系统管理指南：Solaris Containers－资源管理和 Solaris Zones

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

文件号码 819–6955–12
2008 年 4 月

080411@19860

目录

前言

《系统管理指南：Solaris Containers－资源管理和 Solaris Zones》是多卷集的一部分，该多卷集包含 Solaris™ 操作系统管理信息的重要部分。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。

注 – 此 Solaris 发行版支持使用以下 SPARC® 和 x86 系列处理器体系结构的系统：UltraSPARC®、SPARC64、AMD64、Pentium 和 Xeon EM64T。支持的系统可以在 http://www.sun.com/bigadmin/hcl 上的 Solaris OS: Hardware Compatibility Lists 中找到。本文档列举了在不同类型的平台上进行实现时的所有差别。

关于 Solaris Containers

Solaris Container 是一个完整的应用程序运行时环境。Solaris 10 资源管理器和 Solaris Zones 软件分区技术是该容器的两个组成部分。这两个组件分别代表该容器能够实现的两种不同的功能，两者协同工作可以创建完整的容器环境。该容器的区域部分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 Solaris 操作系统的单个实例也是如此。利用资源管理功能，您可以对工作负荷收到的资源数量进行分配。

该容器建立资源占用（如 CPU）的边界。这些边界可以进行扩展，以适应容器中运行的应用程序的不断变化的处理要求。

Solaris 10 8/07：关于 Solaris Containers for Linux Applications

Solaris Containers for Linux Applications 使用 Sun 的 BrandZ 技术在 Solaris 操作系统上运行 Linux 应用程序。Linux 应用程序在非全局区域功能所提供的安全环境中运行，不会被修改。这样，您可使用 Solaris 系统来开发、测试和部署 Linux 应用程序。

有关使用此功能的信息，请参见第 3 部分。

Solaris 10 11/06 及更高版本：关于在 Solaris Trusted Extensions（高可靠扩展版）系统上使用 Solaris Zones

有关在 Solaris Trusted Extensions（高可靠扩展版）系统上使用区域的信息，请参见《Solaris Trusted Extensions Administrator's Procedures》中的第 10 章 "Managing Zones in Trusted Extensions (Tasks)"。

目标读者

本书适用于所有负责管理一个或多个运行 Solaris 10 发行版的系统的人员。要使用本书，您应当至少具备一到两年的 UNIX® 系统管理经验。

系统管理系列书籍的结构

下面是系统管理指南系列书籍中各本书包含的主题列表。

书名	主题
《系统管理指南：基本管理》	用户帐户和组、服务器和客户机支持、关闭和引导系统、管理服务、管理软件（软件包和修补程序）
《系统管理指南：高级管理》	打印服务、终端和调制解调器、系统资源（磁盘配额、记帐和 crontab）、系统进程，以及对 Solaris 软件问题进行疑难解答
《系统管理指南：设备和文件系统》	可移除介质、磁盘和设备、文件系统、备份和恢复数据
《系统管理指南：IP 服务》	TCP/IP 网络管理、IPv4 和 IPv6 地址管理、DHCP、IPsec、IKE、IP 过滤器、移动 IP、IP 网络多路径 (IP network multipathing, IPMP) 和 IPQoS
《系统管理指南：名称和目录服务（DNS、NIS 和 LDAP）》	DNS、NIS 和 LDAP 命名和目录服务，包括从 NIS 到 LDAP 的转换和从 NIS+ 到 LDAP 的转换
《System Administration Guide: Naming and Directory Services (NIS+)》	NIS+ 命名和目录服务
《系统管理指南：网络服务》	Web 高速缓存服务器、与时间相关的服务、网络文件系统（NFS 和 Autofs）、邮件、SLP 和 PPP
《系统管理指南：安全性服务》	审计、设备管理、文件安全、BART、Kerberos 服务、PAM、Solaris 加密框架、权限、RBAC、SAS L和 Solaris 安全 Shell

书名	主题
《系统管理指南：Solaris Containers－资源管理和 Solaris Zones》	资源管理主题项目和任务、扩展记帐、资源控制、公平份额调度器 (fair share scheduler, FSS)、使用资源上限设置守护进程 (rcapd) 的物理内存控制，以及资源池；使用 Solaris Zones 软件分区技术的虚拟化
《Solaris ZFS 管理指南》	ZFS 存储池和文件系统创建及管理、快照、克隆、备份、使用访问控制列表 (access control list, ACL) 保护 ZFS 文件、在安装了区域的 Solaris 系统上使用 Solaris ZFS、模仿卷、疑难解答和数据恢复

相关的第三方 Web 站点引用

本文档引用了第三方 URL 以提供其他相关信息。

注 – Sun 对本文档中提到的第三方 Web 站点的可用性不承担任何责任。对于此类站点或资源中的（或通过它们获得的）任何内容、广告、产品或其他资料，Sun 并不表示认可，也不承担任何责任。对于因使用或依靠此类站点或资源中的（或通过它们获得的）任何内容、产品或服务而造成的、宣称的或连带产生的实际或名义损坏或损失，Sun 概不负责，也不承担任何责任。

文档、支持和培训

Sun Web 站点提供有关以下附加资源的信息：

- 文档 (http://www.sun.com/documentation/)
- 支持 (http://www.sun.com/support/)
- 培训 (http://www.sun.com/training/)

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.`
`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

下表列出了 C shell、Bourne shell 和 Korn shell 的缺省 UNIX 系统提示符和超级用户提示符。

表 P–2 shell 提示符

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

第 1 部 分

资源管理

此部分介绍 Solaris 10 Resource Manager，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 Solaris 10 Resource Manager 介绍

资源管理功能是 Solaris™ Container 环境的一个组件。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。其中包括：

- 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，Solaris 操作系统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 Solaris 资源管理功能，您可以分别对待各个工作负荷。其中包括：

- 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。其中包括：

- 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 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

Solaris 操作系统可用的三种控制机制为约束、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 章中介绍的资源控制功能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。基础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 Solaris Zones 交互

资源管理功能可与 Solaris 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 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冲区和输入设备，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（任务图）

以下任务图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 39 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。	请参考您的业务服务目标。
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。	第 178 页中的 “按系统进行监视” 以及 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，请参考 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 74 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” 以及 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，也可以使用 Solaris Management Console 设置某些配置参数。	第 39 页中的 “project 数据库”、第 40 页中的 “本地 /etc/project 文件格式”、第 74 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
为项目附加的进程集所占用的物理内存资源设置上限。	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 39 页中的 “project 数据库” 和第 10 章
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 39 页中的 “project 数据库” 和第 12 章
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 106 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 “如何激活进程、任务和流的扩展记帐” 和 acctadm(1M) 手册页
（可选）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）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，来开发自定义报告和摘录脚本。	wracct(1M) 手册页和第 61 页中的 “libexacct 的 Perl 接口”

◆ ◆ ◆ 第 2 章

项目和任务（概述）

本章讨论 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。

Solaris 10 在项目数据库和资源控制命令方面的新增功能

Solaris 10 增强功能包括：

- 针对资源控制值和命令扩展了对值和单元修饰符的支持
- 对项目属性字段进行验证和处理更加简便
- 修改了 `prctl` 和 `projects` 命令的输出格式并引入了新选项
- 可通过 `useradd` 命令设置用户的缺省项目，并使用 `usermod` 和 `passmgmt` 命令修改信息

除了本章和第 6 章中包含的信息外，还可以参见以下手册页：

- `passmgmt`(1M)
- `projadd`(1M)
- `projmod`(1M)
- `useradd`(1M)
- `usermod`(1M)
- `resource_controls`(5)

Solaris 10 5/08 增强功能包括为 `projmod` 命令添加了 `-A` 选项。请参见第 44 页中的 “用于项目和任务的命令” 。

有关 Solaris 10 新增功能的完整列表以及 Solaris 发行版的说明，请参见《Solaris 10 新增功能》。

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 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 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. 如果用户在 `/etc/user_attr` 扩展用户属性数据库中具有定义了 `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、usermod 和 passmgmt 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：

passmgmt　修改用户信息

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 passmgmt(1M)、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、网络信息服务 (Network Information Service, NIS) 项目映射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。

项目的帐户管理 PAM 模块在 pam_projects(5) 手册页中进行介绍。有关 PAM 的概述，请参见《系统管理指南：安全性服务》中的第 16 章 “使用 PAM”。

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《系统管理指南：名称和目录服务（DNS、NIS 和 LDAP）》中的第 2 章 “名称服务转换器（概述）”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，则登录进程会在 /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 (-) 和圆点 (.) 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）。*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。不会分配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:System:::
user.root:1:Super-User:::
noproject:2:No Project:::
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:System:::
user.root:1:Super-User:::
noproject:2:No Project:::
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 “限制物理内存使用率的属性”。
- 要将 project.pool 属性添加到项目条目，请参见第 168 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《系统管理指南：名称和目录服务（DNS、NIS 和 LDAP）》中的第 4 章 “网络信息服务 (Network Information Service, NIS)（概述）”。

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《系统管理指南：名称和目录服务（DNS、NIS 和 LDAP）》中的第 8 章 “LDAP 名称服务介绍（概述/参考）”。有关 LDAP 数据库中项目条目结构的更多信息，请参见《系统管理指南：名称和目录服务（DNS、NIS 和 LDAP）》中的“Solaris 架构”。

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进程的新**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。

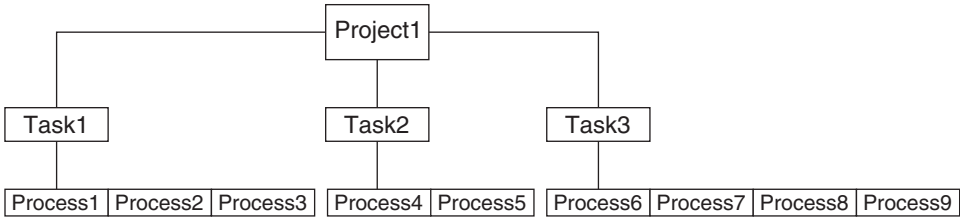


图 2–1 项目和任务树

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。
passmgmt(1M)	更新口令文件中的信息。与 -K *key*=*value* 选项一起使用，可在本地文件中添加或替换用户属性。
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。 从 Solaris 10 5/08 发行版开始，可与 -A 选项一起使用，以便将在项目数据库中找到的资源控制值应用到活动项目。与 project 文件中定义的值不匹配的现有值（例如通过 prctl 命令手动设置的值）都会被删除。
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。
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 Solaris 资源管理中的项目和任务功能。

本章包含以下主题：

有关项目和任务功能的概述，请参见第 2 章。

注 – 如果在安装了区域的 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（任务图）

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 46 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 48 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 51 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 52 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 53 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 53 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 53 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 54 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
 21634 jtd      5512K 4848K cpu0    44    0   0:00.00 0.3% prstat/1
   324 root       29M   75M sleep   59    0   0:08.27 0.2% Xsun/1
 15497 jtd        48M   41M sleep   49    0   0:08.26 0.1% adeptedit/1
   328 root     2856K 2600K sleep   58    0   0:00.00 0.0% mibiisa/11
  1979 jtd      1568K 1352K sleep   49    0   0:00.00 0.0% csh/1
  1977 jtd      7256K 5512K sleep   49    0   0:00.00 0.0% dtterm/1
   192 root     3680K 2856K sleep   58    0   0:00.36 0.0% automountd/5
  1845 jtd        24M   22M sleep   49    0   0:00.29 0.0% dtmail/11
  1009 jtd      9864K 8384K sleep   49    0   0:00.59 0.0% dtwm/8
   114 root     1640K  704K sleep   58    0   0:01.16 0.0% in.routed/1
   180 daemon   2704K 1944K sleep   58    0   0:00.00 0.0% statd/4
   145 root     2120K 1520K sleep   58    0   0:00.00 0.0% ypbind/1
   181 root     1864K 1336K sleep   51    0   0:00.00 0.0% lockd/1
   173 root     2584K 2136K sleep   58    0   0:00.00 0.0% inetd/1
   135 root     2960K 1424K sleep    0    0   0:00.00 0.0% keyserv/4
PROJID    NPROC  SIZE   RSS MEMORY      TIME  CPU PROJECT
    10       52  400M  271M    68%   0:11.45 0.4% booksite
     0       35  113M  129M    32%   0:10.46 0.2% system

Total: 87 processes, 205 lwps, load averages: 0.05, 0.02, 0.02
```

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
 23023 root       26M   20M sleep   59    0   0:03:18 0.6% Xsun/1
 23476 jtd        51M   45M sleep   49    0   0:04:31 0.5% adeptedit/1
 23432 jtd      6928K 5064K sleep   59    0   0:00:00 0.1% dtterm/1
 28959 jtd        26M   18M sleep   49    0   0:00:18 0.0% .netscape.bin/1
 23116 jtd      9232K 8104K sleep   59    0   0:00:27 0.0% dtwm/5
 29010 jtd      5144K 4664K cpu0    59    0   0:00:00 0.0% prstat/1
   200 root     3096K 1024K sleep   59    0   0:00:00 0.0% lpsched/1
   161 root     2120K 1600K sleep   59    0   0:00:00 0.0% lockd/2
   170 root     5888K 4248K sleep   59    0   0:03:10 0.0% automountd/3
   132 root     2120K 1408K sleep   59    0   0:00:00 0.0% ypbind/1
   162 daemon   2504K 1936K sleep   59    0   0:00:00 0.0% statd/2
   146 root     2560K 2008K sleep   59    0   0:00:00 0.0% inetd/1
```

```
   122 root     2336K 1264K sleep   59    0   0:00:00 0.0% keyserv/2
   119 root     2336K 1496K sleep   59    0   0:00:02 0.0% rpcbind/1
   104 root     1664K  672K sleep   59    0   0:00:03 0.0% in.rdisc/1
TASKID    NPROC  SIZE   RSS MEMORY      TIME  CPU PROJECT
   222       30  229M  161M    44%   0:05:54 0.6% group.staff
   223        1   26M   20M   5.3%   0:03:18 0.6% group.staff
    12        1   61M   33M   8.9%   0:00:31 0.0% group.staff
     1       33   85M   53M    14%   0:03:33 0.0% system

Total: 65 processes, 154 lwps, load averages: 0.04, 0.05, 0.06
```

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 成为超级用户或承担等效角色。

角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 使用 `projects -l` 查看系统上缺省的 `/etc/project` 文件。

```
# projects -l
system:0::::
user.root:1::::
noproject:2::::
default:3::::
group.staff:10::::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 163 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. **成为超级用户或承担等效角色。**

   角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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，请以超级用户的身份登录，并键入以下内容：

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

注 – `ypcat project | projmod -f –` 命令尚未实现。

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 *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。要执行此操作，您必须是超级用户，或者是进程属主并且是新项目的成员。

1 **成为超级用户或承担等效角色。**

角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

注 – 如果您是进程属主或新项目成员，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 **成为超级用户或承担等效角色。**

角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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。此值包含权限级别、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

此过程假设具有以下值：

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

1. **成为超级用户或承担等效角色。**
   角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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. **成为超级用户或承担等效角色。**
   角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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. **成为超级用户或承担等效角色。**
   角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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 **成为超级用户或承担等效角色。**

角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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，请参见第 64 页中的 “如何激活进程、任务和流的扩展记帐”。

Solaris 10 在扩展记帐方面的新增功能

现在可以生成进程记帐的 mstate 数据。请参见第 65 页中的 “如何查看可用的记帐资源”。

有关 Solaris 10 新增功能的完整列表以及 Solaris 发行版的说明，请参见《Solaris 10 新增功能》。

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《系统管理指南：IP 服务》中的第 36 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。Solaris 操作系统中的扩展记帐功能提供了一种灵活方法，可按任务或进程或者按 IPQoS `flowacct` 模块提供的选定器来记录系统和网络资源占用情况。有关更多信息，请参见 `ipqos`(7IPP)。

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

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，可以开发自定义报告和摘录脚本。请参见第 61 页中的“`libexacct` 的 Perl 接口”。

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。

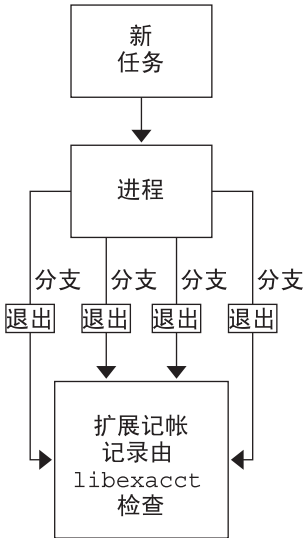


图 4–1 在激活了扩展记帐情况下的任务跟踪

可扩展的格式

扩展记帐格式实际上比 SunOS™ 传统系统记帐软件格式（请参见《系统管理指南：高级管理》中的“什么是系统记帐？”）更具可扩展性。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)）创建嵌入在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 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。全局管理员还可以确定每个区域的资源占用情况。有关更多信息，请参见第 317 页中的 “安装了区域的 Solaris 系统上的扩展记帐”。

扩展记帐配置

/etc/acctadm.conf 文件中包含当前的扩展记帐配置。此文件通过 acctadm 接口而不是用户进行编辑。

目录 /var/adm/exacct 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)。

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、任务和流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 46 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)。

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。

此接口与基础 C API 在功能上是等效的。如果可能，通过基础 C API 获取的数据将显示为 Perl 数据类型。使用此功能，可以更轻松地访问数据，并且无需进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 Sun 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。

基础 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	此模块提供了访问项目操作函数 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) 的功能。	Project(3PERL)
Sun::Solaris::Task	此模块提供了访问任务操作函数 gettaskid(2) 和 settaskid(2) 的功能。	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)

模块（不应包含空格）	说明	更多信息
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) 函数的一些方法。	Exacct::Object(3PERL)
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。

管理扩展记帐功能（任务图）

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 “如何激活进程、任务和流的扩展记帐” 、第 64 页中的 “如何使用启动脚本激活扩展记帐”
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”
取消激活进程、任务和流的记帐功能。	禁用扩展记帐功能。	第 65 页中的 “如何取消激活进程记帐、任务记帐和流记帐”
将 Perl 接口用于扩展记帐功能。	使用 Perl 接口开发自定义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

▼ 如何激活进程、任务和流的扩展记帐

要激活任务、进程和流的扩展记帐功能，请使用 acctadm 命令。acctadm 的可选最终参数表示此命令是应该针对扩展记帐功能的进程记帐组件、系统任务记帐组件还是流记帐组件执行。

1 **成为超级用户或承担等效角色。**

角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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
```

另请参见 有关更多信息，请参见 acctadm(1M)。

如何使用启动脚本激活扩展记帐

通过将 /etc/init.d/acctadm 脚本链接到 /etc/rc2.d，可以在运行时激活扩展记帐。

```
# ln -s /etc/init.d/acctadm /etc/rc2.d/Snacctadm
# ln -s /etc/init.d/acctadm /etc/rc2.d/Knacctadm
```

将 *n* 变量以数字替换。

必须至少手动激活一次扩展记帐以设置配置。

有关记帐配置的信息，请参见第 60 页中的 “扩展记帐配置”。

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。

```
# 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。

```
# 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
```

▼ 如何取消激活进程记帐、任务记帐和流记帐

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

1. **成为超级用户或承担等效角色。**

   角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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. **检验是否已禁用任务记帐、进程记帐和流记帐。**

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

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 exact 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。

Solaris 10 在资源控制方面的新增功能

以下一组资源控制取代了 System V 进程间通信 (interprocess communication, IPC) `/etc/system` 可调参数：

- `project.max-shm-ids`
- `project.max-msg-ids`
- `project.max-sem-ids`
- `project.max-shm-memory`
- `process.max-sem-nsems`
- `process.max-sem-ops`
- `process.max-msg-qbytes`

已添加了以下事件端口资源控制：

- `project.max-device-locked-memory`
- `project.max-port-ids`
- `process.max-port-events`

已添加了以下加密资源控制：

- `project.max-crypto-memory`

已添加了以下其他资源控制：

- `project.max-lwps`
- `project.max-tasks`
- `project.max-contracts`

有关更多信息，请参见第 74 页中的 “可用的资源控制” 。

有关 Solaris 10 新增功能的完整列表以及 Solaris 发行版的说明，请参见《Solaris 10 新增功能》。

资源控制概念

在 Solaris 操作系统中，每进程资源限制的概念已扩展到第 2 章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 74 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 210 页中的 “资源类型属性” 。

有关此发行版中提供的标准资源控制的列表，请参见第 74 页中的 “可用的资源控制” 。

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、每个进程的核心转储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。在 Solaris 10 之前的发行版中，IPC 可调参数是通过向 `/etc/system` 文件中添加条目来设置的。现在，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。

此 Solaris 系统上的 `/etc/system` 文件中可能包含过时参数。如果是这样，这些参数将像在以前的 Solaris 发行版中一样用来初始化缺省的资源控制值。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 Solaris 系统调优的信息，请参见《Solaris Tunable Parameters Reference Manual》。

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。

配置资源控制和属性

资源控制功能通过 project 数据库来配置。请参见第 2 章。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权限级别、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 207 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) 系统调用分配给进程的最高文件编号。

表 6–1 标准资源控制

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。 请注意，在 Solaris 10 8/07 发行版中，此资源控制取代了 project.max-device-locked-memory，后者已被删除。	大小（字节）
project.max-port-ids	允许的最大事件端口数。	数量（事件端口数）
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）
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-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)
process.max-cpu-time	此进程可用的最长 CPU 时间。	时间（秒）

表 6–1 标准资源控制 （续）

控制名称	说明	缺省单位
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转储文件大小。	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 201 页中的 “设置区域范围的资源控制”和第 221 页中的 “如何配置区域”中所述。

表 6–2 区域范围的资源控制

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-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-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 210 页中的 “资源类型属性”和第 221 页中的 “如何配置区域”。要在 lx 标记区域中使用区域范围的资源控制，请参见第 387 页中的 “如何配置、检验和提交 lx 标记区域”。

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 17 章和第 345 页中的 “在安装了区域的 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}
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	无	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。

资源控制值和权限级别

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权限级别相关联。权限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属主修改
- 特权，此类型的权限级别仅可由特权（超级用户）调用方修改
- 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权限值，但仅允许定义一个基本值。缺省情况下，将为没有指定权限值时执行的操作指定基本权限。

资源控制值的权限级别在资源控制块（如 RCTL_BASIC、RCTL_PRIVILEGED 或 RCTL_SYSTEM）的权限字段中定义。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。在 Solaris 10 5/08 发行版中，为无法配置全局操作的资源控制添加了级别 `n/a`。

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	备注
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。	

表 6–3 可用于资源控制值的信号 （续）

信号	说明	备注
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 不需要超级用户权限来减小此控制的权限值。

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 不能设置权限类型为 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。

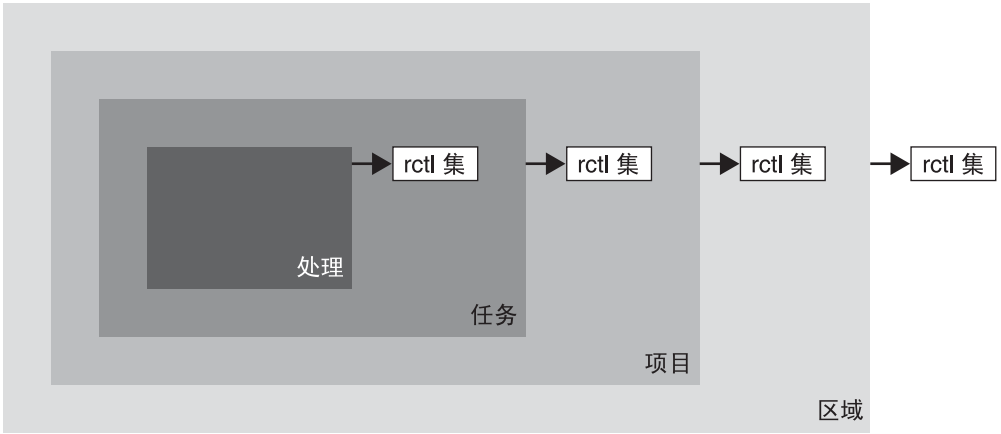


图 6–1 进程集合、容器关系及其资源控制集

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权限值，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，但是具有超级用户权限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

7 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。

管理资源控制（任务图）

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 94 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 **成为超级用户或承担等效角色。**

角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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，如果超级用户在项目 x-files 中创建新任务（通过 newtask 加入项目），则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
        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 **成为超级用户或承担等效角色。**

角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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     文件中的一行
```

