ 如何使用故障安全模式进入区域

当指向区域的连接被拒绝时，可以使用带有 -S 选项的 zlogin 命令进入区域的最小环境。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**
2. **在全局区域中，使用带有 -S 选项的 zlogin 命令来访问区域（例如 my-zone）。**

```
global# zlogin -S my-zone
```

▼ 如何使用 zlogin 关闭区域

注 – 如果在全局区域中运行 init 0 干净地关闭 Oracle Solaris 系统，也会在系统上的每个非全局区域中运行 init 0。请注意，init 0 在系统关闭之前不会警告本地和远程用户注销。

使用此过程可以干净地关闭区域。有关如何在不运行关闭脚本的情况下停止区域，请参见第 265 页中的 “如何停止区域”。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**
2. **登录到要关闭的区域，例如 my-zone，并将 shutdown 指定为实用程序的名称，将 init 0 指定为状态。**

```
global# zlogin my-zone shutdown -i 0
```

您的站点可能具有自己的适用于特定环境的关闭脚本。

启用服务

您可以在区域中启用或禁用各项服务。

列显当前区域的名称

zonename(1) 手册页中所述的 zonename 命令可列显当前区域的名称。以下示例显示了在全局区域中使用 zonename 时的输出。

```
# zonename
global
```

◆ ◆ ◆ 第 22 章

22 关于区域迁移和 zonep2vchk 工具

本章概述了以下内容：

- 物理到虚拟迁移，用于将系统迁移到非全局区域
- 虚拟到虚拟迁移，用于将现有区域迁移到新系统

本章还论述系统到区域迁移所使用的 zonep2vchk 工具。

物理转换为虚拟和虚拟转换为虚拟概念

P2V 和 V2V 可用于以下操作：

- 将多个应用程序整合到单台服务器上
- 使工作负荷重新平衡
- 更换服务器
- 灾难恢复

选择迁移策略

可以重新配置基于 SAN 的存储，使 zonepath 在新主机上可见。

如果必须将一个系统上的所有区域全部移动到另外一个系统，复制流具有保留快照和克隆的优点。pkg、beadm create 和 zoneadm clone 命令会大量使用快照和克隆。

执行 P2V 或 V2V 迁移分为 5 个步骤。

1. 对于 P2V，分析源主机的 Oracle Solaris 配置：
   - 根据联网要求确定非全局区域的 IP 类型（专用 IP 还是共享 IP）。
   - 确定目标主机的全局区域中是否还需要任何其他配置。
   - 决定将如何迁移应用程序数据和文件系统。

   -b 选项所执行的 zonep2vchk 基本分析可确定与源全局区域所使用的 Oracle Solaris 配置或功能相关的基本问题。使用 -s 选项的 zonep2vchk 静态分析可帮助确定与源全

局区域上的特定应用程序相关的问题。-r 执行的 zonep2vchk 运行时分析可检测当前正在执行的应用程序中是否存在可能无法在区域中运行的操作。

2. 归档源系统或区域。该 Oracle Solaris 实例归档潜在排除要单独迁移的数据。
   - 要归档 Oracle Solaris 10 全局区域，可使用 flarcreate。

     请参见第 377 页中的 “如何使用 flarcreate 创建映像”。
   - 要归档 Oracle Solaris 10 系统和非全局区域，可使用 flarcreate 和 -R 或 -L *archiver*，以便将某些文件从归档文件中排除。确保先停止区域。

     请参见第 377 页中的 “如何使用 flarcreate 排除特定数据”。
   - 对于 Oracle Solaris 11 全局区域，可使用 zfs send 归档根池。
   - 对于 Oracle Solaris 11 非全局区域，可使用 zfs send 归档区域的 zonepath 数据集。
   - 对于驻留在共享存储（如 SAN）的 zpool 中的 solaris10 或 solaris 区域，V2V 迁移策略不要求创建归档文件。可以重新配置基于 SAN 的存储，以便在新主机上显示 zonepath。要进行重新配置，请按以下指示操作：
     - 导出 zpool，然后在目标全局区域上导入。
     - 在目标系统上使用 zoneadm install（首选）或 attach。（请参见本节中的步骤 5。）

   另请参见共享存储上的区域。
3. 为其他数据和文件系统选择迁移策略，如：
   - 在归档文件中包括数据（请参见本节中的步骤 2）。
   - 使用喜欢的归档文件格式（如 zfs send）单独归档数据，迁移后，在区域中恢复这些数据。
   - 迁移 SAN 数据，方法是从目标全局区域访问 SAN 存储，然后使用 zonecfg add fs 使得数据可供区域使用。
   - 可迁移 ZFS zpools 中的存储，方法是导出源主机上的 zpool，移动存储，然后在目标全局区域上导入 zpool。然后可使用 zonecfg add dataset 或 zonecfg add fs 将这些 ZFS 文件系统添加到目标区域。请注意，按照这种方法还可以迁移 SAN 存储设备上的 zpools。
4. 为目标主机上的目标区域创建区域配置 (zonecfg)。
   - 对于 P2V，使用带 -c 选项的 zonep2vchk 命令来协助创建配置。
   - 对于 V2V，在源主机上使用 zonecfg -z source_zone export 命令。将 Oracle Solaris 10 Containers 迁移到 Oracle Solaris 10 Zones 时，确保将标记设置为 solaris10。

   根据需要检查并修改导出的 zonecfg，例如更新联网资源。
5. 在目标主机上使用归档文件安装或附加区域。首次引导时，可能会提供新的 sysconfig 配置文件，或者运行 sysconfig 实用程序。

使用 zonep2vchk 工具准备系统迁移

本节介绍 zonep2vchk 工具。该工具的主要文档为 zonep2vchk(1M) 手册页。

关于 zonep2vchk 工具

P2V 进程包括归档全局区域（源），然后使用该归档文件安装非全局区域（目标）。zonep2vchk 实用程序必须以有效的用户 ID 0 运行。

该实用程序执行以下任务：

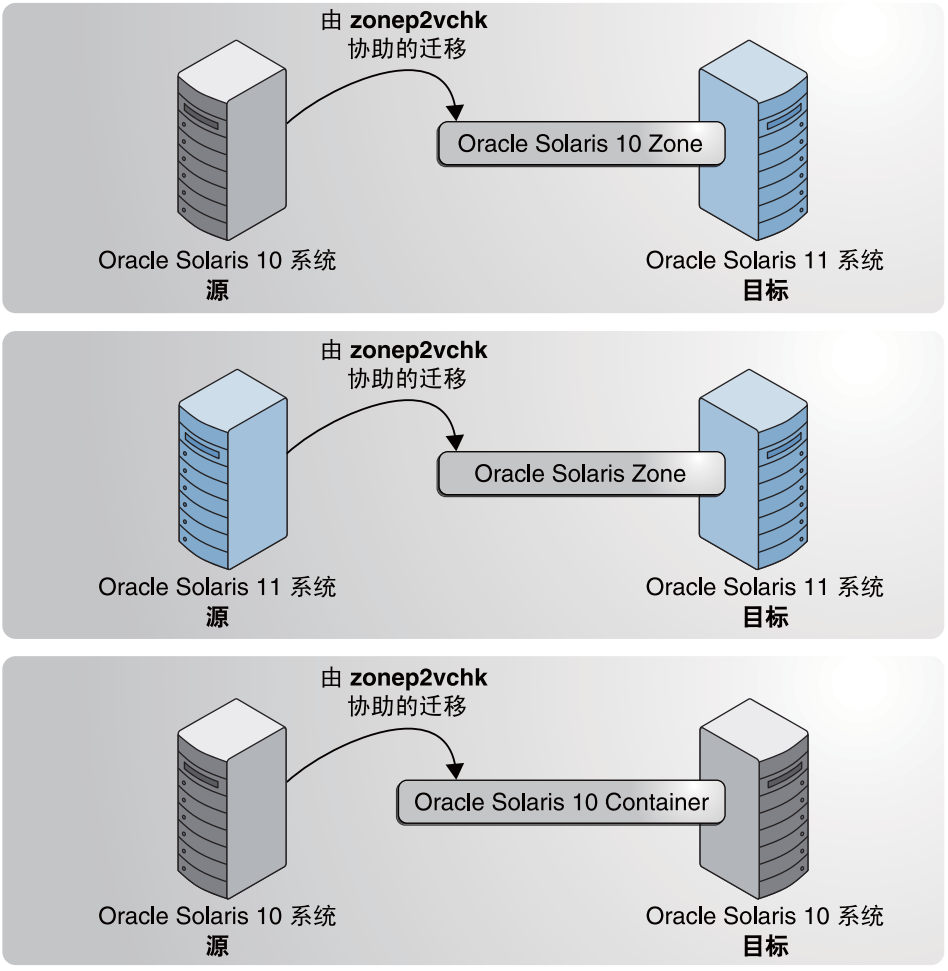
- 确定源系统配置中的问题区域
- 使必需的手动重新配置任务量降至最低
- 支持将 Oracle Solaris 10 和 Oracle Solaris 11 系统映像迁移到 Oracle Solaris 11 发行版的区域中
- 支持原始系统映像中的复杂网络配置，包括多个 IP 接口、IP 多路径和 VLAN

此工具可用于将 Oracle Solaris 11 物理系统或 Oracle Solaris 10 物理系统迁移到此发行版的非全局区域：

- 将 Oracle Solaris 11 系统迁移到 solaris 标记区域
- 将 Oracle Solaris 10 系统迁移到 solaris10 标记区域

对于 Oracle Solaris 11 目标系统，针对源系统上的每个网络资源，在 zonecfg 输出中都包括一个 add anet 资源 (VNIC)。缺省情况下，在将 Oracle Solaris 11 系统或 Oracle Solaris 10 系统迁移到 Oracle Solaris 11 系统上的非全局区域时，网络类型为专用 IP。

图 22–1 zonep2vchk 实用程序



分析类型

基本分析（-b 选项）检查是否使用了可能受 P2V 迁移影响的 Oracle Solaris 功能。

静态分析（-s 选项）检测二进制文件中是否存在可能无法在区域中运行的系统和库调用。

运行时分析（-r 选项）检测当前正在执行的应用程序中是否存在可能无法在区域中运行的操作。

生成的信息

分析将提供两大类信息：

- 能够使用特定的区域配置或全局区域中的配置更改来解决的问题
- 识别在区域内部无法运行的功能

例如，如果某个应用程序设置系统时钟，则可通过向区域中添加相应的特权启用此设置，但如果某个应用程序访问内核内存，则在区域内部永远也不允许这样操作。输出将区分这两类问题。

缺省情况下，实用程序以用户可阅读的形式打印消息。要以计算机可解析形式打印消息，请使用 `-P` 选项。有关可用选项以及命令调用和输出的完整信息，请参见 zonep2vchk(1M) 手册页。

◆ ◆ ◆ 第 23 章

23 迁移 Oracle Solaris 系统和迁移非全局区域（任务）

本章介绍如何将 Oracle Solaris 11 系统迁移到目标 Oracle Solaris 11 计算机上的非全局区域。此外，本章还介绍了如何在迁移源系统前将源系统上的任何现有 `solaris` 区域迁移到新的目标系统。

这些信息还适用于迁移 `solaris10` 标记区域。有关 `solaris10` 标记区域的信息，请参见第 3 部分。

将非全局区域迁移到其他计算机

关于迁移区域

`zonecfg` 和 `zoneadm` 命令可用于将现有非全局区域从一个系统迁移到另一个系统。需要停止区域并使其与当前主机分离。`zonepath` 将移动到它所附加的目标主机。

区域迁移需要满足以下要求：

- 迁移前，必须删除源系统上的所有非活动 BE。
- 目标系统上的全局区域所运行的 Oracle Solaris 11 发行版必须等于或高于原始源主机。
- 为确保区域可以正常运行，目标系统上安装的必需的操作系统软件包必须与原始源主机上安装的软件包相同或版本更高。

  其他软件包（如用于第三方产品的软件包）可以有所不同。
- 如果新主机具有依赖于区域的软件包的更高版本，将 `zoneadm attach` 与 `-u` 或 `-U` 选项结合使用，更新区域内的这些软件包，使其与新主机匹配。对附加软件的更新可查看将要迁移的区域，并确定必须更新哪些软件包以匹配新主机。仅更新那些软件包。其余的软件包可以因区域而有所不同。任何在该区域内安装但未在全局区域中安装的软件包均将被忽略并保留原样。

- 如果存储对象包含预先存在的分区、zpool 或 UFS 文件系统，则 attach 会失败并显示错误消息。要继续 attach 操作和覆盖所有预先存在的数据，可以使用 zoneadm attach 的 -x 选项。

  ```
  -x force-zpool-import
  -x force-zpool-create=zpoolname
  -x force-zpool-create=zpoolname1,zpoolname2,zpoolname3
  -x force-zpool-create-all
  ```

  该选项和 zpool create -f 命令类似。

  -x force-zpool-create=*zpoolname* 选项可以使用一次或多次。

zoneadm detach 进程用于创建在其他系统上附加区域所需的信息。zoneadm attach 进程用于检验目标计算机是否具有托管区域所需的正确配置。

由于可以通过多种方式来使 zonepath 在新主机上可用，因此 zonepath 从一个系统到另一个系统的实际移动是由全局管理员执行的手动进程。

在附加到新系统时，区域处于已安装状态。

▼ 如何使用 ZFS 归档文件迁移非全局区域

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

本例介绍如何创建区域的归档文件，然后将该归档文件附加到另外一个系统。假定源和目标主机上的管理员能够访问共享 NFS 服务器进行临时文件存储。如果共享的临时空间不可用，可使用其他方式（如 scp 安全复制，一个远程文件复制程序）在源和目标计算机之间复制文件。如果需要提供口令或口令短语进行验证，scp 程序会请您提供口令或口令短语。

1 成为 root 用户或承担等效角色。

2 关闭要迁移的区域，在此过程中为 my-zone。

```
host1# zoneadm -z my-zone shutdown
```

3 （可选的）分离该区域。

```
host1# zoneadm -z my-zone detach
```

分离的区域现在处于已配置状态。下次引导全局区域时，该区域将不会自动引导。

4 导出区域配置。

```
host1# mkdir /net/server/zonearchives/my-zone
host1# zonecfg -z my-zone export > /net/server/zonearchives/my-zone/my-zone.zonecfg
```

5 创建 gzip ZFS 归档文件。

```
host1# zfs list -H -o name /zones/my-zone
rpool/zones/my-zone
host1# zfs snapshot -r rpool/zones/my-zone@v2v
host1# zfs send -rc rpool/zones/my-zone@v2v | gzip > /net/server/zonearchives/my-zone/my-zone.zfs.gz
```

您可以选择使用压缩功能，这样通常会更快一些，因为在写入及后续读取归档文件期间，占用的 I/O 较少。有关更多信息，请参见《Oracle Solaris 11.1 管理：ZFS 文件系统》。

6 在新主机上，对该区域进行配置。

```
host2# zonecfg -z my-zone -f /net/server/zonearchives/my-zone/my-zone.zonecfg
```

将显示以下系统消息：

```
my-zone: No such zone configured
Use 'create' to begin configuring a new zone.
```

7 （可选的）查看配置。

```
host2# zonecfg:my-zone> info
zonename: my-zone
zonepath: /zones/my-zone
autoboot: false
pool:
net:
         address: 192.168.0.90
         physical: bge0
```

8 对配置进行所需的任何调整。

例如，新主机上的网络物理设备有所不同，或者属于配置组成部分的设备在新主机上可能具有不同的名称。

```
host2# zonecfg -z my-zone
zonecfg:my-zone> select net physical=bge0
zonecfg:my-zone:net> set physical=e1000g0
zonecfg:my-zone:net> end
```

9 提交配置并退出。

```
zonecfg:my-zone> commit
zonecfg:my-zone> exit
```

10 使用下列方法之一在新主机上安装区域。建议使用 install 子命令。

- 安装区域，执行保证 install 成功所必需的最小更新：

```
host2# zoneadm -z my-zone install -p -a /net/server/zonearchives/my-zone/my-zone.zfs.gz
```

在此发行版中，您也可以对区域执行 attach，执行保证附加成功所必需的最小更新。如果允许更新，则会在执行 zoneadm attach 期间刷新发布者的目录。

```
host2# zoneadm -z my-zone attach -u -a /net/server/zonearchives/my-zone/my-zone.zfs.gz
```

- **安装区域，将区域中的所有软件包更新到与全局区域相兼容的最新版本。**

```
host2# zoneadm -z my-zone install -U -p -a /net/server/zonearchives/my-zone/my-zone.zfs.gz
```

在此发行版中，您也可以对区域执行 attach，将区域中的所有软件更新到与全局区域相兼容的最新版本。

```
host2# zoneadm -z my-zone install -U -a /net/server/zonearchives/my-zone/my-zone.zfs.gz
```

- **向新主机附加区域，但不更新任何软件。**

```
host2# zoneadm -z my-zone attach -a /net/server/zonearchives/my-zone/my-zone.zfs.gz
```

注 – 在未来的 Oracle Solaris 发行版中，attach 子命令的 -a 和 -d 选项可能被删除。建议使用 install 子命令。

故障排除 如果存储对象包含预先存在的分区、zpool 或 UFS 文件系统，则 install 会失败并显示错误消息。要继续安装和覆盖所有预先存在的数据，请使用 zoneadm install 的相应 -x 选项。

从不可用的计算机上迁移区域

托管非全局区域的计算机可能会变得不可用。但是，如果该区域所在的存储器（如 SAN）仍然可用，则仍可以将区域成功迁移到新主机。可将区域的 zonepath 移动到新主机。在某些情况下（如 SAN），zonepath 数据实际上可能未移动。可能只需对 SAN 进行重新配置，便可在新主机上显示 zonepath。由于没有正确分离区域，因此必须首先使用 zonecfg 命令在新主机上创建该区域。完成此操作后，在新主机上附加该区域。

执行此任务的过程在第 296 页中的 “如何使用 ZFS 归档文件迁移非全局区域”中介绍。

将 Oracle Solaris 系统迁移到非全局区域

由于区域不可嵌套，因此 P2V 过程将导致迁移的系统内的任何现有区域在目标区域中不可用。必须在迁移全局区域的系统映像前迁移源系统中的现有非全局区域。

关于将 Oracle Solaris 系统迁移到 solaris 非全局区域

现有的 Oracle Solaris 11 系统可直接迁移到 Oracle Solaris 11 系统上的 solaris 标记区域。在源系统上使用 zonep2vchk 和 zfs 命令准备迁移和归档系统映像。在目标系统上使用 zonecfg 和 zoneadm 命令配置归档文件并将其安装到目标区域。

将全局区域迁移到非全局区域存在以下限制：

- 目标系统上的全局区域所运行的 Oracle Solaris 11 发行版必须等于或高于原始源主机。

- 要确保区域正常运行，目标系统必须安装有所要求的操作系统软件包版本或更高版本。其他软件包（如用于第三方产品的软件包）可以有所不同。

有关更多信息，请参见 zonep2vchk(1M)、zfs(1M)、zonecfg(1M)、zoneadm(1M) 和 solaris(5) 手册页。

▼ 使用 zonep2vchk 扫描源系统

1. **成为管理员。**
2. **运行带 -b 选项的 zonep2vchk 工具执行基本分析，检查是否使用了可能受 P2V 迁移影响的 Oracle Solaris 功能。**

```
source# zonep2vchk -b 11
```

3. **运行带 -s 选项的 zonep2vchk 工具执行对应用程序文件的静态分析。该分析可检测 ELF 二进制文件中是否存在可能影响区域内操作的系统和库调用。**

```
source# zonep2vchk -s /opt/myapp/bin,/opt/myapp/lib
```

4. **运行带 -r 选项的 zonep2vchk 工具执行运行时检查，查找无法在区域内成功执行的进程。**

```
source# zonep2vchk -r 2h
```

5. **在源系统上运行带 -c 选项的 zonep2vchk 工具生成模板 zonecfg 脚本，在此过程中名为 s11-zone.config。**

```
source# zonep2vchk -c > /net/somehost/p2v/s11-zone.config
```

此配置将包含基于源主机的物理资源和联网配置的资源限制和网络配置。

▼ 如何在网络设备上创建系统映像的归档文件

归档全局区域中的文件系统。确认源系统中没有安装非全局区域。支持多种归档文件格式，包括：cpio、使用 -x xustar (XUSTAR) 格式创建的 pax 归档文件以及 zfs。本节中的示例使用 zfs send 命令来创建归档文件。这些示例假定根池的名称为 rpool。

1. **成为管理员。**
2. **创建整个根池的快照，在此过程中名为 *rpool@p2v*。**

```
source# zfs snapshot -r rpool@p2v
```

3. **销毁与交换设备和转储设备关联的快照，目标系统上不需要这些快照。**

```
source# zfs destroy rpool/swap@p2v

source# zfs destroy rpool/dump@p2v
```

4 **归档系统。**

- **生成 ZFS 复制流归档文件，该文件采用 `gzip` 压缩并存储于远程 NFS 服务器上。**

  ```
  source# zfs send -R rpool@p2v | gzip > /net/somehost/p2v/s11-zfs.gz
  ```

- **可使用以下替代命令来避免保存中间快照，从而减小归档文件的大小。**

  ```
  source# zfs send -rc rpool@p2v
  ```

另请参见 有关更多信息，请参见 cpio(1)、pax(1) 和 zfs(1M) 手册页。

▼ 如何配置目标系统上的区域

zonep2vchk 工具生成的模板 zonecfg 脚本定义源系统配置中必须受目标区域配置支持的各个方面。必须手动提供其他目标系统相关信息以完整地定义区域。

在此过程中配置文件名为 s11-zone.config。

1 **成为管理员。**

2 **查看 `zonecfg` 脚本的内容以熟悉源系统的配置参数。**

```
target# less /net/somehost/p2v/s11-zone.config
```

此脚本中 zonepath 的初始值基于源系统的主机名。如果目标区域的名称与源系统的主机名不同，可以更改 zonepath 目录。

注释掉的命令反映原始物理系统环境的参数，包括内存容量、CPU 数量以及网卡 MAC 地址。可在目标区域中取消注释这些行以实现其他资源控制。

3 **在目标系统的全局区域中使用以下命令来查看当前链路配置。**

```
target# dladm show-link
target# dladm show-phys
target# ipadm show-addr
```

缺省情况下，zonecfg 脚本为源系统中配置的每个物理网络接口定义一个具有 anet 资源的独占 IP 网络配置。目标系统在区域引导时自动为每个 anet 资源创建一个 VNIC。使用 VNIC 可实现多个区域共享同一物理网络接口。zonecfg 命令最初将 anet 资源的 lower-link 名称设为 *change-me*。必须手动将此字段设为目标系统上的某个数据链路名称。可以指定可作为有效 VNIC lower-link 的任何链路。

4 **将 `zonecfg` 脚本复制到目标系统。**

```
target# cp /net/somehost/p2v/s11-zone.config .
```

5 **使用文本编辑器（如 `vi`）对配置文件进行任何更改。**

```
target# vi s11-zone.config
```

6 使用 **zonecfg** 命令配置 *s11-zone* 区域。

```
target# zonecfg -z s11-zone -f s11-zone.config
```

▼ 在目标系统上安装区域

此示例在安装期间不改变原始系统配置。

1 成为管理员。

2 使用在源系统上创建的归档文件安装区域。

```
target# zoneadm -z s11-zone install -a /net/somehost/p2v/s11-zfs.gz -p
```

◆ ◆ ◆ 第 24 章

关于安装了区域的 Oracle Solaris 11.1 系统上的自动安装和软件包

您可在 AI 客户机安装过程中指定非全局区域的安装和配置。本发行版支持映像包管理系统 (Image Packaging System, IPS)。本章论述在安装区域的情况下，通过使用 IPS 包管理来安装和维护操作系统。

有关 `solaris10` 和 `native` 区域中使用的 SVR4 包管理和修补的信息，请参见《System Administration Guide: Oracle Solaris Containers-Resource Management and Oracle Solaris Zones》中的“第 25 章，关于安装了区域的 Oracle Solaris 系统上的软件包（概述）”和“第 26 章，在安装了区域的 Oracle Solaris 系统上添加和删除软件包和修补程序（任务）。这是本指南的 Oracle Solaris 10 版本。

运行 Oracle Solaris 11.1 发行版的系统上的映像包管理系统软件

利用图形和命令行工具可从系统信息库中下载及安装软件包。本章提供关于将软件包添加到已安装的非全局区域的信息。同时还包含有关删除软件包的信息。本章中的材料是对现有 Oracle Solaris 安装和包管理文档的补充。有关更多信息，请参见《Oracle Solaris 管理：常见任务》和《添加和更新 Oracle Solaris 11.1 软件包》中的第 4 章 “安装和更新软件包”。