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: 100337: -sh
NAME    PRIVILEGE       VALUE    FLAG   ACTION                     RECIPIENT
```

```
process.max-port-events
        privileged      65.5K       -   deny                                 -
        system          2.15G     max   deny                                 -
process.crypto-buffer-limit
        system          16.0EB    max   deny                                 -
process.max-crypto-sessions
        system          18.4E     max   deny                                 -
process.add-crypto-sessions
        privileged        100       -   deny                                 -
        system          18.4E     max   deny                                 -
process.min-crypto-sessions
        privileged         20       -   deny                                 -
        system          18.4E     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                             100337
        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           8.00MB      -   deny                             100337
        privileged      8.00EB      -   deny                                 -
        system          8.00EB    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
        system          18.4Es    inf   none                           -
task.max-lwps
        system          2.15G     max   deny                           -
project.max-contracts
        privileged      10.0K       -   deny                           -
        system          2.15G     max   deny                           -
project.max-device-locked-memory
        privileged       499MB      -   deny                           -
        system          16.0EB    max   deny                           -
project.max-port-ids
        privileged      8.19K       -   deny                           -
        system          65.5K     max   deny                           -
project.max-shm-memory
        privileged      1.95GB      -   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-tasks
        system          2.15G     max   deny                           -
project.max-lwps
        system          2.15G     max   deny                           -
project.cpu-shares
        privileged          1       -   none                           -
        system          65.5K     max   none                           -
zone.max-lwps
        system          2.15G     max   deny                           -
zone.cpu-shares
        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            110453
```

```
        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权限值，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. **成为超级用户或承担等效角色。**

   角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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 添加一个新的权限值，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
           privileged       3       -   deny                     -
           system       2.15G     max   deny                     -
   ```

▼ 如何使用 prctl 降低资源控制值

1. **成为超级用户或承担等效角色。**

   角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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 **成为超级用户或承担等效角色。**

角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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
        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
        privileged         10       -   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 命令对包含 httpd 进程的任务设置特权（超级用户拥有）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。

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。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系统上使用 `project.cpu-shares` 资源控制，请参见第 207 页中的 “区域配置数据”、第 316 页中的 “在非全局区域中使用的资源控制”和第 345 页中的 “在安装了区域的 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 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。

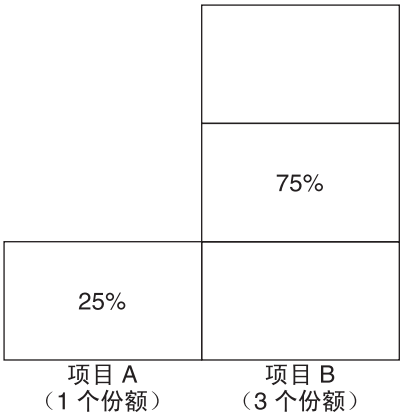
在传统的 TS 调度程序下，会为系统上正在运行的各个工作负荷平均地提供相同的 CPU 资源量。每个工作负荷将使用 50% 的系统容量。

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。

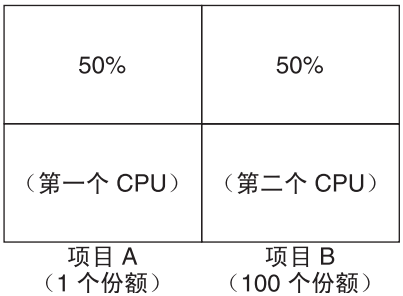
示例 1：每个项目中有两个计算密集型 (CPU-bound) 进程

如果 *A* 和 *B* 各有两个计算密集型 (CPU-bound) 进程，并且 $S_A = 1$，$S_B = 3$，则份额总数是 1 + 3 = 4。在此配置中，假设具有足够的 CPU，则为项目 *A* 和 *B* 分配的 CPU 资源分别为 25% 和 75%。



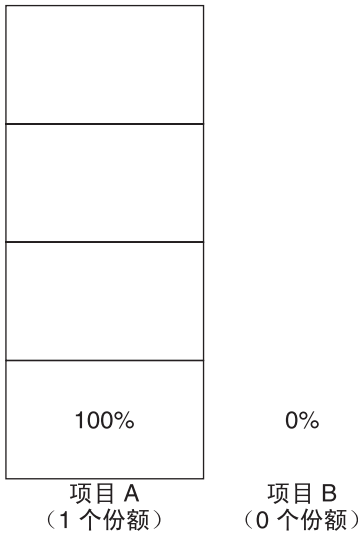
示例 2：项目之间没有争用

如果 *A* 和 *B* 都只有一个计算密集型 (CPU-bound) 进程，并且 $S_A = 1$，$S_B = 100$，则份额总数是 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 39 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。

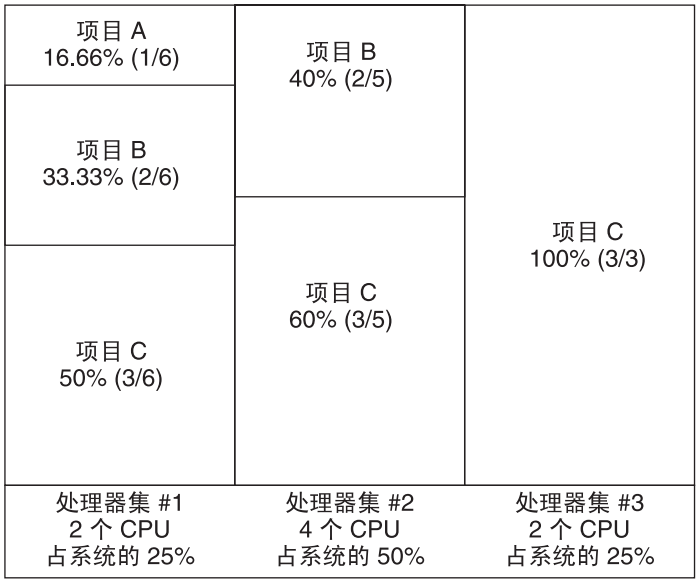
在处理器集边界内运行的应用程序的性能和可用性不会受到新处理器集引入的影响。应用程序也不会受到对其他处理器集中运行的项目的份额分配所做更改的影响。

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 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 资源的分配将有所不同，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。

Solaris 系统还为拥有超级用户权限的用户提供了实时 (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，则常规用户进程便会争用 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 106 页中的 “如何将 FSS 设置为缺省调度程序类”、第 198 页中的 “区域中的调度类”和 dispadmin(1M)。要将正在运行的进程移至其他调度类，请参见第 106 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。有关使用区域时调度类的更多信息，请参见第 198 页中的 “区域中的调度类”。

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 26–5 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)	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

9 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。有关使用区域时调度类的信息，请参见第 198 页中的“区域中的调度类”。

管理公平份额调度器（任务图）

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 106 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 107 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 107 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 108 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。

▼ 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
```

▼ 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 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)。

▼ 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. **成为超级用户或承担等效角色。**

   角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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
```

▼ 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 **成为超级用户或承担等效角色。**

角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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 **成为超级用户或承担等效角色。**

角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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 **成为超级用户或承担等效角色。**

角色包含授权和具有一定权限的命令。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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。这些值会覆写内核中的当前值。键入以下命令：

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

◆ ◆ ◆ 第 10 章

10 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。

Solaris 10 8/07：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 18 章。

本章包含以下主题：

有关使用 rcapd 功能的过程，请参见第 11 章。

在使用资源上限设置守护进程控制物理内存方面的新增功能

Solaris 10：现在可以使用 projmod 命令在 /etc/project 文件中设置 rcap.max-rss 属性。

Solaris 10 11/06：增加了有关将资源上限设置守护进程作为 Solaris 服务管理工具 (service management facility, SMF) 中的一项服务进行启用和禁用的信息。

有关 Solaris 10 新增功能的完整列表以及 Solaris 发行版的说明，请参见《Solaris 10 新增功能》。

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）占用设定的上限。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 章和 `project`(4) 手册页。有关资源控制的信息，请参见第 6 章。

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 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。

限制物理内存使用率的属性

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。

您可以使用 projmod 命令在 /etc/project 文件中设置 rcap.max-rss 属性：

```
# projmod -s -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
```

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，您必须拥有超级用户权限，或您的配置文件列表中有进程管理配置文件。进程管理角色和系统管理员角色都包含进程管理配置文件。

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。有关更多信息，请参见第 199 页中的 “Solaris 10 8/07： 物理内存控制和 capped-memory 资源”。您可以**在**区域（包括全局区域）中运行 rcapd，以便对该区域中的项目执行内存上限。

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，则必须在此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。

注 – 您不能在 `lx` 标记区域中运行 `rcapd`。但是，您可以从全局区域中使用该守护进程来在标记区域中设置内存上限。

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 120 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率。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

守护进程无法确定哪些内存页与其他进程共享，或者哪些内存页在同一个进程内多次映射。由于 rcapd 假设每个页面都是唯一的，因此会导致报告（估算）的 RSS 与实际的 RSS 之间出现差异。

特定的工作负荷（如数据库）广泛使用共享内存。对于这些工作负荷，您可以对项目的常规使用进行抽样，以便确定适当的初始上限值。应使用带有 -J 选项的 prstat 命令的输出。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。

操作	缺省间隔值（秒）	说明
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 120 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率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 122 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。
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 “限制物理内存使用率的属性”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。

命令参考	说明
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。

配置和使用资源上限设置守护进程（任务图）

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 120 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 120 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 121 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 121 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 122 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 123 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 123 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 124 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。

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 114 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. **成为超级用户，或者承担包括进程管理配置文件的角色。**

   系统管理员角色包括进程管理配置文件。有关如何创建该角色并将其指定给用户的信息，请参见《系统管理指南：安全性服务》中的“管理 RBAC（任务列表）”。

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 124 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. **成为超级用户，或者承担包括进程管理配置文件的角色。**

   系统管理员角色包括进程管理配置文件。有关如何创建该角色并将其指定给用户的信息，请参见《系统管理指南：安全性服务》中的“管理 RBAC（任务列表）”。

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 **成为超级用户，或者承担包括进程管理配置文件的角色。**

系统管理员角色包括进程管理配置文件。有关如何创建该角色并将其指定给用户的信息，请参见《系统管理指南：安全性服务》中的“管理 RBAC（任务列表）”。

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
```

- 要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，请键入：

```
# 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 **成为超级用户，或者承担包括进程管理配置文件的角色。**

系统管理员角色包括进程管理配置文件。有关如何创建该角色并将其指定给用户的信息，请参见《系统管理指南：安全性服务》中的“管理 RBAC（任务列表）”。

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 svcadm 命令或带有 -D 的 rcapadm 命令可以安全地禁用 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。

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

以下示例显示了项目的 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页面错误，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，便会出现页面错误。

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页面错误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）

本章讨论以下功能：

- 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

从 Solaris 10 11/06 发行版开始，资源池和动态资源池已成为 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。

资源池和动态资源池的新增功能

Solaris 10：现在，资源池提供了一种机制，可调整每个池的资源分配，以响应系统事件和应用程序负荷的更改。动态资源池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。

现在，您可以使用 `projmod` 命令在 `/etc/project` 文件中设置 `project.pool` 属性。

有关 Solaris 10 新增功能的完整列表以及 Solaris 发行版的说明，请参见《Solaris 10 新增功能》。

Solaris 10 11/06：资源池和动态资源池现在是 SMF 服务。

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)，还可选择性分配调度类。

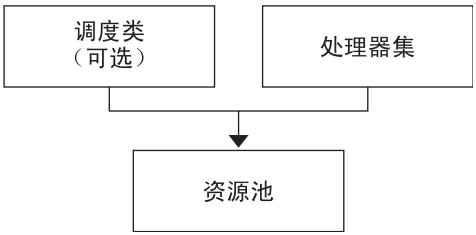


图 12–1 资源池框架

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。可以创建其他用户自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。可以销毁用户自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。

用户自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 146 页中的 “启用和禁用池功能”。

区域中使用的资源池

提示 – Solaris 10 8/07：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 197 页中的 “Solaris 10 8/07： 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分配给特定区域。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 221 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 实用程序仅显示与此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 143 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分配。
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 129 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 131 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 136 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 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 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 146 页中的 “启用和禁用池功能”。如果正在使用用户自定义的池或资源，则不能禁用池功能。

要配置资源池，您必须拥有超级用户权限，或者在配置文件列表中拥有进程管理配置文件。系统管理员角色包括进程管理配置文件。

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。

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 140 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 161 页中的 “传送资源”。

请注意，资源传送可能会触发 poold 执行操作。有关更多信息，请参见第 132 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分配，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

有关池属性的更多信息，请参见 libpool(3LIB) 手册页和第 130 页中的 “池属性” 。

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 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 权限的管理员手动执行。

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。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《编程接口指南》中的“地址组概述”。

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
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

可配置的 pool 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 pool，从命令行控制日志级别。

pool 监视间隔

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

pool 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.pool.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 pool 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 pool 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 139 页中的 “日志位置”。

有关如何配置 pool 日志的更多信息，请参见第 160 页中的 “如何设置 pool 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 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 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 133 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 133 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 137 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 165 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

您可以自定义 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 127 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。

管理动态资源池（任务图）

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 146 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 146 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 129 页中的 “资源池框架” 。	第 151 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 132 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”

任务	说明	参考
设置日志级别。	指定 poold 生成的日志信息的级别。	第 160 页中的 “如何设置 poold 日志级别”
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 161 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 162 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 162 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 163 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 164 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 164 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 165 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 165 页中的 “报告资源集统计信息”

启用和禁用池功能

从 Solaris 10 11/06 发行版开始，您可以使用 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。

▼ Solaris 10 11/06 及更高版本：如何使用 svcadm 启用资源池服务

1. **成为超级用户，或者承担包括进程管理配置文件的角色。**

   系统管理员角色包括进程管理配置文件。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **启用资源池服务。**

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

▼ Solaris 10 11/06 及更高版本：如何使用 svcadm 禁用资源池服务

1. **成为超级用户，或者承担包括进程管理配置文件的角色。**

   系统管理员角色包括进程管理配置文件。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **禁用资源池服务。**

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

▼ Solaris 10 11/06 及更高版本：如何使用 svcadm 启用动态资源池服务

1. **成为超级用户，或者承担包括服务管理权限配置文件的角色。**

   角色包含授权和具有一定权限的命令。有关如何创建该角色并将其指定给用户的信息，请参见《系统管理指南：安全性服务》中的“配置 RBAC（任务列表）”和《系统管理指南：安全性服务》中的“管理 RBAC（任务列表）”。

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       10:32:26 svc:/system/pools/dynamic:default
disabled       10:32:26 svc:/system/pools:default
```

启用动态资源池：

```
# svcadm enable svc:/system/pools/dynamic:default
# svcs -a | grep pool
disabled       10:39:00 svc:/system/pools:default
offline        10:39:12 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 Wed 25 Jan 2006 10:39:00 AM GMT
Reason: Disabled by an administrator.
   See: http://sun.com/msg/SMF-8000-05
   See: libpool(3LIB)
   See: pooladm(1M)
   See: poolbind(1M)
   See: poolcfg(1M)
   See: poolstat(1M)
   See: /var/svc/log/system-pools:default.log
Impact: 1 dependent service is not running.  (Use -v for list.)

svc:/system/pools/dynamic:default (dynamic resource pools)
 State: offline since Wed 25 Jan 2006 10:39:12 AM GMT
Reason: Service svc:/system/pools:default is disabled.
   See: http://sun.com/msg/SMF-8000-GE
   See: poold(1M)
   See: /var/svc/log/system-pools-dynamic:default.log
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         10:40:27 svc:/system/pools:default
online         10:40:27 svc:/system/pools/dynamic: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       10:41:05 svc:/system/pools:default
online         10:40:27 svc:/system/pools/dynamic:default
# svcs *pool*
STATE          STIME    FMRI
disabled       10:41:05 svc:/system/pools:default
online         10:40:27 svc:/system/pools/dynamic:default
```

但最终，DRP 服务将转入 offline，原因是资源池服务已被禁用：

```
# svcs *pool*
STATE          STIME    FMRI
disabled       10:41:05 svc:/system/pools:default
offline        10:41:12 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 Wed 25 Jan 2006 10:41:05 AM GMT
Reason: Disabled by an administrator.
   See: http://sun.com/msg/SMF-8000-05
   See: libpool(3LIB)
   See: pooladm(1M)
   See: poolbind(1M)
   See: poolcfg(1M)
   See: poolstat(1M)
   See: /var/svc/log/system-pools:default.log
Impact: 1 dependent service is not running.  (Use -v for list.)

svc:/system/pools/dynamic:default (dynamic resource pools)
 State: offline since Wed 25 Jan 2006 10:41:12 AM GMT
```

```
Reason: Service svc:/system/pools:default is disabled.
   See: http://sun.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         10:42:23 svc:/system/pools:default
online         10:42:24 svc:/system/pools/dynamic:default
```

▼ Solaris 10 11/06 及更高版本：如何使用 svcadm 禁用动态资源池服务

1. **成为超级用户，或者承担包括进程管理配置文件的角色。**

   系统管理员角色包括进程管理配置文件。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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. **成为超级用户，或者承担包括进程管理配置文件的角色。**

   系统管理员角色包括进程管理配置文件。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **启用池功能。**

   ```
   # pooladm -e
   ```

▼ 如何使用 pooladm 禁用资源池

1 **成为超级用户，或者承担包括进程管理配置文件的角色。**

系统管理员角色包括进程管理配置文件。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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。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，应优先使用新功能 pooladm -s，再考虑使用以前的功能 poolcfg -c discover。

开始之前 在系统上启用池。

1 **成为超级用户，或者承担包括进程管理配置文件的角色。**

系统管理员角色包括进程管理配置文件。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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）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 **成为超级用户，或者承担包括进程管理配置文件的角色。**

系统管理员角色包括进程管理配置文件。有关角色的更多信息，请参见《系统管理指南：基本管理》中的"使用 RBAC 和 Solaris 管理工具（任务图）"。

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）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 **成为超级用户，或者承担包括进程管理配置文件的角色。**

系统管理员角色包括进程管理配置文件。有关如何创建该角色并将其指定给用户的信息，请参见《系统管理指南：安全性服务》中的“管理 RBAC（任务列表）”。

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）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 **成为超级用户，或者承担包括进程管理配置文件的角色。**

系统管理员角色包括进程管理配置文件。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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 **成为超级用户，或者承担包括进程管理配置文件的角色。**

系统管理员角色包括进程管理配置文件。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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）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。有关信息，请参见第 137 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 **成为超级用户，或者承担包括进程管理配置文件的角色。**

系统管理员角色包括进程管理配置文件。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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 137 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 **成为超级用户，或者承担包括进程管理配置文件的角色。**

系统管理员角色包括进程管理配置文件。有关如何创建该角色并将其指定给用户的信息，请参见《系统管理指南：安全性服务》中的“管理 RBAC”。

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 **成为超级用户，或者承担包括进程管理配置文件的角色。**

系统管理员角色包括进程管理配置文件。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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. **成为超级用户，或者承担包括进程管理配置文件的角色。**

   系统管理员角色包括进程管理配置文件。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **提交 `/etc/pooladm.conf` 中的配置。**

   ```
   # pooladm -c
   ```

3. **（可选）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. **成为超级用户，或者承担包括进程管理配置文件的角色。**

   系统管理员角色包括进程管理配置文件。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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. **成为超级用户，或者承担包括进程管理配置文件的角色。**

   系统管理员角色包括进程管理配置文件。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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 107 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. **成为超级用户，或者承担包括进程管理配置文件的角色。**

   系统管理员角色包括进程管理配置文件。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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. **成为超级用户，或者承担包括进程管理配置文件的角色。**

   系统管理员角色包括进程管理配置文件。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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. **成为超级用户，或者承担包括进程管理配置文件的角色。**

   系统管理员角色包括进程管理配置文件。有关角色的更多信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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 141 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旨在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

以下配置用于将应用程序整合到单个系统中。

- 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)     与先前的行保持一致
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。

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。

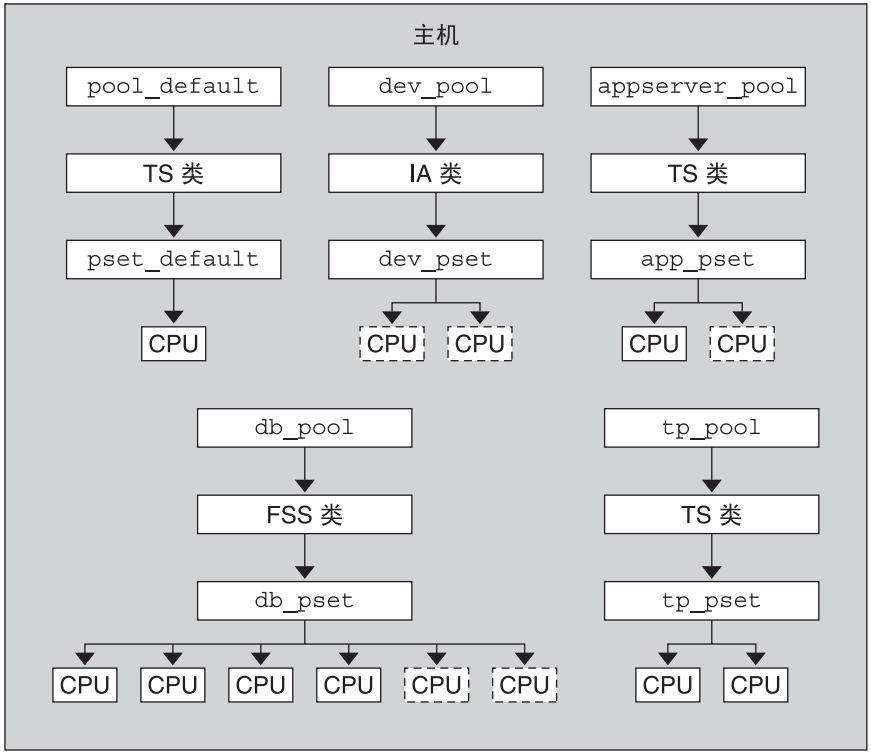


图 14–1 服务器整合配置

注 – 在池 `db_pool` 中，保证为独立数据库实例留出 75% 的 CPU 资源。

◆ ◆ ◆ 第 15 章

15 Solaris Management Console 中的资源控制功能

本章介绍 Solaris Management Console 中的资源控制和性能监视功能。使用此控制台仅可控制一部分资源管理功能。

您可以使用此控制台监视系统性能，并输入表 15–1 中所示的项目、任务和进程的资源控制值。此控制台提供了一种方便、安全的方法，可以替代命令行界面 (command-line interface, CLI) 来管理分布在多个系统中的数以百计的配置参数。每个系统都单独进行管理。此控制台的图形界面支持所有的体验级别。

本章包含以下主题：

使用控制台（任务图）

任务	说明	参考
使用控制台	在本地环境、名称服务或目录服务环境中启动 Solaris Management Console。请注意，性能工具不能用于名称服务环境中。	《系统管理指南：基本管理》中的“启动 Solaris Management Console”和《系统管理指南：基本管理》中的“在名称服务环境中使用 Solaris 管理工具（任务图）”
监视系统性能	访问 “系统状态” 下的性能工具。	第 177 页中的 “如何访问性能工具”

任务	说明	参考
向项目中添加资源控制	访问“系统配置”下的“资源控制”选项卡。	第 180 页中的 “如何访问“资源控制”选项卡”

控制台概述

资源管理功能是 Solaris Management Console 的一个组件。此控制台是基于 GUI 的管理工具的容器，这些管理工具存储在称为工具箱的集合中。有关此控制台及其使用方法的信息，请参见《系统管理指南：基本管理》中的第 2 章 “使用 Solaris Management Console（任务）”。

使用控制台及其工具时，文档主要来源是控制台本身包含的联机帮助系统。有关联机帮助中提供的文档的说明，请参见《系统管理指南：基本管理》中的“Solaris Management Console（概述）”。