区域包管理概述

`solaris` 包管理系统信息库用于管理区域环境。

当您使用 `pkg` 命令将系统升级到新版本 Oracle Solaris 时，区域会自动更新。

`pkg`(5) 中所述的映像包管理系统 (Image Packaging System, IPS) 是用于提供软件包的安装、升级和删除等软件生命周期管理的框架。IPS 可用于创建软件包、创建和管理包管理系统信息库及镜像现有包管理系统信息库。

初始安装 Oracle Solaris 操作系统后，可通过映像包管理系统 CLI 和 GUI（软件包管理器）客户机从包管理系统信息库安装其他软件应用程序。

在系统中安装软件包之后，可以使用 IPS 客户端对其进行搜索、升级和管理。IPS 客户机还可用来将整个系统升级到 Oracle Solaris 的新发行版、创建和管理系统信息库以及镜像现有的系统信息库。

如果安装了 IPS 的系统能够访问 Internet，则客户机可以从 Oracle Solaris 11.1 软件包系统信息库（缺省的 `solaris` 发布者）访问及安装软件，网址为 `http://pkg oracle.com/solaris/release/`。

在本文档介绍的限制范围之内，区域管理员可以使用包管理工具来管理安装在非全局区域中的任何软件。

安装区域时，将应用以下一般原则：

- 如果某个软件包已安装在全局区域中，则非全局区域可从全局区域的系统信息库服务安装该软件包，而不必使用网络安装该软件包。如果该软件包尚未安装在全局区域中，则区域需要使用区域代理服务访问发布者，以通过网络使用全局区域安装该软件包。
- 全局管理员或拥有相应授权的用户可管理系统上各个区域中的软件。
- 通过使用 Oracle Solaris 包管理工具，可以从全局区域管理非全局区域的根文件系统。在非全局区域中支持使用 Oracle Solaris 包管理工具管理共同打包（捆绑）产品、独立（非捆绑）产品或第三方产品。
- 包管理工具在启用了区域的环境中工作。这些工具还允许将软件包安装在非全局区域中。

注 – 当执行某些软件包操作时，将针对此类型的其他操作暂时锁定区域。系统也可能会在继续执行请求的操作之前向管理员确认。

关于软件包和区域

安装在 `solaris` 标记区域中的软件（如 `brands(5)`中所述）必须与全局区域中安装的软件相兼容。`pkg` 命令自动强制执行此兼容性。如果在全局区域中运行 `pkg update` 命令以更新软件，则还会更新各区域，以使各区域与全局区域保持同步。非全局区域和全局区域可安装不同的软件。还可在某个区域中使用 `pkg` 命令，以管理该区域中的软件。

如果 `pkg update` 命令（未指定任何 FMRI）正在全局区域中运行，`pkg` 将更新系统上全局区域和任何非全局区域中的所有软件。

您可在 Oracle Solaris Zones 中使用 `pkg install` 的试运行（也称为预运行）安装功能。

通过使用区域软件包变体，将软件包内的不同组件具体标记为只能安装在全局区域 (`global`) 或非全局区域 (`nonglobal`)。给定软件包会包含一个标记的文件，这样便不会将此软件包安装到非全局区域。

当安装非全局区域时，仅会完全复制全局区域中安装的部分 Oracle Solaris 软件包。例如，很多包含 Oracle Solaris 内核的软件包在非全局区域中是不需要的。所有非全局区域隐含共享全局区域中的同一内核。

有关更多信息，请参见《添加和更新 Oracle Solaris 11.1 软件包》中的“使用非全局区域”和《安装 Oracle Solaris 11.1 系统》。

注 – 更新包含多个非全局区域的系统上的全局区域时，对于这些区域，系统可能会出现两次显示软件包下载信息的情况。实际上，只会下载软件包一次。

关于在安装了区域的系统中添加软件包

在 Oracle Solaris 11 发行版中，使用 `pkg install` 命令。

```
# pkg install package_name
```

在全局区域中使用 pkg

在全局区域中使用 `pkg install` 命令可将软件包仅添加到全局区域。软件包不会传播到其他任何区域。

在非全局区域中使用 pkg install 命令

区域管理员在非全局区域中使用 `pkg install` 命令可将软件包仅添加到非全局区域。要在指定的非全局区域中添加软件包，请以区域管理员身份执行 `pkg install` 命令。

软件包相关项将在 IPS 中自动处理。

使用定制 AI 清单在区域中添加其他软件包

可通过修订 AI 清单实现在安装期间自动在区域中添加额外软件。将安装指定的软件包及其依赖的软件包。将从 AI 清单获取缺省的软件包列表。缺省 AI 清单是 `/usr/share/auto_install/manifest/zone_default.xml`。有关查找和使用软件包的信息，请参见《添加和更新 Oracle Solaris 11.1 软件包》。

示例 24–1 修订清单

以下过程将 `mercurial` 和 `vim` 编辑器的完整安装添加到名为 *my-zone* 的已配置区域。（请注意，缺省情况下，只安装最小的 `vim-core`，vim-core 是 `solaris-small-server` 的一部分。

示例 24–1 修订清单 （续）

1. 将缺省 AI 清单复制到用于编辑此文件的位置，使此文件可写。

```
# cp /usr/share/auto_install/manifest/zone_default.xml ~/my-zone-ai.xml
# chmod 644 ~/my-zone-ai.xml
```

2. 编辑此文件，将 `mercurial` 和 `vim` 软件包添加到 `software_data` 部分，如下所示：

```
      <software_data action="install">
              <name>pkg:/group/system/solaris-small-server</name>
              <name>pkg:/developer/versioning/mercurial</name>
              <name>pkg:/editor/vim</name>
            </software_data>
```

3. 安装区域。

```
# zoneadm -z my-zone install -m ~/my-zone-ai.xml
```

系统将显示：

```
A ZFS file system has been created for this zone.
Progress being logged to /var/log/zones/zoneadm.20111113T004303Z.my-zone.install
       Image: Preparing at /zones/my-zone/root.

 Install Log: /system/volatile/install.15496/install_log
 AI Manifest: /tmp/manifest.xml.XfaWpE
  SC Profile: /usr/share/auto_install/sc_profiles/enable_sci.xml
    Zonename: my-zone
Installation: Starting ...

              Creating IPS image
              Installing packages from:
                  solaris
                      origin:  http://localhost:1008/solaris/54453f3545de891d4daa841ddb3c844fe8804f55/

DOWNLOAD                                  PKGS       FILES    XFER (MB)
Completed                              169/169 34047/34047  185.6/185.6

PHASE                                        ACTIONS
Install Phase                              46498/46498

PHASE                                          ITEMS
Package State Update Phase                 169/169
Image State Update Phase                       2/2
Installation: Succeeded
...
```

关于在区域中删除软件包

可使用 `pkg uninstall` 命令删除安装了区域的系统上的软件包。

```
# pkg uninstall package_name
```

软件包信息查询

可使用 pkg info 命令查询安装了区域的系统上的软件包数据库。

可以在全局区域中使用该命令来查询仅位于全局区域中的软件包数据库。可以在非全局区域中使用该命令来查询仅位于非全局区域中的软件包数据库。

在安装了区域的系统上进行代理配置

应按Chapter 5, Configuring Installed Images,《添加和更新 Oracle Solaris 11.1 软件包》中的第 5 章 "配置已安装的映像" 选项在映像中设置持久性代理。如果未使用持久性映像代理配置，且运行 pkg 命令时始终使用 http_proxy 和 https_proxy 环境变量访问系统信息库，则还应通过 SMF system-repository 服务属性配置 system-repository 服务使用相同的代理。请参见 pkg(1) 手册页。

在全局区域中配置的系统信息库访问可使用 system-repository 服务提供给非全局区域。在全局区域中对源代理所做的任何更新将自动加入到 system-repository 配置中。使用这种方法，无需对 system-repository SMF 服务进行任何修改。

也可以对 system-repository SMF 服务使用的代理进行配置，覆盖在全局区域中为发布者配置的代理。可以使用 config/http_proxy 或 config/https_proxy SMF 属性设置 system-repository 代理。

有关更多信息，请参见 pkg.sysrepo(1M) 手册页和《添加和更新 Oracle Solaris 11.1 软件包》。

在全局区域中配置代理

可以直接在全局区域中配置代理，在全局区域中对源代理所做的任何更新将自动加入到 system-repository 配置中。system-repository 服务并不需要修改。

示例 24–2 在全局区域中配置代理

```
# pkg set-publisher --proxy http://www-proxy -g http://pkg-server pub
```

除非代理在 80 以外的端口上接受连接，否则不需要指定端口。

如果系统中有区域，将重新启动 system-repository 服务，并使用代理提供对 *pkg-server* 的访问。

使用 https_proxy 和 http_proxy 覆盖 system-repository 代理

建议在映像中设置代理，且只设置 system-repository 服务代理。应在运行 pkg 命令时在环境中设置 https_proxy 和 http_proxy。

本节中的过程用于在不直接连接到 IPS 发布者系统信息库的内部子网中设置 system-repository 服务代理。使用此过程可覆盖使用 pkg 命令在全局区域中配置的代理。非全局区域通过 HTTP 与 system-repository 通信。system-repository 然后按照全局区域中的配置，使用该系统信息库的访问协议访问发布者。

此配置还允许 solaris 非全局区域联系全局区域中的发布者集。递归到 solaris 区域的 pkg 操作将会成功。

示例 24–3 使用 https_proxy 和 http_proxy 覆盖全局区域代理

例如，假设运行 solaris 非全局区域的系统上的软件由 IPS 管理，且该软件需要使用代理服务器 http_proxy=http:// *129.156.243.243:3128* 来访问 http 和 https URL。以下步骤显示了如何使用 http_proxy 和 https_proxy 环境变量和 SMF 服务属性来允许全局区域和非全局区域访问 IPS 系统信息库。

请注意，这些变量会覆盖在源上设置的代理配置，除非用户从连接到系统发布者的统一资源标识符 (Universal Resource Identifier, URI) 的非全局区域运行 pkg 命令。在这种情况下，该命令遍历 system-repository。

还可以使用可解析的主机名。

1. 在 shell 中键入以下行，为全局区域设置代理：

   ```
   # export http_proxy=http://129.156.243.243:3128
   # export https_proxy=http://129.156.243.243:3128
   ```

   设置代理可允许 pkg 命令通过代理服务器访问发布者。这会影响使用 https 或 http URL 而不通过 system-repository 访问全局区域的 pkg 操作。

2. 要允许系统上的 solaris 区域使用所配置的可从全局区域直接访问的系统发布者，请执行以下命令：

```
# svccfg -s system-repository:default setprop config/http_proxy = http://129.156.243.243:3128
# svccfg -s system-repository:default setprop config/https_proxy = http://129.156.243.243:3128
```

3. 要使更改在实时 SMF 系统信息库中生效，请运行：

   ```
   # svcadm refresh system-repository
   ```

4. 要确认设置是否正常工作，请运行：

   ```
   # svcprop -p config/http_proxy system-repository
   # svcprop -p config/https_proxy system-repository
   ```

有关 pkg 命令的更多信息，请参见 pkg(1) 手册页。

并行区域更新

可以将区域配置为并行更新而不是串行更新。并行更新可大幅缩短更新系统上的所有区域所需的时间。有关其他信息和配置示例，请参见《添加和更新 Oracle Solaris 11.1 软件包》中的“同时更新多个非全局区域”。

区域状态对软件包操作有何影响

下表介绍了在非全局区域处于各种状态的系统中使用包管理命令时将发生的情况。

区域状态	对软件包操作的影响
已配置	可运行软件包工具。尚未安装任何软件。
未完成	如果 zoneadm 正在对区域进行操作，不应使用软件包工具。如果没有 zoneadm 进程在对区域进行操作，软件包操作可以安全运行，但不会更改该区域中的软件，且该区域中的任何软件都不会影响相关项解析。
不可用	无法访问区域中的软件映像。不会更改软件映像，也不会影响相关项解析。
已安装	可运行软件包工具。 请注意，在执行完 zoneadm -z *zonename* install 后，区域也会立即移至已安装状态。
就绪	可运行软件包工具。
正在运行	可运行软件包工具。

当非全局区域的存储不可访问，或该区域的映像与全局区域的映像不同步时（如 pkg(5) 中所述），该区域将转换到不可用状态。发生该状态转换是为了避免影响非全局区域的问题导致全局区域中的软件包操作受阻。

当区域的存储暂时不可用时，如果发生了更改已安装软件版本的软件包操作，在修复存储问题后，可能需要使用某个 solaris 标记的 attach 选项附加该区域，以便允许更新。例如，可能需要使用 zoneadm -z *zonename* attach -u 同步全局区域和处于不可用状态的非全局区域的关键软件版本。

◆ ◆ ◆ 第 25 章

Oracle Solaris Zones 管理（概述）

本章介绍以下常规区域管理主题：

有关 solaris10 标记区域的信息，请参见第 3 部分。

全局区域可见性和访问权限

全局区域既可作为系统的缺省区域，也可作为在系统范围内实施管理控制的区域。这种双重角色会引起管理问题。由于全局区域内的应用程序有权访问其他区域中的进程和其他系统对象，因此，管理操作的影响范围会比预期的范围更广。例如，服务关闭脚本通常使用 `pkill` 来通知退出具有给定名称的进程。在全局区域中运行此脚本时，将通知退出系统中所有区域内的所有此类进程。

通常需要将整个系统作为考虑范围。例如，要监视系统范围内的资源使用情况，必须查看整个系统中的进程统计信息。如果仅查看全局区域活动，则会遗漏系统中可能正在共享部分或全部系统资源的其他区域的相关信息。在没有使用资源管理功能对系统资源（如 CPU）进行严格分区的情况下，此类查看尤为重要。

因此，全局区域中的进程可以查看非全局区域中的进程和其他对象。这样，此类进程便可查看整个系统范围的内容。控制信号或将信号发送到其他区域中进程的功能由特权 `PRIV_PROC_ZONE` 加以限制。此特权类似于 `PRIV_PROC_OWNER`，因为它允许进程覆盖对非特权进程设定的限制。在这种情况下，所谓的限制是指全局区域中的非特权进程无法向其他区域中的进程发送信号或控制这些进程。即使进程的用户 ID 相匹配或者正在运行的进程拥有 `PRIV_PROC_OWNER` 特权，也会存在上述限制。可以删除其他特权进程的 `PRIV_PROC_ZONE` 特权，以将操作限制为仅对全局区域有效。

有关使用 `zoneidlist` 匹配进程的信息，请参见 `pgrep(1)` `pkill(1)` 手册页。

区域中的进程 ID 可见性

只有同一区域中的进程才能通过使用进程 ID 的系统调用接口（例如 `kill` 和 `priocntl` 命令）进行查看。有关更多信息，请参见 `kill(1)` 和 `priocntl(1)` 手册页。

区域中的系统可查看性

对 `ps` 命令进行了以下修改：

- `-o` 选项用于指定输出格式。使用此选项，可以列显进程的区域 ID 或运行此进程的区域名称。
- `-z` *zonelist* 选项用于仅列出指定区域中的进程。可以通过区域名称或区域 ID 指定区域。只有在全局区域中执行命令时，此选项才有用。
- `-Z` 选项用于列显与进程关联的区域名称。区域名称在列标题 `ZONE` 下列显。

有关更多信息，请参见 `ps(1)` 手册页。

已将 `-z` *zonename* 选项添加到以下 Oracle Solaris 实用程序。可以使用此选项将信息过滤为仅包括指定的一个或多个区域。

- `ipcs`（请参见 `ipcs(1)` 手册页）

- pgrep（请参见 pgrep(1) 手册页）
- ptree（请参见 proc(1) 手册页）
- prstat（请参见 prstat(1M) 手册页）

有关对命令所做的更改的完整列表，请参见表 25–5。

利用 zonestat 实用程序报告活动区域统计信息

要使用 zonestat 实用程序，请参见 zonestat(1) 手册页和第 343 页中的 “在非全局区域中使用 zonestat 实用程序”。

zonestat 实用程序会报告当前正在运行的区域的 CPU、内存和资源控制使用情况。zonestat 实用程序以指定的时间间隔打印一系列报告。该实用程序也可以打印一个或多个摘要报告。

zonestat 实用程序还可报告专用 IP 区域中的网络带宽使用情况。专用 IP 区域具有其自己的 IP 相关状态以及一个或多个专用数据链路。

在非全局区域内运行时，只会报告对区域可见的处理器集。非全局区域输出包含所有内存资源以及限制资源。

全局区域内的 zonestat 服务必须处于联机状态才能在非全局区域内使用 zonestat 服务。每个非全局区域内的 zonestat 服务都从全局区域内的 zonestat 服务读取系统配置和使用情况数据。

zonestatd 系统守护进程在系统引导期间启动。守护进程监视区域使用系统资源的情况，以及区域和系统的配置信息，如 psrset 处理器集、池处理器集和资源控制设置。没有可以配置的组件。

使用 fsstat 实用程序监视非全局区域

fsstat 实用程序收集并输出每个区域的 kstat，包括聚合。缺省情况下，实用程序报告所有正在运行的区域的聚合。每个 fstype kstat 为每个区域而生成。全局区域 kstat 报告其专用活动。全局区域可以看到系统上的所有区域的 kstat。非全局区域只能看到与实用程序运行的区域相关的 kstats。非全局区域不能监视其他区域的文件系统活动。

有关更多信息，请参见 fsstat(1M) 手册页和第 346 页中的 “报告所有区域的每区域 fstype 统计信息”。

非全局区域节点名称

节点名称是系统名称的本地源。节点名称必须唯一，如区域名称。节点名称可以由区域管理员设置。

```
# hostname myhostname
```

要查看主机名，请键入主机名。

```
# hostname
...
myhostname
```

在区域内运行 NFS 服务器

NFS 服务器包 `svc:/network/nfs/server:default` 必须安装在区域中才能在区域内创建 NFS 共享。无法在区域创建期间安装 NFS 服务器软件包。

可以在区域配置中禁止 `sys_share` 特权，以防止在区域内共享 NFS。请参见表 25–1。

约束和限制包括以下各项：

- 无法从区域共享跨区域 LOFS 挂载。
- 无法从全局区域共享在区域内挂载的文件系统。
- 区域内不支持使用远程直接内存访问 (Remote Direct Memory Access, RDMA) 的 NFS。
- 区域内不支持 Oracle Sun Cluster HA for NFS (HANFS) 故障转移。

请参见《Oracle Solaris 管理：网络服务》。

文件系统和非全局区域

本节介绍有关安装了区域的 Oracle Solaris 系统上文件系统问题的相关信息。每个区域都有自己的文件系统分层结构部分，根目录称为区域 `root`。区域中的进程仅可访问区域根目录下的分层结构部分中的文件。`chroot` 实用程序可以在区域中使用，但是仅用于将进程限制在区域内的根路径。有关 `chroot` 的更多信息，请参见 `chroot(1M)`。

`-o nosuid` 选项

`mount` 实用程序的 `-o nosuid` 选项具有以下功能：

- 在使用 `nosetuid` 选项挂载的文件系统上，`setuid` 二进制命令中的进程无法使用 `setuid` 二进制命令特权运行。而是使用执行此二进制命令的用户特权运行。

  例如，如果用户执行属于 `root` 的 `setuid` 二进制命令，则进程使用此用户的特权运行。

- 不允许打开文件系统中的特定设备项。此行为相当于指定 nodevices 选项。

所有可使用 mount 实用程序（如 mount(1M) 手册页中所述）挂载的 Oracle Solaris 文件系统都可以使用这一特定于文件系统的选项。在本指南中，这些文件系统在第 315 页中的“在区域中挂载文件系统”中列出。同时也对挂载功能进行了说明。有关 -o nosuid 选项的更多信息，请参见《Oracle Solaris 管理：网络服务》中的“访问网络文件系统（参考信息）”。

在区域中挂载文件系统

从区域中挂载文件系统时，将应用 nodevices 选项。例如，如果区域被授予访问对应于 UFS 文件系统的块设备 (/dev/dsk/c0t0d0s7) 和原始设备 (/dev/rdsk/c0t0d0s7) 的权限，则从区域中挂载此文件系统时，会自动使用 nodevices 选项挂载。此规则不适用于通过 zonecfg 配置指定的挂载。

下表介绍用于在非全局区域中挂载文件系统的选项。其他挂载方法过程在第 231 页中的“配置、检验并提交区域”和第 348 页中的“在正在运行的非全局区域中挂载文件系统”中介绍。

对于未在此表中列出的任意文件系统类型，如果它在 /usr/lib/*fstype*/mount 中具有挂载二进制命令，则可以在配置中指定此文件系统类型。

要挂载除了非全局区域中 HSFS 和 NFS 之外的文件系统类型，也使用 zonecfg fs-allowed 属性将文件系统类型添加到配置。

如果允许挂载缺省文件系统之外的其他文件系统，则区域管理员可以影响系统。

文件系统	非全局区域中的挂载选项
AutoFS	不能使用 zonecfg 挂载。可以在区域中挂载。
CacheFS	不能在非全局区域中使用。
FDFS	可以使用 zonecfg 挂载，可以从区域中挂载。
HSFS	可以使用 zonecfg 挂载，可以从区域中挂载。
LOFS	可以使用 zonecfg 挂载，可以从区域中挂载。
MNTFS	不能使用 zonecfg 挂载。可以在区域中挂载。
NFS	不能使用 zonecfg 挂载。当前区域所支持的版本 V2、V3 和 V4 可以在区域中挂载。
PCFS	可以使用 zonecfg 挂载，可以从区域中挂载。
PROCFS	不能使用 zonecfg 挂载。可以在区域中挂载。
TMPFS	可以使用 zonecfg 挂载，可以从区域中挂载。

文件系统	非全局区域中的挂载选项
UDFS	可以使用 zonecfg 挂载，可以从区域中挂载。
UFS	可以使用 zonecfg 挂载，可以从区域中挂载。 **注 –** quota(1M) 中所述的 quota 命令不能用来检索通过 zonecfg add fs 资源添加的 UFS 文件系统的配额信息。 如果使用 add fs，必须在全局区域内安装 system/file-system/ufs 软件包。要在非全局区域内通过 zonecfg 命令使用 UFS 文件系统，必须在安装后或通过 AI 清单脚本将该软件包安装到区域中。 以下内容键入到一行中： `global# pkg -R /tank/zones/my-zone/root \ install system/file-system/ufs`
VxFS	可以使用 zonecfg 挂载，可以从区域中挂载。
ZFS	可以使用 zonecfg dataset 和 fs 资源类型进行挂载。

有关更多信息，请参见第 232 页中的 “如何配置区域”、第 348 页中的 “在正在运行的非全局区域中挂载文件系统”和 mount(1M) 手册页。

在区域中卸载文件系统

卸载文件系统的能力将取决于执行初始挂载的人员。如果使用 zonecfg 命令将文件系统指定为区域配置的一部分，则全局区域将拥有此挂载，而非全局区域管理员无法卸载该文件系统。如果从非全局区域内挂载文件系统（例如在区域 /etc/vfstab 文件中指定挂载），则非全局区域管理员可以卸载该文件系统。

安全限制和文件系统行为

在区域中挂载某些文件系统时存在安全限制。其他文件系统在区域中挂载时会显示出特殊行为。已修改的文件系统列表如下。

AutoFS
: Autofs 是一项可自动挂载相应文件系统的客户端服务。当客户机尝试访问目前未挂载的文件系统时，AutoFS 文件系统会拦截请求并调用 automountd 以挂载请求的目录。在区域中建立的 AutoFS 挂载对于此区域而言是本地挂载。不能从其他区域（包括全局区域）访问这些挂载。在停止或重新引导区域时，将删除挂载。有关 AutoFS 的更多信息，请参见《Oracle Solaris 管理：网络服务》中的“Autofs 如何工作”。

每个区域都运行自己的 automountd 副本。自动映射和超时由区域管理员控制。不能跨越非全局区域的 AutoFS 挂载点从全局区域触发其他区域中的挂载。

触发其他挂载时，便会在内核中创建某些 AutoFS 挂载。此类挂载不能使用常规 umount 接口删除，因为它们必须作为一个组进行挂载或卸载。请注意，提供此功能是为了关闭区域。

MNTFS
: MNTFS 是一款虚拟文件系统，可提供本地系统中已挂载文件系统表的只读访问权限。在非全局区域中使用 mnttab 可查看的一组文件系统是该区域中已挂载的一组文件系统和一个根 (/) 项。具有无法在区域中访问的特殊设备的挂载点（例如 /dev/rdsk/c0t0d0s0）都将其特殊设备的挂载点设置为与此挂载点相同。系统中的所有挂载都可从全局区域的 /etc/mnttab 表中查看。有关 MNTFS 的更多信息，请参见《Oracle Solaris 11.1 管理：设备和文件系统》中的“挂载和取消挂载 Oracle Solaris 文件系统”。