管理范围

术语**管理范围**是指选择用于选定的管理工具的名称服务环境。对于资源控制工具和性能工具，可选的管理范围为 `/etc/project` 本地文件或 NIS。

控制台会话期间选择的管理范围应当与 `/etc/nsswitch.conf` 文件中标识的主名称服务相对应。

性能工具

性能工具可用于监视资源的利用率。可以汇总系统的资源利用率，也可以查看每个项目或每个用户的资源利用率。

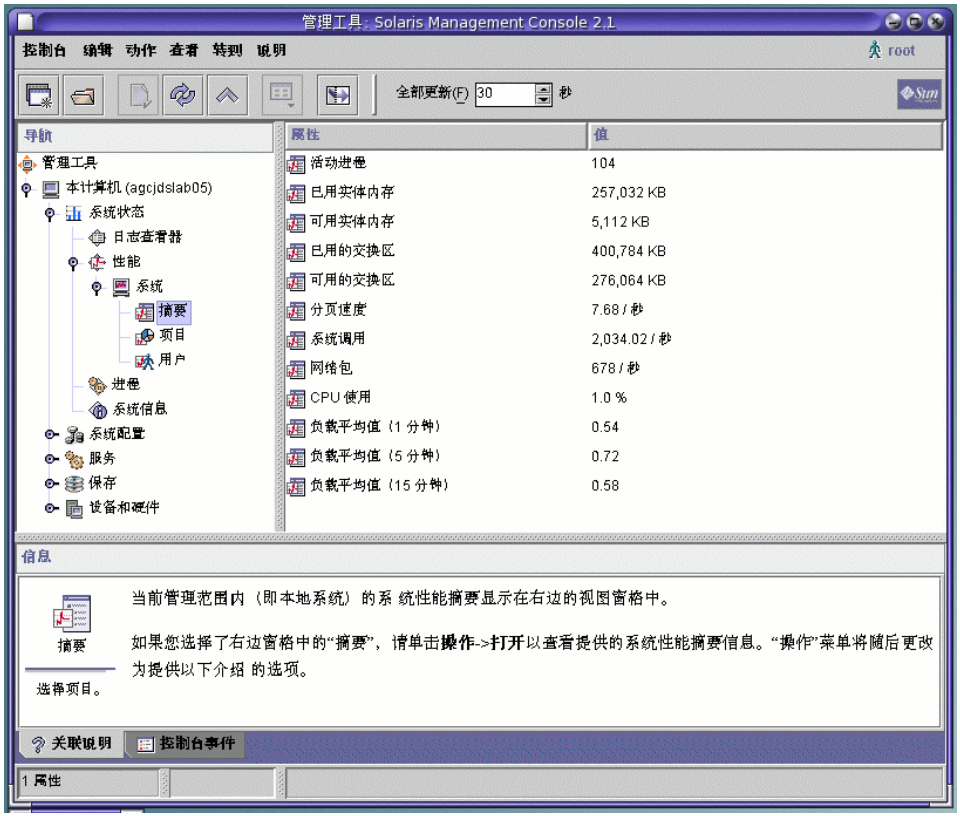


图 15–1 Solaris Management Console 中的性能工具

▼ 如何访问性能工具

性能工具位于“导航”窗格中的“系统状态”下。要访问性能工具，请执行以下操作：

1. **在“ 导航 ”窗格中单击“ 系统状态 ”控制实体。**
   控制实体用于展开“导航”窗格中的菜单项。
2. **单击“ 性能 ”控制实体。**
3. **单击“ 系统 ”控制实体。**
4. **双击“ 摘要 ”、“ 项目 ”或“ 用户 ”。**
   您的选择取决于要监视的使用情况。

按系统进行监视

以下属性的值显示如下。

属性	说明
活动进程	系统上处于活动状态的进程数
已用物理内存	正在使用的系统内存量
可用物理内存	可用的系统内存量
已用交换区	正在使用的系统交换空间量
可用交换区	可用的系统交换空间量
分页速率	系统分页活动的速率
系统调用数	每秒的系统调用数
网络包数	每秒传送的网络包数
CPU 使用率	当前正在使用的 CPU 的百分比
负载平均数	系统运行队列中的进程数，此进程数是最近 1 分钟、5 分钟和 15 分钟内的平均值

按项目名或用户名进行监视

以下属性的值显示如下。

属性	短名称	说明
输入块数	`inblk`	读取的块数
写入块数	`oublk`	写入的块数
读取/写入字符数	`ioch`	读取和写入的字符数
数据页面错误休眠时间	`dftime`	处理数据页面错误所用的时间
偶然上下文切换数	`ictx`	偶然上下文切换数
系统模式时间	`stime`	内核模式下所用的时间
主要页面错误数	`majfl`	主要页面错误数
接收的消息数	`mrcv`	接收的消息数
发送的消息数	`msend`	发送的消息数

属性	短名称	说明
次要页面错误数	minf	次要页面错误数
进程数	nprocs	用户或项目拥有的进程数
LWP 数	count	轻量进程数
其他休眠时间	slptime	tftime、dftime、kftime 和 ltime 以外的休眠时间
CPU 时间	pctcpu	进程、用户或项目使用的最近 CPU 时间的百分比
已用内存	pctmem	进程、用户或项目使用的系统内存的百分比
堆大小	brksize	为进程数据段分配的内存量
驻留集大小	rsssize	进程要求的当前内存量
进程映像大小	size	进程映像的大小 (KB)
接收的信号数	sigs	接收的信号数
停止时间	stoptime	停止状态下所用的时间
交换操作数	swaps	正在进行的交换操作数
已完成的系统调用数	sysc	在上一个时间间隔中执行的系统调用数
系统页面错误休眠时间	kftime	处理页面错误所用的时间
系统陷阱时间	ttime	处理系统陷阱所用的时间
文本页面错误休眠时间	tftime	处理文本页面错误所用的时间
用户锁等待休眠时间	ltime	等待用户锁所用的时间
用户模式时间	utime	用户模式下所用的时间
用户和系统模式时间	time	CPU 的累积执行时间
主动上下文切换数	vctx	主动上下文切换数
等待 CPU 时间	wtime	等待 CPU 所用的时间（延迟）

“ 资源控制 ”选项卡

使用资源控制，可以将项目与一组资源约束进行关联。这些约束可确定项目上下文中运行的任务和进程允许使用的资源。

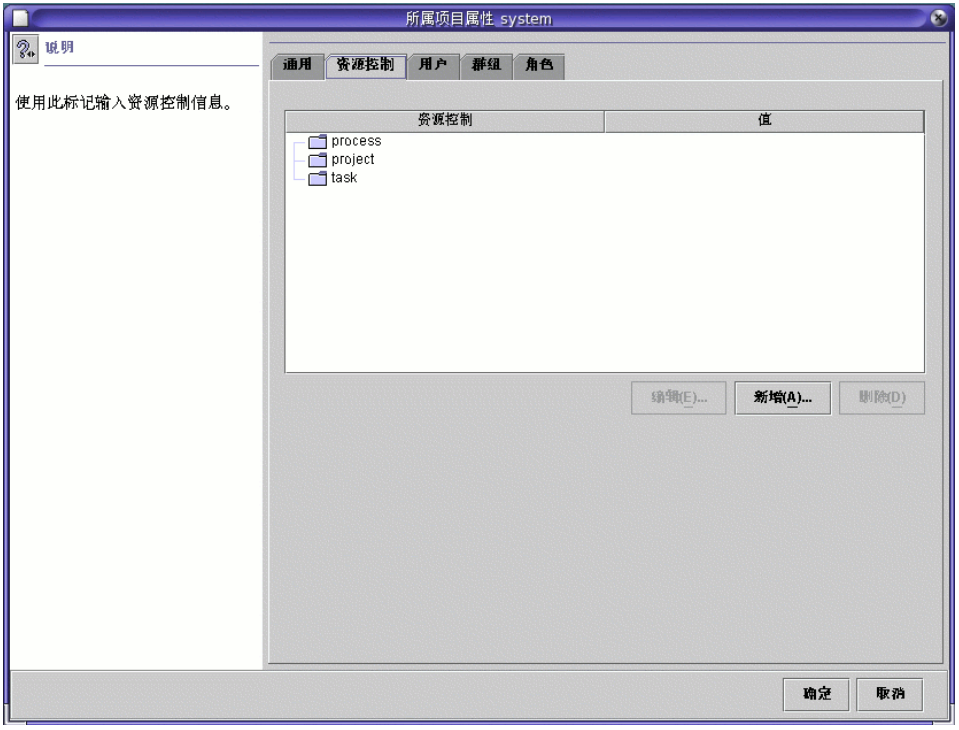


图 15–2 Solaris Management Console 中的“资源控制”选项卡

▼ 如何访问“ 资源控制 ”选项卡

“资源控制”选项卡位于“导航”窗格中的“系统配置”下。要访问“资源控制”，请执行以下操作：

1. **在“ 导航 ”窗格中单击“ 系统配置 ”控制实体。**
2. **双击“ 项目 ”。**
3. **单击控制台主窗口中的某个项目将其选定。**
4. **从“ 操作 ”菜单中选择“ 属性 ”。**

5 **单击“资源控制”选项卡。**

可查看、添加、编辑或删除进程、项目和任务的资源控制值。

可以设置的资源控制

下表显示了可以在控制台中设置的资源控制。该表描述了每个控制约束的资源。表中还标识了 `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) 系统调用分配给进程的最高文件编号。

表 15–1 Solaris Management Console 中的可用标准资源控制

控制名称	说明	缺省单位
`project.cpu-shares`	授予此项目的 CPU 份额数，用于公平份额调度器 (fair share scheduler, FSS)（请参见 FSS(7) 手册页）	数量（份额）
`task.max-cpu-time`	此任务进程可用的最多 CPU 时间	时间（秒）
`task.max-lwps`	此任务进程可同时使用的最大 LWP 数	数量 (LWP)
`process.max-cpu-time`	此进程可用的 CPU 时间最大值	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转储文件大小	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）

设置值

您可以查看、添加、编辑或删除进程、项目和任务的资源控制值。这些操作通过控制台中的对话框执行。

可以在控制台的表中查看资源控制及其值。“资源控制”列列出了可以设置的资源控制。“值”列显示了与每个资源控制关联的属性。在表中，这些值括在括号中，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阈值、一个权限级别、一个信号以及一个与特定阈值关联的本地操作。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。

注 – 在正在运行的系统中，通过控制台在 `project` 数据库中更改的值仅对一个项目中启动的新任务生效。

控制台参考

有关项目和任务的信息，请参见第 2 章。有关资源控制的信息，请参见第 6 章。有关公平份额调度器 (fair share scheduler, FSS) 的信息，请参见第 8 章。

注 – 并非所有的资源控制都可在控制台中设置。有关可以在控制台中设置的各个控制的列表，请参见表 15–1。

第 2 部 分

Zones

此部分介绍 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 16 章

16 Solaris Zones 介绍

Solaris 操作系统中的 Solaris™ Zones 功能提供了一个隔离环境，可在其中运行系统上的应用程序。Solaris Zones 是 Solaris Container 环境的一个组件。

本章包含以下主题：

如果您可以开始在系统上创建区域，请跳至第 17 章。

区域概述

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。**区域**就是在 Solaris 操作系统的某个实例中创建的一个虚拟的操作系统环境。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。即使运行的进程具有超级用户凭证，也不能查看或影响其他区域中的活动。

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 Solaris 10 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区域的数量由所有区域中运行的应用程序软件的总资源需求确定。

有两种类型的非全局区域根文件系统模型：稀疏根和完全根。**稀疏根区域** (sparse root zone) 模型优化对象共享。**完全根区域** (whole root zone) 模型提供最大配置能力。这些概念在第 18 章中介绍。

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。

下图显示了具有四个区域的系统。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自定义的服务。

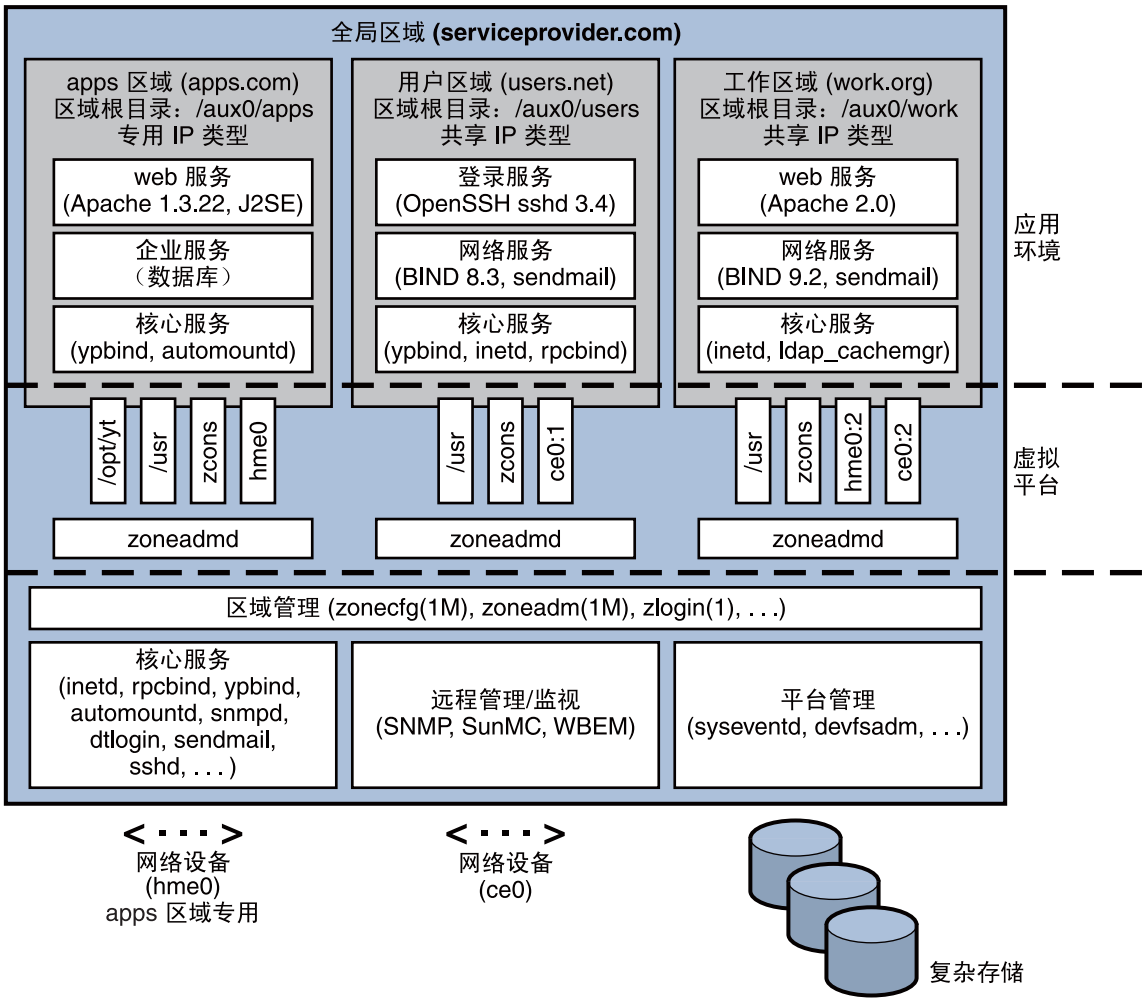


图 16–1 区域服务器整合示例

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容器。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。Solaris 区域使用灵活、软件自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 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。

从 Solaris 10 8/07 开始，IP 网络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。有关区域中 IP 类型的更多信息，请参见第 199 页中的 “区域网络接口”。有关配置信息，请参见第 221 页中的 “如何配置区域”。

每个 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**未创建任何**非全局**区域（简称为区域），则所有进程都在全局区域中运行。

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权限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 204 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 305 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 204 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 198 页中的 “区域中的调度类” 。

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。

区域功能总结

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 Solaris 内核的单个实例 ■ 包含 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

区域类型	特征
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 Solaris 内核下的操作 ■ 包含完整 Solaris 操作系统软件包中已安装的一部分 ■ 包含从全局区域共享的 Solaris 软件包 ■ 可以包含未从全局区域共享的其他已安装的软件包 ■ 可以包含在非全局区域上创建的，未通过软件包安装或者未从全局区域共享的其他软件、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非全局区域中的所有软件组件的有关信息，无论这些组件存在于非全局 区域上还是从全局区域中只读共享 ■ 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权限或主管理员角色。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。全局管理员为区域管理员指定区域管理配置文件。区域管理员的权限仅限于某个非全局区域。

如何创建非全局区域

全局管理员使用 `zonecfg` 命令，通过为区域虚拟平台和应用程序环境指定各种参数来配置区域。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。全局管理员可以使用 `zlogin` 命令登录到已安装的区域。首次登录时，会完成区域的内部配置。然后使用 `zoneadm` 命令引导区域。

有关区域配置的信息，请参见第 17 章。有关区域安装的信息，请参见第 19 章。有关区域登录的信息，请参见第 21 章。

非全局区域状态模型

非全局区域可以处于以下六种状态之一：

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。
已安装	已在系统上实例化区域配置。`zoneadm` 命令用于检验是否可以在指定的 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 20 章和 `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 16–1 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`
	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`

表 16–1 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）
	`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`、`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。无法更改 `inherit-pkg-dir` 资源。

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`、`dedicated-cpu`、`device`、`fs`、`ip-type`、`limitpriv`、`net`、`rctl` 或 `scheduling-class` 属性。无法更改 `inherit-pkg-dir` 资源。

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。
	`zonecfg -z` *zonename* 可用于添加或删除 `attr`、`bootargs`、`capped-memory`、`dataset`、`dedicated-cpu`、`device`、`fs`、`ip-type`、`limitpriv`、`net`、`rctl` 或 `scheduling-class` 属性。无法更改 `zonepath` 和 `inherit-pkg-dir` 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自定义的服务。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。

非全局区域提供的功能

非全局区域可提供以下功能：

安全　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。区域中可用的权限是整个系统中可用权限的一部分。

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。例如，使用与每个区域关联的特定 IP 地址或通配符地址，可以将同一系统的不同共享 IP 区域中运行的多个应用程序绑定到同一网络端口。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　如果需要在网络上的 IP 层隔离某个区域，例如，将其连接到与全局区域和其他非全局区域不同的 VLAN 或不同的 LAN，此时出于安全原因考虑，该区域可以有一个专用 IP。专用 IP 区域可用于整合必须在不同子网（这些子网位于不同的 VLAN 或不同的 LAN）上通信的应用程序。

也可以将区域配置为共享 IP 区域。这些区域将连接到与全局区域相同的 VLAN 或相同的 LAN，并与全局区域共享 IP 路由配置。共享 IP 区域具有单独的 IP 地址，但共享 IP 的其他部分。

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 191 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 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（任务图）

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 ■ 建议每个区域安装 100 MB 可用磁盘空间的可用性	请参见第 217 页中的 “评估当前的系统设置”。

任务	说明	参考
确定是否将资源池与区域结合使用，以便创建容器。	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，从 Solaris 10 8/07 发行版开始，您可以使用 `zonecfg` 属性将区域范围的资源控制和池功能快速添加到区域中。	请参见第 221 页中的 “如何配置区域” 和第 13 章。
执行预配置任务。	确定区域名称和区域路径。确定此区域是共享 IP 区域还是专用 IP 区域，然后获取 IP 地址或数据链路名称。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。确定在标准缺省权限集不充足的情况下，应为区域内的进程设置的权限集。请注意，有些 `zonecfg` 设置会自动添加权限。例如，`ip-type=exclusive` 会自动添加配置和管理网络栈所需的多个权限。	有关区域名称和路径、IP 类型、IP 地址、文件系统、设备、调度类以及权限的信息，请参见第 17 章和第 217 页中的 “评估当前的系统设置”。有关缺省权限和可在非全局区域中配置的权限的列表，请参见第 318 页中的 “非全局区域中的权限”。有关 IP 功能可用性的信息，请参见第 311 页中的 “共享 IP 非全局区域中的联网” 和第 313 页中的 “Solaris 10 8/07：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 221 页中的 “配置、检验并提交区域” 和 `zonecfg`(1M) 手册页。
以全局管理员身份检验和安装已配置的区域。	必须在登录之前检验和安装区域。	请参见第 19 章和第 20 章。
以全局管理员身份，使用带有 `-C` 选项的 `zlogin` 命令登录到每个非全局区域，或将 `sysidcfg` 文件放置到区域的 `/etc` 目录中。		请参见第 21 章和第 22 章。
以全局管理员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 19 章和第 20 章。
为生产使用准备新区域。	创建用户帐户，添加其他软件，并自定义区域配置。	请参阅用于设置新安装的计算机的文档。本指南中介绍了适用于区域环境的特殊注意事项。

◆ ◆ ◆ 第 17 章

17 非全局区域配置（概述）

本章介绍非全局区域配置。

本章包含以下主题：

了解区域配置之后，请转至第 18 章以配置要在系统上安装的非全局区域。

有关 lx 标记区域配置的信息，请参见第 31 章和第 32 章。

本章新增内容

Solaris 10 6/06：添加了对 Zettabyte 文件系统 (Zettabyte File System, ZFS) 的支持，包括在本地非全局区域中添加数据集资源的功能。有关更多信息，请参见第 210 页中的 “资源类型属性”。

Solaris 10 11/06：添加了对可配置权限的支持。请参见第 203 页中的 “*Solaris 10 11/06*：可配置权限”。

Solaris 10 8/07：向 zonecfg 命令中添加了对以下功能的支持：

- 更好地集成了资源管理功能和区域。zonecfg 命令现在可用于配置临时池、内存限制、区域的缺省调度类和资源控制别名。您不再需要执行任何手动步骤来设置资源管理。添加了一些新的资源控制：
  - zone.max-locked-memory

  - zone.max-msg-ids
  - zone.max-sem-ids
  - zone.max-shm-ids
  - zone.max-shm-memory
  - zone.max-swap
- 可以在全局区域中使用 zonecfg 命令。
- 可以为区域指定一个 IP 类型。可用于非全局区域的两种 IP 类型是共享 IP 和专用 IP。
- 通过使用 limitpriv 属性添加必需的权限，可以在区域中使用 DTrace。
- 通过 bootargs 属性可以在区域中使用引导参数。

有关 Solaris 10 新增功能的完整列表以及 Solaris 发行版的说明，请参见《Solaris 10 新增功能》。

关于区域中的资源

包含资源管理功能的区域称为容器。可以在容器中控制的资源包括：

- 资源池或分配的 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，可让您通过相关份额在区域中控制可用 CPU 资源的分配。您可以通过分配给给定区域的 CPU 资源的份额数量来强调该区域中工作负荷的重要性。

预安装配置过程

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

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区域组件。第 207 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。

资源池关联

如果按第 13 章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。

从 Solaris 10 8/07 发行版开始，如果尚未配置资源池，您仍可以使用 `dedicated-cpu` 资源指定在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。

Solaris 10 8/07 ： `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）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 147 页中的 “Solaris 10 11/06 及更高版本：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**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 133 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。

Solaris 10 5/08 ： 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。

区域中的调度类

可以使用**公平份额调度器** (fair share scheduler, FSS)，根据区域中各工作负荷的重要性控制可用 CPU 资源在区域之间的分配。这种工作负荷重要性通过您为每个区域分配的 CPU 资源**份额**数来表示。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 :

- 在 Solaris 10 8/07 发行版中，可以在 zonecfg 中使用 scheduling-class 属性来为区域设置调度类。
- 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 126 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。

Solaris 10 8/07：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。每个限制均为可选项，但至少要设置一个限制。

- 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。

有关更多信息，请参见第 10 章、第 11 章和第 221 页中的 “如何配置区域” 。

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 命令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。共享 IP 区域可共享网络接口，专用 IP 区域必须具有一个专用网络接口。

有关每种类型中 IP 功能的信息，请参见第 311 页中的 “共享 IP 非全局区域中的联网” 和第 313 页中的 “Solaris 10 8/07：专用 IP 非全局区域中的联网” 。

共享 IP 非全局区域

共享 IP 区域是缺省类型。该区域必须有一个或多个专用 IP 地址。共享 IP 区域可与全局区域共享 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：