NFS
: 在区域中建立的 NFS 挂载对于此区域而言是本地挂载。不能从其他区域（包括全局区域）访问这些挂载。在停止或重新引导区域时，将删除挂载。

  在区域中，NFS 挂载如同使用 nodevices 选项进行挂载。

  nfsstat 命令输出仅与运行此命令的区域有关。例如，如果在全局区域中运行此命令，则仅报告有关此全局区域的信息。有关 nfsstat 命令的更多信息，请参见 nfsstat(1M)。

PROCFS
: /proc 文件系统（或 PROCFS）提供进程可见性和访问限制，同时还提供有关进程的区域关联的信息。通过 /proc 只能查看同一区域中的进程。

  全局区域中的进程可以查看非全局区域中的进程和其他对象。这样，此类进程便可查看整个系统范围的内容。

  在区域中，procfs 挂载如同使用 nodevices 选项进行挂载。有关 procfs 的更多信息，请参见 proc(4) 手册页。

LOFS
: 通过 LOFS 进行挂载的范围被限制为区域中可见的文件系统部分。因此，对区域中的 LOFS 挂载没有任何限制。

UFS、UDFS、PCFS 以及其他基于存储的文件系统
: 使用 zonecfg 命令配置具有 fsck 二进制命令的基于存储的文件系统（例如 UFS）时，区域管理员必须指定 raw 参数。该参数指明原始（字符）设备，如 /dev/rdsk/c0t0d0s7。zoneadmd 守护进程自动在清理模式下运行 fsck 命令 (fsck -p)，该命令在挂载文件系统之前以非交互方式检查并修复文件系统。如果 fsck 失败，则 zoneadmd 无法使区域达到就绪状态。由 raw 指定的路径不能是相对路径。

  对于没有在 /usr/lib/fs/ *fstype*/fsck 中提供 fsck 二进制代码的文件系统，不能为 fsck 指定设备。如果此文件系统具有 fsck 二进制命令，则必须为 fsck 指定设备。

有关更多信息，请参见第 253 页中的“zoneadmd 守护进程”和 fsck(1M) 命令。

ZFS

除了第 201 页中的“在区域中挂载的文件系统”中所述的缺省数据集外，您还可以使用 zonecfg 命令以及 add dataset 资源将 ZFS 数据集添加到非全局区域。此数据集在非全局区域中进行挂载并显示，并且在全局区域中也可见。区域管理员可以在此数据集中创建和销毁文件系统，并可修改此数据集的属性。

zfs 的 zoned 属性指明是否已将数据集添加到非全局区域。

```
# zfs get zoned tank/sales
NAME          PROPERTY    VALUE       SOURCE
tank/sales    zoned       on          local
```

通过数据集资源委托到非全局区域中的每个数据集都具有别名。数据集布局在区域内不可见。每个具有别名的数据集在区域中的显示方式与在池中一样。数据集的缺省别名是数据集名称中最后的部分。例如，如果委托数据集 tank/sales 使用缺省别名，则区域将看到名为 sales 的虚拟 ZFS 池。在数据集资源内设置别名属性，可以将别名定制为其他值。

每个非全局区域的 zonepath 数据集中都有一个名为 rpool 的数据集。对于所有非全局区域，该区域 rpool 数据集别名为 rpool。

```
my-zone# zfs list -o name,zoned,mounted,mountpoint
NAME                 ZONED  MOUNTED  MOUNTPOINT
rpool                   on       no  /rpool
rpool/ROOT              on       no  legacy
rpool/ROOT/solaris      on      yes  /
rpool/export            on       no  /export
rpool/export/home       on       no  /export/home
```

数据集别名的名称限制与 ZFS 池的名称限制相同。这些限制在 zpool(1M) 手册页中介绍。

如果要共享全局区域中的数据集，可以使用具有 add fs 子命令的 zonecfg 命令来添加通过 LOFS 方式挂载的 ZFS 文件系统。全局管理员或授予了相应权限的用户负责设置和控制数据集的属性。

有关 ZFS 的更多信息，请参见《Oracle Solaris 11.1 管理：ZFS 文件系统》中的第 9 章 “Oracle Solaris ZFS 高级主题”。

作为 NFS 客户机的非全局区域

区域可以是 NFS 客户机。支持版本 2、版本 3 和版本 4 协议。有关这些 NFS 版本的信息，请参见《Oracle Solaris 管理：网络服务》中的“NFS 服务的功能”。

缺省版本为 NFS 版本 4。可以使用以下方法之一在客户机上启用其他 NFS 版本：

- 可以使用 sharectl(1M) 设置属性。设置 NFS_CLIENT_VERSMAX=*number*，以使区域在缺省情况下使用指定的版本。请参见《Oracle Solaris 管理：网络服务》中的“设置 NFS 服务”。按照《在 Oracle Solaris 11.1 中管理网络文件系统》中的“如何在客户机上选择不同的 NFS 版本”过程操作。
- 可以手动创建版本挂载。该方法将覆盖 sharectl 设置。请参见《Oracle Solaris 管理：网络服务》中的“设置 NFS 服务”。按照《在 Oracle Solaris 11.1 中管理网络文件系统》中的“如何在客户机上选择不同的 NFS 版本”过程操作。

在区域中禁止使用 mknod

请注意，不能使用 mknod(1M) 手册页中所述的 mknod 命令在非全局区域中创建特殊文件。

遍历文件系统

区域的文件系统名称空间是可从全局区域访问的名称空间的子集。可以通过以下方式，防止全局区域中的非特权进程遍历非全局区域的文件系统分层结构：

- 指定区域根目录的父目录仅可由根拥有、读取、写入和执行
- 限制访问由 /proc 导出的目录

请注意，尝试访问为其他区域挂载的 AutoFS 节点将失败。全局管理员不必具有向下派生到其他区域的自动映射。

从全局区域中访问非全局区域的限制

安装了非全局区域之后，除了系统备份实用程序之外，此区域永远不能通过其他任何命令从全局区域中直接访问。此外，当非全局区域向未知环境公开之后，便不再将其视为安全区域。例如放置在可公共访问的网络上的区域，这种情况下可能会危及区域的安全并且可能会改变其文件系统的内容。如果存在任何危及区域安全的可能性，全局管理员便应将此区域视为不可信区域。

任何可通过 -R 或 -b 选项（或等效选项）接受备用根的命令，在以下情况成立时**不**得使用：

- 命令在全局区域中运行。
- 备用根指非全局区域中的任何路径，此路径既可以是当前运行的系统的全局区域的相对路径，也可以是备用根中全局区域的相对路径。

例如通过非全局区域根路径在全局区域中运行的 pkgadd 实用程序的 -R *root_path* 选项。

以下是通过备用根路径使用 -R 的命令、程序和实用程序的列表：

- auditreduce

- bart
- installf
- localeadm
- makeuuid
- metaroot
- pkg
- prodreg
- removef
- routeadm
- showrev
- syseventadm

以下是通过备用根路径使用 -b 的命令和程序的列表：

- add_drv
- pprosetup
- rem_drv
- roleadd
- update_drv
- useradd

共享 IP 非全局区域中的联网

在安装了区域的 Oracle Solaris 系统中，区域之间可以通过网络相互通信。所有区域都有单独的绑定或连接，并且所有区域都可运行自己的服务器守护进程。这些守护进程可以侦听相同的端口号而不会引起冲突。IP 栈通过分析传入连接的 IP 地址来解决冲突。IP 地址标识区域。

要使用共享 IP 类型，全局区域中的网络配置必须通过 ipadm 来完成，而非通过自动网络配置完成。如果使用了 ipadm，以下命令应返回 DefaultFixed。

```
# svcprop -p netcfg/active_ncp svc:/network/physical:default
DefaultFixed
```

共享 IP 区域分区

共享 IP 不是缺省类型，但受支持。

在支持区域的系统中，IP 栈对区域之间的网络通信流量执行隔离。接收 IP 通信流量的应用程序只能接收发送到同一区域的通信流量。

系统上的每个逻辑接口都属于特定的区域，缺省情况下属于全局区域。借助 zonecfg 实用程序指定给区域的逻辑网络接口用于在网络上进行通信。每个流和连接都属于打开它的进程所在的区域。

上层流和逻辑接口之间的绑定会受到限制。流只能与同一区域中的逻辑接口建立绑定。同样，来自逻辑接口的包只能传递到此逻辑接口所在区域中的上层流。

每个区域都有自己的一组绑定。每个区域都可以运行侦听同一端口号的相同应用程序，而且绑定不会失败，因为地址已处于使用状态。每个区域都可以运行各种网络服务中自己的版本，如：

- 具有完整配置文件的 Internet 服务守护进程（请参见 `inetd(1M)` 手册页）
- `sendmail`（请参见 `sendmail(1M)` 手册页）
- `apache`

除全局区域之外的区域拥有受限的网络访问权限。标准 TCP 和 UDP 套接字接口均可用，但是 `SOCK_RAW` 套接字接口被限制为网际控制报文协议 (Internet Control Message Protocol, ICMP)。ICMP 是检测和报告网络错误状态或使用 `ping` 命令时所必需的。

共享 IP 网络接口

每个需要网络连接的非全局区域都有一个或多个专用 IP 地址。这些地址与可以放入区域中的逻辑网络接口关联。引导区域时，将在其中自动设置并放置通过 `zonecfg` 配置的区域网络接口。运行区域时，可使用 `ipadm` 命令添加或删除逻辑接口。只有全局管理员或者授予了相应权限的用户才能修改接口配置和网络路由。

在非全局区域内，只有该区域的接口才对 `ipadm` 命令可见。

有关更多信息，请参见 `ipadm(1M)` 和 `if_tcp(7P)` 手册页。

同一计算机上共享 IP 区域之间的 IP 通信

如果在目标的转发表中有该目标可用的路由，则共享 IP 区域可以连接任何指定的 IP 目标。要查看转发表，在区域内使用带有 `-r` 选项的 `netstat` 命令。在其他区域或其他系统中，各 IP 目标的 IP 转发规则相同。

共享 IP 区域中的 Oracle Solaris IP 过滤器

Oracle Solaris IP 过滤器可提供有状态包过滤器和网络地址转换 (network address translation, NAT) 功能。有状态包过滤器可以监视活动连接的状态，并使用获得的信息确定允许哪些网络包通过防火墙。Oracle Solaris IP 过滤器还包括无状态包过滤器以及创建和管理地址池的功能。有关其他信息，请参见《在 Oracle Solaris 11.1 中保护网络安全》中的第 4 章 “Oracle Solaris 中的 IP 过滤器（概述）”。

在非全局区域中，可以通过启用回送过滤功能来启用 Oracle Solaris IP 过滤器，如《在 Oracle Solaris 11.1 中保护网络安全》中的第 5 章 “IP 过滤器（任务）”中所述。

Oracle Solaris IP 过滤器是从开源 IP 过滤器软件派生的。

共享 IP 区域中的 IP 网络多路径

IP 网络多路径 (IP network multipathing, IPMP) 为在同一 IP 链路上具有多个接口的系统提供物理接口故障检测和透明网络访问故障转移功能。IPMP 还为具有多个接口的系统提供了包负荷分配。

所有网络配置均在全局区域中完成。可以在全局区域中配置 IPMP，然后将功能扩展到非全局区域。当配置非全局区域时，将此区域的地址放入 IPMP 组中即可实现功能扩展。此后，如果全局区域中有一个接口出现故障，则非全局区域地址将迁移到其他网络接口卡。

在给定的非全局区域中，只有与此区域关联的接口才能通过 `ipadm` 命令进行查看。

请参见第 353 页中的 “如何将 IP 网络多路径功能扩展到共享 IP 非全局区域”。区域配置过程在第 232 页中的 “如何配置区域”中介绍。有关 IPMP 功能、组件和使用情况的信息，请参见《管理 Oracle Solaris 11.1 网络性能》中的第 5 章 “IPMP 介绍”。

专用 IP 非全局区域中的联网

专用 IP 区域具有自己的与 IP 相关的状态。配置该区域时，系统会为该区域指定其自己的数据链路集合。

包在物理链路中传送。然后，类似以太网交换机或 IP 路由器的设备可将包转发到其目的地，该目的地可能位于发送者所用的同一台计算机上的不同区域。

对于虚拟链路，该包首先发送到虚拟交换机。如果目标链路在同一设备上（如在相同物理链路或 etherstub 上的 VNIC），该包将直接发往目标 VNIC。否则，包将脱离 VNIC 底层的物理链路。

有关可在独占 IP 非全局区域中使用的功能的信息，请参见第 199 页中的 “专用 IP 非全局区域”。

专用 IP 区域分区

专用 IP 区域具有单独的 TCP/IP 栈，因此可以隔离数据链路层及其上的所有层。全局管理员可以将一个或多个数据链路名称（可以是 NIC 或 NIC 上的 VLAN）指定给一个专用 IP 区域。区域管理员可以配置这些数据链路上的 IP，其灵活性和选项与全局区域中的相同。

专用 IP 数据链路接口

必须将一个数据链路名称专门指定给单个区域。

可以使用 `dladm show-link` 命令显示指定给正在运行的区域的数据链路。

```
sol-t2000-10{pennyc}1: dladm show-link
LINK                CLASS     MTU    STATE    OVER
vsw0                phys      1500   up       --
e1000g0             phys      1500   up       --
e1000g2             phys      1500   up       --
e1000g1             phys      1500   up       --
e1000g3             phys      1500   up       --
zoneA/net0          vnic      1500   up       e1000g0
zoneB/net0          vnic      1500   up       e1000g0
aggr1               aggr      1500   up       e1000g2 e1000g3
vnic0               vnic      1500   up       e1000g1
zoneA/vnic0         vnic      1500   up       e1000g1
vnic1               vnic      1500   up       e1000g1
zoneB/vnic1         vnic      1500   up       e1000g1
vnic3               vnic      1500   up       aggr1
vnic4               vnic      1500   up       aggr1
zoneB/vnic4         vnic      1500   up       aggr1
```

有关更多信息，请参见 dladm(1M)。

同一计算机上专用 IP 区域之间的 IP 通信

专用 IP 区域之间不存在 IP 数据包内部回送。所有包都向下发送到数据链路。通常，这意味着包通过网络接口发出。然后，类似以太网交换机或 IP 路由器的设备可将包转发到其目的地，该目的地可能位于发送者所用的同一台计算机上的不同区域。

专用 IP 区域中的 Oracle Solaris IP 过滤器

在专用 IP 区域中可以使用在全局区域中使用的相同 IP 过滤器功能。而且在专用 IP 区域中配置 IP 过滤器的方法与在全局区域中使用的方法相同。

专用 IP 区域中的 IP 网络多路径

IP 网络多路径 (IP network multipathing, IPMP) 为在同一 IP 链路上具有多个接口的系统提供物理接口故障检测和透明网络访问故障转移功能。IPMP 还为具有多个接口的系统提供了包负荷分配。

数据链路配置在全局区域中完成。首先，使用 zonecfg 将多个数据链路接口指定给某个区域。这些数据链路接口必须连接到相同的 IP 子网。然后，区域管理员便可在专用 IP 区域内配置 IPMP。

非全局区域中的设备使用

对区域中可用的一组设备进行了限制，以防止某个区域中的进程干扰在其他区域中运行的进程。例如，区域中的进程不能修改内核内存，也不能修改根磁盘的内容。因此，缺省情况下，只提供被视为可以在区域中安全使用的特定伪设备。在特定区域内，可以使用 `zonecfg` 实用程序使其他设备变得可用。

/dev 和 /devices 名称空间

Oracle Solaris 系统使用 devfs(7FS) 手册页中所述的 `devfs` 文件系统来管理 `/devices`。此名称空间中的每个元素都表示指向硬件设备、伪设备或 nexus 设备的物理路径。名称空间是设备树的一种表现形式。同样，文件系统由目录和特定于设备的文件分层结构填充。

设备根据相对 `/dev` 分层结构来分组。例如，全局区域中 `/dev` 下的所有设备都分组为全局区域设备。对于非全局区域，设备分组到区域的根路径下面的 `/dev` 目录中。每个组都是一个挂载的 `/dev` 文件系统实例，挂载在 `/dev` 目录下面。因此，全局区域设备挂载在 `/dev` 下面，而非全局区域 `my-zone` 的设备则挂载在 `/my-zone/root/dev` 下面。

`/dev` 文件分层结构由 dev(7FS) 手册页中所述的 `dev` 文件系统来管理。

注意 – 依赖于 `/devices` 路径名的子系统不能在非全局区域中运行。子系统必须更新才能使用 `/dev` 路径名。

注意 – 如果非全局区域具有的设备资源有一个包含 `/dev/zvol` 内的设备的匹配项，则在非全局区域内可能出现名称空间冲突。有关更多信息，请参见 dev(7FS) 手册页。

专用设备

可能拥有需要指定给特定区域的设备。允许非特权用户访问块设备可能会导致通过使用这些设备造成系统出现紧急情况、总线复位或其他不良影响。在进行此类指定之前，请考虑以下问题：

- 在为特定区域指定 SCSI 磁带设备之前，请查看 sgen(7D) 手册页。
- 将物理设备放入多个区域可以在区域之间创建隐蔽信道。使用此类设备的全局区域应用程序可能会存在非全局区域危及数据或损坏数据的风险。

设备驱动程序管理

在非全局区域中，可以使用 modinfo(1M) 手册页中所述的 `modinfo` 命令来检查已装入的内核模块的列表。

大多数与内核、设备和平台管理相关的操作都不能在非全局区域内部执行，因为修改平台硬件配置会破坏区域安全模型。这些操作包括：

- 添加和删除驱动程序
- 明确装入和卸载内核模块
- 启动动态重新配置 (dynamic reconfiguration, DR) 操作
- 使用影响物理平台状态的功能

在非全局区域中无法使用或者修改的实用程序

无法在非全局区域中使用的实用程序

以下实用程序无法在区域中使用，因为它们所依赖的设备通常不存在：

- `add_drv`（请参见 `add_drv(1M)` 手册页）
- `disks`（请参见 `disks(1M)` 手册页）
- `prtconf`（请参见 `prtconf(1M)` 手册页）
- `prtdiag`（请参见 `prtdiag(1M)` 手册页）
- `rem_drv`（请参见 `rem_drv(1M)` 手册页）

SPARC: 修改为可在非全局区域中使用的实用程序

`eeprom` 实用程序可用于查看区域中的设置，但不能用于更改设置。有关更多信息，请参见 `eeprom(1M)` 和 `openprom(7D)` 手册页。

允许具有安全含义的实用程序

如果启用了 `allowed-raw-io`，则可在区域中使用以下实用程序。注意，必须对安全注意事项进行评估。添加设备前，请参见第 324 页中的 “非全局区域中的设备使用”、第 325 页中的 “在非全局区域中运行应用程序”和第 327 页中的 “非全局区域中的特权”，以了解限制和有关安全的注意事项。

- `cdrecord`（请参见 `cderecord(1)` 手册页）。
- `cdrw`（请参见 `cdrw(1)`手册页）。
- `rmformat`（请参见 `rmformat(1)`手册页）。

在非全局区域中运行应用程序

通常，所有应用程序均可在非全局区域中运行。但是，以下应用程序类型可能不适用于此环境：

- 使用影响系统整体的特权操作的应用程序。例如设置全局系统时钟或锁定物理内存的操作。
- 依赖于非全局区域中不存在的某些设备的极少数应用程序，例如 `/dev/kmem`。

- 在共享 IP 区域中，应用程序依赖于 /dev/ip 中的设备。

在非全局区域中使用的资源控制

有关在区域中使用资源管理功能的其他信息，另请参阅第 1 部分中介绍此功能的章节。

资源管理章节中所述的任何资源控制和属性都可以在全局和非全局区域 /etc/project 文件、NIS 映射或 LDAP 目录服务中设置。给定区域的设置仅影响此区域。在不同区域中独立运行的项目可以在每个区域中分别设置控制。例如，项目 A 在全局区域中可以设置 project.cpu-shares=10，而在非全局区域中可以设置 project.cpu-shares=5。系统中可能同时运行若干个 rcapd 实例，而每个实例都仅在自己的区域中运行。

某个区域中用于在该区域中控制项目、任务和进程的资源控制和属性还要满足其他与池和区域范围资源控制相关的要求。

非全局区域可以与一个资源池关联，虽然不需要将该池专门指定给特定区域。多个非全局区域可以共享一个池的资源。但是，全局区域中的进程可以由拥有足够特权的进程绑定到任意池。资源控制器 poold 仅在全局区域中运行，它可以在其中的多个池中运行。在非全局区域中运行的 poolstat 实用程序仅显示与该区域关联的池的相关信息。在非全局区域中运行的不带参数的 pooladm 命令仅显示与此区域关联的池的相关信息。

区域范围的资源控制在 project 文件中设置时不会生效。区域范围的资源控制通过 zonecfg 实用程序设置。

安装了区域的系统上的公平份额调度器

本节介绍如何在区域中使用公平份额调度器 (fair share scheduler, FSS)。

全局或非全局区域中的 FSS 份额分配

区域的 FSS CPU 份额是分层的。全局和非全局区域的份额由全局管理员通过区域范围的资源控制 zone.cpu-shares 设置。然后，可以为该区域中的每个项目定义 project.cpu-shares 资源控制，以便进一步细分通过区域范围的控制设置的份额。

要使用 zonecfg 命令指定区域份额，请参见第 245 页中的 “如何在全局区域中设置 zone.cpu-shares”。有关 project.cpu-shares 的更多信息，请参见第 73 页中的 “可用的资源控制”。有关说明如何设置临时份额的示例过程，另请参见第 356 页中的 “在安装了区域的 Oracle Solaris 系统上使用公平份额调度器”。

区域之间的份额平衡

在全局区域和非全局区域中，可以使用 zone.cpu-shares 指定 FSS 份额。如果 FSS 是您系统中的缺省调度程序，并且尚未指定任何份额，则缺省情况下，会分配给每个区域一个份额。如果系统上有一个非全局区域，则将通过 zone.cpu-shares（定义非全局区域将相对于全局区域接到的 CPU 比例）为此区域提供两个份额。这两个区域之间的 CPU 比例为 2:1。

安装了区域的系统上的扩展记帐

当扩展记帐子系统在全局区域中运行时，它会收集和报告整个系统（包括非全局区域）的信息。全局管理员还可以确定每个区域的资源占用情况。

扩展记帐子系统允许每个区域针对基于进程和基于任务的记帐具有不同的记帐设置和文件。对于进程，exacct 记录可以使用区域名称 EXD PROC ZONENAME 进行标记；对于任务，则可以使用区域名称 EXD TASK ZONENAME 进行标记。记帐记录将写入全局区域的记帐文件以及每个区域的记帐文件。EXD TASK HOSTNAME、EXD PROC HOSTNAME 和 EXD HOSTNAME 记录包含用于执行进程或任务的区域的 uname -n 值，而不是全局区域的节点名称。

有关 IPQoS 流记帐的信息，请参见《在 Oracle Solaris 11.1 中管理 IP 服务质量》中的第 5 章 “使用流记帐和统计信息收集功能（任务）”。

非全局区域中的特权

仅允许进程拥有部分特权。特权限制可防止某个区域执行可能会影响其他区域的操作。通过特权设置，可以限制区域内特权用户的功能。要显示指定区域内可用特权的列表，请使用 ppriv 实用程序。

下表列出了所有 Oracle Solaris 特权以及每个特权相对于区域的状态。缺省特权集不包含可选的特权，但可以通过 limitpriv 属性指定它们。最终的特权集中必须包含必需特权。最终的特权集中不能包含禁止特权。

表 25–1 区域中特权的状态

特权	状态	附注
cpc_cpu	可选的	访问某些 cpc(3CPC) 计数器的权限
dtrace_proc	可选的	fasttrap 和 pid 提供器；plockstat(1M)
dtrace_user	可选的	profile 和 syscall 提供器
graphics_access	可选的	访问 agpgart_io(7I) 的 ioctl(2)

表 25–1 区域中特权的状态 （续）

特权	状态	附注
graphics_map	可选的	访问 agpgart_io(7I) 的 mmap(2)
net_rawaccess	在共享 IP 区域中为可选的。 在专用 IP 区域中为缺省值。	原始 PF_INET/PF_INET6 包访问权限
proc_clock_highres	可选的	使用高精度计时器
proc_priocntl	可选的	调度控制；priocntl(1)
sys_ipc_config	可选的	增加 IPC 消息队列缓冲区大小
sys_time	可选的	系统时间处理；xntp(1M)
dtrace_kernel	禁止	当前不支持
proc_zone	禁止	当前不支持
sys_config	禁止	当前不支持
sys_devices	禁止	当前不支持
sys_dl_config	禁止	当前不支持
sys_linkdir	禁止	当前不支持
sys_net_config	禁止	当前不支持
sys_res_config	禁止	当前不支持
sys_smb	禁止	当前不支持
sys_suser_compat	禁止	当前不支持
proc_exec	必需，缺省	用于启动 init(1M)
proc_fork	必需，缺省	用于启动 init(1M)
sys_mount	必需，缺省	需要用于挂载必需的文件系统
sys_flow_config	在专用 IP 区域中为必需、缺省权限。 在共享 IP 区域中为禁止权限。	配置流时需要
sys_ip_config	在专用 IP 区域中为必需、缺省权限。 在共享 IP 区域中为禁止权限。	在专用 IP 区域中需要用于引导和初始化 IP 联网