- 区域将连接到相同的数据链路，即，区域位于与全局区域相同的 IP 子网或子网上。
- 您不想使用专用 IP 区域提供的其他功能。

使用 zonecfg 命令可为共享 IP 区域分配一个或多个 IP 地址。数据链路名称也必须在全局区域中配置。

这些地址与逻辑网络接口关联。可以从全局区域中使用 ifconfig 命令来在运行的区域中添加或删除逻辑接口。有关更多信息，请参见第 312 页中的 “共享 IP 网络接口” 。

Solaris 10 8/07：专用 IP 非全局区域

在专用 IP 区域中可使用 IP 级别的全部功能。

专用 IP 区域具有其自己的与 IP 相关的状态。

这包括可以在专用 IP 区域中使用下列功能：

- 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
- `ndd`，用于设置 TCP/UDP/SCTP 和 IP/ARP 级别按钮
- IP 安全 (IPsec) 和 IKE，可自动提供用于 IPsec 安全关联的验证加密材料

使用 `zonecfg` 命令可为专用 IP 区域分配其自己的数据链路集合。使用 `net` 资源的 `physical` 属性，可为该区域指定一个数据链路名称，如 `xge0`、`e1000g1` 或 `bge32001`。没有设置 `net` 资源的 `address` 属性。

注意，通过分配的数据链路，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，以将其他数据链路分配给正在运行的区域。有关用法示例，请参见第 343 页中的 “Solaris 10 8/07：在专用 IP 非全局区域中管理数据链路”。

在正在运行的专用 IP 区域内，`ifconfig` 命令可用于配置 IP，包括添加或删除逻辑接口。通过使用 `sysidcfg`(4) 中所述的 `sysidtools`，可以按全局区域的设置方式对区域中的 IP 配置进行设置。

注 – 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。下面是一个示例。

```
global# zlogin zone1 ifconfig -a
```

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` 实用程序分配给该区域的包。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，专用 IP 区域中的超级用户可以通过这些数据链路发送欺骗型包，就像可以在全局区域中发送一样。

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

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，这可以包括以下文件系统：

- 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
- 区域管理员明确执行的挂载

将对在应用程序环境中执行的挂载设定特定限制。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 305 页中的 “文件系统和非全局区域”。

区域中的已配置设备

`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 221 页中的 “如何配置区域”。

设置区域范围的资源控制

全局管理员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 221 页中的 “如何配置区域”。

从 Solaris 10 8/07 发行版开始，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。

Solaris 10 5/08： `zone.cpu-cap` 资源控制用于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 `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。

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.capped-cpu` 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 345 页中的 “在安装了区域的 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` 资源控制对可用于区域的锁定物理内存量加以限制。区域中各项目之间的锁定内存资源分配可通过 `project.max-locked-memory` 资源控制来控制。有关更多信息，请参见表 6–1。

`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-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 17–1 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`		Solaris 10 5/08：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。	

表 17–1 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.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-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-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）	
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 345 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。

Solaris 10 11/06： 可配置权限

引导区域时，配置中包括**安全**权限的缺省集合。这些权限被视为安全权限，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：

- 将权限添加至缺省权限集，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。

- 从缺省权限集中删除权限，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权限才能运行的话）。

注 – 目前，有些权限不能从区域的缺省权限集中删除，还有一些权限不能添加到缺省权限集中。

有关更多信息，请参见第 318 页中的 “非全局区域中的权限”、第 221 页中的 “如何配置区域”和 `privileges`(5)。

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 221 页中的 “如何配置区域”。

使用 zonecfg 命令

可以使用 `zonecfg`(1M) 手册页中所述的 `zonecfg` 命令配置非全局区域。在 Solaris 10 8/07 发行版中，此命令还可以用来为全局区域持久指定资源管理设置。

`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 18 章。

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) 手册页中有关选项的内容。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:inherit-pkg-dir> 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 Sun 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，用于覆写现有配置。
- 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 221 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` **Solaris 10 8/07**：清除可选设置的值。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。

`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 205 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。

资源和属性类型

资源和属性类型如下所述：

区域名称

区域名称用于标识配置实用程序的区域。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 64 个字符。
- 名称 global 和所有以 SUNW 开头的名称均保留，不能使用。

zonepath

zonepath 属性是区域根目录的路径。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。它必须由 root 拥有，并且模式为 700。

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 (/) 具有相同的拥有权和权限。区域目录必须由 root 拥有，并且模式为 755。这些目录是使用正确的权限自动创建的，并且不需要区域管理员进行检验。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。

路径	说明
/home/export/my-zone	zonecfg zonepath

路径	说明
/home/export/my-zone/root	区域的根目录
/home/export/my-zone/dev	为区域创建的设备目录

有关此问题的进一步讨论，请参见第 310 页中的 “遍历文件系统”。

注 – 有关此发行版的 ZFS 限制，请参见第 353 页中的 “Solaris 10 6/06、Solaris 10 11/06、Solaris 10 8/07 和 Solaris 10 5/08：不要将非全局区域的根文件系统放置在 ZFS 上”。

`autoboot`

如果此属性设置为 true，则引导全局区域时会自动引导区域。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
```

`bootargs`

Solaris 10 8/07：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 241 页中的 “Solaris 10 8/07：区域引导参数”。

`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 197 页中的 “Solaris 10 8/07：`dedicated-cpu` 资源”。

`limitpriv`

Solaris 10 11/06 及更高版本：此属性用于指定缺省权限集之外的权限掩码。请参见第 318 页中的 “非全局区域中的权限”。

通过指定权限名称可添加权限，权限名称中可包含或不包含前导 `priv_`。在权限名称前添加破折号 (`-`) 或感叹号 (`!`) 可以排除权限。权限值以逗号分隔，并放在引号 (") 内。

如 `priv_str_to_set`(3C) 中所述，特殊权限集 `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，因此不能使用特殊权限集 `zone`。由于常见用法是通过添加或删除某些权限来更改缺省权限集，因此特殊权限集 `default` 将映射为缺省权限集。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` 权限：

```
global# zonecfg -z userzone
zonecfg:userzone> set limitpriv="default,dtrace_proc,dtrace_user"
```

如果区域的权限集包含不允许的权限、缺少必需权限或包含未知权限，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`
Solaris 10 8/07：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 198 页中的 “区域中的调度类”。

`ip-type`
Solaris 10 8/07：仅当区域是专用 IP 区域时，才需要设置此属性。请参见第 200 页中的 “Solaris 10 8/07： 专用 IP 非全局区域”和第 221 页中的 “如何配置区域”。

`dedicated-cpu`
Solaris 10 8/07：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）提供限制。有关更多信息，请参见第 197 页中的 “Solaris 10 8/07： `dedicated-cpu` 资源”。

`capped-cpu` 资源
Solaris 10 5/08：此资源对区域在运行时可占用的 CPU 资源量设置限制。此资源可为 `ncpus` 提供限制。

`capped-memory` 资源
Solaris 10 8/07：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。

`dataset`
Solaris 10 6/06：通过添加 ZFS 文件系统数据集资源，可以将存储管理委托给非全局区域。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。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。

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《Solaris ZFS 管理指南》中的第 8 章 “ZFS 高级主题”以及第 305 页中的 “文件系统和非全局区域”。

有关数据集问题的信息，另请参见第 29 章。

`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 305 页中的 “文件系统和非全局区域”。

`inherit-pkg-dir`
不应在完全根区域中配置此资源。

在稀疏根区域中，inherit-pkg-dir 资源用于表示包含非全局区域与全局区域共享的软件包的目录。

非全局区域以只读模式继承传送到 inherit-pkg-dir 目录的软件包内容。区域的打包数据库将更新，以反映软件包。使用 zoneadm 安装区域之后，便不能修改或删除这些资源。

注 – 配置中包含四种缺省 inherit-pkg-dir 资源。这些目录资源表示哪些目录应从全局区域中继承关联的软件包。资源通过只读回送文件系统挂载来实现。

- /lib
- /platform
- /sbin
- /usr

---

net — 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。

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。

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。

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。

dedicated-cpu — ncpus、importance

指定 CPU 个数以及池的相对重要性（可选）。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。每个限制均为可选项，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
```

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

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

以下示例的几行代码指定全局区域中的 /dev/dsk/c0t0d0s2 将作为要配置的区域中的 /mnt 进行挂载。raw 属性指定一个在尝试挂载文件系统之前运行 fsck 命令的可选设备。所使用的文件系统类型为 UFS。添加了选项 nodevices 和 logging。

```
zonecfg:my-zone> add fs
zonecfg:my-zone:fs> set dir=/mnt
zonecfg:my-zone:fs> set special=/dev/dsk/c0t0d0s2
zonecfg:my-zone:fs> set raw=/dev/rdsk/c0t0d0s2
zonecfg:my-zone:fs> set type=ufs
zonecfg:my-zone:fs> add options [nodevices,logging]
zonecfg:my-zone:fs> end
```

有关更多信息，请参见第 305 页中的 “-o nosuid 选项” 、第 307 页中的 “安全限制和文件系统行为” 以及 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 – 要使用 fs 资源属性添加 ZFS 文件系统，请参见《Solaris ZFS 管理指南》中的“向非全局区域中添加 ZFS 文件系统”。

dataset　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
```

inherit-pkg-dir　dir

以下示例的几行代码指定 /opt/sfw 从全局区域中进行回送挂载。

```
zonecfg:my-zone> add inherit-pkg-dir
zonecfg:my-zone:inherit-pkg-dir> set dir=/opt/sfw
zonecfg:my-zone:inherit-pkg-dir> end
```

net　address、physical

注 – 对于共享 IP 区域，可以同时指定 IP 地址和设备。对于专用 IP 区域，只能指定物理接口。

在以下针对共享 IP 区域的示例中，IP 地址 192.168.0.1 将添加到该区域中。hme0 卡用于物理接口。要确定所使用的物理接口，请在系统上键入 ifconfig -a。每一个输出行（回送驱动程序行除外）都以系统上安装的卡的名称开头。说明中包含 LOOPBACK 的行不适用于卡。

```
zonecfg:my-zone> add net
zonecfg:my-zone:net> set physical=hme0
zonecfg:my-zone:net> set address=192.168.0.1
zonecfg:my-zone:net> end
```

在专用 IP 区域的以下示例中，bge32001 链接用于物理接口。要确定哪些数据链路可用，请使用命令 dladm show-link。只有 GLDv3

数据链路才能用于专用 IP 区域，非 GLDv3 数据链路在 dladm show-link 输出中显示为 type: legacy。请注意，还必须指定 ip-type=exclusive。

```
zonecfg:my-zone> set ip-type=exclusive
zonecfg:my-zone> add net
zonecfg:my-zone:net> set physical=bge32001
zonecfg:my-zone:net> end
```

device　match

在以下示例中，/dev/pts 设备包括在区域中。

```
zonecfg:my-zone> add device
zonecfg:my-zone:device> set match=/dev/pts*
zonecfg:my-zone:device> end
```

rctl　name、value

Solaris 10 8/07：此发行版的新增资源控制包括 zone.max-locked-memory、zone.max-msg-ids、zone.max-sem-ids、zone.max-shm-ids、zone.max-shm-memory 和 zone.max-swap。

以下是可用的区域范围的资源控制：

- zone.cpu-shares（首选：cpu-shares）
- zone.max-locked-memory
- zone.max-lwps（首选：max-lwps）
- zone.max-msg-ids（首选：max-msg-ids）
- 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 221 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 章和第 316 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。

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。

Tecla 命令行编辑库在以下手册页中进行介绍：

- `enhance`(1)
- `libtecla`(3LIB)
- `ef_expand_file`(3TECLA)
- `gl_get_line`(3TECLA)
- `gl_io_mode`(3TECLA)
- `pca_lookup_file`(3TECLA)
- `tecla`(5)

◆ ◆ ◆ 第 18 章

18 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 17 章。

规划和配置非全局区域（任务图）

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。以下任务图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。	请参阅历史使用情况。另请参见第 217 页中的 “磁盘空间需求” 和第 127 页中的 “区域中使用的资源池” 。
确定区域名称。	基于命名约定决定区域的名称。	请参见第 207 页中的 “区域配置数据” 和第 218 页中的 “区域主机名” 。
确定区域路径。	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。	请参见第 207 页中的 “区域配置数据” 。

任务	说明	参考
如果没有配置资源池，请评估是否需要进行 CPU 限制。	查看您的应用程序要求。	请参见第 197 页中的 “Solaris 10 8/07： 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、第 11 章和第 199 页中的 “Solaris 10 8/07： 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、第 198 页中的 “区域中的调度类” 。
确定区域是共享 IP 区域，还是专用 IP 区域。	对于共享 IP 区域（缺省区域），可获取或配置区域的 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。 对于专用 IP 区域，请确定要分配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。该接口可以是单独的 LAN（如 bge1），也可以是单独的 VLAN（如bge2000）。数据链路必须是 GLDv3。非 GLDv3 的数据链路在 dladm show-link 命令的输出中会被标识为 type: legacy。	请参见第 218 页中的 “确定区域主机名并获取网络地址” 、第 221 页中的 “如何配置区域” 和《系统管理指南： IP 服务》 。 有关 GLDv3 接口的更多信息，请参见《系统管理指南： IP 服务》中的“Solaris OS 接口类型”。
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 312 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。	检查权限集： 缺省权限集、可以添加和删除的权限，以及目前不能使用的权限。	请参见第 318 页中的 “非全局区域中的权限” 。
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 221 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 221 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 Solaris 10 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。全局管理员负责限制空间。全局管理员必须确保本地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。

全局区域中安装的软件包的性质影响所创建的非全局区域的空间需求。软件包的数量和空间需求为相关因素。

稀疏根区域

拥有 `inherit-pkg-dir` 资源的非全局区域称为稀疏根区域。

稀疏根区域模型通过以下方法优化对象共享：

- 只有安装在全局区域中的部分软件包会直接安装在非全局区域中。
- 标识为 `inherit-pkg-dir` 资源的只读回送文件系统用于获取对其他文件的访问权限。

在该模型中，所有的软件包都安装在非全局区域中。将完整安装不向只读回送挂载文件系统提供内容的软件包。无需安装提供给只读回送挂载文件系统的内容，因为这些内容可以从全局区域中继承（并可见）。

- 一般情况下，当全局区域安装了所有标准的 Solaris 软件包时，每个区域大约需要 100 MB 的可用磁盘空间。
- 缺省情况下，全局区域中安装的所有附加软件包也将填充非全局区域。所需磁盘空间可能会相应增加，具体取决于驻留在 `inherit-pkg-dir` 资源空间的附加软件包是否提供文件。

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。

完全根区域

完全根区域模型提供最大配置能力。所有需要的和任何选定的可选 Solaris 软件包都安装到此区域的专用文件系统中。该模型的优势之一是全局管理员可以自定义其区域文件系统布局。例如，可以执行此操作来添加任意非绑定的软件包或第三方软件包。

该模型的磁盘需求由当前安装在全局区域中的软件包使用的磁盘空间决定。

注 – 如果您创建包含以下 `inherit-pkg-dir` 目录的稀疏根区域，则必须**在安装区域之前**从非全局区域配置中删除这些目录，以拥有一个完全根区域：

- `/lib`
- `/platform`
- `/sbin`
- `/usr`

请参见第 221 页中的 “如何配置区域” 。

限制区域大小

可以使用以下选项限制区域大小：

- 您可以将区域放置在挂载了 `lofi` 的分区上。此操作会将区域占用的空间量限制为 `lofi` 使用的文件所占用的空间量。有关更多信息，请参见 `lofiadm`(1M) 和 `lofi`(7D) 手册页。
- 您可以使用软分区将磁盘分片或将逻辑卷分为多个分区。您可以将这些分区用作区域根目录，从而限制每个区域的磁盘占用量。软分区限制为 8192 个分区。有关更多信息，请参见《Solaris Volume Manager Administration Guide》中的第 12 章 “Soft Partitions (Overview)”。
- 您可以将磁盘的标准分区用作区域根目录，从而限制每个区域的磁盘占用量。

确定区域主机名并获取网络地址

您必须确定区域的主机名。然后，如果要使区域具有网络连接，则必须为其指定一个 IPv4 地址，或手动配置并指定一个 IPv6 地址。

区域主机名

您为区域选择的主机名必须在 `hosts` 数据库或 `/etc/inet/hosts` 数据库（在全局区域中的 `/etc/nsswitch.conf` 文件中指定）中定义。网络数据库是指提供网络配置信息的文件。`nsswitch.conf` 文件指定要使用的命名服务。

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，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《系统管理指南：IP 服务》中的“TCP/IP 配置文件”和《系统管理指南：IP 服务》中的“网络数据库和 nsswitch.conf 文件”。

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。

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：

链路本地地址

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带有 `-a6` 选项的 `ifconfig` 命令，也可以获得该前缀。

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 `inet6`(7P) 手册页。

专用 IP 区域网络地址

在专用 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。

有关更多信息，请参见 `sysidcfg`(4)。

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 virtual 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 305 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。

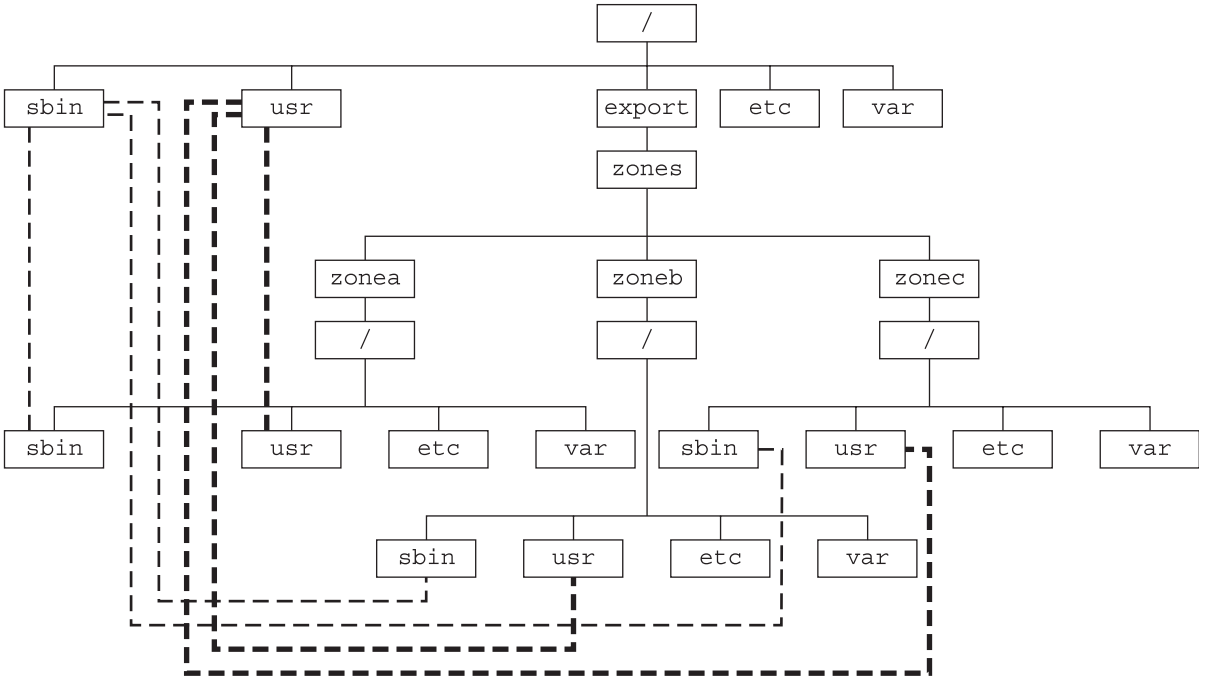


图 18–1 回送挂载的文件系统

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（任务图）

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 221 页中的 “配置、检验并提交区域” 、第 226 页中的 “配置多个区域的脚本”
修改区域配置。	使用此过程可以修改区域配置中的资源类型，或在区域中添加专用设备。	第 229 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 233 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 235 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 233 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 226 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 229 页中的 “如何显示非全局区域的配置” 。

▼ 如何配置区域

请注意，创建本地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 207 页中的 “区域配置数据” 。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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 Sun 的缺省设置。

```
zonecfg:my-zone> create
```

4 **设置区域路径，在此过程中为 /export/home/my-zone。**

```
zonecfg:my-zone> set zonepath=/export/home/my-zone
```

对于该发行版，不要为 ZFS 设置 zonepath。

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: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）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 **修改缺省权限集。**

```
zonecfg:my-zone> set limitpriv="default,sys_time"
```

该行添加了将系统时钟设置为缺省权限集的功能。

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=50m
```

b. **设置交换内存上限。**

```
zonecfg:my-zone:capped-memory> set swap=100m
```

c. **设置锁定内存上限。**

```
zonecfg:my-zone:capped-memory> set locked=30m
```

d. **结束内存上限指定。**

```
zonecfg:my-zone:capped-memory> end
```

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 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 dataset 指定。

```
zonecfg:my-zone> end
```

13 （仅用于稀疏根区域）添加一个从全局区域回送挂载的共享文件系统。

不要执行此步骤来创建一个没有任何共享文件系统的完全根区域。请参见第 217 页中的“磁盘空间需求”中对完全根区域的讨论。

```
zonecfg:my-zone> add inherit-pkg-dir
```

a. 指定在正在配置的区域中以只读模式挂载全局区域中的 /opt/sfw。

```
zonecfg:my-zone:inherit-pkg-dir> set dir=/opt/sfw
```

注 – 区域的打包数据库将更新，以反映软件包。使用 zoneadm 安装区域之后，便不能修改或删除这些资源。

---

b. 结束 inherit-pkg-dir 指定。

```
zonecfg:my-zone:inherit-pkg-dir> end
```

可多次执行此步骤来添加多个共享文件系统。

注 – 如果要创建一个完全根区域，并且已经使用 inherit-pkg-dir 添加了缺省的共享文件系统资源，则必须使用 zonecfg 删除以下缺省的 inherit-pkg-dir 资源，才能安装区域：

- zonecfg:my-zone> **remove inherit-pkg-dir dir=/lib**
- zonecfg:my-zone> **remove inherit-pkg-dir dir=/platform**
- zonecfg:my-zone> **remove inherit-pkg-dir dir=/sbin**
- zonecfg:my-zone> **remove inherit-pkg-dir dir=/usr**

---

14 （可选）如果要创建一个专用 IP 区域，请设置 ip-type。

```
zonecfg:my-zone> set ip-type=exclusive
```

注 – 在 add net 步骤中只会指定物理设备类型。

---

15 添加网络接口。

```
zonecfg:my-zone> add net
```

a. （仅共享 IP）设置网络接口的 IP 地址，在此过程中为 192.168.0.1。

```
zonecfg:my-zone:net> set address=192.168.0.1
```

b. 设置网络接口的物理设备类型，在此过程中为 hme 设备。

```
zonecfg:my-zone:net> set physical=hme0
```

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。

16 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。

17 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。

18 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
```

19 **检验区域的配置。**

```
zonecfg:my-zone> verify
```

20 **提交区域的配置。**

```
zonecfg:my-zone> commit
```

21 **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=/export/home/my-zone"
```

下一步执行的操作

请参见第 244 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。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

要执行此脚本，您必须是全局区域中的全局管理员。全局管理员在全局区域中拥有超级用户权限或承担主管理员角色。

```
#!/bin/ksh
#
# Copyright 2006 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
#ident    "%Z%%M%    %I%    %E% SMI"

if [[ -z "$1" || -z "$2" || -z "$3" ]]; then
```

```
        echo "usage: $0 <#-of-zones> <zonename-prefix> <basedir>"
        exit 2
fi

if [[ ! -d $3 ]]; then
        echo "$3 is not a directory"
        exit 1
fi

nprocs=‘psrinfo | wc -l‘
nzones=$1
prefix=$2
dir=$3

ip_addrs_per_if=‘ndd /dev/ip ip_addrs_per_if‘
if [ $ip_addrs_per_if -lt $nzones ]; then
        echo "ndd parameter ip_addrs_per_if is too low ($ip_addrs_per_if)"
        echo "set it higher with ’ndd -set /dev/ip ip_addrs_per_if <num>"
        exit 1
fi

i=1
while [ $i -le $nzones ]; do
    zoneadm -z $prefix$i list > /dev/null 2>&1
    if [ $? != 0 ]; then
        echo configuring $prefix$i
        F=$dir/$prefix$i.config
        rm -f $F
        echo "create" > $F
        echo "set zonepath=$dir/$prefix$i" >> $F
        zonecfg -z $prefix$i -f $dir/$prefix$i.config 2>&1 | \
            sed ’s/^/    /g’
    else
        echo "skipping $prefix$i, already configured"
    fi
    i=‘expr $i + 1‘
done

i=1
while [ $i -le $nzones ]; do
    j=1
    while [ $j -le $nprocs ]; do
        if [ $i -le $nzones ]; then
            if [ ‘zoneadm -z $prefix$i list -p | \
                cut -d’:’ -f 3‘ != "configured" ]; then
                echo "skipping $prefix$i, already installed"
            else
                echo installing $prefix$i
```

```
                mkdir -pm 0700 $dir/$prefix$i
                chmod 700 $dir/$prefix$i
                zoneadm -z $prefix$i install > /dev/null 2>&1 &
                sleep 1    # spread things out just a tad
            fi
        fi
        i=‘expr $i + 1‘
        j=‘expr $j + 1‘
    done
    wait
done

i=1
while [ $i -le $nzones ]; do
    echo setting up sysid for $prefix$i
    cfg=$dir/$prefix$i/root/etc/sysidcfg
    rm -f $cfg
    echo "network_interface=NONE {hostname=$prefix$i}" > $cfg
    echo "system_locale=C" >> $cfg
    echo "terminal=xterms" >> $cfg
    echo "security_policy=NONE" >> $cfg
    echo "name_service=NONE" >> $cfg
    echo "timezone=US/Pacific" >> $cfg
    echo "root_password=Qexr7Y/wzkSbc" >> $cfg  # ’l1a’
    i=‘expr $i + 1‘
done

i=1
para=‘expr $nprocs \* 2‘
while [ $i -le $nzones ]; do
    date
    j=1
    while [ $j -le $para ]; do
        if [ $i -le $nzones ]; then
            echo booting $prefix$i
            zoneadm -z $prefix$i boot &
        fi
        j=‘expr $j + 1‘
        i=‘expr $i + 1‘
    done
    wait
done
```

▼ 如何显示非全局区域的配置

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**
   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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

▼ 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。

请注意，在使用 zoneadm 安装区域之后，不能修改或删除 inherit-pkg-dir 目录中软件包的内容。

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**
   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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。

▼ Solaris 10 8/07：如何清除区域配置中的属性类型

使用此过程可以重置一个独立属性。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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。

▼ Solaris 10 3/05 至 Solaris 10 11/06：如何修改区域配置中的属性类型

使用此过程可以重置一个独立属性，该独立属性没有其他相关属性要配置。例如，要删除现有的池关联，您可以将 pool 资源重置为 null。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **选择要修改的区域，在此过程中为 my-zone。**

```
global# zonecfg -z my-zone
```

3 **重置要更改的属性，在此过程中为现有的池关联。**

```
zonecfg:my-zone> set 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。

▼ Solaris 10 8/07：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。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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。

▼ 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。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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。

要执行此过程，您必须是全局区域中的全局管理员。

1 成为超级用户或承担主管理员角色。

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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 19 章

19 关于安装、停止、克隆和卸载非全局区域（概述）

本章介绍如何在您的 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 20 章。

有关 `lx` 标记区域安装的信息，请参见第 33 章和第 34 章。

本章新增内容

Solaris 10 11/06：现在可以克隆非全局区域。请参见第 253 页中的 “Solaris 10 11/06：在同一系统中克隆非全局区域”。

Solaris 10 8/07：还添加了有关引导参数的信息。请参见第 241 页中的 “Solaris 10 8/07：区域引导参数”。

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。可以使用 zoneadm 命令执行以下任务：

- 检验区域
- 安装区域
- 引导区域，类似于引导常规的 Solaris 系统
- 显示有关正在运行的区域的信息
- 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过程，请参见第 20 章和 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。有关区域配置过程，请参见第 18 章和 zonecfg(1M) 手册页。区域状态在第 189 页中的 “非全局区域状态模型”中进行了介绍。

如果您打算为区域生成 Solaris 审计记录，请在安装非全局区域之前先阅读第 322 页中的 “在区域中使用 Solaris 审计”。

区域构建

本节适用于初始区域构建，不适用于现有区域的克隆。

在配置了非全局区域之后，应检验是否可以在系统配置中安全安装此区域。然后您可以安装此区域。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。

非全局区域是使用开放式网络配置 (generic_open.xml) 安装的。网络配置类型在《系统管理指南：基本管理》中的第 15 章 “管理服务（任务）”中介绍。区域管理员可以使用 netservices 命令将区域切换到受限的网络配置 (generic_limited_net.xml)。可通过使用 SMF 命令启用或禁用特定服务。

成功安装了区域之后，便可进行初始登录和引导。

在 Solaris 安装中用于初始安装软件包的方法也可用于填充非全局区域。

全局区域必须包含填充非全局区域所需的所有数据。填充区域包括创建目录、复制文件以及提供配置信息。

从全局区域中填充区域时，只会使用在全局区域中通过软件包创建的信息或数据。有关更多信息，请参见 pkgparam(1) 和 pkginfo(4) 手册页。

安装区域时，不引用或复制以下数据：

- 未安装的软件包
- 修补程序
- CD 和 DVD 上的数据
- 网络安装映像
- 区域的任何原型或其他实例

此外，以下信息类型（如果在全局区域中存在）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
- 网络服务（例如 DHCP 地址分配、UUCP（UNIX 对 UNIX 复制）或 sendmail）的配置
- 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 Solaris 审计，则可能需要对从全局区域复制的审计文件进行修改。有关更多信息，请参见第 322 页中的 “在区域中使用 Solaris 审计” 。

不能在非全局区域中配置以下功能：

- Solaris Live Upgrade™ 引导环境
- Solaris Volume Manager 元设备
- 共享 IP 区域中的 DHCP 地址分配
- SSL 代理服务器

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。

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。有关更多信息，请参见 devfsadmd(1M) 手册页。
- 设置虚拟网络接口。
- 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。

在首次引导非全局区域之前，必须创建区域的内部配置。内部配置指定要使用的命名服务、缺省语言环境 (locale) 和时区、区域的超级用户口令，以及应用程序环境的其他方面。通过响应出现在区域控制台上的一系列提示来建立应用程序环境，如第 256 页中的 “内部区域配置”中所述。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。

关于停止、重新引导和卸载区域

本节概述了停止、重新引导和卸载区域的过程，还提供了区域在需要时无法停止的疑难解答提示。

停止区域

`zoneadm halt` 命令用于删除区域的应用程序环境和虚拟平台。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 266 页中的 “如何使用 `zlogin` 关闭区域” 。

停止操作失败时，请参见第 354 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。

Solaris 10 8/07 ： 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 248 页中的 “如何引导区域”和第 249 页中的 “如何在单用户模式下引导区域”。

有关 Solaris 服务管理工具 (service management facility, SMF) 和 `init` 的信息，请参见《系统管理指南：基本管理》中的第 14 章“管理服务（概述）”以及 `svc.startd`(1M) 和 `init`(1M)。

区域 autoboot

如果您在区域配置中将 `autoboot` 资源属性设置为 `true`，则引导全局区域时便会自动引导此区域。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`。

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。

Solaris 10 11/06 及更高版本：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。

- 克隆区域是安装区域的一种比较快速的方法。
- 新区域将包括因自定义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。

有关更多信息，请参见第 253 页中的 “Solaris 10 11/06：在同一系统中克隆非全局区域”。

◆ ◆ ◆ 第 20 章

20 安装、引导、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），同时提供了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 19 章。

有关 lx 标记区域安装和克隆的信息，请参见第 33 章和第 34 章。

区域安装（任务图）

任务	说明	参考
（可选）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 244 页中的 “（可选）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 245 页中的 “如何安装已配置的区域”
Solaris 8/07：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 246 页中的 “Solaris 10 8/07：如何获取已安装的非全局区域的 UUID”
（可选）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 247 页中的 “（可选）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。请注意，首次在引导后登录到区域时，必须执行内部区域配置。	第 248 页中的 “如何引导区域” 、第 256 页中的 “内部区域配置” 、第 260 页中的 “执行初始内部区域配置”
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 249 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) 手册页中所述的 `zoneadm` 命令可以为非全局区域执行安装任务。要执行区域安装，您必须是全局管理员。本章中的示例使用在第 221 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）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。如果您跳过此过程，则会在安装区域时自动执行检验。

要执行此过程，您必须是全局区域中的全局管理员。

1 成为超级用户或承担主管理员角色。

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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: /export/home/my-zone does not exist, so it cannot be verified.
When 'zoneadm install' is run, 'install' will try to create
/export/home1/my-zone, and 'verify' will be tried again,
but the 'verify' may fail if:
the parent directory of /export/home/my-zone is group- or other-writable
or
/export/home1/my-zone overlaps with any other installed zones.
```

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。

▼ 如何安装已配置的区域

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **使用带有 `-z install` 选项的 `zoneadm` 命令安装已配置的区域 `my-zone`。**

```
global# zoneadm -z my-zone install
```

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）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 -z my-zone list -v
```

- 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 **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      /                              native     shared
 -  my-zone  installed    /export/home/my-zone           native     shared
```

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。

接下来的操作 缺省情况下，此区域是使用开放式网络配置安装的，该配置在《系统管理指南：基本管理》中的第 15 章“管理服务（任务）”中进行介绍。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 267 页中的 “将非全局区域切换到其他网络服务配置”。

▼ Solaris 10 8/07：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 `-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 -p
```

将显示以下类似信息：

```
0:global:running:/:
6:my-zone:running:/export/home/my-zone:61901255-35cf-40d6-d501-f37dc84eb504
```

示例 20–1 如何在命令中使用 UUID

```
global# zoneadm -z my-zone -u 61901255-35cf-40d6-d501-f37dc84eb504 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)。

▼ Solaris 10 8/07：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。

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**

   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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      /                             native     shared
 -  my-zone  installed    /export/home/my-zone          native     shared
 -  testzone incomplete   /export/home/testzone         native     shared
```

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 252 页中的 “如何卸载区域” 。

▼ （可选）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。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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      /                              native     shared
 1  my-zone  ready        /export/home/my-zone           native     shared
```

请注意，系统已指定唯一的区域 ID 1。

▼ 如何引导区域

引导区域会将该区域置于运行状态。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。

提示 – 请注意，首次登录到区域时，即会执行内部区域配置。这在第 256 页中的 “内部区域配置”中介绍。

如果您打算使用 `/etc/sysidcfg` 文件来执行初始区域配置（如第 262 页中的 “如何使用 /etc/sysidcfg 文件执行初始区域配置”中所述），请创建 `sysidcfg` 文件并将其放入区域的 `/etc` 目录中，然后再引导区域。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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      /                              native     shared
 1  my-zone  running      /export/home/my-zone           native     shared
```

示例 20–2 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
```

故障排除 如果显示一条消息，指出系统找不到要用于在区域配置中指定的 IP 地址的网络掩码，请参见第 355 页中的 “引导区域时显示 netmasks 警告”。请注意，此消息只是警告，而命令已成功执行。

▼ 如何在单用户模式下引导区域

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**
   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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 21 章和第 22 章。

停止、重新引导、卸载、克隆和删除非全局区域（任务图）

任务	说明	参考
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 266 页中的 “如何使用 zlogin 关闭区域”。	第 250 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 251 页中的 “如何重新引导区域”
卸载区域。	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 252 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 253 页中的 “Solaris 10 11/06：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 254 页中的 “从系统中删除非全局区域”

停止、重新引导和卸载区域

▼ 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 266 页中的 “如何使用 zlogin 关闭区域”。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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      /                              native     shared
 1  my-zone  running      /export/home/my-zone           native     shared
```

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      /                              native     shared
 -  my-zone  installed    /export/home/my-zone           native     shared
```

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 停止操作失败时，请参见第 354 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

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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      /                              native     shared
 1  my-zone  running      /export/home/my-zone           native     shared
```

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      /                             native     shared
 2  my-zone  running      /export/home/my-zone          native     shared
```

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 – 使用此过程时应谨慎。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。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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      /                             native     shared
 -  my-zone  installed    /export/home/my-zone          native     shared
```

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
```

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 -v
```

将显示以下类似信息：

```
ID  NAME     STATUS       PATH                           BRAND      IP
 0  global   running      /                              native     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。

使用 `uninstall` 命令时应谨慎，因为此操作是无法恢复的。

Solaris 10 11/06：在同一系统中克隆非全局区域

克隆用于通过从源 `zonepath` 向目标 `zonepath` 复制数据来在系统中置备新区域。

▼ 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。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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 /export/zones/master
```

注 – 也可以通过使用第 221 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 /export/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 /export/home/my-zone...
```

7 **列出系统上的区域。**

```
ID  NAME     STATUS       PATH                          BRAND      IP
 0  global   running      /                             native     shared
 -  my-zone  installed    /export/home/my-zone          native     shared
 -  zone1    installed    /export/home/zone1            native     shared
```

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 **关闭区域 my-zone。**

```
global# zlogin my-zone shutdown
```

2 **删除 my-zone 的根文件系统。**

```
global# zoneadm -z my-zone uninstall -F
```

3 **删除 my-zone 的配置。**

```
global# zonecfg -z my-zone delete -F
```

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      /                             native     shared
```

◆ ◆ ◆ 第 21 章

21 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 22 章。

zlogin 命令

安装区域之后，您必须登录到该区域来完成应用程序环境的配置。您还可以登录到区域来执行管理任务。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 265 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。

只有全局区域中的全局管理员才能使用 zlogin 命令。有关更多信息，请参见 zlogin(1) 手册页。

内部区域配置

安装后，区域处于未配置状态。此时区域没有进行命名服务的内部配置，未设置语言环境和时区，也尚未执行各种其他的配置任务。因此，首次使用区域控制台登录时，会运行 `sysidtool` 程序。有关更多信息，请参见 `sysidtool`(1M) 手册页。

可以使用以下两种方法执行所需的配置：

- 区域控制台登录，它启动了一系列系统问题。请准备对以下各项作出响应：
  - 语言
  - 所用终端的类型
  - 主机名
  - 安全策略（Kerberos 或标准 UNIX）
  - 命名服务类型（有效响应为 `None`）
  - 命名服务域
  - 名称服务器
  - 缺省时区
  - 超级用户口令

  此过程在第 260 页中的 “执行初始内部区域配置”中介绍。
- `/etc/sysidcfg` 文件，您可以在首次引导区域之前创建它并将其放入区域内。有关更多信息，请参见 `sysidcfg`(4) 手册页。

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。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 257 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。

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。

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。缺省情况下，通过 rlogin、ssh 和 telnet 的登录可正常运行。有关这些命令的更多信息，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。

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。

◆ ◆ ◆ 第 22 章

22 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。

有关区域登录进程的介绍，请参见第 21 章。

初始区域引导与区域登录过程（任务图）

任务	说明	参考
执行内部配置。	登录到区域控制台或使用 /etc/sysidcfg 文件来执行初始区域配置。	第 260 页中的 “执行初始内部区域配置”
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 263 页中的 “登录到区域”
退出非全局区域。	从非全局区域断开。	第 265 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 266 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 268 页中的 “列显当前区域的名称”

执行初始内部区域配置

您必须使用以下方法之一来配置区域：

- 按照第 256 页中的 “内部区域配置”中所述，登录到区域并对其进行配置。
- 按照第 262 页中的 “如何使用 /etc/sysidcfg 文件执行初始区域配置”中所述，使用 /etc/sysidcfg 文件配置区域。

提示 – 执行完内部配置之后，最好复制非全局区域的配置。将来您可以使用此备份来恢复区域。以超级用户或主管理员的身份，将区域 my-zone 的配置列显到文件。以下示例使用名为 my-zone.config 的文件。

```
global# zonecfg -z my-zone export > my-zone.config
```

有关更多信息，请参见第 349 页中的 “如何恢复单个非全局区域”。

▼ 如何登录到区域控制台以执行内部区域配置

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**

   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **使用带有 -C 选项和区域名称（在此过程中为 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 窗口中显示以下类似信息：

   ```
   [NOTICE: Zone booting up]
   ```

4. **首次登录到控制台时，系统会提示您回答一系列问题。将显示以下类似信息：**

   ```
   SunOS Release 5.10 Version Generic 64-bit
   Copyright 1983-2006 Sun Microsystems, Inc.  All rights reserved.
   Use is subject to license terms.

   Hostname: my-zone
   Loading smf(5) service descriptions:
   Select a Language

        1. English
   ```

```
     2. es
     2. fr
Please make a choice (0 - 1), or press h or ? for help:

Select a Locale

      1. English (C - 7-bit ASCII)
      2. Canada (English) (UTF-8)
      4. U.S.A. (UTF-8)
      5. U.S.A. (en_US.ISO8859-1)
      6. U.S.A. (en_US.ISO8859-15)
      7. Go Back to Previous Screen
Please make a choice (0 - 9), or press h or ? for help:

What type of terminal are you using?
      1) ANSI Standard CRT
      2) DEC VT52
      3) DEC VT100
      4) Heathkit 19
      5) Lear Siegler ADM31
      6) PC Console
      7) Sun Command Tool
      8) Sun Workstation
      9) Televideo 910
      10) Televideo 925
      11) Wyse Model 50
      12) X Terminal Emulator (xterms)
      13) CDE Terminal Emulator (dtterm)
      14) Other
Type the number of your choice and press Return:
13
.
.
.
```

有关必须回答的问题的完整列表，请参见第 256 页中的 “内部区域配置” 。

5 （可选）如果未按步骤 3 所述使用两个窗口，则可能已错过关于配置信息的初始提示。如果您在区域登录时看到的不是提示而是以下系统消息：

```
[connected to zone zonename console]
```

请按回车键再次显示提示。

如果输入错误响应并尝试重新启动配置，则可能会在再次尝试此过程时遇到问题。这是因为 `sysidtools` 可以存储先前的响应。

如果发生这种情况，请在全局区域中使用以下解决方法来重新启动配置过程。

```
global# zlogin -S zonename /usr/sbin/sys-unconfig
```

有关 sys-unconfig 命令的更多信息，请参见 sys-unconfig(1M) 手册页。

▼ 如何使用 /etc/sysidcfg 文件执行初始区域配置

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **在全局区域中，转到非全局区域的 /etc 目录：**

```
global# cd /export/home/my-zone/root/etc
```

3 **创建 sysidcfg 文件并将其放入此目录中。**

将显示以下类似信息：

- **对于共享 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)
}
timezone=US/Central
root_password=m4qtoWN
```

- **对于带有静态 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
}
timezone=US/Central
root_password=m4qtoWN
```

- **对于带有 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
}
timezone=US/Central
root_password=m4qtoWN
```

4 **缺省情况下，单独的模块将请求 `nfsmapid` 命令使用的 NFSv4 域参数。要完成脱离手动干预的初始区域配置，请编辑文件 `default/nfs`，取消对 `NFSMAPID_DOMAIN` 参数的注释，并将域设置为所需的 NFSv4 域：**

```
global# vi default/nfs
        .
        .
        .
        NFSMAPID_DOMAIN=domain
```

有关 NFSv4 域参数的更多信息，请参见 `nfsmapid`(1M) 手册页。

5 **在此目录中创建文件 `.NFS4inst_state.domain`，以表明已经设置 NFSv4 域：**

```
global# touch .NFS4inst_state.domain
```

6 **引导区域。**

另请参见 有关更多信息，请参见 `sysidcfg`(4) 手册页。

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 257 页中的 “远程登录”中所述。

▼ 如何登录到区域控制台

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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
```

注 – 如果您在发出 `zoneadm boot` 命令之后立即启动 `zlogin` 会话，则会显示区域的引导消息：

```
SunOS Release 5.10 Version Generic 64-bit
Copyright 1983-2005 Sun Microsystems, Inc. All rights reserved.
Use is subject to license terms.
starting rpc services: rpcbind done.
syslog service starting.
The system is ready.
```

---

3 **当显示区域控制台时，以 `root` 身份登录，按回车键，并在提示时键入超级用户口令。**

```
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。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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
Sun Microsystems Inc. SunOS 5.10 Generic June 2004
```

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。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的"使用 RBAC 和 Solaris 管理工具（任务图）"。

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
```

您会看到以下输出：

```
my-zone
```

▼ 如何退出非全局区域

- **要退出区域，请键入：**

```
zonename# exit
```

将显示以下类似信息：

```
[Connection to zone 'my-zone' pts/6 closed]
```

您还可以通过键入波浪号 (~) 字符和句点从非全局区域断开。

```
zonename# ~.
```

另请参见 有关 `zlogin` 命令选项的更多信息，请参见 `zlogin`(1)。

▼ 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。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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 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 250 页中的 “如何停止区域”。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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。

更多信息 在非交互模式下使用 shutdown

当前，不能在非交互模式下使用 `shutdown` 命令将区域置于单用户状态。有关更多信息，请参见 CR 6214427。

您可以使用第 264 页中的 “如何使用交互模式访问区域”中介绍的交互式登录。

将非全局区域切换到其他网络服务配置

此区域安装时使用了《系统管理指南：基本管理》中的第 15 章“管理服务（任务）”介绍的开放式网络配置。您可以将区域切换到受限的网络配置，也可以在区域中启用或禁用个别服务。

▼ 如何将区域切换到受限的网络服务配置

1 从全局区域登录到某个区域（例如 my-zone）。

```
global# zlogin my-zone
```

2 运行 netservices 命令，将区域切换到受限的网络配置。

```
my-zone# /usr/sbin/netservices limited
```

将显示以下类似信息。输入 *y* 响应提示，以重新启动 dtlogin。

```
restarting syslogd
restarting sendmail
dtlogin needs to be restarted. Restart now? [Y] y
restarting dtlogin
```

▼ 如何在区域中启用特定服务

1 从全局区域登录到某个区域（例如 my-zone）。

```
global# zlogin my-zone
```

2 运行 svcadm 命令，以使用资源上限设置守护进程启用物理内存控制。

```
my-zone# svcadm enable svc:/system/rcap:default
```

3 列出服务，以验证是否已启用 rcapd。

```
my-zone# svcs -a
.
.
.
online    14:04:21 svc:/system/rcap:default
.
.
.
```

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 23 章

23 移动和迁移非全局区域（任务）

本章为 Solaris 10 11/06 发行版的新增内容。后续发行版中还添加了其他功能。

本章介绍如何：

- 将现有的非全局区域移动到同一计算机上的新位置
- 在执行实际迁移前验证将在非全局区域迁移中发生的情况。
- 将现有的非全局区域迁移到新计算机中

Solaris 10 11/06：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。新 zonepath 必须位于本地文件系统中。需要满足第 207 页中的 “资源和属性类型”中介绍的标准 zonepath 条件。

▼ 如何移动区域

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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** /export/zones/db-zone。

```
global# zoneadm -z db-zone move /export/zones/db-zone
```

4 检验路径。

```
ID   NAME      STATUS       PATH                          BRAND      IP
 0   global    running      /                             native     shared
 -   my-zone   installed    /export/home/my-zone          native     shared
 -   db-zone   installed    /export/zones/db-zone         native     shared
```

Solaris 10 11/06：将非全局区域迁移到其他计算机

请注意，使用 Solaris 10 5/08 发行版时，可以在将区域实际移动到其他计算机之前执行区域迁移试验。有关更多信息，请参见第 273 页中的 “Solaris 10 5/08：关于在执行迁移之前验证区域迁移”。

关于迁移区域

自 Solaris 10 11/06 发行版以来，本节添加了新信息。

`zonecfg` 和 `zoneadm` 命令可用于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。

以下限制适用于区域迁移：

- 目标系统上的全局区域必须与原始主机运行相同的 Solaris 发行版。
- 为确保区域可以正常运行，目标系统上所需安装的以下操作系统软件包和修补程序必须与原始主机上安装的软件包和修补程序具有相同的版本。
  - 在 `inherit-pkg-dir` 资源下提供文件的软件包
  - `SUNW_PKG_ALLZONES=true` 的软件包