表 25–1 区域中特权的状态 （续）

特权	状态	附注
sys_iptun_config	在专用 IP 区域中为必需、缺省权限。 在共享 IP 区域中为禁止权限。	配置 IP 隧道链路
contract_event	缺省值	供合约文件系统使用
contract_identity	缺省值	设置进程合同模板的服务 FMRI 值
contract_observer	缺省值	合约调查，不考虑 UID
file_chown	缺省值	文件所有权更改
file_chown_self	缺省值	拥有文件的所有者/组更改
file_dac_execute	缺省值	执行访问权限，不考虑模式/ACL
file_dac_read	缺省值	读取访问权限，不考虑模式/ACL
file_dac_search	缺省值	搜索访问权限，不考虑模式/ACL
file_dac_write	缺省值	写入访问权限，不考虑模式/ACL
file_link_any	缺省值	链接访问权限，不考虑所有者
file_owner	缺省值	其他访问权限，不考虑所有者
file_setid	缺省值	更改 setid、setgid 和 setuid 文件的权限
ipc_dac_read	缺省值	IPC 读取访问权限，不考虑模式
ipc_dac_owner	缺省值	IPC 写入访问权限，不考虑模式
ipc_owner	缺省值	IPC 其他访问权限，不考虑模式
net_icmpaccess	缺省值	ICMP 包访问权限：ping(1M)
net_privaddr	缺省值	绑定到特权端口
proc_audit	缺省值	生成审计记录
proc_chroot	缺省值	更改 root 目录
proc_info	缺省值	检查进程
proc_lock_memory	缺省值	锁定内存；shmctl(2) 和 mlock(3C) 如果系统管理员要将此特权指定给非全局区域，请同时考虑设置 zone.max-locked-memory 资源控制以防止区域锁定所有内存。
proc_owner	缺省值	控制进程，不考虑所有者

表 25–1 区域中特权的状态 （续）

特权	状态	附注
proc_session	缺省值	控制进程，不考虑会话
proc_setid	缺省值	任意设置用户/组 ID
proc_taskid	缺省值	将任务 ID 指定给调用者
sys_acct	缺省值	记帐管理
sys_admin	缺省值	简单的系统管理任务
sys_audit	缺省值	审计管理
sys_nfs	缺省值	NFS 客户端支持
sys_ppp_config	在专用 IP 区域中为缺省权限 在共享 IP 区域中为禁止权限	创建和销毁 PPP (sppp) 接口，配置 PPP 隧道 (sppptun)
sys_resource	缺省值	资源限制处理
sys_share	缺省值	允许共享文件系统所需的 sharefs 系统调用。可以在区域配置中禁止特权，以防止在区域内共享 NFS。

下表列出了区域中所有 Oracle Solaris Trusted Extensions（高可靠扩展版）特权，以及相对于区域每个特权的状态。缺省特权集不包含可选的特权，但可以通过 limitpriv 属性指定它们。

注 – 仅当使用 Oracle Trusted Extensions（高可靠扩展版）配置了系统时，才会解释 Oracle Trusted Solaris 特权。

表 25–2 区域中 Oracle Solaris Trusted Extensions（高可靠扩展版）特权的状态

Oracle Solaris Trusted Extensions（高可靠扩展版）特权	状态	附注
file_downgrade_sl	可选的	将文件或目录的敏感标签设置为不影响现有敏感标签的敏感标签
file_upgrade_sl	可选的	将文件或目录的敏感标签设置为影响现有敏感标签的敏感标签
sys_trans_label	可选的	转换优先级低于敏感标签的标签
win_colormap	可选的	颜色映射限制覆盖
win_config	可选的	配置或销毁 X 服务器永久保留的资源

表 25–2 区域中 Oracle Solaris Trusted Extensions（高可靠扩展版）特权的状态 （续）

Oracle Solaris Trusted Extensions（高可靠扩展版）特权	状态	附注
win_dac_read	可选的	从非客户机用户 ID 拥有的窗口资源中进行读取
win_dac_write	可选的	写入或创建非客户机用户 ID 拥有的窗口资源
win_devices	可选的	在输入设备上执行操作。
win_dga	可选的	使用直接图形访问 X 协议扩展；需要帧缓存器特权
win_downgrade_sl	可选的	将窗口资源的敏感标签更改为优先级低于现有标签的新标签
win_fontpath	可选的	添加其他字体路径
win_mac_read	可选的	从其标签优先级高于客户机标签的窗口资源中进行读取
win_mac_write	可选的	写入其标签优先级与客户机标签优先级不同的窗口资源
win_selection	可选的	请求移动数据，而无需确认者介入
win_upgrade_sl	可选的	将窗口资源的敏感标签更改为优先级不低于现有标签的新标签
net_bindmlp	缺省值	允许绑定到多级别端口 (MLP)
net_mac_aware	缺省值	允许通过 NFS 向下读取

要在配置非全局区域过程中更改特权，请参见第 231 页中的 “配置、检验并提交区域”。

要检查特权集，请参见第 341 页中的 “使用 ppriv 实用程序”。有关特权的更多信息，请参见 ppriv(1) 手册页和《系统管理指南：安全性服务》。

在区域中使用 IP 安全体系结构

可提供 IP 数据报保护的 Internet 协议安全体系结构 (Internet Protocol Security Architecture, IPsec) 在《在 Oracle Solaris 11.1 中保护网络安全》中的第 8 章 “IP 安全体系结构（参考信息）”中进行了介绍。Internet 密钥交换 (Internet Key Exchange, IKE) 协议用于自动管理进行验证和加密所需的加密材料。

有关更多信息，请参见 ipsecconf(1M) 和 ipseckey(1M) 手册页。

共享 IP 区域中的 IP 安全体系结构

IPsec 可以在全局区域中使用。但是，非全局区域中的 IPsec 不能使用 IKE。因此，您必须在全局区域中使用 Internet 密钥交换 (Internet Key Exchange, IKE) 协议来为非全局区域管理 IPsec 密钥和策略。请使用对应于要配置的非全局区域的源地址。

专用 IP 区域中的 IP 安全体系结构

IPsec 可以在专用 IP 区域中使用。

在区域中使用 Oracle Solaris 审计

审计记录对事件进行描述，如登录到系统或写入文件的事件。Oracle Solaris 审计在运行区域的系统中提供了以下两种审计模式：

- 所有区域都从全局区域审计。当所有区域都由全局区域来管理（例如为通过区域实现服务隔离）时，使用该模式。
- 每个区域都独立于全局区域审计。当每个区域都单独进行管理（例如为实现按区域整合服务器）时，使用该模式。

Oracle Solaris 审计在《Oracle Solaris 11.1 管理：安全服务》中的第 26 章 “审计（概述）”中进行了介绍。有关与审计相关的区域注意事项，请参见《Oracle Solaris 11.1 管理：安全服务》中的“在具有 Oracle Solaris 区域的系统上审计”和《Oracle Solaris 11.1 管理：安全服务》中的“在区域中配置审计服务（任务）”。有关更多信息，另请参见 `auditconfig(1M)`、`auditreduce(1M)`、`usermod(1M)` 和 `user_attr(4)` 手册页。

注 – 还可以使用临时激活但未在系统信息库中设置的审计策略。

有关其他信息，请参见《Oracle Solaris 11.1 管理：安全服务》中的“如何更改审计策略”后面的示例。

区域中的核心文件

`coreadm` 命令用于指定因异常终止进程而生成的核心文件的名称和位置。通过指定 `%z` 变量，可以生成核心文件路径，此路径包括执行进程的区域的 *zonename*。路径名相对于区域的根目录。

有关更多信息，请参见 `coreadm(1M)` 和 `core(4)` 手册页。

在非全局区域中运行 DTrace

只需要 dtrace_proc 和 dtrace_user 特权的 DTrace 程序可以在非全局区域中运行。要将这些特权添加到非全局区域中的可用特权的集合中，请使用 zonecfg limitpriv 属性。有关说明，请参见第 347 页中的 “如何使用 DTrace”。

通过 dtrace_proc 支持的提供器是 fasttrap 和 pid。通过 dtrace_user 支持的提供器是 profile 和 syscall。DTrace 提供器和操作的范围限制在区域内。

有关更多信息，请参见第 327 页中的 “非全局区域中的特权”。

关于备份安装了区域的 Oracle Solaris 系统

可以在单个非全局区域中执行备份，也可以在全局区域中备份整个系统。

备份回送文件系统目录

不要从非全局区域内备份回送文件系统 (lofs)。

如果从非全局区域内备份并恢复 read/write 回送文件系统，应注意，这些文件系统还可以从全局区域以及以 read/write 方式挂载这些文件系统的任何其他区域写入。仅从全局区域备份及恢复这些文件系统，以避免多次复制。

在全局区域中备份系统

在以下情况下，可能会选择在全局区域中执行备份：

- 需要备份非全局区域的配置以及应用程序数据。
- 主要关注从灾难中恢复的功能。如果需要恢复系统上的所有内容或者绝大部分内容（包括区域的根文件系统及其配置和全局区域中的数据），则应在全局区域中执行备份。
- 有商业网络备份软件。

注 – 如有可能，网络备份软件应配置为跳过所有继承的 lofs 文件系统。应在区域及其应用程序处于静态时对要备份的数据执行备份。

在系统上备份单个非全局区域

在以下情况下，可能会决定在非全局区域内执行备份。

- 非全局区域管理员要求可以从不太严重的故障中恢复，或者恢复特定于某区域的应用程序数据或用户数据。
- 要使用按文件备份的程序，例如 `tar` 或 `cpio`。请参见 `tar(1)` 和 `cpio(1)` 手册页。
- 使用区域中运行的特定应用程序或服务的备份软件。可能很难在全局区域中执行备份软件，因为全局区域和非全局区域中的应用程序环境（例如目录路径和已安装的软件）不同。

  如果应用程序可以按照自己的备份计划在每个非全局区域中执行快照，并将这些备份存储在从全局区域导出的可写目录中，则作为备份策略的一部分，全局区域管理员可以从全局区域中选取这些单个备份。

创建 Oracle Solaris ZFS 备份

ZFS `send` 命令创建写入标准输出中的 ZFS 快照的流表示。缺省情况下，生成完整的流。可以将输出重定向到文件或其他系统。ZFS `receive` 命令创建其内容在标准输入所提供的流中指定的快照。如果接收了完整的流，那么同时会创建一个新文件系统。可通过这些命令来发送 ZFS 快照数据并接收 ZFS 快照数据和文件系统。

除了 ZFS `send` 和 `receive` 命令外，您还可以使用归档实用程序（如 `tar` 和 `cpio` 命令）来保存 ZFS 文件。这些实用程序可以保存和恢复 ZFS 文件属性和访问控制列表 (access control list, ACL)。在手册页中检查 `tar` 和 `cpio` 命令的相应选项。

相关信息及示例，请参见《Oracle Solaris 11.1 管理：ZFS 文件系统》中的第 6 章 “使用 Oracle Solaris ZFS 快照和克隆”。

确定在非全局区域中备份的内容

可以在非全局区域中备份所有内容，或者，如果区域的配置更改并不频繁，也可以仅对应用程序数据执行备份。

仅备份应用程序数据

如果应用程序数据保存在文件系统的特定部分，则可以决定仅对此数据执行常规备份。可以不必经常备份区域的根文件系统，因为其更改并不频繁。

必须确定应用程序放置其文件的位置。可以存储文件的位置如下：

- 用户的起始目录
- `/etc`（对于配置数据文件）
- `/var`

假设应用程序管理员知道数据的存储位置，则可以创建一个系统，其中每个区域均可使用其各自的可写目录。然后，每个区域可以存储自己的备份，而全局管理员或授予了相应权限的用户则可以将此位置作为系统上的备份位置之一。

常规数据库备份操作

如果数据库应用程序数据不在自己的目录下，则应用以下规则：

- 首先确保数据库处于一致的状态。

  数据库必须处于静态，因为它们具有要刷新到磁盘的内部缓冲区。请确保非全局区域中的数据库处于静态，然后从全局区域中开始备份。
- 在每个区域内，使用文件系统功能创建数据快照，然后直接从全局区域备份快照。

  此进程将最大程度缩短备份窗口所用的时间，并且不需要备份所有区域中的客户机/模块。

磁带备份

每个非全局区域都可以对自己的专用文件系统捕获快照，前提是此区域便于执行快照并且应用程序处于暂时静止状态。随后，全局区域可以备份每个快照，并在应用程序再次使用之后将备份放置在磁带上。

此方法具有如下优点：

- 需要较少的磁带设备。
- 不需要在非全局区域之间进行协调。
- 不需要直接为区域指定设备，从而提高了安全性。
- 通常，此方法保持在全局区域中执行系统管理，这是首选方法。

关于恢复非全局区域

如果恢复在全局区域中执行的备份，则全局管理员或授予了相应权限的用户可以重新安装受影响的区域，然后恢复这些区域的文件。请注意，上述情况以下面的假设为前提：

- 所要恢复的区域的配置与执行备份时的配置相同。
- 从备份完毕到恢复区域这段时间内，全局区域未更新。

否则，恢复操作可能会覆盖某些应手动合并的文件。

注 – 如果全局区域中的所有文件系统均已丢失，则只要备份非全局区域各自的根文件系统，恢复全局区域中的所有内容时也会恢复非全局区域。

在安装了区域的系统上使用的命令

表 25–3 中列出的命令提供了区域功能的主要管理接口。

表 25–3 用于管理和监视区域的命令

命令参考	说明
zlogin(1)	登录到非全局区域
zonename(1)	显示当前区域的名称
zonestat(1)	用于观察区域资源使用情况。
zoneadm(1M)	管理系统上的区域
zonecfg(1M)	用于设置区域配置
getzoneid(3C)	用于在区域 ID 和区域名称之间进行映射
zones(5)	提供区域功能的说明
zcons(7D)	区域控制台设备驱动程序

zoneadmd 守护进程是管理区域虚拟平台的主要进程。zoneadmd 守护进程的手册页为 zoneadmd(1M)。此守护进程并没有构成编程接口。

下表中的命令可与资源上限设置守护进程结合使用。

表 25–4 用于 rcapd 的命令

命令参考	说明
rcapstat(1)	监视具有上限的项目的资源利用率。
rcapadm(1M)	配置资源上限设置守护进程，显示已配置的资源上限设置守护进程的当前状态，以及启用或禁用资源上限设置。
rcapd(1M)	资源上限设置守护进程。

下表中介绍的命令已修改为可在安装了区域的 Oracle Solaris 系统上使用。这些命令具有的选项特定于区域或者以不同的方式显示信息。这些命令将在手册页中列出。

表 25–5 修改为可在安装了区域的 Oracle Solaris 系统上使用的命令

命令参考	说明
ipcrm(1)	添加了 -z *zone* 选项。只有在全局区域中执行命令时，此选项才有用。

表 25–5 修改为可在安装了区域的 Oracle Solaris 系统上使用的命令 （续）

命令参考	说明
ipcs(1)	添加了 -z *zone* 选项。只有在全局区域中执行命令时，此选项才有用。
pgrep(1)	添加了 -z *zoneidlist* 选项。只有在全局区域中执行命令时，此选项才有用。
ppriv(1)	添加了表达式 zone，以便与 -l 选项一起使用来列出当前区域中的所有可用特权。还可以在 zone 后使用选项 -v 来获取详细输出。
priocntl(1)	可以在 *idlist* 和 -i *idtype* 中使用区域 ID 来指定进程。在非全局区域中，可以使用 priocntl -i *zoneid* 命令将正在运行的进程移至其他调度类。
proc(1)	仅向 ptree 中添加了 -z *zone* 选项。只有在全局区域中执行命令时，此选项才有用。
ps(1)	向与 -o 选项一起使用的已识别 format 名称的列表中添加了 *zonename* 和 *zoneid*。 添加了 -z *zonelist* 以便仅列出指定区域中的进程。可以通过区域名称或区域 ID 指定区域。只有在全局区域中执行命令时，此选项才有用。 添加了 -Z 以便显示与进程关联的区域的名称。区域名称在另一个列标题 ZONE 下显示。
renice(1)	向与 -i 选项一起使用的有效参数的列表中添加了 *zoneid*。
sar(1)	如果在启用了池功能的非全局区域中执行，则 -b、-c、-g、-m、-p、-u、-w 和 -y 选项仅针对绑定有区域的池的处理器集中的处理器显示值。
auditconfig(1M)	添加了 zonename 标记。
auditreduce(1M)	添加了 -z *zone-name* 选项。新增了获取区域审计日志的功能。
coreadm(1M)	添加了变量 %z 以便标识执行进程的区域。
df(1M)	添加了 -Z 选项以便显示所有可见区域中的挂载。该选项在非全局区域中无效。
dladm(1M)	将 -Z 选项添加到 show 子命令中，从而在缺省命令输出中加入区域列。区域列指示资源当前所指定到的区域。
dlstat(1M)	将 -Z 选项添加到 show 子命令中，从而在缺省命令输出中加入区域列。区域列指示资源当前所指定到的区域。

表 25–5 修改为可在安装了区域的 Oracle Solaris 系统上使用的命令 （续）

命令参考	说明
fsstat(1M)	添加 -z 选项以报告每个区域的文件系统活动。多个 -z 选项可以用来监视选定区域的活动。如果仅用来监视 mountpoints 而不监视 fstypes，则该选项没有效果。 添加 -A 选项来报告所有区域中指定的 fstypes 的聚合文件系统活动。如果 -z 和 -Z 选项均未使用，则此为缺省行为。如果仅用来监视 mountpoints 而不监视 fstypes， -A 选项没有效果。 当与 -z 或 -Z 选项结合使用时， -A 选项在单独一行中显示所有区域的指定 fstypes 的聚合。 添加 -Z 选项以报告系统上的所有区域的文件系统活动。如果与 -z 选项一起使用，则该选项没有效果。如果仅用来监视 mountpoints 而不监视 fstypes，则该选项没有效果。
iostat(1M)	如果在启用了池功能的非全局区域中执行，则仅针对绑定有区域的池的处理器集中的那些处理器提供信息。
ipadm(1M)	配置 Internet 协议网络接口和 TCP/IP 可调参数。 from-gz 类型仅在非全局区域中显示，并且表示该地址是根据 allowed-address 属性配置的，而此属性是从全局区域为非全局专用 IP 区域配置的。 zone 地址属性指定应放置 allowed-address 所引用的全部地址的区域。该区域必须配置为共享 IP 区域。
kstat(1M)	如果在全局区域中执行，将针对所有区域显示 kstat。如果在非全局区域中执行，则只显示具有匹配 *zoneid* 的 kstat。
mpstat(1M)	如果在启用了池功能的非全局区域中执行，则命令仅针对绑定有区域的池的处理器集中的处理器显示行。
ndd(1M)	在全局区域中使用时，会显示所有区域的信息。在专用 IP 区域中，对 TCP/IP 模块执行的 ndd 只显示该区域的信息。
netstat(1M)	仅显示当前区域的信息。
nfsstat(1M)	仅显示当前区域的统计信息。
poolbind(1M)	添加了 *zoneid* 列表。有关使用具有资源池的区域的信息，另请参见第 129 页中的 “区域中使用的资源池”。
prstat(1M)	添加了 -z *zoneidlist* 选项。还添加了 -Z 选项。 如果在启用了池功能的非全局区域中执行，则仅针对绑定有区域的池的处理器集中的处理器显示进程所用最近 CPU 时间的百分比。 -a 、 -t 、 -T 、 -J 和 -Z 选项的输出显示 SWAP，而不是 SIZE 列。报告的交换是区域进程和 tmpfs 挂载所使用的总交换量。此值有助于监视每个区域预留的交换空间，可用于选择合理的 zone.max-swap 设置。
psrinfo(1M)	如果在非全局区域中执行，则仅显示有关区域可见的处理器的信息。

表 25–5 修改为可在安装了区域的 Oracle Solaris 系统上使用的命令 （续）

命令参考	说明
traceroute(1M)	用法更改。在非全局区域中指定时，-F 选项不起作用，因为始终设置了“不要分段”位。
vmstat(1M)	在启用了池功能的非全局区域中执行时，仅针对绑定有区域的池的处理器集中的处理器报告统计信息。应用于 -p 选项以及 page、faults 和 cpu 等报告字段的输出。
priocntl(2)	添加了 P_ZONEID *id* 参数。
processor_info(2)	如果调用者位于非全局区域中并且启用了池功能，但是处理器不在绑定有区域的池的处理器集中，则会返回错误。
p_online(2)	如果调用者位于非全局区域中并且启用了池功能，但是处理器不在绑定有区域的池的处理器集中，则会返回错误。
pset_bind(2)	添加了 P_ZONEID 作为 *idtype*。添加了区域作为可能的 P_MYID 规范选项。向 EINVAL 错误说明中的有效 *idtype* 列表中添加了 P_ZONEID。
pset_info(2)	如果调用者位于非全局区域中并且启用了池功能，但是处理器不在绑定有区域的池的处理器集中，则会返回错误。
pset_list(2)	如果调用者位于非全局区域中并且启用了池功能，但是处理器不在绑定有区域的池的处理器集中，则会返回错误。
pset_setattr(2)	如果调用者位于非全局区域中并且启用了池功能，但是处理器不在绑定有区域的池的处理器集中，则会返回错误。
sysinfo(2)	将 PRIV_SYS_CONFIG 更改为 PRIV_SYS_ADMIN。
umount(2)	如果 *file* 指向的文件不是绝对路径，则会返回 ENOENT。
getloadavg(3C)	如果调用者位于非全局区域中并且启用了池功能，则此行为相当于使用 PS_MYID 的 psetid 进行调用。
getpriority(3C)	向可以指定的目标进程中添加了区域 ID。向 EINVAL 错误说明中添加了区域 ID。
priv_str_to_set(3C)	针对调用者区域内的所有可用特权的集合添加了 "zone" 字符串。
pset_getloadavg(3C)	如果调用者位于非全局区域中并且启用了池功能，但是处理器不在绑定有区域的池的处理器集中，则会返回错误。
sysconf(3C)	如果调用者位于非全局区域中并且启用了池功能，则 sysconf(_SC_NPROCESSORS_CONF) 和 sysconf(_SC_NPROCESSORS_ONLN) 分别返回绑定了区域的池的处理器集中的处理器总数和联机处理器数。
ucred_get(3C)	添加了 ucred_getzoneid() 函数，此函数将返回处理器的区域 ID 或 -1（如果未提供区域 ID）。
core(4)	添加了 n_type: NT_ZONENAME。此项包含一个描述运行进程的区域名称的字符串。

表 25–5 修改为可在安装了区域的 Oracle Solaris 系统上使用的命令 （续）

命令参考	说明
pkginfo(4)	现在，提供了可选的参数和一个环境变量来支持区域。
proc(4)	添加了获取区域中所运行进程的相关信息的功能。
audit_syslog(5)	添加了在设置 zonename 审计策略时使用的 in<*zone name*> 字段。
privileges(5)	添加了 PRIV_PROC_ZONE，它允许某个进程跟踪其他区域中的进程或向这些进程发送信号。请参见 zones (5)。
if_tcp(7P)	添加了区域 ioctl() 调用。
cmn_err(9F)	添加了区域参数。
ddi_cred(9F)	添加了 crgetzoneid()，它将从 cr 指向的用户证书中返回区域 ID。

◆ ◆ ◆ 第 26 章

管理 Oracle Solaris Zones（任务）

本章介绍一般管理任务并提供用法示例。

有关常规区域管理主题，请参见第 25 章，Oracle Solaris Zones 管理（概述）。

使用 ppriv 实用程序

使用 ppriv 实用程序可以显示区域的特权。

▼ 如何列出全局区域中的 Oracle Solaris 特权

可使用 ppriv 实用程序的 -l 选项列出该系统中可用的特权。

- **在提示符下，键入 ppriv -l zone 报告区域中的一组可用特权。**

```
global# ppriv -l zone
```

将显示以下类似信息：

```
contract_event
contract_observer
cpc_cpu
.
.
.
```

▼ 如何列出非全局区域的特权集

可以使用带有 -l 选项和表达式 zone 的 ppriv 实用程序列出区域的特权。

1. **登录到非全局区域。此示例使用名为 *my-zone* 的区域。**
2. **在提示符下，键入 ppriv -l zone 报告区域中的一组可用特权。**

```
my-zone# ppriv -l zone
```

将显示以下类似信息：

```
contract_event
contract_identity
contract_observer
file_chown

.
.
.
```

▼ 如何列出带有详细输出的非全局区域的特权集

可以使用带有 -l 选项、表达式 zone，以及 -v 选项的 ppriv 实用程序列出区域的特权。

1. **登录到非全局区域。此示例使用名为 *my-zone* 的区域。**
2. **在提示符下，键入 ppriv -l -v zone 报告区域中的一组可用特权，同时给出每个特权的说明。**

```
my-zone# ppriv -lv zone
```

将显示以下类似信息：

```
contract_event
        Allows a process to request critical events without limitation.
        Allows a process to request reliable delivery of all events on
        any event queue.
contract_identity
        Allows a process to set the service FMRI value of a process
        contract template.
```

```
contract_observer
        Allows a process to observe contract events generated by
        contracts created and owned by users other than the process's
        effective user ID.
        Allows a process to open contract event endpoints belonging to
        contracts created and owned by users other than the process's
        effective user ID.
file_chown
        Allows a process to change a file's owner user ID.
        Allows a process to change a file's group ID to one other than
        the process' effective group ID or one of the process'
        supplemental group IDs.
.
.
.
```

在非全局区域中使用 zonestat 实用程序

zonestat 实用程序会报告当前正在运行的区域的 CPU、内存、网络和资源控制使用情况。后面提供了使用示例。

有关完整信息，请参见 zonestat(1)。

zonestat 网络组件按区域显示 PHYS、AGGR、Etherstub 和 SIMNET 数据链路上虚拟网络 (VNIC) 资源的使用状况。有关其他数据链路（如网桥和隧道）的信息可通过使用 dladm(1M) 和 dlstat(1M) 手册页中介绍的联网实用程序获得。

也可以在非全局区域内调用所有 zonestat 选项和资源类型，以显示该区域的统计信息。

```
root@zoneA:~# zonestat -z global -r physical-memory 2
```

注 – 当在非全局区域中使用 zonestat 时，与在全局区域中使用一样，将报告所有其他区域（包括全局区域）的综合资源使用情况。zonestat 的非全局区域用户并未发现有其他区域正在共享系统。

▼ 如何使用 zonestat 实用程序显示 CPU 和内存使用率摘要

1. **成为 root 用户或承担等效角色。**
2. **每 5 秒显示一次 CPU 和内存使用率摘要。**

```
# zonestat -z global -r physical-memory 5
Collecting data for first interval...
Interval: 1, Duration: 0:00:05
```

```
PHYSICAL-MEMORY              SYSTEM MEMORY
mem_default                          2046M
                                ZONE  USED %USED   CAP  %CAP
                             [total] 1020M 49.8%     -     -
                            [system]  782M 38.2%     -     -
                              global  185M 9.06%     -     -

Interval: 2, Duration: 0:00:10
PHYSICAL-MEMORY              SYSTEM MEMORY
mem_default                          2046M
                                ZONE  USED %USED   CAP  %CAP
                             [total] 1020M 49.8%     -     -
                            [system]  782M 38.2%     -     -
                              global  185M 9.06%     -     -
...
```

▼ 如何使用 zonestat 实用程序报告缺省 pset

1. 成为 root 用户或承担等效角色。
2. 每一秒报告一次缺省 pset，持续 1 分钟：

```
# zonestat -r default-pset 1 1m
Collecting data for first interval...
Interval: 1, Duration: 0:00:01
PROCESSOR_SET                    TYPE  ONLINE/CPUS     MIN/MAX
pset_default            default-pset          2/2         1/-
                                ZONE  USED   PCT   CAP  %CAP   SHRS  %SHR %SHRU
                             [total]  0.02 1.10%     -     -      -     -     -
                            [system]  0.00 0.19%     -     -      -     -     -
                              global  0.01 0.77%     -     -      -     -     -
                                zone1  0.00 0.07%     -     -      -     -     -
                                zone2  0.00 0.06%     -     -      -     -     -

...
Interval: 60, Duration: 0:01:00
PROCESSOR_SET                    TYPE  ONLINE/CPUS     MIN/MAX
pset_default            default-pset          2/2         1/-
                                ZONE  USED   PCT   CAP  %CAP   SHRS  %SHR %SHRU
                             [total]  0.06 3.26%     -     -      -     -     -
                            [system]  0.00 0.18%     -     -      -     -     -
                              global  0.05 2.94%     -     -      -     -     -
                                zone1  0.00 0.06%     -     -      -     -     -
                                zone2  0.00 0.06%     -     -      -     -     -
```

▼ 使用 zonestat 报告总使用率和最高使用率

1. 成为 root 用户或承担等效角色。

2 **按 10 秒间隔以无提示方式进行监视，持续 3 分钟，然后生成一份有关总使用率和最高使用率的报告。**

```
# zonestat -q -R total,high 10s 3m 3m
Report: Total Usage
    Start: Fri Aug 26 07:32:22 PDT 2011
      End: Fri Aug 26 07:35:22 PDT 2011
    Intervals: 18, Duration: 0:03:00
SUMMARY                   Cpus/Online: 2/2   PhysMem: 2046M  VirtMem: 3069M
                     ---CPU----  --PhysMem-- --VirtMem-- --PhysNet--
                ZONE  USED %PART  USED %USED  USED %USED PBYTE %PUSE
             [total]  0.01 0.62% 1020M 49.8% 1305M 42.5%    14 0.00%
            [system]  0.00 0.23%  782M 38.2% 1061M 34.5%     -     -
              global  0.00 0.38%  185M 9.06%  208M 6.77%     0 0.00%
               test2  0.00 0.00% 52.4M 2.56% 36.6M 1.19%     0 0.00%

Report: High Usage
    Start: Fri Aug 26 07:32:22 PDT 2011
      End: Fri Aug 26 07:35:22 PDT 2011
    Intervals: 18, Duration: 0:03:00
SUMMARY                   Cpus/Online: 2/2   PhysMem: 2046M  VirtMem: 3069M
                     ---CPU----  --PhysMem-- --VirtMem-- --PhysNet--
                ZONE  USED %PART  USED %USED  USED %USED PBYTE %PUSE
             [total]  0.01 0.82% 1020M 49.8% 1305M 42.5%  2063 0.00%
            [system]  0.00 0.26%  782M 38.2% 1061M 34.5%     -     -
              global  0.01 0.55%  185M 9.06%  207M 6.77%     0 0.00%
               test2  0.00 0.00% 52.4M 2.56% 36.6M 1.19%     0 0.00%
```

▼ 如何获得专用 IP 区域的网络带宽使用率

使用带 -r 选项和 network 资源类型的 zonestat 命令可显示在每个网络设备上每个区域的使用率。

使用此过程可显示每个区域使用了多少 VNIC（虚拟网络接口卡）形式的数据链路带宽。例如，显示在 e1000g0 之下的 zoneB 表明，此区域使用的 VNIC 形式资源为 e1000g0。还可以通过添加 -x 选项显示具体的 VNIC。

1 **成为 root 用户管理员。**

2 **在带有 -r 选项的 zonestat 命令中使用 network 资源类型，以一次性显示利用率。**

```
    # zonestat -r network 1 1
Collecting data for first interval...
Interval: 1, Duration: 0:00:01

    NETWORK-DEVICE                  SPEED        STATE        TYPE
    aggr1                        2000mbps           up        AGGR
                    ZONE TOBYTE  MAXBW %MAXBW PRBYTE %PRBYTE POBYTE %POBYTE
                  global  1196K      -      -   710K   0.28%   438K   0.18%

    e1000g0                      1000mbps           up        PHYS
                    ZONE TOBYTE  MAXBW %MAXBW PRBYTE %PRBYTE POBYTE %POBYTE
                 [total]  7672K      -      -  6112K   4.89%  1756K   1.40%
                  global  5344K  100m*  42.6%  2414K   1.93%  1616K   1.40%
```

```
                zoneB   992K   100m  15.8%  1336K   0.76%   140K   0.13%
                zoneA  1336K    50m  10.6%   950K   1.07%      0   0.00%

    e1000g1                    1000mbps           up        PHYS
                  ZONE TOBYTE  MAXBW %MAXBW PRBYTE %PRBYTE POBYTE %POBYTE
                global   126M      -      -    63M   6.30%    63M   6.30%

    etherstub1                      n/a          n/a   ETHERSTUB
                  ZONE TOBYTE  MAXBW %MAXBW PRBYTE %PRBYTE POBYTE %POBYTE
               [total]  3920K      -      -      0       -      0       -
                global  1960K  100M*  1.96%      0       -      0       -
                 zoneA  1960K    50M  3.92%      0       -      0       -
```

更多信息 **非全局区域中的示例命令**

在非全局区域中使用的命令：

```
root@zoneA:~# zonestat -r network -x 1 1
```

报告所有区域的每区域 fstype 统计信息

使用 -z 选项来报告每个区域的文件系统活动。多个 -z 选项可以用来监视选定区域的活动。

使用 -A 选项来报告所有区域中指定的 fstypes 的聚合文件系统活动。如果 -z 和 -Z 选项均未使用，则此为缺省行为。

当与 -z 或 -Z 选项结合使用时， -A 选项在单独一行中显示所有区域的指定 fstypes 的聚合。

使用 -Z 选项来报告系统上的所有区域的文件系统活动。如果与 -z 选项一起使用，则该选项没有效果。如果仅用来监视 mountpoints 而不监视 fstypes，则该选项没有效果。

▼ 如何使用 -z 选项来监视指定区域的活动。

- **使用多个 -z 选项来监视 *s10* 和 *s10u9* 区域中的活动。**

```
$ fsstat -z s10 -z s10u9 zfs tmpfs
 new  name   name  attr  attr lookup rddir  read read  write write
 file remov  chng   get   set    ops   ops   ops bytes   ops bytes
   93    82     6  163K   110   507K   148 69.7K 67.9M 4.62K 13.7M zfs:s10
  248   237   158  188K   101   612K   283 70.6K 68.6M 4.71K 15.2M zfs:s10u9
12.0K 1.90K 10.1K 35.4K    12  60.3K     4 25.7K 29.8M 36.6K 31.0M tmpfs:s10
12.0K 1.90K 10.1K 35.6K    14  60.2K     2 28.4K 32.1M 36.5K 30.9M tmpfs:S10u9
```

▼ 如何显示所有区域的每区域 fstype 统计信息

- **获取系统上运行的每个区域的 `tmpfs` 和 `zfs` 文件系统类型的每区域统计数据，并且显示 `tmpfs` 和 `zfs` 文件系统类型的系统范围聚合：**

```
$  fsstat -A -Z zfs tmpfs
 new  name   name  attr  attr lookup rddir  read read  write write
 file remov  chng   get   set    ops   ops   ops bytes   ops bytes
 360K 1.79K 20.2K 4.20M 1.02M  25.0M  145K 5.42M 2.00G 1.07M 8.10G zfs
 359K 1.48K 20.1K 4.04M 1.02M  24.5M  144K 5.31M 1.88G 1.06M 8.08G zfs:global
   93    82     6 74.8K   107   250K   144 54.8K 60.5M 4.61K 13.7M zfs:s10
  248   237   158 90.2K   101   336K   283 53.0K 58.3M 4.71K 15.2M zfs:s10u9
60.0K 41.9K 17.7K  410K   515   216K   426 1022K 1.02G  343K  330M tmpfs
49.4K 38.1K 11.0K  366K   489   172K   420  968K  979M  283K  273M tmpfs:global
5.28K 1.90K 3.36K 21.9K    12  21.7K     4 25.7K 29.8M 29.9K 28.3M tmpfs:s10
5.25K 1.90K 3.34K 22.1K    14  21.6K     2 28.4K 32.1M 29.8K 28.2M tmpfs:s10u9
```

在输出中，系统上的非全局区域是 `S10` 和 `S10u9`。

在非全局区域中使用 DTrace

执行以下步骤，以使用第 333 页中的 “在非全局区域中运行 DTrace”中所述的 DTrace 功能。

▼ 如何使用 DTrace

1. **使用 `zonecfg limitpriv` 属性添加 `dtrace_proc` 和 `dtrace_user` 特权。**

```
global# zonecfg -z my-zone
zonecfg:my-zone> set limitpriv="default,dtrace_proc,dtrace_user"
zonecfg:my-zone> exit
```

注 – 可以根据需要添加其中一个特权或同时添加这两个特权。

---

2. **引导区域。**

```
global# zoneadm -z my-zone boot
```

3. **登录到区域。**

```
global# zlogin my-zone
```

4. **运行 DTrace 程序。**

```
my-zone# dtrace -l
```

检查非全局区域中的 SMF 服务的状态

要检查非全局区域中的 SMF 服务的状态，请使用 `zlogin` 命令。

▼ 如何从命令行检查 SMF 服务的状态

1. **成为 root 用户或承担等效角色。**
2. **在命令行中键入以下内容，以显示所有服务，包括禁用的服务。**

```
global# zlogin my-zone svcs -a
```

另请参见 有关更多信息，请参见第 21 章，登录到非全局区域（任务）和 svcs(1)。

▼ 如何从区域内检查 SMF 服务的状态

1. **成为 root 用户或承担等效角色。**
2. **登录到区域。**

```
global# zlogin my-zone
```

3. **运行带有 -a 选项的 svcs 命令，以显示所有服务，包括禁用的服务。**

```
my-zone# svcs -a
```

另请参见 有关更多信息，请参见第 21 章，登录到非全局区域（任务）和 svcs(1)。

在正在运行的非全局区域中挂载文件系统

可以在正在运行的非全局区域中挂载文件系统。包括以下过程。

- 作为全局管理员或在全局区域中拥有相应授权的用户，您可以将原始和块设备导入到非全局区域。导入设备之后，区域管理员便可访问磁盘。然后，区域管理员可以在磁盘上创建一个新的文件系统，并执行以下操作之一：
  - 手动挂载文件系统
  - 将文件系统放在 `/etc/vfstab` 中，以便在引导区域时挂载
- 作为全局管理员或拥有相应授权的用户，您还可以将文件系统从全局区域挂载到非全局区域。

  在将文件系统从全局区域挂载到非全局区域之前，请注意非全局区域应处于就绪状态，或已经引导。否则，接下来使区域处于就绪状态或引导区域的操作将失败。另外，任何从全局区域挂载到非全局区域的文件系统在区域停止时会取消挂载。

▼ 如何使用 LOFS 挂载文件系统

可以通过使用 LOFS 挂载在全局区域和非全局区域之间共享文件系统。此过程使用 `zonecfg` 命令将全局区域 `/export/datafiles` 文件系统的 LOFS 挂载添加到 `my-zone` 配置中。此示例没有定制挂载选项。

您必须是全局管理员，或在全局区域中具有区域安全权限配置文件的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**
2. **使用 `zonecfg` 命令。**

   ```
   global# zonecfg -z my-zone
   ```

3. **将文件系统添加到配置中。**

   ```
   zonecfg:my-zone> add fs
   ```

4. **为文件系统设置挂载点，在 `my-zone` 中为 `/datafiles`。**

   ```
   zonecfg:my-zone:fs> set dir=/datafiles
   ```

5. **指定全局区域中的 `/export/datafiles` 在 `my-zone` 中挂载为 `/datafiles`。**

   ```
   zonecfg:my-zone:fs> set special=/export/datafiles
   ```

6. **设置文件系统类型。**

   ```
   zonecfg:my-zone:fs> set type=lofs
   ```

7. **结束指定。**

   ```
   zonecfg:my-zone:fs> end
   ```

8. **检验并提交配置。**

   ```
   zonecfg:my-zone> verify
   zonecfg:my-zone> commit
   ```

更多信息 临时挂载

可从全局区域添加 LOFS 文件系统挂载，而不重新引导非全局区域：

```
global# mount -F lofs /export/datafiles /export/my-zone/root/datafiles
```

要在每次引导区域时都执行此挂载，必须使用 `zonecfg` 命令修改区域的配置。

▼ 如何将 ZFS 数据集委托到非全局区域

请使用以下过程将 ZFS 数据集委托到非全局区域。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1 **成为 root 用户或承担等效角色。**

2 **在全局区域中，在名为 poolA 的现有 ZFS 池上新建一个名为 fs2 的 ZFS 文件系统：**

```
global# zfs create poolA/fs2
```

3 **（可选的）将 poolA/fs2 文件系统的 mountpoint 属性设置为 /fs-del/fs2。**

```
global# zfs set mountpoint=/fs-del/fs2 poolA/fs2
```

不要求设置 mountpoint。如果未指定 mountpoint 属性，缺省情况下将在区域内的 /alias 挂载数据集。为 mountpoint 和 canmount 属性指定非缺省值可改变此行为，如 zfs (1M) 手册页中所述。

4 **检验对于此文件系统，mountpoint 属性的源现在为 local。**

```
global# zfs get mountpoint poolA/fs2
NAME       PROPERTY    VALUE        SOURCE
poolA/fs2  mountpoint  /fs-del/fs2  local
```

5 **委托 poolA/fs2 文件系统或指定一个有别名的数据集：**

- **将 poolA/fs2 文件系统委托到区域：**

```
# zonecfg -z my-zone
zonecfg:my-zone> add dataset
zonecfg:my-zone:dataset> set name=poolA/fs2
zonecfg:my-zone:dataset> end
```

- **指定一个有别名的数据集：**

```
# zonecfg -z my-zone
zonecfg:my-zone> add dataset
zonecfg:my-zone:dataset> set name=poolA/fs2
zonecfg:my-zone:dataset> set alias=delegated
zonecfg:my-zone:dataset> end
```

6 **重新引导区域并显示所有 poolA 文件系统的 zoned 属性：**

```
global# zfs get -r zoned poolA
NAME       PROPERTY  VALUE   SOURCE
poolA      zoned     off     default
poolA/fs2 zoned      on      default
```

请注意，poolA/fs2 的 zoned 属性已设置为 on。此 ZFS 文件系统已委托到非全局区域，挂载在区域中，并在区域管理员的控制之下。ZFS 使用 zoned 属性来指示已在某一时刻将数据集委托给非全局区域。

在全局区域中添加非全局区域对特定文件系统的访问权限

▼ 如何在非全局区域中添加对 CD 或 DVD 介质的访问权限

借助此过程，您可以在非全局区域中添加对 CD 或 DVD 介质的只读访问权限。在全局区域中，使用 Volume Management 文件系统来挂载介质。然后可以使用 CD 或 DVD 在非全局区域中安装产品。此过程使用名为 `jes_05q4_dvd` 的 DVD。

1 **成为 root 用户或承担等效角色。**

2 **确定 Volume Management 文件系统是否正在全局区域中运行。**

```
global# svcs rmvolmgr
STATE          STIME    FMRI
online         Sep_29   svc:/system/filesystem/volfs:default
```

3 **（可选的）如果 Volume Management 文件系统没有在全局区域中运行，则启动它。**

```
global# svcadm rmvolmgr enable
```

4 **插入介质。**

5 **检查驱动器中的介质。**

```
global# volcheck
```

6 **测试 DVD 是否自动挂载。**

```
global# ls /media
```

将显示以下类似信息：

```
cdrom   cdrom1   jes_05q4_dvd
```

7 **在非全局区域中使用选项 `ro,nodevices`（只读并且无设备）来回送挂载文件系统。**

```
global# zonecfg -z my-zone
zonecfg:my-zone> add fs
zonecfg:my-zone:fs> set dir=/cdrom
zonecfg:my-zone:fs> set special=/cdrom
zonecfg:my-zone:fs> set type=lofs
zonecfg:my-zone:fs> add options [ro,nodevices]
zonecfg:my-zone:fs> end
zonecfg:my-zone> commit
zonecfg:my-zone> exit
```

8 **重新引导非全局区域。**

```
global# zoneadm -z my-zone reboot
```

9 **使用带有 -v 选项的 zoneadm list 命令来检验状态。**

```
global# zoneadm list -v
```

将显示以下类似信息：

```
ID  NAME     STATUS       PATH                           BRAND      IP
 0  global   running      /                              solaris    shared
 1  my-zone  running      /zones/my-zone                 solaris    excl
```

10 **登录到非全局区域。**

```
global#  my-zone
```

11 **检验 DVD-ROM 挂载。**

```
my-zone# ls /cdrom
```

将显示以下类似信息：

```
cdrom   cdrom1   jes_05q4_dvd
```

12 **按照产品安装指南中的介绍安装产品。**

13 **退出非全局区域。**

```
my-zone# exit
```

提示 – 您可能需要在非全局区域中保留 /cdrom 文件系统。挂载始终反映 CD-ROM 驱动器的当前内容，如果驱动器为空，则反映为一个空目录。

---

14 **（可选的）如果想要从非全局区域中删除 /cdrom 文件系统，请使用以下过程。**

```
global# zonecfg -z my-zone
zonecfg:my-zone> remove fs dir=/cdrom
zonecfg:my-zone> commit
zonecfg:my-zone> exit
```

在安装了区域的 Oracle Solaris 系统上使用 IP 网络多路径

▼ 如何在专用 IP 非全局区域中使用 IP 网络多路径

可以按在全局区域中的配置方式在专用 IP 区域中配置 IP 网络多路径 (IP Network Multipathing, IPMP)。要使用 IPMP，专用 IP 区域至少需要两个 zonecfg add net 资源。从区域内在这些数据链路上配置 IPMP。

您可以将一个或多个物理接口配置到一个 IP 多路径组或 IPMP 组中。配置 IPMP 后，系统将自动监视 IPMP 组中的接口是否出现故障。如果该组中的接口出现故障或被删除以进行维护，则 IPMP 会自动迁移或故障转移故障接口的 IP 地址。这些地址的接收者是

故障接口的 IPMP 组中的工作接口。IPMP 的故障转移组件可保持连通性，防止任何现有连接发生中断。此外，通过自动在 IPMP 组中的一组接口中分配网络通信流量，IPMP 提高了总体网络性能。此过程称作负荷分配。

1 **成为 root 用户或承担等效角色。**

2 **配置 IPMP 组，相关操作在《 管理 Oracle Solaris 11.1 网络性能 》中的“配置 IPMP 组”中有介绍。**

▼ 如何将 IP 网络多路径功能扩展到共享 IP 非全局区域

使用此过程可以在全局区域中配置 IPMP，并将 IPMP 功能扩展到非全局区域。

当您配置区域时，每个地址或逻辑接口都应当与非全局区域相关联。有关说明，请参见第 207 页中的 “使用 zonecfg 命令”和第 232 页中的 “如何配置区域”。

此过程将实现以下内容：

- 同时在一个组中配置 `bge0` 卡和 `hme0` 卡。
- 地址 192.168.0.1 与非全局区域 *my-zone* 相关联。
- `bge0` 卡设置为物理接口。这样，IP 地址驻留在包含 `bge0` 卡和 `hme0` 卡的组中。

在正在运行的区域中，可以使用 `ipadm` 命令来建立关联。有关更多信息，请参见第 321 页中的 “共享 IP 网络接口”和 ipadm(1M) 手册页。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1 **成为 root 用户或承担等效角色。**

2 **在全局区域中配置 IPMP 组，相关操作在《 管理 Oracle Solaris 11.1 网络性能 》中的“配置 IPMP 组”中有介绍。**

3 **使用 `zonecfg` 命令配置区域。当您配置 `net` 资源时，请将地址 `192.168.0.1` 和物理接口 `bge0` 添加到区域 *my-zone*：**

```
zonecfg:my-zone> add net
zonecfg:my-zone:net> set address=192.168.0.1
zonecfg:my-zone:net> set physical=bge0
zonecfg:my-zone:net> end
```

在非全局区域 *my-zone* 中只有 `bge0` 可见。

更多信息 如果 bge0 随后出现故障

如果 `bge0` 随后出现故障，并且 `bge0` 数据地址故障转移到全局区域中的 `hme0`，*my-zone* 地址也会迁移。

如果地址 192.168.0.1 移至 hme0，此时在非全局区域 *my-zone* 中只有 hme0 可见。该卡将与地址 192.168.0.1 相关联，并且 bge0 将不再可见。

在独占 IP 非全局区域中管理数据链路

在全局区域中可使用 dladm 命令管理数据链路。

▼ 如何使用 dladm show-linkprop

可以将 dladm 命令与 show-linkprop 子命令一起使用，以显示正在运行的专用 IP 区域的数据链路分配。

您必须是全局管理员或在全局区域中具有相应授权的用户才能管理数据链路。

1 成为 root 用户或承担等效角色。

2 显示系统中数据链路的分配。

```
global# dladm show-linkprop
```

示例 26–1 将 dladm 与 show-linkprop 子命令一起使用

1. 在第一个屏幕中，没有引导指定了 bge0 的区域 49bge。

```
global# dladm show-linkprop
LINK     PROPERTY        PERM   VALUE        DEFAULT        POSSIBLE
bge0     zone            rw     --           --             --
vsw0     speed           r-     1000         1000           --
vsw0     autopush        rw     --           --             --
vsw0     zone            rw     --           --             --
vsw0     duplex          r-     full         full           half,full
vsw0     state           r-     up           up             up,down
vsw0     adv_autoneg_cap --     --           0              1,0
vsw0     mtu             rw     1500         1500           1500
vsw0     flowctrl        --     --           no             no,tx,rx,bi,pfc,
                                                            auto
...
```