  其他软件包和修补程序（例如用于第三方产品的软件包和修补程序）可以有所不同。
- 主机系统和目标系统必须具有相同的计算机体系结构。

要检验 Solaris 发行版本和计算机体系结构，请键入：

```
#uname -m
```

`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。

▼ 如何迁移非全局区域

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **停止要迁移的区域，在此过程中为** `my-zone`**。**

```
host1# zoneadm -z my-zone halt
```

3 **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。

4 **将** `my-zone` **的** `zonepath` **移动到新主机。**

有关更多信息，请参见第 272 页中的 “如何将 zonepath 移动到新主机”。

5 **在新主机上，对该区域进行配置。**

```
host2# zonecfg -z my-zone
```

您会看到以下系统消息：

```
my-zone: No such zone configured
Use 'create' to begin configuring a new zone.
```

6 **要在新主机上创建区域** `my-zone`**，请使用带有** `-a` **选项以及新主机上的** `zonepath` **的** `zonecfg` **命令。**

```
zonecfg:my-zone> create -a /export/zones/my-zone
```

7 **（可选）查看配置。**

```
zonecfg:my-zone> info
zonename: my-zone
zonepath: /export/zones/my-zone
autoboot: false
pool:
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
```

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。

```
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 **在新主机上附加区域。**

- **附加区域，并进行验证检查。**

  ```
  host2# zoneadm -z my-zone attach
  ```

  当发生下列一种或两种情况时，将向系统管理员通知所需执行的操作：

  - 新计算机中不存在所需软件包和修补程序。
  - 计算机之间的软件级别不同。

- **强制执行附加操作，而不执行验证。**

  ```
  host2# zoneadm -z my-zone attach -F
  ```

注意 – `-F` 选项可以在不进行验证的情况下强制执行 `attach`。这在某些情况下（例如在群集环境中或在执行备份和恢复操作时）很有用，但要求对系统进行托管区域所需的正确配置。不正确的配置以后可能会导致未定义的行为。

▼ 如何将 zonepath 移动到新主机

创建 `zonepath` 的归档的方法有很多种。例如，可以使用 `cpio`(1) 和 `pax`(1) 手册页中所述的 `cpio` 或 `pax` 命令。

将归档传送至新主机的方法也有很多种。用于将 `zonepath` 从源主机传送到目标主机的机制取决于本地配置。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`。在其他情况下，可能要将 `zonepath` 写入磁带，再将磁带邮寄至新站点。

由于上述原因，此步骤不能自动执行。系统管理员必须选择最合适的方法来将 `zonepath` 移动到新主机。

1. **成为超级用户或承担主管理员角色。**
   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **将 zonepath 移动到新主机。您可以使用本过程中介绍的方法，也可以使用您选择的其他方法。**

示例 23–1 使用 tar 命令归档和移动 zonepath

1. 在 host1 上创建 zonepath 的 tar 文件，并使用 sftp 命令将其传送到 host2。

```
host1# cd /export/zones
host1# tar cf my-zone.tar my-zone
host1# sftp host2
Connecting to host2...
Password:
sftp> cd /export/zones
sftp> put my-zone.tar
Uploading my-zone.tar to /export/zones/my-zone.tar
sftp> quit
```

2. 在 host2上，解压缩 tar 文件。

```
host2# cd /export/zones
host2# tar xf my-zone.tar
```

有关更多信息，请参见 sftp(1) 和 tar(1)。

故障排除 有关以下问题的疑难解答信息，请参见第 355 页中的 “使用 zoneadm attach 操作解决问题”：

- 修补程序和软件包不同步。
- 操作系统发行版不匹配。

接下来的操作 如果已复制数据而未重新配置 SAN，那么即使该区域现在处于已配置状态，zonepath 数据在源主机上也仍然可见。您可以在将数据移动到新主机后从源主机上手动删除 zonepath，也可以将该区域重新附加到源主机，并使用 zoneadm uninstall 命令删除 zonepath。

Solaris 10 5/08：关于在执行迁移之前验证区域迁移

可以在将区域移动到新计算机之前使用“不执行”选项 -n 执行试验。

zoneadm detach 子命令与 -n 选项结合使用，可在运行的区域上生成清单，而不实际分离该区域。源系统中区域的状态不会改变。区域清单会被发送到 stdout。全局管理员

可以将此输出定向到某一文件，或将此输出传输到远程命令以便立即在目标主机上进行验证。zoneadm attach 子命令与 -n 选项结合使用，可读取该清单并检验目标计算机是否具有托管区域所需的正确配置，而不实际执行附加。

在执行试验性附加之前，不必在新主机上配置目标系统中的区域。

▼ Solaris 10 5/08：如何在执行迁移之前验证区域迁移

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**

   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **使用以下方法之一。**

   - **在名为 my-zone 的源主机上生成清单，并将输出传输到将立即验证目标主机的远程命令：**

     ```
     global# zoneadm -z my-zone detach -n | ssh remotehost zoneadm attach -n -
     ```

     行尾的连字符 (–) 指定了路径为 stdin。

   - **在名为 my-zone 的源主机上生成清单，并将输出定向到某一文件：**

     ```
     global# zoneadm -z my-zone detach -n
     ```

     如第 272 页中的 “如何将 zonepath 移动到新主机” 中所述将清单复制到新主机系统，并执行验证：

     ```
     global# zoneadm attach -n path_to_manifest
     ```

     路径可以是 –，以便指定 stdin。

从不可用的计算机上迁移区域

托管本地 Solaris 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，那么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。尽管新主机将告知没有正确分离区域，但系统仍将尝试附加。

执行此任务的过程在第 271 页中的 “如何迁移非全局区域” 的步骤 4 到 8 中介绍。另请参见第 272 页中的 “如何将 zonepath 移动到新主机” 。

◆ ◆ ◆ 第 24 章

关于安装了区域的 Solaris 系统上的软件包和修补程序（概述）

Solaris 10 1/06：对本章进行了彻底修订。

本章介绍如何在安装区域后维护 Solaris 操作系统。提供了有关在全局区域和所有已安装的非全局区域中向操作系统添加软件包和修补程序的信息，同时还包含有关删除软件包和修补程序的信息。本章中的材料是对现有 Solaris 安装和修补程序文档的补充。有关更多信息，请参见 Solaris 10 Release and Installation Collection - Simplified Chinese 和《系统管理指南：基本管理》。

本章包含以下主题：

安装区域时打包和修补方面的新增功能

Solaris 10 1/06：从 Solaris 10 开始，为了记录安装了非全局区域的系统上的软件包命令和修补程序命令的当前行为，重新编写了本章内容。

Solaris 10 6/06：有关 `SUNW_PKG_ALLZONES`、`SUNW_PKG_HOLLOW` 和 `SUNW_PKG_THISZONE` 软件包参数的信息已修订。请参见第 276 页中的 “打包工具和修补程序工具概述” 和第 284 页中的 “软件包参数信息”。

Solaris 10 6/06 及更高发行版：有关如何注册系统或如何使用 Sun Connection（以前称为 Sun Update Connection）管理软件更新的信息，请访问 BigAdmin 站点中的 Sun Connection 信息中心。

Solaris 10 8/07 及更高发行版：

- 使用 `patchadd` 命令向软件包中添加修补程序时，如果该软件包是使用带有 `-G` 选项的 `pkgadd` 命令安装的，则 `patchadd` 命令将不再需要 `-G` 选项。
- 添加了一个表，此表介绍在非全局区域处于各种状态的系统中使用 `pkgadd`、`pkgrm`、`patchadd` 和 `patchrm` 命令时将会发生的情况。请参见第 279 页中的 “区域状态对修补程序和软件包操作有何影响”。
- 添加了对 patchadd `-G` 和 `pkginfo` 变量之间的交互的阐述。请参见第 293 页中的 “安装有区域的系统上的 patchadd `-G` 和 `pkginfo` 变量之间的交互”。
- 添加了关于延迟激活修补的信息。请参见第 291 页中的 “Solaris 10 8/07：延迟激活修补”。
- 删除了关于 `pkgrm` 命令的 `-G` 选项的信息。

Solaris 10 5/08 及更高的更新发行版：PatchPro 的 EOF。对 PatchPro 的支持于 2007 年 9 月结束。PatchPro 曾使用修补程序数据库和修补程序工具修补全局和非全局区域中安装的软件。有关最新进程的信息，请参见 Sun xVM Ops Center。

Solaris 10 5/08：尽管是在 Solaris 10 5/08 发行版中添加了此信息，但此信息适用于所有 Solaris 10 系统。

要注册您的 Solaris 系统，请访问 `https://inventory.sun.com/inventory/`。有关如何使用 Sun™ Inventory 注册硬件、软件和操作系统的信息，请参见 Sun Inventory 信息中心。

如果使用 Sun xVM Ops Center 在数据中心中置备、更新和管理系统，请参见 Sun xVM 信息中心 (`http://wikis.sun.com/display/xVM/Sun+xVM+Ops+Center`)，了解有关如何向 Sun xVM Ops Center 注册软件的信息。

有关 Solaris 10 新增功能的完整列表以及 Solaris 发行版的说明，请参见《Solaris 10 新增功能》。

打包工具和修补程序工具概述

Solaris 打包工具用于管理区域环境。全局管理员可以将系统升级到新版本的 Solaris，此操作会同时更新全局区域和非全局区域。

Solaris Live Upgrade、标准 Solaris 交互式安装程序或自定义 JumpStart 安装程序可用于在全局区域中对包含非全局区域的系统进行升级。

在本文档介绍的限制范围之内，区域管理员可以使用打包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：

- 全局区域管理员可以管理系统上每个区域中的软件。
- 通过使用 Solaris 打包和修补程序工具，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 Solaris 打包和修补程序工具管理共同打包（捆绑）产品、独立（非绑定）产品或第三方产品。
- 打包和修补程序工具在启用了区域的环境中工作。使用这些工具，还可以将全局区域中安装的软件包或修补程序安装在非全局区域中。
- SUNW_PKG_ALLZONES 软件包参数定义软件包的**区域范围**。此范围确定了可以安装单独软件包的区域类型。有关此参数的更多信息，请参见第 286 页中的“SUNW_PKG_ALLZONES 软件包参数”。
- 需要在所有区域中安装某软件包并要求此软件包在所有区域中均相同时，可使用 SUNW_PKG_HOLLOW 软件包参数定义软件包的**可见性**。有关此参数的信息，请参见第 288 页中的“SUNW_PKG_HOLLOW 软件包参数”。
- SUNW_PKG_THISZONE 软件包参数定义是否必须将软件包仅安装在当前区域中。有关此参数的信息，请参见第 289 页中的“SUNW_PKG_THISZONE 软件包参数”。
- 未定义区域软件包参数值的软件包具有缺省设置 `false`。
- 非全局区域中可见的打包信息与使用 Solaris 打包和修补程序工具已安装在该区域中的文件一致。打包信息与 `inherit-pkg-dir` 目录保持同步。
- 可以将某项更改（例如添加到全局区域中的修补程序或软件包）应用到所有区域。此功能可保持全局区域和每个非全局区域之间的一致性。
- 软件包命令可以添加、删除和询问软件包。修补程序命令可以添加和删除修补程序。

注 – 当执行某些软件包和修补程序操作时，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

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 Solaris 软件包。例如，很多包含 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 Solaris 内核。但是，即使非全局区域不需要或者不使用软件包的数据，非全局区域也可能需要在全局区域中安装软件包的信息。利用此信息，可以通过全局区域正确解析非全局区域中软件包的相关性。

软件包的参数可以控制软件包内容的分发方式，并使这些内容在安装了非全局区域的系统上可见。SUNW_PKG_ALLZONES、SUNW_PKG_HOLLOW 和 SUNW_PKG_THISZONE 软件包参数定义安装了区域的系统上的软件包的特征。如果需要，在区域环境中应用或删除软件包时，系统管理员可以检查这些软件包参数的设置，以检验软件包的适用性。可以使

用 pkgparam 命令查看这些参数的值。有关参数的更多信息，请参见第 284 页中的 “软件包参数信息”。有关使用说明，请参见第 302 页中的 “在安装了区域的系统上检查软件包参数设置”。

有关软件包特征和参数的信息，请参见 pkginfo(4) 手册页。有关显示软件包参数值的信息，请参见 pkgparam(1) 手册页。

针对软件包生成的修补程序

针对任何软件包生成修补程序时，必须将参数设置为与原始软件包相同的值。

交互式软件包

任何必须为交互式的软件包（这意味着该软件包具有请求脚本）仅添加到当前区域。软件包不会传播到其他任何区域。如果将交互式软件包添加到全局区域中，则对该软件包的处理类似于使用带有 -G 选项的 pkgadd 命令进行添加。有关此选项的更多信息，请参见第 280 页中的 “关于在区域中添加软件包”。

保持区域同步

最好使非全局区域中安装的软件与全局区域中安装的软件尽可能保持同步。此做法可最大限度地降低管理安装了多个区域的系统的难度。

要实现此目标，在全局区域中添加或删除软件包时，软件包工具会强制执行以下规则。

全局区域中可能的软件包操作

如果软件包当前既没有安装在全局区域中，也没有安装在任何非全局区域中，则可将其安装在以下位置：

- 仅安装在全局区域中，前提是 SUNW_PKG_ALLZONES=false
- 仅安装在当前（全局）区域中，前提是 SUNW_PKG_THISZONE=true
- 安装在全局区域和所有非全局区域中

如果软件包当前仅安装在全局区域中：

- 该软件包可以安装在所有非全局区域中。
- 该软件包可以从全局区域中删除。

如果软件包当前安装在全局区域和仅部分非全局区域中：

- SUNW_PKG_ALLZONES 必须设置为 false。

- 该软件包可以安装在所有非全局区域中。任何非区域中的现有实例都会更新到安装的修订版。
- 该软件包可以从全局区域中删除。
- 该软件包可以从全局区域和所有非全局区域中删除。

如果软件包当前安装在全局区域和所有非全局区域中，则该软件包可以从全局区域和所有非全局区域中删除。

这些规则可以确保以下情况：

- 安装在全局区域中的软件包仅安装在全局区域中，或是安装在全局区域和所有非全局区域中。
- 安装在全局区域和任何非全局区域中的软件包在所有区域中均相同。

非全局区域中可能的软件包操作

任何非全局区域中可能的软件包操作包括：

- 如果软件包当前未安装在非全局区域中，则仅当 `SUNW_PKG_ALLZONES=false` 时，才能安装软件包。
- 软件包可以安装在当前（非全局）区域中，前提是 `SUNW_PKG_THISZONE=true`。
- 如果软件包当前安装在非全局区域中：
  - 仅当 `SUNW_PKG_ALLZONES=false` 时，才可以通过软件包的现有实例安装软件包。
  - 仅当 `SUNW_PKG_ALLZONES=false` 时，才能从非全局区域中删除软件包。

区域状态对修补程序和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 `pkgadd`、`pkgrm`、`patchadd` 和 `patchrm` 命令时将发生的情况。

请注意，已针对 Solaris 10 5/08 发行版修订了此表中已安装状态的说明。

区域状态	对软件包和修补程序操作的影响
已配置	修补程序和软件包工具可以运行。尚未安装任何软件。

区域状态	对软件包和修补程序操作的影响
已安装	修补程序和软件包工具可以运行。在修补或打包操作期间，系统会将区域从已安装状态移至称为已挂载的新内部状态。完成修补后，区域将恢复为已安装状态。 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。处于已安装状态并且从未引导的区域不能进行修补，也不能运行打包命令。至少要将该区域引导至正在运行状态一次。在至少引导一次区域、然后通过 `zoneadm halt` 再返回已安装状态后，便可运行修补程序和打包命令。
就绪	修补程序和软件包工具可以运行。
正在运行	修补程序和软件包工具可以运行。
未完成	`zoneadm` 正在安装或删除区域。无法使用软件包和修补程序工具。这些工具不能使区域进入使用工具所需的相应状态。

关于在区域中添加软件包

使用 pkgadd(1M) 手册页中所述的 pkgadd 系统实用程序，可以在安装了区域的 Solaris 系统上添加软件包。

在全局区域中使用 pkgadd

pkgadd 实用程序可以在全局区域中与 -G 选项一起使用，以仅向全局区域中添加软件包。软件包不会传播到其他任何区域。请注意，如果 SUNW_PKG_THISZONE=true，则不必使用 -G 选项。如果 SUNW_PKG_THISZONE=false，则 -G 选项会将其覆盖。

在全局区域中运行 pkgadd 实用程序时，将会进行以下操作。

- pkgadd 实用程序可以将软件包添加到以下位置：
  - 仅全局区域，除非软件包设置为 SUNW_PKG_ALLZONES=true
  - 全局区域和所有非全局区域
  - 仅所有非全局区域，前提是软件包已安装在全局区域中
  - 仅当前区域，前提是 SUNW_PKG_THISZONE=true
- pkgadd 实用程序不能将软件包添加到以下位置：
  - 非全局区域的任何部分
  - 所有非全局区域，除非软件包已安装在全局区域中

- 如果在不带 -G 选项并且 SUNW_PKG_THISZONE=false 的情况下运行 pkgadd 实用程序，则缺省情况下会将指定软件包添加到所有区域。软件包不会标记为仅安装在全局区域中。
- 如果在不带 -G 选项并且 SUNW_PKG_THISZONE=true 的情况下运行 pkgadd 实用程序，则缺省情况下会将指定软件包添加到当前（全局）区域。软件包标记为仅安装在全局区域中。
- 如果使用 -G 选项，则 pkgadd 实用程序将指定的软件包仅添加到全局区域。软件包标记为仅安装在全局区域中。如果安装了任何非全局区域，则不会安装软件包。

添加软件包到全局区域和所有非全局区域

要将软件包添加到全局区域和所有非全局区域，请在全局区域中执行 pkgadd 实用程序。以全局管理员身份运行不带 -G 选项的 pkgadd。

可以将软件包添加到全局区域和所有非全局区域中，而不用考虑受其影响的区域。

pkgadd 实用程序会执行以下步骤：

- 检查全局区域和所有非全局区域中软件包的相关性。如果任何区域中都没有安装所需的软件包，则相关性检查将失败。系统会通知全局管理员，提示其是否继续。
- 将软件包添加到全局区域。
- 更新全局区域中的软件包数据库。
- 将软件包添加到每个非全局区域中并更新全局区域中的数据库。
- 更新每个非全局区域中的软件包数据库。

仅向全局区域中添加软件包

要仅向全局区域中添加软件包，请在全局区域中以全局管理员身份执行仅带有 -G 选项的 pkgadd 实用程序。

如果以下情况成立，则可以将软件包添加到全局区域中：

- 该软件包的内容不会影响全局区域中与任何非全局区域共享的任何区域。
- 该软件包设置为 SUNW_PKG_ALLZONES=false。

pkgadd 实用程序会执行以下步骤：

- 如果该软件包的内容影响全局区域中与任何非全局区域共享的任何区域，或者软件包设置为 SUNW_PKG_ALLZONES=true，则 pkgadd 将失败。将出现错误消息，告知必须将软件包添加到全局区域和所有非全局区域中。
- 仅对全局区域执行软件包的相关性检查。如果未安装所需的软件包，则相关性检查将失败。系统会通知全局管理员，提示其是否继续。
- 将软件包添加到全局区域。
- 更新全局区域中的软件包数据库。

- 注释全局区域中的软件包信息，指明仅在全局区域中安装该软件包。如果将来安装非全局区域，则不会安装该软件包。

向所有非全局区域中添加安装在全局区域中的软件包

要向所有非全局区域中添加已安装在全局区域中的软件包，当前必须从全局区域中删除该软件包，然后在所有区域中重新安装。

以下是用于向所有非全局区域中添加已安装在全局区域中的软件包的步骤：

1. 在全局区域中，使用 pkgrm 删除软件包。
2. 在不使用 -G 选项的情况下添加软件包。

在非全局区域中使用 pkgadd

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不带任何选项的 pkgadd 实用程序。需要满足以下条件：

- pkgadd 实用程序只能将软件包添加到使用该实用程序的非全局区域中。
- 该软件包不能影响该区域中从全局区域共享的任何区域。
- 该软件包必须设置为 SUNW_PKG_ALLZONES=false。

pkgadd 实用程序会执行以下步骤：

- 添加软件包之前，针对非全局区域的软件包数据库检查软件包的相关性。如果未安装所需的软件包，则相关性检查将失败。系统会通知非全局区域管理员，提示其是否继续。如果以下任何一种情况成立，检查将会失败。
  - 软件包的任何组件影响到该区域中从全局区域共享的任何区域。
  - 软件包设置为 SUNW_PKG_ALLZONES=true。
- 将软件包添加到区域中。
- 更新区域中的软件包数据库。

关于在区域中删除软件包

pkgrm(1M) 手册页中所述的 pkgrm 实用程序支持在安装了区域的 Solaris 系统上删除软件包。

在全局区域中使用 pkgrm

在全局区域中使用 pkgrm 实用程序时，将进行以下操作。

- pkgrm 可以从全局区域和所有非全局区域中删除软件包；如果仅在全局区域中安装了软件包，则 pkgrm 只能从全局区域中删除软件包。

- 如果软件包还安装在非全局区域中，则 pkgrm 不能仅从全局区域中删除软件包，也不能从非全局区域的任何部分中删除软件包。

请注意，只有在以下情况成立时，才能由非全局区域中的区域管理员将软件包从该区域中删除。

- 该软件包不会影响非全局区域中从全局区域共享的任何区域。
- 该软件包设置为 SUNW_PKG_ALLZONES=false。

从全局区域和所有非全局区域中删除软件包

要从全局区域和所有非全局区域中删除软件包，请以全局管理员的身份在全局区域中执行 pkgrm 实用程序。

可以从全局区域和所有非全局区域中删除软件包，而不用考虑受其影响的区域。

pkgrm 实用程序会执行以下步骤：

- 检查全局区域和所有非全局区域中软件包的相关性。如果相关性检查失败，则 pkgrm 也会失败。系统会通知全局管理员，提示其是否继续。
- 从每个非全局区域中删除软件包。
- 更新每个非全局区域中的软件包数据库。
- 从全局区域中删除软件包。
- 更新全局区域中的软件包数据库。

在非全局区域中使用 pkgrm

以区域管理员身份在非全局区域中使用 pkgrm 实用程序来删除软件包。将应用以下限制：

- pkgrm 仅能从非全局区域中删除软件包。
- 该软件包不能影响该区域中从全局区域共享的任何区域。
- 该软件包必须设置为 SUNW_PKG_ALLZONES=false。

pkgrm 实用程序会执行以下步骤：

- 针对非全局区域中的软件包数据库检查相关性。如果相关性检查失败，则 pkgrm 也会失败并通知区域管理员。如果以下任何一种情况成立，检查将会失败。
  - 软件包的任何组件影响到该区域中从全局区域共享的任何区域。
  - 软件包设置为 SUNW_PKG_ALLZONES=true。
- 将软件包从区域中删除。
- 更新区域中的软件包数据库。

软件包参数信息

设置区域的软件包参数

SUNW_PKG_ALLZONES、SUNW_PKG_HOLLOW 和 SUNW_PKG_THISZONE 软件包参数定义安装了区域的系统上的软件包的特征。必须设置这些参数，以便可在安装了非全局区域的系统上管理软件包。

下表列出了设置软件包参数的四种有效组合。如果您选择的设置组合未在下表中列出，则这些设置无效并且将无法安装软件包。

请确保您已设置了全部三个软件包参数。可以将这三个软件包参数保留为空。软件包工具会将缺少的区域软件包参数解释成设置为 false，但是绝对不建议不设置这些参数。通过设置全部三个软件包参数，可以指定安装或删除软件包时，软件包工具应当表现的确切行为。

表 24–1 有效的软件包参数设置

SUNW_PKG_ALLZONES 设置	SUNW_PKG_HOLLOW 设置	SUNW_PKG_THISZONE 设置	软件包说明
false	false	false	此为软件包的缺省设置，该设置不会指定所有区域软件包参数的值。 具有这些设置的软件包既可安装在全局区域中，也可安装在非全局区域中。 ■ 如果在全局区域中运行 pkgadd 命令，则会将软件包安装在全局区域和所有非全局区域中。 ■ 如果在非全局区域中运行 pkgadd 命令，则仅将软件包安装在非全局区域中。 在这两种情况下，软件包的所有内容都会在其安装所在的所有区域中可见。

表 24–1 有效的软件包参数设置 （续）

SUNW_PKG_ALLZONES 设置	SUNW_PKG_HOLLOW 设置	SUNW_PKG_THISZONE 设置	软件包说明
false	false	true	具有这些设置的软件包既可安装在全局区域中，也可安装在非全局区域中。如果在安装软件包之后创建新的非全局区域，则软件包不会传播到这些新的非全局区域。 ■ 如果在全局区域中运行 pkgadd 命令，则仅将软件包安装在全局区域中。 ■ 如果在非全局区域中运行 pkgadd 命令，则仅将软件包安装在非全局区域中。 在这两种情况下，软件包的所有内容都会在其安装所在的区域中可见。
true	false	false	具有这些设置的软件包只能安装在全局区域中。运行 pkgadd 命令时，会将软件包安装在全局区域和所有非全局区域中。软件包的所有内容在所有区域中可见。 **注 –** 任何将软件包安装在非全局区域中的尝试都会失败。

表 24–1 有效的软件包参数设置 （续）

SUNW_PKG_ALLZONES 设置	SUNW_PKG_HOLLOW 设置	SUNW_PKG_THISZONE 设置	软件包说明
true	true	false	具有这些设置的软件包只能由全局管理员安装在全局区域中。运行 pkgadd 命令时，软件包的内容会全部安装在全局区域中。如果软件包的软件包参数设置为这些值，则不会在任何非全局区域中提供软件包内容本身。非全局区域中仅会安装使软件包显示为已安装状态所必需的软件包安装信息。这将安装依赖于该软件包的要安装的其他软件包。 为了检查软件包的相关性，该软件包显示为已安装在所有区域中。 ■ 在全局区域中，该软件包的所有内容均可见。 ■ 在完全根非全局区域中，该软件包的所有内容均不可见。 ■ 当非全局区域从全局区域中继承文件系统时，安装在该文件系统中的软件包在非全局区域中可见，而该软件包所提供的所有其他文件在非全局区域中均不可见。 例如，稀疏根非全局区域 (sparse root non-global zone) 与全局区域共享某些目录。这些目录为只读目录。稀疏根非全局区域与其他区域共享 /platform 文件系统。另一个示例为软件包提供仅与引导硬件有关的文件。 **注 –** 任何将软件包安装在非全局区域中的尝试都会失败。

SUNW_PKG_ALLZONES 软件包参数

可选的 SUNW_PKG_ALLZONES 软件包参数说明软件包的区域范围。此参数定义了以下内容：

- 是否需要在所有区域中安装软件包
- 是否需要所有区域中的软件包均相同

SUNW_PKG_ALLZONES 软件包参数有两个允许的值。这些值为 true 和 false。缺省值为 false。如果未设置此参数或将其设置为除 true 或 false 以外的值，则会使用值 false。

对于在所有区域中**必须**是同一软件包版本并属于同一修补程序修订版级别的软件包，应将 SUNW_PKG_ALLZONES 参数设置为 true。对于所提供的功能依赖于某个特定 Solaris 内核（例如 Solaris 10）的任何软件包，应将此参数设置为 true。任何用于软件包的修补程序都必须将 SUNW_PKG_ALLZONES 参数值设置为与正在修补的已安装软件包中设置的值相同。对于将此参数设置为 true 的任何软件包，其修补程序修订版级别在所有区域中都必须相同。

对于所提供的功能不依赖于某个特定 Solaris 内核（例如第三方软件包或 Sun 编译器）的软件包，应将此参数设置为 false。对于将此参数设置为 false 的软件包，其修补程序也必须将此参数设置为 false。对于任何将此参数设置为 false 的软件包，其软件包版本或修补程序修订版级别在不同区域中可以不同。例如，两个非全局区域可以安装不同版本的 Web 服务器。

下表描述了 SUNW_PKG_ALLZONES 软件包的参数值。

表 24–2 SUNW_PKG_ALLZONES 软件包参数值

值	说明
false	该软件包仅能从全局区域安装到全局区域，或从全局区域安装到全局区域和所有非全局区域。该软件包还可以从任何非全局区域安装到同一非全局区域。 ■ 全局管理员仅能将软件包安装在全局区域中。 ■ 全局管理员可以将软件包安装在全局区域和所有非全局区域中。 ■ 区域管理员可将软件包安装在非全局区域中。 如果将软件包从全局区域中删除，则不会将其从其他区域中删除。可以将软件包从单个非全局区域中删除。 ■ 不要求将软件包安装在全局区域中。 ■ 不要求将软件包安装在任何非全局区域中。 ■ 不要求软件包在所有区域中均相同。各区域可以有不同版本的软件包。 ■ 软件包提供的软件不在所有区域中隐含共享。这意味着软件包并不是操作系统特定的。大多数应用程序级的软件都属于这一类别。例如 StarSuite™ 产品或 Web 服务器。

表 24–2 SUNW_PKG_ALLZONES 软件包参数值 （续）

值	说明
true	如果将软件包安装在全局区域中，则还必须将其安装在所有非全局区域中。如果将软件包从全局区域中删除，则还必须将其从所有非全局区域中删除。 ■ 如果安装软件包，必须将其安装在全局区域中。然后，该软件包会自动安装在所有非全局区域中。 ■ 软件包的版本在所有区域中均必须相同。 ■ 软件包提供的软件在所有区域中隐含共享。软件包依赖于在所有区域中隐含共享的软件版本。软件包应当在所有非全局区域中可见。例如内核模块。 利用这些软件包，非全局区域可以通过获取所有非全局区域中安装的整个软件包，来解析安装在全局区域中的软件包的相关性。 ■ 仅有全局管理员才能安装软件包。区域管理员不能在非全局区域中安装软件包。

SUNW_PKG_HOLLOW 软件包参数

SUNW_PKG_HOLLOW 软件包参数定义了在要求将软件包安装在所有区域中并且在所有区域内均相同的情况下，该软件包是否应当在任何非全局区域中可见。

SUNW_PKG_HOLLOW 软件包参数有两个允许的值 true 或 false。

- 如果未设置 SUNW_PKG_HOLLOW 或将其设置为除 true 和 false 以外的值，则会使用值 false。
- 如果 SUNW_PKG_ALLZONES 设置为 false，则会忽略 SUNW_PKG_HOLLOW 参数。
- 如果 SUNW_PKG_ALLZONES 设置为 false，则 SUNW_PKG_HOLLOW 不能设置为 true。

下表描述了 SUNW_PKG_HOLLOW 软件包的参数值。

表 24–3 SUNW_PKG_HOLLOW 软件包参数值

值	说明
false	这不是“空”软件包： ■ 如果将软件包安装在全局区域中，则必须在所有非全局区域中提供软件包的内容和安装信息。 ■ 该软件包提供的软件应在所有非全局区域中可见。例如提供 truss 命令的软件包。 ■ 除 SUNW_PKG_ALLZONES 软件包参数的当前设置的限制以外，未定义其他限制。

表 24–3 SUNW_PKG_HOLLOW 软件包参数值 （续）

值	说明
true	这是“空”软件包： ■ 任何非全局区域中均未显示软件包的内容。但是，要求在所有非全局区域中提供软件包的安装信息。 ■ 软件包提供的软件不应在所有非全局区域中可见。例如仅在全局区域中运行的内核驱动程序和系统配置文件。利用该设置，非全局区域可以解析仅安装在全局区域中的软件包的相关性，而无需实际安装软件包数据。 ■ 为了使依赖于正在安装的软件包的其他软件包进行相关性检查，该软件包识别为安装在所有区域中。 ■ 该软件包设置包括针对将 SUNW_PKG_ALLZONES 设置为 true 所定义的所有限制。 ■ 在全局区域中，软件包识别为已安装，并且安装了该软件包的所有组件。安装软件包时，会创建目录、安装文件，并相应地运行类操作和其他脚本。 ■ 在非全局区域中，软件包识别为已安装，但是未安装该软件包的任何组件。安装软件包时，不创建任何目录、不安装任何文件，也不运行任何类操作或其他安装脚本。 ■ 从全局区域中删除软件包时，系统会将该软件包识别为已完全安装。删除软件包时，会删除相应的目录和文件、运行类操作或其他安装脚本。

SUNW_PKG_THISZONE 软件包参数

SUNW_PKG_THISZONE 软件包参数定义了是否必须将软件包仅安装在当前区域（全局或非全局区域）中。SUNW_PKG_THISZONE 软件包参数有两个允许的值。这些值为 true 和 false。缺省值为 false。

下表描述了 SUNW_PKG_THISZONE 软件包的参数值。

表 24–4 SUNW_PKG_THISZONE 软件包参数值

值	说明
false	■ 如果在非全局区域中运行 pkgadd，则软件包仅安装在当前区域中。 ■ 如果在全局区域中运行 pkgadd，则软件包会安装在全局区域以及所有当前安装的非全局区域中。此外，软件包将传播到所有将来新安装的非全局区域。

表 24–4 SUNW_PKG_THISZONE 软件包参数值 （续）

值	说明
true	■ 软件包仅安装在当前区域中。 ■ 如果将软件包安装在全局区域中，则不会将该软件包添加到任何当前现有的或待创建的非全局区域中。这与将 -G 选项指定到 pkgadd 时出现的行为相同。

软件包信息查询

pkginfo(1) 手册页中所述的 pkginfo 实用程序支持查询安装了区域的 Solaris 系统上的软件包数据库。有关该数据库的信息，请参见第 294 页中的 “产品数据库”。

可以在全局区域中使用 pkginfo 实用程序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 pkginfo 实用程序来查询仅位于非全局区域中的软件包数据库。

关于在区域中添加修补程序

通常，修补程序由以下几个部分组成：

- 修补程序信息：
  - 标识，即修补程序版本和修补程序 ID
  - 适用性，即操作系统类型、操作系统版本和体系结构
  - 相关性，如需要的兼容性和需要卸载的其他包
  - 属性，如需要随后重新引导
- 要修补的一个或多个软件包，其中每个软件包包含以下内容：
  - 可应用修补程序的软件包的版本
  - 修补程序信息，如 ID、需要卸载的其他包和需要的兼容性
  - 要修补的一个或多个软件包组件

使用 patchadd 命令应用修补程序时，修补程序信息用于确定该修补程序是否适用于当前正在运行的系统。如果确定不适用，则不应用该修补程序。还会针对系统上的所有区域检查修补程序的相关性。如果不满足任一所需的相关性，则不应用该修补程序。这可能包括已安装更高版本的修补程序的情况。

修补程序所包含的每个软件包都会进行检查。如果任何区域中都没有安装软件包，则会跳过软件包，不对其进行修补。

如果满足所有相关性，则会使用任何区域中安装的修补程序内的所有软件包来修补系统。软件包和修补程序数据库也会进行更新。

注 – Solaris 10 3/05 至 Solaris 10 11/06：如果软件包是使用 `pkgadd -G` 安装的，或者其 `pkginfo` 设置是 `SUNW_PKG_THISZONE=true`，则**只能**使用 `patchadd -G` 修补该软件包。Solaris 8/07 发行版中已取消此限制。

Solaris 10 8/07：延迟激活修补

从修补程序 119254-41 和 119255-41 开始，对 `patchadd` 和 `patchrm` 修补程序安装实用程序进行了修改，从而改变了某些修补程序提供功能的处理方式。此修改将影响上述修补程序在任何 Solaris 10 发行版上的安装。这些延迟激活的修补程序能够更好地处理功能修补程序（例如与 Solaris 10 3/05 发行版之后的 Solaris 10 发行版相关联的内核修补程序）中提供的大范围更改。

延迟激活的修补使用回送文件系统 (loopback file system, lofs) 来确保正在运行的系统的稳定性。将修补程序应用于正在运行的系统时，lofs 可以保持修补过程中的稳定性。这些较大的内核修补程序始终需要重新引导，但现在所需的重新引导将激活 lofs 所做的更改。修补程序 README 提供了有关哪些修补程序需要重新引导的说明。

如果正在运行非全局区域或已禁用 lofs，请在安装或删除延迟激活的修补程序时考虑以下几点：

- 要执行此修补程序操作，所有非全局区域都必须处于已停止状态。在应用修补程序之前，必须先停止非全局区域。
- 延迟激活的修补需要回送文件系统 (loopback file system, lofs)。由于启用 lofs 对 HA-NFS 功能的限制，正在运行 Sun Cluster 3.1 或 Sun Cluster 3.2 的系统很可能已禁用 lofs。因此，在安装延迟激活的修补程序之前，必须通过在 `/etc/system` 文件中删除或注释掉以下行来重新启用回送文件系统：