2. 引导区域 49bge。

```
global# zoneadm -z 49bge boot
```

3. 再次执行命令 dladm show-linkprop。请注意，bge0 链路现在分配给了 49bge。

```
global# dladm show-linkprop
LINK     PROPERTY        PERM   VALUE        DEFAULT        POSSIBLE
bge0     zone            rw     49bge        --             --
vsw0     speed           r-     1000         1000           --
vsw0     autopush        rw     --           --             --
vsw0     zone            rw     --           --             --
vsw0     duplex          r-     full         full           half,full
```

```
vsw0     state           r-      up          up          up,down
vsw0     adv_autoneg_cap --      --          0           1,0
vsw0     mtu             rw      1500        1500        1500
vsw0     flowctrl        --      --          no          no,tx,rx,bi,pfc,
                                                         auto
...
```

示例 26–2 如何在使用虚命名时显示数据链路和物理位置

设备物理位置显示在 LOCATION 字段中。要查看设备的数据链路名称和物理位置信息，请使用 -L 选项。

```
global# dladm show-phys -L
LINK         DEVICE       LOCATION
net0         e1000g0      MB
net1         e1000g1      MB
net2         e1000g2      MB
net3         e1000g3      MB
net4         ibp0         MB/RISER0/PCIE0/PORT1
net5         ibp1         MB/RISER0/PCIE0/PORT2
net6         eoib2        MB/RISER0/PCIE0/PORT1/cloud-nm2gw-2/1A-ETH-2
net7         eoib4        MB/RISER0/PCIE0/PORT2/cloud-nm2gw-2/1A-ETH-2
```

▼ 如何使用 dladm 指定临时数据链路

可以将 dladm 命令与 set-linkprop 子命令一起使用，以临时向正在运行的专用 IP 区域指定数据链路。必须使用 zonecfg 命令进行持久性指定。

您必须是全局管理员或在全局区域中具有相应授权的用户才能管理数据链路。

1. **成为 root 用户或承担等效角色。**

2. **使用带有 -t 的 dladm set-linkprop 将 bge0 添加到正在运行的名为 zoneA 的区域。**

```
global# dladm set-linkprop -t -p zone bge0
LINK     PROPERTY        PERM    VALUE        DEFAULT        POSSIBLE
bge0     zone            rw      zoneA        --             --
```

提示 – -p 选项会生成一个显示内容，其格式为稳定的机器可解析格式。

▼ 如何使用 dladm reset-linkprop

可以将 dladm 命令与 reset-linkprop 子命令一起使用，以将 bge0 链路值重置为未指定状态。

1. **成为 root 用户或承担等效角色。**

2 **使用带有 -t 选项的 dladm reset-linkprop 撤消对 bge0 设备的区域指定。**

```
global# dladm reset-linkprop -t -p zone bge0
LINK     PROPERTY        PERM    VALUE        DEFAULT        POSSIBLE
bge0     zone            rw      zoneA        --             --
```

提示 – -p 选项会生成一个显示内容，其格式为稳定的机器可解析格式。

故障排除 如果正在运行的区域在使用该设备，则重新指定将失败，并显示一条错误消息。请参见第 363 页中的 “专用 IP 区域正在使用设备，因此 dladm reset-linkprop 失败”。

在安装了区域的 Oracle Solaris 系统上使用公平份额调度器

通过 prctl 命令指定的限制不是持久的。在重新引导系统后，此限制将失效。要在区域中设置永久性份额，请参见第 232 页中的 “如何配置区域”和第 245 页中的 “如何在全局区域中设置 zone.cpu-shares”。

▼ 如何使用 prctl 命令在全局区域中设置 FSS 份额

缺省情况下，为全局区域提供一个份额。可以使用此过程来更改缺省分配。请注意，只要重新引导系统，就必须重置通过 prctl 命令分配的份额。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1 **成为 root 用户或承担等效角色。**

2 **使用 prctl 实用程序为全局区域指定两个份额：**

```
# prctl -n zone.cpu-shares -v 2 -r -i zone global
```

3 **（可选的）要检验为全局区域指定的份额数，请键入：**

```
# prctl -n zone.cpu-shares -i zone global
```

另请参见 有关 prctl 实用程序的更多信息，请参见 prctl(1) 手册页。

▼ 如何在区域中动态更改 zone.cpu-shares 的值

可以在全局区域或非全局区域使用此过程。

1 **成为 root 用户或承担等效角色。**

2 **使用 prctl 命令为 cpu-shares 指定一个新值。**

```
# prctl  -n zone.cpu-shares -r -v value -i zone zonename
```

idtype 为 *zonename* 或 *zoneid*。*value* 为新值。

在区域管理中使用权限配置文件

本节介绍与在非全局区域中使用权限配置文件相关联的任务。

▼ 如何指定区域管理配置文件

区域管理配置文件授予用户管理系统上所有非全局区域的权力。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为超级用户，或拥有等效授权。**

   有关角色的更多信息，请参见《Oracle Solaris 11.1 管理：安全服务》中的第 III 部分, “角色、权限配置文件和特权”。

2. **创建一个包括区域管理权限配置文件的角色并将其分配给用户。**

备份安装了区域的 Oracle Solaris 系统

以下过程可以用于在区域中备份文件。同时，请记住还要备份区域的配置文件。

▼ 如何使用 ZFSsend 执行备份

1. **成为 root 用户或承担等效角色。**

2. **获得区域的 `zonepath`：**

```
global# zonecfg -z my-zone info zonepath
zonepath: /zones/my-zone
```

3. **使用 `zfs list` 命令获得 `zonepath` 数据集：**

```
global# zfs list -H -o name /zones/my-zone
rpool/zones/my-zone
```

4. **使用 ZFS 快照创建区域的归档文件：**

```
global# zfs  snapshot -r rpool/zones/my-zone@snap
global# zfs snapshot -r rpool/zones/my-zone@snap
global# zfs zfs send -rc rpool/zones/my-zone@snap > /path/to/save/archive
global# zfs destroy -r rpool/zones/my-zone@snap
```

   将显示以下类似信息：

```
-rwxr-xr-x   1 root     root     99680256 Aug 10 16:13 backup/my-zone.cpio
```

▼ 如何列显区域配置的副本

您应当创建非全局区域配置的备份文件。如有必要，将来可以使用备份来重新创建区域。在您首次登录到区域，并回答了 sysidtool 的问题之后，创建区域配置的副本。此过程使用名为 my-zone 的区域和名为 my-zone.config 的备份文件来显示过程。

1. **成为 root 用户或承担等效角色。**
2. **将区域 my-zone 的配置列显到名为 my-zone.config 的文件。**

```
global# zonecfg -z my-zone export > my-zone.config
```

重新创建非全局区域

▼ 如何重新创建单个非全局区域

如有必要，可以使用非全局区域配置的备份文件来重新创建非全局区域。此过程使用名为 my-zone 的区域和名为 my-zone.config 的备份文件来说明重新创建区域的过程。

1. **成为 root 用户或承担等效角色。**
2. **指定 my-zone.config 作为 zonecfg 命令文件来重新创建区域 my-zone。**

```
global# zonecfg -z my-zone -f my-zone.config
```

3. **安装区域。**

```
global# zoneadm -z my-zone install -a /path/to/archive  options
```

4. **如果您需要恢复任何区域特定的文件（例如应用程序数据），请将这些文件从备份手动恢复（并可能手动合并）到新创建的区域根文件系统。**

◆ ◆ ◆ 第 27 章

27 配置和管理不可编辑的区域

不可编辑的区域为 solaris 非全局区域提供只读文件系统配置文件。

只读区域概述

如果区域的根目录为只读的，则该区域称为不可编辑的区域。solaris 不可编辑的区域通过为非全局区域实现只读 root 文件系统来保留区域的配置。此区域通过向运行时环境添加更多限制来扩展区域安全运行时界限。除非作为特定维护操作执行，否则对系统二进制文件或系统配置的修改将被阻止。

强制写入访问控制 (mandatory write access control, MWAC) 内核策略用于通过 zonecfg file-mac-profile 属性强制执行文件系统写入特权。由于全局区域不受 MWAC 策略的制约，全局区域可写入非全局区域的文件系统进行安装、映像更新和维护。

区域进入就绪状态时，下载 MWAC 策略。该策略将在区域引导时启用。要执行安装后装配和配置，使用临时可写入 root 文件系统引导序列。对区域 MWAC 配置的修改仅在区域重新引导时才生效。

有关配置、安装及引导区域的一般信息，请参见第 17 章，规划和配置非全局区域（任务）和第 19 章，安装、引导、关闭、停止、卸载和克隆非全局区域（任务）

配置只读区域

zonecfg file-mac-profile 属性

缺省情况下，不在非全局区域中设置 zonecfg file-mac-profile 属性。将区域配置为具有可写根数据集。

在 solaris 只读区域中，file-mac-profile 属性用于配置只读区域根目录。只读根目录限制访问区域内部的运行时环境。

通过 zonecfg 实用程序，可将 file-mac-profile 设置为以下值之一。所有配置文件（none 除外）均会造成 /var/pkg 目录及其内容在区域内部为只读状态。

none	标准读写非全局区域，在现有区域限制之外没有其他保护。将该值设置为 none 相当于没有设置 file-mac-profile 属性。
strict	只读文件系统，没有例外。 ■ 无法安装 IPS 软件包。 ■ 持久启用的 SMF 服务被修复。 ■ 从缺省位置无法添加 SMF 清单。 ■ 日志记录和审计配置文件被修复。只能远程记录数据。
fixed-configuration	允许对 /var/* 目录进行更新（包含系统配置组件的目录除外）。 ■ 无法安装 IPS 软件包（包括新软件包）。 ■ 持久启用的 SMF 服务被修复。 ■ 从缺省位置无法添加 SMF 清单。 ■ 日志记录和审计配置文件可以是本地文件。syslog 和审计配置被修复。
flexible-configuration	允许修改 /etc/ * 目录中的文件、更改根目录的起始目录及更新 /var/* 目录。此配置提供的功能与《System Administration Guide: Oracle Solaris Containers-Resource Management and Oracle Solaris Zones》中所述的 Oracle Solaris 10 native 稀疏根区域的功能最为相近。这是本指南的 Oracle Solaris 10 版本。 ■ 无法安装 IPS 软件包（包括新软件包）。 ■ 持久启用的 SMF 服务被修复。 ■ 从缺省位置无法添加 SMF 清单。 ■ 日志记录和审计配置文件可以是本地文件。可更改 syslog 和审计配置。

zonecfg add dataset 资源策略

通过 add dataset 资源添加到区域的数据集不受 MWAC 策略的制约。被委托附加数据集的区域对这些数据集具有完全控制权。平台数据集是可见的，但其中的数据及属性为只读，除非引导的区域为读/写。

zonecfg add fs 资源策略

通过 add fs 资源添加到区域的文件系统不受 MWAC 策略的制约。可将文件系统挂载为只读。

管理只读区域

盘上配置只能通过全局区域进行管理。在运行着的区域内，管理仅限于设置运行时状态，除非将该区域引导为可写。因此，通过 svcadm(1M) 和 svccfg(1M) 手册页中所述的 SMF 命令进行的配置更改仅适用于临时的实时 SMF 数据库，不适用于盘上 SMF 数据库。对区域 MWAC 配置的修改在重新引导区域时才生效。

进行初始安装或随后的更新时，区域将引导瞬态读写，直到到达 self-assembly-complete 里程碑。随后在只读模式下重新引导区域。

zoneadm list -p 显示

可解析输出显示 R/W 列和 file-mac-profile 列：

```
global# zoneadm list -p
0:global:running:/:UUID:solaris:shared:-:none
    5:testzone2:running:/export/zones/testzone2:UUID \
        :solaris:shared:R:fixed-configuration
    12:testzone3:running:/export/zones/testzone3:UUID \
        :solaris:shared:R:fixed-configuration
    13:testzone1:running:/export/zones/testzone1:UUID \
        :solaris:excl:W:fixed-configuration
    -:testzone:installed:/export/zones/testzone:UUID \
        :solaris:excl:-:fixed-configuration
```

定义了以下 R 和 W 选项：

- R 指示具有 file-mac-profile 并引导为只读的区域。
- W 指示具有 file-mac-profile 并引导为读写的区域。
- – 指示未运行，也不具有 file-mac-profile 的区域。

用于通过可写根文件系统引导只读区域的选项

zoneadm boot 子命令提供了两种允许全局区域管理员手动引导只读区域的方法：使用可写根文件系统和使用瞬态可写根文件系统。请注意，只有下次重新引导时，区域才会处于可写模式。

-w 使用可写 root 文件系统手动引导区域。

-W 使用瞬态可写 root 文件系统手动引导区域。到达 self-assembly-complete 里程碑时，将自动重新引导系统。

重新引导将区域重新置于 MWAC 策略的控制之下。当区域具有 none MWAC 策略时，才允许此选项。

非 ROZR 区域将忽略 -W 和 - w 选项。

◆ ◆ ◆ 第 28 章

28 各种 Oracle Solaris Zones 问题的故障排除

本章包含区域的疑难解答信息。

专用 IP 区域正在使用设备，因此 dladm reset-linkprop 失败

如果显示以下错误消息：

```
dladm: warning: cannot reset link property 'zone' on 'bge0': operation failed
```

则表明尝试使用 dladm reset-linkprop 失败，请参阅第 355 页中的 “如何使用 dladm reset-linkprop”。运行中的区域 excl 正在使用该设备。

要重置值：

1. 键入：

   ```
   global# ipadm delete-ip bge0
   ```

2. 重新运行 dladm 命令。

在区域配置中指定的特权集不正确

如果区域的特权集包含不允许的特权、缺少必需特权或包含未知特权名称，则检验、准备或引导该区域的尝试都将失败，并将显示如下所示的错误消息：

```
zonecfg:zone5> set limitpriv="basic"
.
.
.
global# zoneadm -z zone5 boot
     required privilege "sys_mount" is missing from the zone's privilege set
     zoneadm: zone zone5 failed to verify
```

区域无法停止

如果无法破坏与区域关联的系统状态，则停止操作会中途失败。区域便会陷于中间状态，即介于正在运行和已安装状态之间。在此状态下，不存在任何活动的用户进程或内核线程，也无法创建它们。当停止操作失败时，您必须手动干预来完成此过程。

最常见的故障原因是系统无法卸载所有的文件系统。与破坏系统状态的传统 Oracle Solaris 系统关闭不同，区域一旦停止，就必须确保在引导区域或继续进行区域操作时没有执行任何挂载。即使 `zoneadm` 可确保区域中没有执行任何进程，但是如果全局区域中的进程在此区域中具有打开的文件，则卸载操作也会失败。请使用 `proc`(1)（请参见 `pfiles`）和 `fuser`(1M) 手册页中所述的工具来查找这些进程，并采取相应的操作。处理了这些进程之后，重新调用 `zoneadm halt` 应完全停止区域。

第 3 部分

Oracle Solaris 10 Zones

Oracle Solaris 10 Zones 是 `solaris10` 标记区域，用于托管在 Oracle Solaris 11 内核上运行的 x86 和 SPARC Solaris 10 9/10（或以后发行的 Oracle Solaris 10 更新）用户环境。请注意，可以在原始系统上使用早期 Oracle Solaris 10 发行版（如果首先安装内核修补程序 142909-17 (SPARC) 或 142910-17 (x86/x64)），或者使用更新版本。

◆ ◆ ◆ 第 29 章

29 Oracle Solaris 10 Zones 介绍

BrandZ 提供了创建标记区域的框架，标记区域用于运行不能在 Oracle Solaris 11 环境中运行的应用程序。此处介绍的标记是 `solaris10` 标记，即 Oracle Solaris 10 Zones。在这些 `solaris10` 标记区域内运行的工作负荷可以利用 Oracle Solaris 内核的增强功能，并使用一些仅适用于 Oracle Solaris 11 发行版的创新技术，如虚拟 NIC (Virtual NIC, VNIC) 和 ZFS 重复数据删除技术。

注 – 如果您想立即创建一个 `solaris10` 标记区域，请参见第 30 章，评估 Oracle Solaris 10 系统和创建归档文件。

关于 solaris10 标记

`solaris10` 标记区域在 `solaris10(5)` 手册页中有所介绍，对于运行 Oracle Solaris 10 9/10 操作系统（或之后发布的更新版本）的 SPARC 和 x86 计算机上运行的 Oracle Solaris 10 应用程序而言，它是一个完整的运行时环境。当运行 Oracle Solaris 10 9/10 之前的 Oracle Solaris 10 发行版时，如果首先在原始系统上安装内核修补程序 142909-17 (SPARC) 或 142910-17 (x86/x64)（或更高版本），则可使用早期更新发行版。必须在创建用于安装区域的归档文件之前安装修补程序。迁移到 Oracle Solaris 10 Zones 的先决条件是此发行版的内核修补程序，而不是完整的 Oracle Solaris 10 9/10 或更新发行版。修补程序的软件下载站点为 My Oracle Support (`https://support.oracle.com`)。单击 "Patches & Updates"（修补程序和更新）选项卡。在该站点中，您可以查看下载说明并下载映像。请与您的支持提供商联系以获得有关修补程序的其他信息。

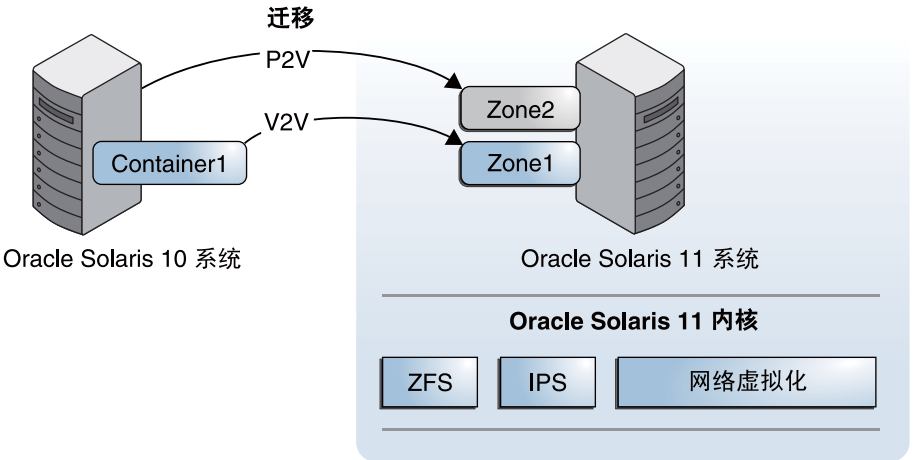
Oracle Solaris 11.1 发行版已定义为受支持平台的所有体系结构均支持在单个全局区域实例中运行的非全局区域。

该标记包括将 Oracle Solaris 10 系统映像安装到非全局区域所必需的工具。您无法直接从 Oracle Solaris 10 介质安装 `solaris10` 标记区域。可以使用物理转换为虚拟 (physical-to-virtual, P2V) 功能将现有系统直接迁移到目标系统上的非全局区域。可使用 `zonep2vchk` 工具生成 P2V 过程需要的信息，并输出模板 zonecfg 命令文件以供在目标系

统上使用。此实用程序将创建一个与源系统配置相匹配的区域。在源 Oracle Solaris 10 1/13 系统上，发行版中包含实用程序 `/usr/sbin/zonep2vchk`。要在 Oracle Solaris 10 早期版本中使用该实用程序，请从位于 `http://www.oracle.com/technetwork/server-storage/solaris10/downloads` 的 Oracle 技术网 (Oracle Technology Network, OTN) 上下载非捆绑软件包。

该标记还支持用于将 Oracle Solaris 10 `native` 区域迁移到 `solaris10` 标记非全局区域的工具。将 Oracle Solaris 10 `native` 非全局区域迁移到 `solaris10` 标记区域的虚拟转换为虚拟 (virtual-to-virtual, V2V) 过程支持与 P2V 相同的归档文件格式。有关更多信息，请参见第 31 章，（可选）将 Oracle Solaris 10 `native` 非全局区域迁移到 Oracle Solaris 10 Zones。

图 29–1 Oracle Solaris 10 Containers 转换为 Oracle Solaris 10 Zones



solaris10 区域支持

`solaris10` 标记区域支持完全根非全局区域模型。所有必需的 Oracle Solaris 10 软件及任何其他软件包都安装在区域的专用文件系统中。

非全局区域必须位于自己的 ZFS 数据集上；仅支持 ZFS。在安装或附加区域时，将自动创建 ZFS 数据集。如果无法创建 ZFS 数据集，也无法安装或附加区域。请注意，区域路径的父目录必须也是 ZFS 数据集，否则将无法创建文件系统。

在 `native` Oracle Solaris 10 非全局区域中执行的任何应用程序或程序在 `solaris10` 标记区域中也应该运行。

请注意，区域不支持静态链接的二进制文件。

注 – 您可以在已启用了标签的 Oracle Solaris Trusted Extensions 系统中创建和安装 `solaris10` 标记区域，但**如果**所引导的标记为**有标签的**标记，只能在此系统配置上引导标记区域。在 Oracle Solaris 10 系统中使用 Oracle Solaris Trusted Extensions 的客户必须转换到经过认证的 Oracle Solaris 系统配置。

Oracle Solaris 10 Zones 中的 SVR4 包管理和修补

关于在 solaris10 标记区域中使用包管理和修补

SVR4 软件包元数据在区域中可用，并且包管理和修补命令可以正确运行。为了能够正确运行，请注意您**必须**在创建归档文件**之前**在 Oracle Solaris 10 系统中安装修补程序 119254-75 (SPARC) 或 119255-75 (x86/x64)，或更新版本。修补程序的软件下载站点为 My Oracle Support (`https://support.oracle.com`)。单击 "Patches & Updates"（修补程序和更新）选项卡可以查看下载说明并下载映像。请与您的支持提供商联系以获得有关修补程序的其他信息。

由于 `solaris10` 标记区域是完全根区域，因此所有包管理和修补操作都会如手册页和其他文档中所描述的一样工作。请注意安装时不使用软件包或修补程序的内核组件。SVR4 软件包仅安装在当前区域中。有关在 `solaris10` 和 `native` 区域中使用的 SVR4 包管理的信息，请参见《System Administration Guide: Oracle Solaris Containers-Resource Management and Oracle Solaris Zones》中的第 25 章“关于安装了区域的 Solaris 系统上的软件包（概述）”和第 26 章“在安装了区域的 Solaris 系统上添加和删除软件包和修补程序（任务）”。这是本指南的 Oracle Solaris 10 版本。

有关系统发行版级别的信息，请参见第 29 章，Oracle Solaris 10 Zones 介绍。

关于远程执行包管理和修补操作

对于在 Oracle Solaris 10 Zones 内启动的修补操作，如果远程系统也是 `solaris10` 区域，修补操作会正确运行。但是，如果远程系统是并非 `solaris10` 区域的 miniroot 或 Oracle Solaris 10 系统，操作会产生不确定的结果。同样，如果在并非 Oracle Solaris 10 Zones 的 miniroot 或物理系统中使用修补工具来修补 Oracle Solaris 10 Zones，修补工具也会产生不确定结果。

请注意，通常 `patchadd` 和 `patchrm` 工具允许管理员在运行修补操作时指定备用根。此功能允许管理员修补其**根**目录在 NFS 上为可见的远程系统，如 Oracle Solaris 10 miniroot 和 Oracle Solaris 10 物理系统。例如，如果 Oracle Solaris 10 系统的根目录通过 NFS 挂载到本地系统的 `/net/a-system` 目录，则可从本地系统修补远程 Oracle Solaris 10 系统。

要在远程系统上安装修补程序 142900-04（或更新版本）：

```
# patchadd -R /net/a-system 142900-04
```

有关更多信息，请参见《man pages section 1M: System Administration Commands》中的下列手册页：

- patchadd(1M)，-R 和 -C 选项
- patchrm(1M)

作为 NFS 客户机的非全局区域

区域可以是 NFS 客户机。支持版本 2、版本 3 和版本 4 协议。有关这些 NFS 版本的信息，请参见《Oracle Solaris 管理：网络服务》中的“NFS 服务的功能”。

缺省版本为 NFS 版本 4。可以使用以下方法之一在客户机上启用其他 NFS 版本：

- 可以编辑 /etc/default/nfs 以设置 NFS_CLIENT_VERSMAX=*number*，从而使区域在缺省情况下使用指定的版本。请参见《Oracle Solaris 管理：网络服务》中的“设置 NFS 服务”。请使用任务列表中的“如何通过修改 /etc/default/nfs 文件在客户机上选择不同的 NFS 版本”过程。
- 可以手动创建版本挂载。此方法会覆盖 /etc/default/nfs 的内容。请参见《Oracle Solaris 管理：网络服务》中的“设置 NFS 服务”。使用任务列表中的“如何使用命令行在客户机上选择不同的 NFS 版本”过程。

一般的区域概念

您应该熟悉以下资源管理和区域概念，这些概念在本指南的第 1 部分和第 2 部分中有所讨论。

- zonep2vchk 工具
- 支持的功能和不支持的功能
- 资源控制，使管理员可以控制应用程序如何使用可用的系统资源
- 用于配置、安装和管理区域的命令，主要包括 zonecfg、zoneadm 和 zlogin
- zonecfg 资源和属性类型
- 全局区域和非全局区域
- 完全根非全局区域模型
- 通过 zonecfg 实用程序授予的授权
- 全局管理员和区域管理员
- 区域状态模型
- 区域隔离特性
- 特权

- 联网
- 使用 anet 资源来配置 IPoIB
- 区域共享 IP 和专用 IP 类型
- 在区域中使用资源管理功能（如资源池）
- 公平份额调度器 (fair share scheduler, FSS)，一个使您可以根据份额来分配 CPU 时间的调度类
- 资源上限设置守护进程 (rcapd)，可用于从全局区域中控制标记区域驻留集大小 (resident set size, RSS) 的使用

关于此版本的 Oracle Solaris 10 Zones

运行限制

不能在 solaris10 标记区域中配置 /dev/sound 设备。

用于创建只读区域的 file-mac-profile 属性不可用。

quota(1M) 中介绍的 quota 命令不能用于检索在 solaris10 标记区域中使用的 UFS 文件系统的配额信息。

solaris10 标记区域不能是 NFS 服务器。

Oracle Solaris 10 Zones 中的联网

下面的部分列出了在 Oracle Solaris 10 Zones 中不可用或不同的 Oracle Solaris 10 联网组件。

不受支持的联网组件

- 不支持使用 atun STREAMS 模块的自动隧道。
- solaris10 标记区域不支持下列 ndd 可调参数：
  - ip_squeue_fanout
  - ip_soft_rings_cnt
  - ip_ire_pathmtu_interval
  - tcp_mdt_max_pbufs

不同的联网功能

在具有专用 IP 配置的 `solaris10` 标记区域中，下列功能与在物理 Oracle Solaris 10 系统中有所不同：

- 移动 IP 不可用，因为它在 Oracle Solaris 11 发行版中不可用。
- 在 `solaris10` 标记区域中，当 `tcp`、`udp` 或 `icmp` 套接字处于打开状态时，会忽略 `autopush` 配置。这些套接字缺省情况下映射到模块，而不是 STREAMS 设备。要使用 `autopush`，请使用 `soconfig(1M)` 和 `sock2path.d(4)` 手册页中介绍的 `soconfig` 和 `sock2path.d` 实用程序将这些套接字显式映射到基于 STREAMS 的设备。
- 在从运行 Oracle Solaris 10 9/10 或更早更新的物理系统归档的 `solaris10` 标记区域中，数据链路供应商接口库 (`libdlpi`) 不支持 VNIC 之类的 `/dev/net` 链路。这些链接在 Oracle Solaris 10 8/11 中受支持。在 `libdlpi(3LIB)` 手册页中介绍了该库。

  不使用 Oracle Solaris 10 8/11 中的 `libdlpi` 库，或者 `libpcap` 版本 1.0.0 或更高库的应用程序将无法访问 VNIC 之类的 `/dev/net` 链接。
- 由于 Oracle Solaris 10 Zones 中的 IP 网络多路径 (IP network multipathing, IPMP) 是基于 Oracle Solaris 11 版本的，所以与 Oracle Solaris 10 操作系统中的命令输出相比，`ifconfig` 命令的输出有许多不同之处。但对于 `ifconfig` 命令，所阐述的功能和 IPMP 没有变化。因此，使用所述接口的 Oracle Solaris 10 应用程序不需更改也仍可继续在 Oracle Solaris 10 Zones 中运行。

  以下示例为数据地址为 `192.168.1.3`，底层接口为 `e1000g1` 和 `e1000g2`，测试地址为 `192.168.1.1` 和 `192.168.1.2` 的 IPMP 组 `ipmp0`，显示了在 `solaris10` 标记区域中 `ifconfig` 命令的输出。

```
% ifconfig -a
e1000g1:
flags=9040843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4,NOFAILOVER>
mtu 1500 index 8
        inet 192.168.1.1 netmask ffffff00 broadcast 192.168.1.255
        ether 0:11:22:45:40:a0
e1000g2:
flags=9040843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4,NOFAILOVER>
mtu 1500 index 9
        inet 192.162.1.2 netmask ffffff00 broadcast 192.168.1.255
        ether 0:11:22:45:40:a1
ipmp0: flags=8011000803<UP,BROADCAST,MULTICAST,IPv4,FAILED,IPMP> mtu 68
index 10
        inet 192.168.1.3 netmask ffffff00 broadcast 192.168.1.255
        groupname ipmp0
```

  - 与在 Oracle Solaris 10 系统中生成的显示不同，Oracle Solaris 10 Container 中的 `ifconfig` 命令不会显示底层接口和 IP 地址的绑定。可以使用带有 `-an` 选项的 `arp` 命令来获取此信息。
  - 如果检测到 IPv6 的接口，并且地址配置成功，会为接口授予其自己的全局地址。在 Oracle Solaris 10 系统中，IPMP 组中的每个物理接口都有自己的全局地址，IPMP 组的全局地址与接口一样多。在 Oracle Solaris 10 Zones 中，仅 IPMP 接口有自己的全局地址。底层接口没有自己的全局地址。

- 与 Oracle Solaris 10 操作系统不同，如果 IPMP 组中只有一个接口，其测试地址和数据地址不能相同。

请参见 arp(1M) 和 ifconfig(1M) 手册页，以及第 323 页中的 “专用 IP 区域中的 IP 网络多路径”。

安装了 native 非全局区域时

如果在 Oracle Solaris 10 9/10（或之后发布的更新版本）源物理系统上安装了 native 区域，P2V 过程会多一个步骤。由于各区域并不嵌套，这些系统上的 P2V 过程会使标记区域内的现有区域无法使用。安装区域时系统会检测到现有区域，并发出警告，表明任何嵌套区域将无法使用并且可以恢复磁盘空间。可以使用第 31 章，（可选）将 Oracle Solaris 10 native 非全局区域迁移到 Oracle Solaris 10 Zones中介绍的 V2V 过程先迁移这些区域。

如果在运行较早版本的系统上应用内核修补程序，请在迁移现有区域之前应用修补程序。

◆ ◆ ◆ 第 30 章

30 评估 Oracle Solaris 10 系统和创建归档文件

本章讨论如何获得有关 Oracle Solaris 10 10/09（或最新发布的更新）系统的信息和如何为系统创建归档文件。可以使用物理转换为虚拟 (physical-to-virtual, P2V) 功能将现有 Oracle Solaris 系统直接迁移到目标系统上的非全局区域。还提供了有关目标系统中必要软件包的信息。

源系统和目标系统的先决条件

启用 Oracle Solaris 10 软件包和修补工具

要在 Oracle Solaris 10 Zones 中使用 Oracle Solaris 10 软件包和修补工具，请在创建映像**之前**在源系统中为您的体系结构安装下列修补程序。

- 119254-75、119534-24 和 140914-02 (SPARC)
- 119255-75、119535-24 和 140915-02 (x86/x64)

P2V 过程可以在没有修补程序的情况下进行，但软件包和修补工具在 `solaris10` 标记区域内无法正常运行。

在目标系统中安装必要的 Oracle Solaris 软件包

要在系统中使用 Oracle Solaris 10 Zones，必须在运行 Oracle Solaris 11 的系统上安装 `pkg:/system/zones/brand/brand-solaris10`。

有关系统信息库的更多信息，请参见第 303 页中的 “运行 Oracle Solaris 11.1 发行版的系统上的映像包管理系统软件”。

有关软件包安装的说明，请参见《添加和更新 Oracle Solaris 11.1 软件包》 。

使用 zonep2vchk 实用程序评估要迁移的系统

现有 Oracle Solaris 10 9/10 系统（或后来发布的 Solaris 10 更新）可以直接迁移到 Oracle Solaris 11 系统上的 `solaris10` 标记区域。

开始时，请先使用 zonep2vchk(1M) 和第 22 章，关于区域迁移和 zonep2vchk 工具中所述的 `zonep2vchk` 工具来检查源系统，并收集所需信息。此工具用于评估要迁移的系统，并生成一个包含网络配置的 `zonecfg` 模板。

取决于原有系统执行的服务，全局管理员或具有相应授权的用户可能需要在安装区域后手动对其进行定制。例如，指定给区域的特权可能需要进行修改。此操作无法自动完成。此外，由于并非所有系统服务都在区域内运行，因此并非每个 Oracle Solaris 10 系统都适合迁移到区域中。

注 – 如果要迁移的系统中存在任何 `native` 非全局区域，必须先将其删除，或将其归档并移到新目标系统的区域中。对于稀疏根区域，必须使用处于就绪状态的区域进行归档。有关迁移的其他信息，请参见第 31 章，（可选）将 Oracle Solaris 10 `native` 非全局区域迁移到 Oracle Solaris 10 Zones。有关稀疏根区域的其他信息，请参见 Oracle Solaris 10 文档中的区域概述。

仅 Oracle Solaris 10 系统：获取 zonep2vchk 实用程序

`zonep2vchk` 实用程序在 Oracle Solaris 10 1/13 系统中可用。

要在 Oracle Solaris 10 系统早期版本中使用该实用程序，您可以从 OTN (`http://www.oracle.com/technetwork/server-storage/solaris10/downloads`) 下载非捆绑软件包。未捆绑的软件包安装到 `/opt/SUNWzonep2vchk` 中。

注 – 如果系统随后进行了升级或修补，该非捆绑软件包将不会与 Oracle Solaris 10 1/13 提供的版本冲突。该非捆绑版本安装到 `/opt/SUNWzonep2vchk` 中。对 Oracle Solaris 10 1/13 进行升级或修补时，会在 `/usr/sbin` 中添加捆绑版本。然后可以卸载早期获取的非捆绑软件包。

为将 Oracle Solaris 10 系统直接迁移到区域中创建映像

可以使用 Oracle Solaris Flash 归档工具为能迁移到区域中的已安装系统创建映像。

创建映像之前，可对系统进行完全配置，以包含将在区域中运行的所有软件。安装区域时，安装程序将使用该映像。

▼ 如何使用 flarcreate 创建映像

在具有 ZFS 根的系统中，可以使用 `flarcreate(1M)` Oracle Solaris 10 手册页中介绍的 `flarcreate` 命令创建系统映像。缺省情况下，创建的 `flar` 是一个 ZFS 发送流，如《Oracle Solaris 11.1 管理：ZFS 文件系统》中的“发送和接收 ZFS 数据”中所述。

此示例过程使用 NFS 将 Flash 归档文件放到目标 Oracle Solaris 11 系统上，不过您可以使用任一方法移动这些文件。

您必须是全局管理员或在全局区域中具有所需权限配置文件的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**
2. **登录到要归档的源 Oracle Solaris 10 系统。**
3. **转到根目录。**

   ```
   # cd /
   ```

4. **使用 `flarcreate` 在源系统上创建名为 `s10-system` 的 Flash 归档映像文件，并将该归档文件放到目标 Oracle Solaris 11 系统上：**

   ```
   source-system # flarcreate -n s10-system /net/target/export/archives/s10-system.flar
   ```

▼ 如何使用 flarcreate 排除特定数据

要从归档文件中排除不在 ZFS 数据集范围内的数据，必须将 `cpio` 或 `pax` 与 `flarcreate` 组合使用。可以使用 `-L archiver` 选项将 `cpio` 或 `pax` 指定为归档文件的方法。

此示例过程使用 NFS 将 Flash 归档文件放到目标 Oracle Solaris 11 系统上，不过您可以使用任一方法移动这些文件。

您必须是全局管理员或在全局区域中具有所需权限配置文件的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**
2. **登录到要归档的源 Oracle Solaris 10 系统。**
3. **转到根目录。**

   ```
   # cd /
   ```

4. **使用 `flarcreate` 在源系统上创建名为 `s10-system` 的 Flash 归档映像文件，并将该归档文件放到目标 Oracle Solaris 11 系统上：**

```
source-system # flarcreate -S -n s10-system -x /path/to/exclude -L cpio /net/target/export/archives/s10-system.fl
Determining which filesystems will be included in the archive...
Creating the archive...
cpio: File size of "etc/mnttab" has
```

```
increased by 435
2068650 blocks
1 error(s)
Archive creation complete.
```

提示 – 在某些情况下，`flarcreate` 可能显示 `cpio` 命令的错误。通常，这些消息是诸如 `File size of etc/mnttab has increased by 33` 之类的消息。当这些消息与日志文件或反映系统状态的文件有关时，可将其忽略。请务必要彻底查看所有的错误消息。

创建归档文件的其他方法

您可以使用其他方法创建归档文件。安装程序可以接受以下归档文件格式：

- `cpio` 归档文件
- `gzip` 压缩的 `cpio` 归档文件
- `bzip2` 压缩的 `cpio` 归档文件
- 使用 `-x xustar` (XUSTAR) 格式创建的 `pax` 归档文件
- `ufsdump` 级别零（完整）备份

此外，安装程序仅接受使用归档实用程序创建的文件的目录，该实用程序可以保存和恢复文件权限、所有权和链接。

有关更多信息，请参见 `cpio(1)`、`pax(1)`、`bzip2(1)`、`gzip(1)` 和 `ufsdump(1M)` 手册页。

注 – 如果使用除 Flash 归档文件以外的方法创建 P2V 的归档文件，则必须卸载源系统上依赖于处理器的 `libc.so.1` 挂载 `lofs` 的 (`hwcap`) 硬件功能库，然后才能创建归档文件。否则，安装有归档文件的区域在目标系统中可能无法引导。创建归档文件后，您可以在 `/lib/libc.so.1` 之上通过使用 `lofs` 和挂载 `-O` 选项重新挂载适当的硬件功能库。

```
source-system# umount /lib/libc.so.1
source-system# mount -O -F lofs /lib/libc.so.1
```

主机 ID 仿真

如果将应用程序从独立 Oracle Solaris 系统迁移到新系统上的区域，`hostid` 将更改为新计算机的 `hostid`。

在某些情况下，应用程序会依赖原始的 `hostid`，从而不能更新应用程序配置。这种情况下，可以将区域配置为使用原始系统的 `hostid`。通过设置 `zonecfg` 属性来指定 `hostid` 即可实现这一目的，如第 232 页中的 “如何配置区域”中所述。使用的值应该是在原始系统上运行 `hostid` 命令时的输出。要查看已安装区域中的 `hostid`，也可以使用 `hostid` 命令。

有关主机 ID 的更多信息，请参见 `hostid(1)`。

◆ ◆ ◆ 第 31 章

（可选）将 Oracle Solaris 10 native 非全局区域迁移到 Oracle Solaris 10 Zones

本章介绍了如何将 Oracle Solaris 10 9/10（或之后发布的更新版本）系统上的 native 非全局区域迁移到运行 Oracle Solaris 11 发行版的系统上的 Oracle Solaris 10 Zones。

只有您要迁移的系统中存在 native 非全局区域时，才需要阅读本章。必须先归档这些区域，并将其移到新目标系统上的标记区域中。

归档注意事项

Oracle Solaris 10 系统上的稀疏根区域将由系统转换为完全根模型，以便实现 solaris10 标记区域迁移。在 V2VA 过程发生之前，稀疏根区域必须在源系统中处于就绪状态。这样将创建归档文件之前挂载任何 inherited-pkg-dir 资源。有关这些概念的更多信息，请参见本指南 Oracle Solaris 10 版本中的区域概述。

在此过程中区域的标记将发生变化。

solaris10 区域迁移过程概述

将 Oracle Solaris 10 native 区域迁移到 solaris10 标记区域的虚拟转换为虚拟 (virtual-to-virtual, V2V) 过程支持与 P2V 相同的归档文件格式。此过程使用 zoneadm install 子命令。solaris10 标记 install 子命令使用下列选项，这些选项与 attach 子命令中的相同选项相对应。

注 – 在未来的 Oracle Solaris 发行版中，附加子命令的 -a 和 -d 选项可能被删除。建议使用 install 子命令。

选项	说明
-a *path*	指定将解压缩到区域中的归档文件的路径。支持完整 Flash 归档文件以及 pax、cpio、gzip 压缩的 cpio、bzip 压缩的 cpio 和 0 级 ufsdump。

选项	说明
`-d` *path*	将文件树的路径指定为安装的源。
`-d –`	将 `-d` 选项与破折号参数一起使用可指示要在 `zonepath` 中使用的现有目录布局。因此，如果管理员在安装前手动设置 `zonepath` 目录，则 `-d –` 选项可用于指示该目录已经存在。

关于分离和附加 solaris10 区域

可以通过配置目标系统上的区域，然后使用带有 `detach` 和 `attach` 子命令的 `zoneadm` 命令，以及附加归档文件的 `-a` 选项或指定 `zonepath` 的 `-d` 选项，将 `solaris10` 区域迁移到 Oracle Solaris 主机。此过程在第 295 页中的 “关于迁移区域”和第 296 页中的 “如何使用 ZFS 归档文件迁移非全局区域”中介绍。

注 – 在未来的 Oracle Solaris 发行版中，附加子命令的 `-a` 和 `-d` 选项可能被删除。建议使用 `install` 子命令。

迁移 solaris10 标记区域

可以使用 `zonecfg` 和 `zoneadm` 命令将现有的非全局区域从一个系统迁移到另一个系统。需要停止区域并使其与当前主机分离。`zonepath` 将移动到它所附加的目标主机。

`zoneadm detach` 进程用于创建在其他系统上附加区域所需的信息。`zoneadm attach` 进程用于检验目标计算机是否具有托管区域所需的正确配置。

由于可以通过多种方式来使 `zonepath` 在新主机上可用，因此 `zonepath` 从一个系统到另一个系统的实际移动是由全局管理员执行的手动进程。

在附加到新系统时，区域处于已安装状态。

示例 31–1 示例 attach 命令

```
host2# zoneadm -z zonename attach -a /net/machine_name/s10-system.flar
```

迁移 Oracle Solaris 10 系统上的现有区域

必须先归档系统上的任何现有非全局区域，并将其移到新目标系统上的区域，之后才能迁移物理系统。

▼ 如何迁移现有 native 非全局区域

可使用 V2V 进程将 Solaris 10 系统上的现有区域迁移到运行 Oracle Solaris 11 版本的系统上的 `solaris10` 标记区域。

1 打印现有区域的配置。您需要此信息来在目标系统上重新创建区域：

```
source# zonecfg -z my-zone info
zonename: my-zone
zonepath: /zones/my-zone
brand: native
autoboot: false
bootargs:
pool:
limitpriv:
scheduling-class:
ip-type: shared
hostid: 1337833f
inherit-pkg-dir:
         dir: /lib
inherit-pkg-dir:
         dir: /platform
inherit-pkg-dir:
         dir: /sbin
inherit-pkg-dir:
         dir: /usr
net:
         address: 192.168.0.90
         physical: bge0
```

2 使区域停止：

```
source# zoneadm -z my-zone halt
```

不要归档正在运行的区域，因为此区域内的应用程序和系统数据可能是在不一致状态下捕获的。

3 （可选）如果区域是具有 `inherit-pkg-dir` 设置的稀疏根区域，请先准备区域，以便归档继承的目录：

```
source# zoneadm -s my-zone ready
```

4 用 `zonepath /zones/my-zone` 归档区域。

- **为区域创建一个名为 `my-zone.cpio.gz` 的 `gzip` 压缩的 `cpio` 归档文件，在目标系统中其名称仍为 `my-zone`：**

  ```
  source# cd /zones
  source# find my-zone -print | cpio -oP@ | gzip >/zones/my-zone.cpio.gz
  ```

- **如果打算重命名目标系统上的区域，在 `zonepath` 中创建归档文件：**

  ```
  source# cd /zones/my-zone
  source# find root -print | cpio -oP@ | gzip >/zones/my-zone.cpio.gz
  ```

5 **使用诸如以下机制的任何文件传输机制来复制文件，将归档文件传输到目标 Oracle Solaris 11.1 系统：**

- `sftp(1)` 手册页中介绍的 `sftp` 命令
- NFS 挂载
- 可复制文件的任何其他文件传输机制。

6 **在目标系统上重新创建区域。**

```
target# zonecfg -z my-zone
my-zone: No such zone configured
Use 'create' to begin configuring a new zone.
zonecfg:my-zone> create -t SYSsolaris10
zonecfg:my-zone> set zonepath=/zones/my-zone
...
```

注 – 区域的标记必须为 `solaris10` 并且区域不能使用任何 `inherit-pkg-dir` 设置，即使原始系统被配置为稀疏根区域也是如此。有关 `inherit-pkg-dir` 资源的信息，请参见Part II, Oracle Solaris Zones。

如果目标系统的硬件不同、网络接口不同或具有必须在区域中进行配置的其他设备或文件系统，您必须更新区域的配置。请参见第 16 章，非全局区域配置（概述）、第 17 章，规划和配置非全局区域（任务）和第 295 页中的“关于迁移区域”。

---

7 **显示区域的配置：**

```
target# zonecfg -z my-zone info
zonename: my-zone
zonepath: /zones/my-zone
brand: solaris10
autoboot: false
bootargs:
pool:
limitpriv:
scheduling-class:
ip-type: shared
hostid: 1337833f
net:
         address: 192.168.0.90
         physical: bge0
```

8 **安装在源系统上创建的归档文件中的区域，此归档文件已传输至目标系统上的 `/zones` 目录：**

```
target# zoneadm -z my-zone install -a /zones/my-zone.cpio.gz
```

成功安装完区域后，可立即对区域进行引导。

可以保存区域的归档文件以备日后使用，或将其从系统中删除。

要从目标系统中删除归档文件：

```
target# rm /zones/myzone.cpio.gz
```

◆ ◆ ◆ 第 32 章

32 配置 solaris10 标记区域

本章论述如何配置 `solaris10` 标记区域。

预配置任务

您需要具备以下条件：

- 运行 Oracle Solaris 11 版本的受支持的 SPARC 或 x86 系统。
- 缺省设置是具有 `anet` 资源的专用 IP 类型。对于需要网络连接的共享 IP 区域，您需要为每个要创建的区域提供一个或多个唯一 IPv4 地址。还必须指定物理接口。
- 一台运行 Oracle Solaris 10 10/09（或之后发布的更新版本）操作系统的计算机，您要将此操作系统迁移至 `solaris10` 容器。迁移早期更新时可以附带相应的内核修补程序。您可以利用现有系统生成自己的映像。第 376 页中的 “为将 Oracle Solaris 10 系统直接迁移到区域中创建映像”对此过程进行了介绍。

配置中缺省包括的资源

缺省情况下，配置中会包含标记区域中的设备、文件系统和特权。

solaris10 标记区域中的已配置设备

每个区域支持的设备都在与该标记相关的手册页和其他文档中进行了介绍。`solaris10` 区域不允许添加任何不受支持或无法识别的设备。框架可以检测添加不受支持设备的任何尝试，并发出一条错误消息，指出无法检验区域配置。

要了解有关非全局区域中的设备注意事项的更多信息，请参见第 324 页中的 “非全局区域中的设备使用”。

solaris10 标记区域中定义的特权

仅允许进程拥有部分特权。特权限制可防止某个区域执行可能会影响其他区域的操作。通过特权设置，可以限制区域内特权用户的功能。

缺省特权、必需的缺省特权、可选的特权以及禁止特权由每个标记定义。如第 232 页中的 “如何配置区域”的步骤 8 所述，还可以使用 `limitpriv` 属性添加或删除某些特权。表 25–1 列出了区域中的所有 Solaris 特权以及每个特权的状态。

有关特权的更多信息，请参见 `ppriv(1)` 手册页和《系统管理指南：安全性服务》。

solaris10 标记区域配置过程

`zonecfg` 命令可用于执行以下操作：

- 为区域设置标记。
- 为 `solaris10` 区域创建配置。
- 检验配置，以确定虚拟系统中是否允许使用指定的资源和属性，这些资源和属性在内部是否一致。
- 执行特定于标记的检验。

可以使用 `zonep2vchk` 实用程序创建区域配置。

`zonecfg verify` 命令将针对给定配置检验以下内容：

- 确保已指定区域路径
- 确保已为每个资源指定所有必需的属性
- 确保已满足标记要求

有关 `zonecfg` 命令的更多信息，请参见 `zonecfg(1M)` 手册页。

配置目标区域

必须在 Oracle Solaris 11 系统中安装以下区域：`pkg:/system/zones/brand/brand-solaris10`。

通过使用 `zonecfg` 命令在目标系统上创建新的区域配置。

`zonecfg` 提示符的格式如下：

```
zonecfg:zonename>
```

当您配置特定的资源类型（例如文件系统）时，此资源类型也包含在提示符中：

```
zonecfg:zonename:fs>
```

提示 – 如果您知道将要使用 CD 或 DVD 在 `solaris10` 标记区域中安装应用程序，请在最初配置标记区域时，使用 `add fs` 在全局区域内添加对 CD 或 DVD 介质的只读访问权限。然后可以使用 CD 或 DVD 在标记区域中安装产品。有关更多信息，请参见第 351 页中的 “如何在非全局区域中添加对 CD 或 DVD 介质的访问权限”。