  ```
  exclude:lofs
  ```

  然后，重新引导系统并安装修补程序。在完成修补程序的安装操作之后，请在 `/etc/system` 文件中恢复或取消注释该行。然后，必须重新引导，以恢复正常操作。

注 – 使用 Solaris Live Upgrade 管理修补可以避免与修补正在运行的系统相关联的问题。通过在出现问题时提供回退功能，Solaris Live Upgrade 缩短了修补引起的停机时间，同时降低了风险。请参见《Solaris 10 安装指南：Solaris Live Upgrade 和升级规划》中的“使用软件包或修补程序升级系统”。

在安装了区域的 Solaris 系统上应用修补程序

在全局区域级别上应用的所有修补程序会在所有区域中应用。安装非全局区域后，该区域与全局区域处于同一修补程序级别。修补全局区域时，会对所有非全局区域进行类似的修补。此操作保持所有区域中的修补程序级别相同。

使用 patchadd(1M) 手册页中所述的 patchadd 系统实用程序，可以在安装了区域的系统上添加修补程序。

在全局区域中使用 patchadd

要向全局区域和所有非全局区域中添加修补程序，请以全局管理员身份在全局区域中运行 patchadd。

在全局区域中使用 patchadd 时，将应用以下条件：

- patchadd 实用程序可以将一个或多个修补程序仅添加到全局区域和所有非全局区域中。这是缺省操作。
- patchadd 实用程序不能将修补程序仅添加到全局区域中，也不能将其添加到部分非全局区域中。

在向全局区域和所有非全局区域中添加修补程序时，不必考虑该修补程序是否会影响从全局区域共享的区域。

patchadd 实用程序会执行以下步骤：

- 将修补程序添加到全局区域中。
- 更新全局区域中的修补程序数据库。
- 将修补程序添加到每个非全局区域中。
- 更新每个非全局区域中的修补程序数据库。

在非全局区域中使用 patchadd

当区域管理员在某个非全局区域中使用 patchadd 时，此实用程序仅能将修补程序添加到该区域中。在以下情况下，可将修补程序添加到非全局区域中：

- 该修补程序不影响该区域中从全局区域共享的任何区域。
- 该修补程序中的所有软件包都设置为 SUNW_PKG_ALLZONES=false。

patchadd 实用程序会执行以下步骤：

- 将修补程序添加到区域中。
- 更新区域中的修补程序数据库。

安装有区域的系统上的 patchadd -G 和 pkginfo 变量之间的交互

以下列表指定在全局和非全局区域中添加修补程序时 -G 选项和 SUNW_PKG_ALLZONES 变量之间的交互。

全局区域，已指定 -G	如果任意软件包具有 SUNW_PKG_ALLZONES=TRUE 设置，则将产生错误且不执行任何操作。 如果任何软件包都不具有 SUNW_PKG_ALLZONES=TRUE 设置，则修补程序仅应用于全局区域中的软件包。
全局区域，未指定 -G	如果任意软件包具有 SUNW_PKG_ALLZONES=TRUE 设置，则修补程序将应用于所有区域中的相应软件包。 如果任意软件包不具有 SUNW_PKG_ALLZONES=TRUE 设置，则修补程序将应用于所有相应区域中的这些软件包。仅用于全局区域的软件包将只在全局区域中安装。
非全局区域，已指定或未指定 -G	如果任意软件包具有 SUNW_PKG_ALLZONES=TRUE 设置，则将产生错误且不执行任何操作。 如果任何软件包都不具有 SUNW_PKG_ALLZONES=TRUE 设置，则修补程序仅应用于非全局区域中的软件包。

在安装了区域的 Solaris 系统上删除修补程序

使用 patchrm(1M) 手册页中所述的 patchrm 系统实用程序，可以在安装了区域的系统上删除修补程序。

在全局区域中使用 patchrm

您可以在全局区域中以全局管理员身份使用 patchrm 实用程序来删除修补程序。patchrm 实用程序不能仅从全局区域中删除修补程序，也不能从部分非全局区域中删除它们。

在非全局区域中使用 patchrm

您可以在非全局区域中以区域管理员身份使用 patchrm 实用程序仅从该非全局区域中删除修补程序。修补程序不能影响共享的区域。

产品数据库

每个区域各自的软件包、修补程序和产品注册表数据库都详细介绍了区域中可用的所有已安装软件。安装附加软件或修补程序时要执行所有相关性检查，但不会访问其他任何区域的数据库，除非正在全局区域和一个或多个非全局区域中安装或删除软件包或修补程序。在这种情况下，必须访问相应的非全局区域数据库。

有关数据库的更多信息，请参见 pkgadm(1M) 手册页。

◆ ◆ ◆ 第 25 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）

Solaris 10 1/06：在此发行版中，对本章进行了彻底修订。本章介绍安装了非全局区域的系统上的最新软件包和修补程序过程。

Solaris 10 6/06：向第 296 页中的 “如何仅将软件包添加到全局区域”过程中添加了一条注释。

Solaris 10 8/07：从第 300 页中的 “如何仅将修补程序应用于全局区域”任务中删除了一条注释。

有关 Solaris 10 新增功能的完整列表以及 Solaris 发行版的说明，请参见《Solaris 10 新增功能》。

本章介绍如何在安装了区域的系统上添加和删除软件包和修补程序，还介绍了与管理软件包和修补程序关联的其他任务（例如，检查软件包参数设置和获取软件包信息）。有关安装了区域的系统上的修补和打包概念的概述，请参见第 24 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（任务图）

任务	说明	参考
添加软件包。	在安装了区域的系统上添加软件包。	第 296 页中的 “在安装了区域的 Solaris 系统上添加软件包”
检查软件包信息。	在安装了区域的系统上检查软件包信息。	第 298 页中的 “在安装了区域的 Solaris 系统上检查软件包信息”
删除软件包。	在安装了区域的系统上删除软件包。	第 299 页中的 “从安装了区域的 Solaris 系统中删除软件包”

任务	说明	参考
应用修补程序。	在安装了区域的系统上应用修补程序。	第 300 页中的 “将修补程序应用于安装了区域的 Solaris 系统”
删除修补程序。	在安装了区域的系统上删除修补程序。	第 301 页中的 “在安装了区域的系统上删除修补程序”
（可选）检查软件包参数设置。	当添加或删除软件包时，检验软件包参数的设置是否支持您要执行的操作。	第 302 页中的 “在安装了区域的系统上检查软件包参数设置”

在安装了区域的 Solaris 系统上添加软件包

您可以使用 pkgadd(1M) 手册页中所述的 pkgadd 系统实用程序执行以下任务：

- 仅将软件包添加到全局区域
- 将软件包添加到全局区域和所有非全局区域
- 将已安装在全局区域中的软件包添加到非全局区域
- 仅将软件包添加到指定的非全局区域

要添加软件包，SUNW_PKG_ALLZONES 和 SUNW_PKG_HOLLOW 软件包参数设置必须匹配正确的值（true 或 false）。否则，不会获得所需的结果。有关这些软件包参数设置的影响的更多信息，请参见第 277 页中的 “关于软件包和区域”。有关如何检查这些软件包参数设置的更多信息，请参见第 302 页中的 “在安装了区域的系统上检查软件包参数设置”。

▼ 如何仅将软件包添加到全局区域

要仅将软件包添加到全局区域，必须将 SUNW_PKG_ALLZONES 软件包参数设置为 false。

要执行此过程，您必须是全局区域中的全局管理员。

1 成为超级用户或承担主管理员角色。

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 在全局区域中，运行后跟软件包位置、-G 选项以及软件包名称的 pkgadd -d 命令。

- 如果从 CD-ROM 安装软件包，请键入：

```
global# pkgadd -d /cdrom/cdrom0/directory -G package_name
```

- 如果从已将软件包复制到其中的某个目录安装软件包，请键入：

```
global# pkgadd -d disk1/image -G package_name
```

其中，*disk1* 为软件包的复制位置。

注 – 如果在没有 `-G` 选项和 `SUNW_PKG_THISZONE=true` 的情况下运行 `pkgadd` 实用程序，则缺省情况下会将指定的软件包添加到当前（全局）区域。

▼ 如何将软件包添加到全局区域和所有非全局区域

请不要在此过程中使用 `pkgadd` 选项 `-G`。

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**

   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **在全局区域中，运行后跟软件包位置和软件包名称的 `pkgadd -d` 命令。**

   - 如果从 CD-ROM 安装软件包，请键入：

     ```
     global# pkgadd -d /cdrom/cdrom0/directory  package_name
     ```

   - 如果从已将软件包复制到其中的某个目录安装软件包，请键入：

     ```
     global# pkgadd -d disk1/image package_name
     ```

     其中，*disk1* 为软件包的复制位置。

▼ 如何将已安装在全局区域中的软件包添加到所有非全局区域

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**

   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **在全局区域中，使用 `pkgrm` 删除软件包。**

3. **在不使用 `-G` 选项的情况下添加软件包。**

▼ 如何仅将软件包添加到指定的非全局区域

要仅将软件包添加到指定的非全局区域，必须将 SUNW_PKG_ALLZONES 软件包参数设置为 false。请不要在此过程中使用 pkgadd 选项 -G，否则操作会失败。

要执行此过程，您必须是非全局区域中的区域管理员。

1. **以区域管理员的身份登录到非全局区域。**
2. **在非全局区域（此过程中为 my-zone）中，运行后跟软件包位置和软件包名称的 pkgadd -d 命令。**
   - 如果从 CD-ROM 安装软件包，请键入：

     ```
     my-zone# pkgadd -d /cdrom/cdrom0/directory package_name
     ```
   - 如果从已将软件包复制到其中的某个目录安装软件包，请键入：

     ```
     my-zone# pkgadd -d disk1/image package_name
     ```

     其中，*disk1* 为软件包的复制位置。

在安装了区域的 Solaris 系统上检查软件包信息

您可以使用 pkginfo 命令查询全局区域和非全局区域的软件包数据库。有关此命令的更多信息，请参见 pkginfo(1) 手册页。

▼ 如何仅在全局区域中检查软件包信息

- **要仅检查全局区域的软件包数据库，请使用后跟软件包名称的 pkginfo。**

  ```
  global% pkginfo package_name
  ```

示例 25–1 在全局区域中使用 pkginfo 命令

```
global% pkginfo SUNWcsr SUNWcsu
system      SUNWcsr Core Solaris, (Root)
system      SUNWcsu Core Solaris, (Usr)
```

▼ 如何仅在指定的非全局区域中检查软件包信息

- **要在特定的非全局区域中检查软件包数据库，请登录到该非全局区域，并使用后跟软件包名称的 pkginfo。**