▼ 如何配置专用 IP solaris10 标记区域

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**

2. **创建一个区域名为 `s10-zone` 的专用 IP `solaris10` 区域。**

   ```
   global# zonecfg -z s10-zone
   ```

   如果是第一次配置该区域，则可以看到以下系统消息：

   ```
   s10-zone: No such zone configured
   Use 'create' to begin configuring a new zone.
   ```

3. **使用 `SYSsolaris10` 模板创建新的 `solaris10` 区域配置。**

   ```
   zonecfg:s10-zone> create -t SYSsolaris10
   ```

   `SYSsolaris10` 配置文件创建一个专用 IP 区域，该区域缺省情况下包含自动 `anet` 资源。

4. **设置区域路径，在此过程中为 `/zones/s10-zone`。**

   ```
   zonecfg:s10-zone> set zonepath=/zones/s10-zone
   ```

5. **设置自动引导值。**

   ```
   zonecfg:s10-zone> set autoboot=true
   ```

   如果设置为 `true`，则在引导全局区域时将自动引导该区域。缺省值为 `false`。请注意，要自动引导区域，还必须启用区域服务 `svc:/system/zones:default`。可以使用 `svcadm` 命令启用区域服务。

6. **添加与全局区域共享的 ZFS 文件系统。**

   ```
   zonecfg:s10-zone> add fs
   ```

   a. **将类型设置为 `zfs`。**

      ```
      zonecfg:s10-zone:fs> set type=zfs
      ```

   b. **设置目录以便从全局区域挂载。**

      ```
      zonecfg:s10-zone:fs> set special=share/zone/s10-zone
      ```

c. 指定挂载点。

```
zonecfg:s10-zone:fs> set dir=/opt/shared
```

d. 结束指定。

```
zonecfg:s10-zone:fs> end
```

可多次执行此步骤来添加多个文件系统。

7 委托存储池 *tank* 中一个名为 *sales* 的 ZFS 数据集。

```
zonecfg:my-zone> add dataset
```

a. 指定指向 ZFS 数据集 *sales* 的路径。

```
zonecfg:my-zone> set name=tank/sales
```

b. 结束数据集指定。

```
zonecfg:my-zone> end
```

8 将 `hostid` 设置为源系统的 `hostid`。

```
zonecfg:my-zone> set hostid=80f0c086
```

9 检验区域的配置。

```
zonecfg:s10-zone> verify
```

10 提交区域的配置。

```
zonecfg:s10-zone> commit
```

11 退出 `zonecfg` 命令。

```
zonecfg:s10-zone> exit
```

请注意，即使您没有在提示符下明确键入 `commit`，也会在键入 `exit` 或出现 EOF 时自动执行 `commit`。

12 使用 `info` 子命令验证标记是否已设置为 `solaris10`。

```
global# zonecfg -z s10-zone info
```

13 （可选的）使用 `info` 子命令检查 `hostid`：

```
global# zonecfg -z s10-zone info hostid
```

接下来的步骤

提示 – 配置完区域之后，最好复制该区域的配置。将来可以使用此备份来重新创建区域。具有正确配置文件的根用户或管理员可以将区域 *s10-zone* 的配置打印成文件。以下示例使用了名为 `s10-zone.config` 的文件。

```
global# zonecfg -z s10-zone export > s10-zone.config
```

另请参见 有关可以使用 zonecfg 进行配置的其他组件，请参见第 16 章，非全局区域配置（概述）。该指南还提供了在命令行或命令文件模式下使用 zonecfg 命令的信息。请注意，对于共享 IP 区域，必须在 zonecfg net 资源中指定一个静态地址。有关添加 ZFS 文件系统的更多信息，请参见《Oracle Solaris 11.1 管理：ZFS 文件系统》中的“向非全局区域中添加 ZFS 文件系统”。

▼ 如何配置共享 IP solaris10 标记区域

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**

2. **创建一个区域名为 s10-zone 的共享 IP solaris10 区域。**

   ```
   global# zonecfg -z s10-zone
   ```

   如果是第一次配置该区域，则可以看到以下系统消息：

   ```
   s10-zone: No such zone configured
   Use 'create' to begin configuring a new zone.
   ```

3. **创建新的 solaris10 区域配置。**

   ```
   zonecfg:s10-zone> create -b
   set brand=solaris10
   ```

   注 – 不要使用 create -t SYSsolaris10-shared-ip 设置 IP 类型。

4. **设置区域路径，在此过程中为 /zones/s10-zone。**

   ```
   zonecfg:s10-zone> set zonepath=/zones/s10-zone
   ```

5. **设置自动引导值。**

   如果设置为 true，则在引导全局区域时将自动引导该区域。请注意，要自动引导区域，还必须启用区域服务 svc:/system/zones:default。缺省值为 false。

   ```
   zonecfg:s10-zone> set autoboot=true
   ```

6. **使用网络虚拟接口创建共享 IP 区域。**

   ```
   zonecfg:my-zone> set ip-type=shared

   zonecfg:my-zone> add net
   ```

   a. **为网络接口设置 physical 设备类型，在此过程中为 bge 设备。**

      ```
      zonecfg:my-zone:net> Set physical=bge0
      ```

   b. **设置 IP 地址，在此过程中为 *10.6.10.233/24*。**

      ```
      zonecfg:my-zone:net> Set address=10.6.10.233/24
      ```

c. **结束指定。**

```
zonecfg:my-zone:net> end
```

可多次执行此步骤来添加多个网络接口。

7 **添加与全局区域共享的 ZFS 文件系统。**

```
zonecfg:s10-zone> add fs
```

a. **将类型设置为 `zfs`。**

```
zonecfg:s10-zone:fs> set type=zfs
```

b. **设置目录以便从全局区域挂载。**

```
zonecfg:s10-zone:fs> set special=share/zone/s10-zone
```

c. **指定挂载点。**

```
zonecfg:s10-zone:fs> set dir=/opt/shared
```

d. **结束指定。**

```
zonecfg:s10-zone:fs> end
```

可多次执行此步骤来添加多个文件系统。

8 **委托存储池 *tank* 中一个名为 *sales* 的 ZFS 数据集。**

```
zonecfg:my-zone> add dataset
```

a. **指定指向 ZFS 数据集 *sales* 的路径。**

```
zonecfg:my-zone> set name=tank/sales
```

b. **结束数据集指定。**

```
zonecfg:my-zone> end
```

9 **将 `hostid` 设置为源系统的 `hostid`。**

```
zonecfg:my-zone> set hostid=80f0c086
```

10 **检验区域的配置。**

```
zonecfg:s10-zone> verify
```

11 **提交区域的配置。**

```
zonecfg:s10-zone> commit
```

12 **退出 `zonecfg` 命令。**

```
zonecfg:s10-zone> exit
```

请注意，即使您没有在提示符下明确键入 `commit`，也会在键入 `exit` 或出现 EOF 时自动执行 `commit`。

13 使用 **info** 子命令验证标记是否已设置为 **solaris10**。

```
global# zonecfg -z s10-zone info
```

14 （可选的）使用 **info** 子命令检查 **hostid**：

```
global# zonecfg -z s10-zone info hostid
```

接下来的步骤

提示 – 配置完区域之后，最好复制该区域的配置。将来可以使用此备份来重新创建区域。具有正确配置文件的根用户或管理员可以将区域 *s10-zone* 的配置打印成文件。以下示例使用了名为 `s10-zone.config` 的文件。

```
global# zonecfg -z s10-zone export > s10-zone.config
```

另请参见 有关可以使用 `zonecfg` 进行配置的其他组件，请参见第 16 章，非全局区域配置（概述）。该指南还提供了在命令行或命令文件模式下使用 `zonecfg` 命令的信息。请注意，对于共享 IP 区域，必须在 `zonecfg net` 资源中指定一个静态地址。有关添加 ZFS 文件系统的更多信息，请参见《Oracle Solaris 11.1 管理：ZFS 文件系统》中的“向非全局区域中添加 ZFS 文件系统”。

◆ ◆ ◆ 第 33 章

33 安装 solaris10 标记区域

本章介绍如何安装 `solaris10` 标记区域。

区域安装映像

系统映像的类型

- 可以使用一个经过完全配置，包含将在区域中运行的所有软件的 Oracle Solaris 系统映像。请参见第 376 页中的 “为将 Oracle Solaris 10 系统直接迁移到区域中创建映像”。`zoneadm install -a` 命令将对物理系统进行归档。
- 可以使用现有 Oracle Solaris 10 `native` 区域的映像，而非物理系统的映像。请参见第 31 章，（可选）将 Oracle Solaris 10 `native` 非全局区域迁移到 Oracle Solaris 10 Zones。`zoneadm install -a` 命令对区域或物理系统进行归档，而 `zoneadm attach -a` 命令对区域进行归档。

映像 `sysidcfg` 状态

`-c` 可用来传送用于在安装完成后配置区域的 `sysidcfg` 文件。

如果利用现有系统创建了 Oracle Solaris 10 系统归档文件，并在安装区域时使用 `-p`（保留 `sysidcfg`）选项，该区域将与用于创建映像的系统具有相同的标识。

如果在安装目标区域时使用 `-u (sys-unconfig)` 和 `-c` 选项，则生成的区域将不会配置主机名或名称服务。

安装 solaris10 标记区域

第 2 部分和 zoneadm(1M) 手册页中所述的 zoneadm 命令是用于安装和管理非全局区域的主要工具。使用 zoneadm 命令的操作必须从目标系统上的全局区域中运行。

除了从归档文件解压缩文件外，安装过程还会执行检查、必需的后处理以及其他功能，以确保将区域优化为在主机上运行。

如果利用现有系统创建了 Oracle Solaris 系统归档文件，并在安装区域时使用 -p（保留 sysidcfg）选项，该区域将与用于创建映像的系统具有相同的标识。

如果在安装目标区域时使用 -u (sys-unconfig) 选项，则生成的区域将不会配置主机名或名称服务。

注意 – 必须使用 -p 选项或 -u 选项。如果不指定这两个选项之一，将出现错误。

安装程序选项

选项	说明
-a	从中复制系统映像的归档文件位置。支持完整 Flash 归档文件以及 pax、cpio、gzip 压缩的 cpio 、bzip 压缩的 cpio 和 0 级 ufsdump。
-c *path*	传送用于在安装完成后配置区域的 sysidcfg 文件。
-d *path*	从中复制系统映像的目录位置。
-d –	将 -d 选项与破折号参数一起使用可指示要在 zonepath 中使用的现有目录布局。因此，如果管理员在安装前手动设置 zonepath 目录，则 -d – 选项可用于指示该目录已经存在。
-p	保留系统标识。 必须使用 -p 或 -u。
-s	无提示安装。
-u	对区域执行 sys-unconfig。 必须使用 -p 或 -u。 除 -u 选项外，还可以使用 -c 来传送用于在安装完成后配置区域的 sysidcfg 文件。
-v	详细输出。

-a 和 -d 选项相互排斥。

▼ 如何安装 solaris10 标记区域

可使用 zoneadm 命令和 install 子命令来安装已配置的 solaris10 标记区域。

有关创建 Oracle Solaris 10 系统映像的信息，请参见第 376 页中的 “为将 Oracle Solaris 10 系统直接迁移到区域中创建映像”。要在所创建的系统映像中保留 sysidcfg 标识而不改变映像，请在 -install 子命令之后使用 p 选项。要从所创建的系统映像中删除系统标识而不改变映像，请使用 -u 选项。将对目标区域执行 sys-unconfig。-c 选项可用来包含 sysidcfg 文件，该文件所含的信息可用于在安装完成后配置区域。

示例过程显示了如何对所创建的已安装物理 Oracle Solaris 10 系统的归档文件映像使用 -a 选项。

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**

2. **通过将 zoneadm install 命令与 -p 和 -a 选项和归档文件路径组合使用，安装已配置区域 s10-zone：**

   ```
   global# zoneadm -z s10-zone install -p -a /net/machine_name/s10-system.flar -u
   ```

   安装完成时，将显示多条消息。这可能需要一些时间。

3. **（可选的）如果出现一个错误消息，并且无法安装区域，请使用 zoneadm list 命令以及 -c 和 -v 选项来获取区域状态：**

   ```
   global# zoneadm list -civ

   ID  NAME     STATUS       PATH                           BRAND      IP
    0  global   running      /                              solaris    shared
    -  s10-zone configured   /zones/s10-zone                solaris10  shared
   ```

   - 如果显示为已配置状态，请执行消息中指定的更正操作，并再次尝试执行 zoneadm install 命令。
   - 如果显示为未完成状态，请首先执行以下命令：

     ```
     global# zoneadm -z my-zone uninstall
     ```

     然后执行消息中指定的更正操作，并再次尝试 zoneadm install 命令。

4. **当安装完成时，使用带有 -i 和 -v 选项的 list 子命令来列出已安装的区域并检验状态。**

   ```
   global# zoneadm list -iv
   ```

   将显示以下类似信息：

   ```
   ID  NAME     STATUS       PATH                           BRAND      IP
    0  global   running      /                              solaris     shared
    -  s10-zone  installed    /zones/s10-zone                solaris10   shared
   ```

示例 33–1 solaris10 区域安装

```
# zoneadm -z s10sepvar install -p -a /net/data13/tmp/s10u10_sparc_sepvar.flar -u
The following ZFS file system(s) have been created:
    rpool/zones/s10sepvar
Progress being logged to /var/log/zones/zoneadm.20120519T151123Z.s10sepvar.install
    Installing: This may take several minutes...
```

故障排除 如果安装失败，请查看日志文件。如果成功，则日志文件位于区域内的 `/var/log` 中。如果失败，则日志文件位于全局区域内的 `/var/log/zones` 中。

如果区域安装中断或失败，则此区域会处于未完成状态。请使用带有 `-F` 选项的 `uninstall` 命令将区域重置为已配置状态。

◆ ◆ ◆ 第 34 章

34 引导区域、登录和区域迁移

本章介绍如何引导已安装的区域并使用 zlogin 完成内部区域配置。此外，还讨论了如何将区域迁移到另一台计算机。

关于引导 solaris10 标记区域

引导区域会使区域步入运行状态。既可以从就绪状态引导区域，也可以从已安装状态引导区域。处于已安装状态的区域经透明引导，会从就绪状态转换为正在运行状态。允许登录到处于正在运行状态下的区域。

请注意，在初始引导后首次登录到未配置的区域时，即会执行内部区域配置。

映像 sysidcfg 配置文件

如果利用现有系统创建了 Oracle Solaris 10 系统归档文件，并在安装区域时使用 -p（保留 sysidcfg）选项，该区域将与用于创建映像的系统具有相同的标识。

-c 选项可用来包含用于在安装完成后配置区域的 sysidcfg 文件。要安装 solaris10 区域，请在命令行中使用 sysidcfg 文件。请注意必须提供文件的完整路径。

```
# zoneadm -z s10-zone install -a /net/machine_name/s10-system.flar -u -c /path_to/sysidcfg
```

以下样例 sysidcfg 文件使用 net0 网络名称和 timezone 来配置具有静态 IP 配置的专用 IP 区域。

```
system_locale=C
 terminal=xterm
 network_interface=net0 {
    hostname=test7
    ip_address=192.168.0.101
    netmask=255.255.255.0
    default_route=NONE
```

```
    protocol_ipv6=no
 }
 name_service=NONE
 security_policy=NONE
 timezone=US/Pacific
 timeserver=localhost
 nfs4_domain=dynamic
 root_password=FSPXl81aZ7Vyo
 auto_reg=disable
```

以下样例 `sysidcfg` 文件用于配置共享 IP 区域：

```
system_locale=C
terminal=dtterm
network_interface=primary {
hostname=my-zone
}
security_policy=NONE
name_service=NIS {
domain_name=special.example.com
name_server=bird(192.168.112.3)
}
nfs4_domain=domain.com
timezone=US/Central
root_password=m4qtoWN
```

以下样例 `sysidcfg` 文件用于配置具有静态 IP 配置的专用 IP 区域。

```
system_locale=C
terminal=dtterm
network_interface=primary {
hostname=my-zone
default_route=10.10.10.1
ip_address=10.10.10.13
netmask=255.255.255.0
}
nfs4_domain=domain.com
timezone=US/Central
root_password=m4qtoWN
```

以下样例 `sysidcfg` 文件用于配置具有 DHCP 和 IPv6 选项的专用 IP 区域。

```
system_locale=C
terminal=dtterm
network_interface=primary {
dhcp protocol_ipv6=yes
}
security_policy=NONE
name_service=DNS {
domain_name=example.net
name_server=192.168.224.11,192.168.224.33
}
nfs4_domain=domain.com
timezone=US/Central
root_password=m4qtoWN
```

▼ solaris10 标记区域内部配置

未提供配置文件时，则配置工具将在首次使用 `zlogin -C` 时启动。

此过程中区域的名称为 `s10-zone`。

1. **成为 root 用户或承担等效角色。**
2. **在一个终端窗口中，在引导区域前通过以下命令连接到区域控制台（在本过程中为 `s10-zone`）：**

   ```
   # zlogin -C s10-zone
   ```

3. **在第二个窗口中，按照第 397 页中的“如何引导 solaris10 标记区域”中所述引导区域。**

▼ 如何引导 solaris10 标记区域

您必须是全局管理员或在全局区域中具有相应授权的用户才能执行此过程。

1. **成为 root 用户或承担等效角色。**
2. **使用带有 `z` 选项、区域名称 `-s10-zone` 以及 `boot` 子命令的 `zoneadm` 命令引导区域。**

   ```
   global# zoneadm -z s10-zone boot
   ```

3. **当引导完成时，使用带有 `-v` 选项的 `list` 子命令检验状态。**

   ```
   global# zoneadm list -v
   ```

   将显示以下类似信息：

   ```
   ID  NAME     STATUS       PATH                           BRAND      IP
    0  global   running      /                              solaris     shared
    1  s10-zone running      /zone/s10-zone                 solaris10   shared
   ```

另请参见 有关引导区域和引导选项的更多信息，请参见第 19 章，安装、引导、关闭、停止、卸载和克隆非全局区域（任务）。

将 solaris10 标记区域迁移至另一台主机

使用带有 `detach` 和 `attach` 子命令的 `zoneadm` 命令可将 `solaris10` 区域迁移至另一台主机。此过程在第 295 页中的“关于迁移区域”和第 296 页中的“如何使用 ZFS 归档文件迁移非全局区域”中介绍。

请注意，`zoneadm attach -a` 命令将对区域而**不是**物理系统进行归档。

词汇表

blessed（隶属） 在 Perl 中，用来表示对象的类成员关系的术语。

branded zone（标记区域） 一种隔离环境，用于在非全局区域中运行非本机应用程序。

brand（标记） BrandZ 功能的实例，提供了包含用于运行应用程序的非本机操作环境的非全局区域。

capping（上限设置） 针对系统资源使用设定限制的过程。

cap（上限） 针对系统资源使用设定的限制。

data-link（数据链路） OSI 协议栈的第二层接口，在系统中表示为 STREAMS DLPI (v2) 接口。该接口可以在 TCP/IP 等协议栈下检测到。在 Oracle Solaris 10 Zones 环境中，数据链路为物理接口、集合或带 VLAN 标记的接口。数据链路也称为物理接口，例如，涉及 NIC 或 VNIC 时。

default pool（缺省池） 启用池时由系统创建的池。

另请参见 resource pool（资源池）。

default processor set（缺省处理器集） 启用池时由系统创建的处理器集。

另请参见 processor set（处理器集）。

disjoint（不相交） 其成员不重叠并且不重复的一类集合。

dynamic configuration（动态配置） 某一时刻，给定系统中资源池框架内的资源部署的相关信息。

dynamic reconfiguration（动态重新配置） 在基于 SPARC 的系统上，当系统运行时重新配置硬件的功能。也称为 DR。

extended accounting（扩展记帐） 在 Solaris 操作系统中，按任务或进程来记录资源占用情况的一种比较灵活的方法。

fair share scheduler（公平份额调度器） 一个调度类，也称为 FSS，可用于分配基于份额的 CPU 时间。份额定义了分配给某个项目的那一部分系统 CPU 资源。

FSS 请参见 fair share scheduler（公平份额调度器）。

global administrator（全局管理员） root 用户或具有 root 角色的管理员。登录到全局区域后，全局管理员或被授予相应权限的用户可以将系统作为一个整体进行监视和控制。

另请参见 zone administrator（区域管理员）。

global scope（全局范围） 应用于系统上所有资源控制的资源控制值的操作。

global zone（全局区域） 所有 Oracle Solaris 系统上都包含的区域。使用非全局区域时，全局区域既是系统的缺省区域，也是用于系统范围内管理控制的区域。

另请参见 non-global zone（非全局区域）。

heap（堆） 进程分配的临时内存。

local scope（本地范围） 对试图超越控制值的进程采取的本地操作。

locked memory（锁定内存） 不能执行调页操作的内存。

memory cap enforcement threshold（内存上限执行阈值） 系统中的物理内存使用百分比，该值将触发资源上限设置守护进程执行上限。

naming service database（命名服务数据库） 在本文档的“项目和任务（概述）”一章中，指 LDAP 容器和 NIS 映射。

non-global zone administrator（非全局区域管理员） 请参见 zone administrator（区域管理员）。

non-global zone（非全局区域） 在 Oracle Solaris 操作系统的单个实例中创建的虚拟操作系统环境。Oracle Solaris Zones 软件分区技术用于虚拟化操作系统服务。

Oracle Solaris 10 Zones 适用于在运行 Oracle Solaris 11 发行版的系统上的 `solaris10` 标记区域中执行的 Solaris 10 应用程序的完整运行时环境。

Oracle Solaris Zones 用于虚拟化操作系统服务的软件分区技术，提供运行应用程序的安全隔离环境。

pool daemon（池守护进程） 需要动态分配资源时处于活动状态的 `poold` 系统守护进程。

pool（池） 请参见 resource pool（资源池）。

processor set（处理器集） 不相交的 CPU 分组。每个处理器集都可以包含零个或多个处理器。在资源池配置中，一个处理器集表示为一个资源元素。也称为 pset。

另请参见 disjoint（不相交）。

project 相关工作在网络范围内的管理标识符。

read-only zone（只读区域） 配置为只读根的不可编辑区域。

resident set size（驻留集大小） 驻留集的大小。驻留集是驻留在物理内存中的一组页面。

resource capping daemon（资源上限设置守护进程） 一种守护进程，用于调节已定义资源上限的项目中运行的进程所占用的物理内存。

resource consumer（资源使用者） 实际上是指 Solaris 进程。利用进程模型实体（例如项目和任务），可以从总资源占用角度讨论资源占用情况。

resource control（资源控制） 对每个进程、任务或项目设置的资源占用限制。

resource management（资源管理） 可用于控制应用程序如何使用可用系统资源的功能。

resource partition（资源分区） 一个专用的资源子集。资源的所有分区加起来表示正在执行的单个 Solaris 实例中的可用资源总量。

resource pool（资源池） 用于对计算机资源进行分区的配置机制。资源池表示各组可分区资源之间的关联。

resource set（资源集） 可绑定到进程的资源。通常指提供某种分区形式的内核子系统构造的对象。资源集的示例包括调度类和处理器集。

resource（资源） 计算系统的一个方面，可对其进行处理以更改应用程序行为。

RSS 请参见 resident set size（驻留集大小）。

scanner（扫描程序） 标识不常用页面的内核线程。在低内存情况下，扫描程序会回收最近未使用的页面。

static pools configuration（静态池配置） 一种管理员希望如何针对资源池功能对系统进行配置的方法。

task（任务） 在资源管理中，表示一段时间内一组工作的进程集。每项任务都与一个项目关联。

whole root zone（完全根区域） 一种非全局区域类型，在此区域中，所有必需的系统软件和任何附加软件包都安装在该区域的专有文件系统中。

working set size（工作集大小） 工作集的大小。工作集是指在处理项目工作负荷过程中实际使用的一组页面。

workload（工作负荷） 一个或一组应用程序的所有进程的集合。

WSS 另请参见 working set size（工作集大小）。

zone administrator（区域管理员） 区域管理员的特权仅限于某个非全局区域。

另请参见 global administrator（全局管理员）。

zone state（区域状态） 非全局区域的状态。区域状态可以是“已配置”、“未完成”、“已安装”、“就绪”、“不可用”、“正在运行”或“正在关闭”中的一种。

索引