  ```
  my-zone% pkginfo package_name
  ```

示例 25–2 在非全局区域中使用 pkginfo 命令

```
my-zone% pkginfo SUNWcsr SUNWcsu
system      SUNWcsr Core Solaris, (Root)
system      SUNWcsu Core Solaris, (Usr)
```

从安装了区域的 Solaris 系统中删除软件包

您可以使用 pkgrm(1M) 手册页中所述的 pkgrm 系统实用程序执行以下任务：

- 从全局区域和所有非全局区域中删除软件包
- 仅从指定的非全局区域中删除软件包

要删除软件包，SUNW_PKG_ALLZONES 和 SUNW_PKG_HOLLOW 软件包参数设置必须匹配正确的值（true 或 false）。否则，不会获得所需的结果。有关这些软件包参数设置的影响的更多信息，请参见第 277 页中的 “关于软件包和区域”。有关如何检查这些软件包参数设置的更多信息，请参见第 302 页中的 “在安装了区域的系统上检查软件包参数设置”。

▼ 如何从全局区域和所有非全局区域中删除软件包

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**

   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **在全局区域中，运行后跟软件包名称的 pkgrm 命令。**

   ```
   global# pkgrm package_name
   ```

▼ 如何仅从指定的非全局区域中删除软件包

要仅从指定的非全局区域中删除软件包，必须将 SUNW_PKG_ALLZONES 软件包参数设置为 false。

要执行此过程，您必须是非全局区域中的区域管理员。

1. **以区域管理员的身份登录到非全局区域。**

2. **在非全局区域（此过程中为 my-zone）中，运行后跟软件包名称的 pkgrm 命令。**

   ```
   my-zone# pkgrm package_name
   ```

将修补程序应用于安装了区域的 Solaris 系统

您可以使用 patchadd(1M) 手册页中所述的 patchadd 系统实用程序执行以下任务：

- 仅将修补程序应用于全局区域
- 将修补程序应用于全局区域和所有非全局区域
- 仅将修补程序应用于指定的非全局区域

▼ 如何仅将修补程序应用于全局区域

注 – Solaris 10 3/05 至 Solaris 10 11/06：如果要修补使用带有 -G 选项的 pkgadd 命令添加的软件包，则必须使用带有 -G 选项的 patchadd 命令修补该软件包。Solaris 8/07 发行版中已取消此限制。

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**

   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **执行后跟 -G 选项和修补程序 ID 的 patchadd 命令。**

   ```
   global# patchadd -G patch_id
   ```

▼ 如何将修补程序应用于全局区域和所有非全局区域

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**

   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **执行后跟修补程序 ID 的 patchadd 命令。**

   ```
   global# patchadd patch_id
   ```

▼ 如何仅将修补程序应用于指定的非全局区域

要仅将修补程序应用于指定的非全局区域，必须将修补程序集中所有软件包的 SUNW_PKG_ALLZONES 软件包参数设置为 false。

要执行此过程，您必须是非全局区域中的区域管理员。

1. **以区域管理员的身份登录到非全局区域。**
2. **在非全局区域（此过程中为 my-zone）中，执行后跟修补程序 ID 的 patchadd 命令。**

```
my-zone# patchadd patch_id
```

在安装了区域的系统上删除修补程序

您可以使用 patchrm(1M) 手册页中所述的 patchrm 系统实用程序执行以下任务：

- 从全局区域和所有非全局区域中删除修补程序
- 仅从指定的非全局区域中删除修补程序

▼ 如何从全局区域和所有非全局区域中删除修补程序

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**

   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **执行后跟修补程序 ID 的 patchrm 命令。**

```
global# patchrm patch_id
```

▼ 如何仅从指定的非全局区域中删除修补程序

要仅从指定的非全局区域中删除修补程序，必须将修补程序集中所有软件包的 SUNW_PKG_ALLZONES 软件包参数设置为 false。

要执行此过程，您必须是非全局区域中的区域管理员。

1. **以区域管理员的身份登录到非全局区域。**
2. **在非全局区域（此过程中为 my-zone）中，执行后跟修补程序 ID 的 patchrm 命令。**

```
my-zone# patchrm patch_id
```

在安装了区域的系统上检查软件包参数设置

在添加或删除软件包之前，您可以使用 pkgparam 命令来检查软件包参数设置。此步骤是可选的。在解决无法按预期方式添加或删除软件包的问题时，也可以执行此检查。有关显示软件包参数值的信息，请参见 pkgparam(1) 手册页。

▼ （可选）如何检查系统上已安装的软件包的设置

- **要检查已安装在全局区域或非全局区域中的软件包的软件包参数设置，请使用后跟软件包名称和参数名称的 pkgparam。**

```
my-zone% pkgparam package_name SUNW_PKG_ALLZONES
true
my-zone% pkgparam package_name SUNW_PKG_HOLLOW
false
```

▼ （可选）如何检查 CD-ROM 上软件中软件包的设置

- **要检查 CD-ROM 上软件中未安装软件包的参数设置，请使用后跟 CD-ROM 路径、软件包名称以及参数名称的 pkgparam -d。**

```
my-zone% pkgparam -d /cdrom/cdrom0/directory package_name SUNW_PKG_ALLZONES
true
my-zone% pkgparam -d /cdrom/cdrom0/directory package_name SUNW_PKG_HOLLOW
false
```

◆ ◆ ◆ 第 26 章

Solaris Zones 管理（概述）

本章介绍以下常规区域管理主题：

有关 `lx` 标记区域的信息，请参见第 3 部分。

本章新增内容

Solaris 10 1/06：新增了第 307 页中的 “在区域中卸载文件系统”一节。

Solaris 10 1/06：新增了有关区域备份和恢复过程的几节内容。请参见第 323 页中的 “关于备份安装了区域的 Solaris 系统”。

Solaris 10 6/06：ZFS 条目已添加到第 306 页中的 “在区域中挂载文件系统”中的表内。

Solaris 10 8/07：以下信息是此发行版中的新增或更新内容：

- 使用本发行版，非全局区域可以使用两种 IP 类型。新增了有关每种 IP 类型中可用功能的信息。请参见第 311 页中的 “共享 IP 非全局区域中的联网”和第 313 页中的 “Solaris 10 8/07：专用 IP 非全局区域中的联网”。
- 现在，Solaris IP 过滤器可在共享 IP 区域中使用。有关更多信息，请参见第 313 页中的 “共享 IP 区域中的 Solaris IP 过滤器”。
- 已对区域中权限设置的相关信息进行了修订。请参见表 26–1。
- 更新了第 326 页中的 “在安装了区域的 Solaris 系统上使用的命令”中的信息。

有关 Solaris 10 新增功能的完整列表以及 Solaris 发行版的说明，请参见《Solaris 10 新增功能》。

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权限 `PRIV_PROC_ZONE` 加以限制。此权限类似于 `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` 权限，也会存在上述限制。可以删除其他特权进程的 `PRIV_PROC_ZONE` 权限，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) 和 `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 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 26–5。

非全局区域节点名称

区域管理员可以设置 /etc/nodename 中由 uname -n 返回的节点名称。节点名称必须是唯一的。

文件系统和非全局区域

本节介绍有关安装了区域的 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 二进制命令权限运行，而是使用执行此二进制命令的用户权限运行。

  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权限运行。
- 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 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 306 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《系统管理指南：网络服务》中的“访问网络文件系统（参考）”。

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 221 页中的 “配置、检验并提交区域”和第 335 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。

文件系统	非全局区域中的挂载选项
AutoFS	不能使用 zonecfg 挂载，不能从全局区域中手动挂载到非全局区域。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，可以从全局区域中手动挂载到非全局区域，可以在区域中挂载。
HSFS	可以使用 zonecfg 挂载，可以从全局区域中手动挂载到非全局区域，可以在区域中挂载。
LOFS	可以使用 zonecfg 挂载，可以从全局区域中手动挂载到非全局区域，可以在区域中挂载。
MNTFS	不能使用 zonecfg 挂载，不能从全局区域中手动挂载到非全局区域。可以在区域中挂载。

文件系统	非全局区域中的挂载选项
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，可以从全局区域中手动挂载到非全局区域，可以在区域中挂载。
PROCFS	不能使用 zonecfg 挂载，不能从全局区域中手动挂载到非全局区域。可以在区域中挂载。
TMPFS	可以使用 zonecfg 挂载，可以从全局区域中手动挂载到非全局区域，可以在区域中挂载。
UDFS	可以使用 zonecfg 挂载，可以从全局区域中手动挂载到非全局区域，可以在区域中挂载。
UFS	可以使用 zonecfg 挂载，可以从全局区域中手动挂载到非全局区域，可以在区域中挂载。
XMEMFS	可以使用 zonecfg 挂载，可以从全局区域中手动挂载到非全局区域，可以在区域中挂载。 在未来的发行版中，将从 Solaris 系统中删除对此文件系统的支持。
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 221 页中的 “如何配置区域” 、第 335 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，而非全局区域的区域管理员无法卸载该文件系统。如果在非全局区域中挂载文件系统（例如指定区域 /etc/vfstab 文件中的挂载），则非全局区域的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《系统管理指南：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《系统管理指南：设备和文件系统》中的第 19 章 “挂载和取消挂载文件系统（任务）”。

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。

  如 `mount_nfs`(1M) 手册页中所述，NFS 服务器不应尝试挂载自己的文件系统。因此，区域不应对由全局区域导出的文件系统执行 NFS 挂载。区域不能是 NFS 服务器。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)。

  如果 `zlogin` 命令的打开文件或其地址空间的任意部分驻留在 NFS 上，此命令将失败。有关更多信息，请参见第 255 页中的 “`zlogin` 命令” 。

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 在挂载文件系统之前，会自动以非交互、仅检查的模式 (fsck -m) 在此设备上运行 fsck 命令。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/*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 240 页中的 “zoneadmd 守护进程” 和 fsck(1M)。

ZFS
可以使用带有 add dataset 资源的 zonecfg 命令将 ZFS 数据集添加到非全局区域。此数据集将在非全局区域中进行挂载并显示，并且在全局区域中不再可见。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
```

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。全局管理员负责设置和控制数据集的属性。

有关 ZFS 的更多信息，请参见《Solaris ZFS 管理指南》中的第 8 章 “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《系统管理指南：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《系统管理指南：网络服务》中的“设置 NFS 服务”。请使用任务图中的 “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《系统管理指南：网络服务》中的“设置 NFS 服务”。使用任务图中的 “如何使用命令行在客户机上选择不同的 NFS 版本” 过程。

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根路径，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
- flar
- flarcreate
- installf
- localeadm
- makeuuid
- metaroot
- patchadd

- patchrm
- pkgadd
- pkgadm
- pkgask
- pkgchk
- pkgrm
- 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
- sysidconfig
- update_drv
- useradd

共享 IP 非全局区域中的联网

在安装了区域的 Solaris 系统上，区域可通过网络相互通信。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。

共享 IP 区域分区

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自己版本的以下服务：

- 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（请参见 apache(1M) 手册页）

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。这些地址与可以使用 ifconfig 命令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 ifconfig 命令添加或删除逻辑接口。只有全局管理员才能修改接口配置和网络路由。

在非全局区域内，只有此区域的接口才能通过 ifconfig 进行查看。

有关更多信息，请参见 ifconfig(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

在同一计算机上的两个区域之间，仅当转发表中的目标和区域具有“匹配的路由”时，才允许传送包。

匹配信息按如下方式执行：

- 在由匹配路由指定的输出接口上选择包的源地址。
- 缺省情况下，允许地址位于同一子网上的两个区域之间进行通信。在这种情况下，匹配路由即为子网的接口路由。
- 如果给定区域具有缺省路由，并且网关位于此区域的一个子网上，则允许此区域与所有其他区域进行通信。在这种情况下，匹配路由即为缺省路由。
- 如果匹配路由具有 RTF_REJECT 标志，则包会触发 ICMP 不可访问的消息。如果匹配路由具有 RTF_BLACKHOLE 标志，则会放弃包。全局管理员可以使用下表所述的 route 命令选项来创建具有这些标志的路由。

修饰符	标志	说明
-reject	RTF_REJECT	匹配时会发出 ICMP 不可访问的消息。
-blackhole	RTF_BLACKHOLE	在更新过程中以静默方式放弃包。

有关更多信息，请参见 route(1M)。

共享 IP 区域中的 Solaris IP 过滤器

Solaris IP 过滤器可提供有状态包过滤和网络地址转换 (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。Solaris IP 过滤器还具有无状态包过滤功能，并且可以创建和管理地址池。有关其他信息，请参见《系统管理指南： IP 服务》中的第 25 章 “Solaris IP 过滤器（概述）”。

在非全局区域中，可以通过打开回送过滤来启用 Solaris IP 过滤器，如《系统管理指南： IP 服务》中的第 26 章 “Solaris IP 过滤器（任务）”所述。

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

在同一个 IP 链路上，IP 网络多路径 (IP network multipathing, IPMP) 为具有多个接口的系统提供了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 `ifconfig` 命令进行查看。

请参见第 342 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 221 页中的 “如何配置区域”中介绍。有关 IPMP 功能、组件和用法的信息，请参见《系统管理指南： IP 服务》中的第 30 章 “IPMP 介绍（概述）”。

Solaris 10 8/07 ： 专用 IP 非全局区域中的联网

专用 IP 区域具有自己的与 IP 相关的状态和调节变量。配置该区域时，系统会为该区域分配其自己的数据链路集合。

有关可在专用 IP 非全局区域中使用的功能的信息，请参见第 200 页中的 “Solaris 10 8/07 ： 专用 IP 非全局区域”。有关调节 IP `ndd` 变量的信息，请参见《Solaris Tunable Parameters Reference Manual》。

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）分配给一个专用 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

必须以独占方式将一个数据链路名称分配给单个区域。

可以使用 `dladm show-link` 命令显示分配给正在运行的区域的数据链路。

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 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

在同一个 IP 链路上，IP 网络多路径 (IP network multipathing, IPMP) 为具有多个接口的系统提供了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分配给某个区域。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，可使用 `zonecfg` 实用程序使其他设备变得可用。

`/dev` 和 `/devices` 名称空间

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。

现在作为 /（根）文件系统一部分的 /dev 文件分层结构由指向 /devices 中的物理路径的符号链接或逻辑路径组成。应用程序引用指向 /dev 中设备的逻辑路径。/dev 文件系统使用只读挂载回送挂载到区域。

/dev 文件分层结构由以下列出的组件所组成的系统进行管理：

- devfsadm（请参见 devfsadm(1M) 手册页）
- syseventd（请参见 syseventd(1M) 手册页）
- libdevinfo 设备信息库（请参见 libdevinfo(3LIB) 手册页）
- devinfo 驱动程序（请参见 devinfo(7D) 手册页）
- 重新配置协调管理器 (Reconfiguration Coordination Manager, RCM)（请参见《系统管理指南：设备和文件系统》中的“重新配置调整管理器 (Reconfiguration Coordination Manager, RCM) 脚本概述”）

注意 – 在建立 /dev 路径名之前，依赖于 /devices 路径名的子系统不能在非全局区域中运行。

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藏通道。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：

- `prtconf`（请参见 `prtconf`(1M) 手册页）
- `prtdiag`（请参见 `prtdiag`(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。

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`。
- 预期在运行时或者在安装、修补或升级后能够写入 `/usr` 的应用程序。这是因为在缺省情况下，`/usr` 对于非全局区域而言是只读的。有时，无需更改应用程序本身，便可缓解与此应用程序类型关联的问题。
- 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，另请参阅本指南第 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。

“一个区域，一个池”规则适用于非全局区域。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权限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 Solaris 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)。

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 命令分配区域份额，请参见第 232 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 74 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 345 页中的 “在安装了区域的 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 分配 FSS 份额。如果 FSS 是您系统中的缺省调度程序，并且尚未分配任何份额，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 Solaris 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《系统管理指南：IP 服务》中的第 36 章 “使用流记帐和统计信息收集功能（任务）”。

非全局区域中的权限

仅允许进程拥有部分权限。权限限制可防止某个区域执行可能会影响其他区域的操作。通过权限设置，可以限制区域内特权用户的功能。要显示区域内可用权限的列表，请使用 `ppriv` 实用程序。

下表列出了所有 Solaris 权限以及相对于区域每个权限的状态。可选权限并不属于缺省权限集，但可以通过 `limitpriv` 属性来指定。最终的权限集中必须包含必需权限。最终的权限集中不能包含禁止权限。

从 Solaris 10 11/06 发行版开始，`limitpriv` 属性就已经可用了。

表 26–1 区域中权限的状态

权限	状态	备注
`cpc_cpu`	可选	访问某些 `cpc`(3CPC) 计数器的权限
`dtrace_proc`	可选	`fasttrap` 和 `pid` 提供器；`plockstat`(1M)
`dtrace_user`	可选	`profile` 和 `syscall` 提供器
`gart_access`	可选	访问 `agpgart_io`(7I) 的 `ioctl`(2) 权限
`gart_map`	可选	访问 `agpgart_io`(7I) 的 `mmap`(2) 权限
`net_rawaccess`	在共享 IP 区域中为可选。 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	使用高精度计时器
`proc_priocntl`	可选	调度控制；`priocntl`(1)
`sys_ipc_config`	可选	增加 IPC 消息队列缓冲区大小
`sys_time`	可选	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_linkdir`	禁止	当前不支持

表 26–1 区域中权限的状态 （续）

权限	状态	备注
sys_net_config	禁止	当前不支持
sys_res_config	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_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
contract_event	缺省	供合约文件系统使用
contract_observer	缺省	合约调查，不考虑 UID
file_chown	缺省	文件所有权更改
file_chown_self	缺省	拥有文件的属主/组更改
file_dac_execute	缺省	执行访问权限，不考虑模式/ACL
file_dac_read	缺省	读取访问权限，不考虑模式/ACL
file_dac_search	缺省	搜索访问权限，不考虑模式/ACL
file_dac_write	缺省	写入访问权限，不考虑模式/ACL
file_link_any	缺省	链接访问权限，不考虑属主
file_owner	缺省	其他访问权限，不考虑属主
file_setid	缺省	更改 setid、setgid 和 setuid 文件的权限
ipc_dac_read	缺省	IPC 读取访问权限，不考虑模式
ipc_dac_owner	缺省	IPC 写入访问权限，不考虑模式
ipc_owner	缺省	IPC 其他访问权限，不考虑模式
net_icmpaccess	缺省	ICMP 包访问权限：ping(1M)
net_privaddr	缺省	绑定到特权端口
proc_audit	缺省	生成审计记录
proc_chroot	缺省	更改 root 目录

表 26–1 区域中权限的状态 （续）

权限	状态	备注
proc_info	缺省	检查进程
proc_lock_memory	缺省	锁定内存；shmctl(2) 和 mlock(3C) 如果系统管理员要将此权限分配给非全局区域，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	控制进程，不考虑属主
proc_session	缺省	控制进程，不考虑会话
proc_setid	缺省	任意设置用户/组 ID
proc_taskid	缺省	将任务 ID 分配给调用方
sys_acct	缺省	记帐管理
sys_admin	缺省	简单的系统管理任务
sys_audit	缺省	审计管理
sys_nfs	缺省	NFS 客户端支持
sys_resource	缺省	资源限制处理

下表列出了区域中所有 Solaris Trusted Extensions（高可靠扩展版）权限，以及每个权限的状态。缺省权限集不包含可选权限，但可以通过 limitpriv 属性指定它们。

注 – 仅当使用 Solaris Trusted Extensions（高可靠扩展版）配置了系统时，才会解释这些权限。

表 26–2 区域中 Solaris Trusted Extensions（高可靠扩展版）权限的状态

Solaris Trusted Extensions（高可靠扩展版）权限	状态	备注
sys_trans_label	可选	转换优先级低于敏感度标签的标签
win_colormap	可选	颜色映射限制覆盖
win_config	可选	配置或销毁 X 服务器永久保留的资源
win_dac_read	可选	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	写入或创建非客户机用户 ID 拥有的窗口资源

表 26–2 区域中 Solaris Trusted Extensions（高可靠扩展版）权限的状态 （续）

Solaris Trusted Extensions（高可靠扩展版）权限	状态	备注
win_devices	可选	在输入设备上执行操作。
win_dga	可选	使用直接图形访问 X 协议扩展；需要帧缓冲权限
win_downgrade_sl	可选	将窗口资源的敏感度标签更改为优先级低于现有标签的新标签
win_fontpath	可选	添加其他字体路径
win_mac_read	可选	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	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	请求移动数据，而无需确认者介入
win_upgrade_sl	可选	将窗口资源的敏感度标签更改为优先级不低于现有标签的新标签
net_bindmlp	缺省	允许绑定到多级端口 (MLP)
net_mac_aware	缺省	允许通过 NFS 向下读取

要在配置非全局区域过程中更改权限，请参见第 221 页中的 “配置、检验并提交区域”。

要检查权限集，请参见第 332 页中的 “使用 ppriv 实用程序”。有关权限的更多信息，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) 将在《系统管理指南：IP 服务》中的第 19 章 “IP 安全体系结构（概述）”中进行介绍。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，必须从全局区域中运行 ipseckey 和 ipsecconf 命令来管理用于非全局区域的 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。

Solaris 10 8/07：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 Solaris 审计

Solaris 审计将在《系统管理指南：安全性服务》中的第 27 章 “Solaris 审计（概述）”中进行介绍。有关与审计关联的区域注意事项，请参见以下各节：

- 《系统管理指南：安全性服务》中的第 28 章 “规划 Solaris 审计”
- 《系统管理指南：安全性服务》中的“审计和 Solaris Zones”

审计记录用于介绍事件，例如登录到系统或写入文件。记录由作为审计数据集合的标记组成。使用 zonename 标记，可以配置 Solaris 审计来标识每个区域的审计事件。使用 zonename 标记，可以生成以下信息：

- 审计记录，使用生成记录的区域名称进行标记
- 特定区域的审计日志，全局管理员使此日志可用于区域管理员。

在全局区域中配置审计

Solaris 审计跟踪在全局区域中配置。审计策略在全局区域中设置并应用于所有区域中的进程。审计记录可以使用发生事件的区域名称进行标记。要在审计记录中包括区域名称，必须在安装任何非全局区域之前编辑 /etc/security/audit_startup 文件。区域名称选择区分大小写。

要在全局区域中将审计配置为包括所有区域审计记录，请将以下行添加到 /etc/security/audit_startup 文件：

```
/usr/sbin/auditconfig -setpolicy +zonename
```

以全局区域中全局管理员的身份执行 auditconfig 实用程序：

```
global# auditconfig -setpolicy +zonename
```

有关其他信息，请参见 audit_startup(1M) 和 auditconfig(1M) 手册页以及《系统管理指南：安全性服务》中的 “配置审计文件（任务列表）”。

在非全局区域中配置用户审计特征

安装了非全局区域之后，便可将全局区域中的 audit_control 文件和 audit_user 文件复制到此区域的 /etc/security 目录。这些文件可能需要进行修改以反映此区域的审计需求。

例如，可以将每个区域配置为以不同的方式审计某些用户。要按用户应用不同的预选条件，必须编辑 `audit_control` 和 `audit_user` 文件。如有必要，还可能需要修改非全局区域中的 `audit_user` 文件以反映区域的用户基础。由于可以针对审计用户以不同的方式配置每个区域，因此，`audit_user` 文件可能为空。

有关其他信息，请参见 `audit_control`(4) 和 `audit_user`(4) 手册页。

为特定的非全局区域提供审计记录

通过如第 322 页中的 “在全局区域中配置审计”中所述包括 `zonename` 标记，可以按区域对 Solaris 审计记录进行分类。然后，可以使用 `auditreduce` 命令收集来自不同区域的记录，从而为特定区域创建日志。

有关更多信息，请参见 `audit_startup`(1M) 和 `auditreduce`(1M) 手册页。

区域中的核心转储文件

`coreadm` 命令用于指定因异常终止进程而生成的核心转储文件的名称和位置。通过指定 `%z` 变量，可以生成核心转储文件路径，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` 权限的 DTrace 程序可以在非全局区域中运行。要将这些权限添加到非全局区域中的可用权限的集合中，请使用 `zonecfg limitpriv` 属性。有关说明，请参见第 334 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 318 页中的 “非全局区域中的权限”。

关于备份安装了区域的 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

由于许多非全局区域通过使用回送文件系统只读挂载（通常为 /usr、/lib、/sbin 和 /platform）与全局区域共享文件，因此，必须使用全局区域备份方法来备份 lofs 目录。

注意 – 请不要在非全局区域中备份 lofs 文件系统。如果非全局管理员尝试从非全局区域中恢复 lofs 文件系统，则可能会导致严重问题。

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。
- 要使用 ufsdump 命令来执行数据备份。由于将物理磁盘设备导入非全局区域将更改区域的安全配置文件，因此，应仅在全局区域中使用 ufsdump。
- 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。

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数据，并且全局管理员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，使用文件系统功能对数据执行快照，然后直接从全局区域中备份快照。

  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，则全局管理员可以重新安装受影响的区域，然后恢复该区域的文件。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。
- 在执行备份到恢复区域这段时间内，未升级或修补全局区域。

否则，恢复操作可能会覆写某些应手动合并的文件。

例如，如果全局区域在备份之后和恢复非全局区域之前已进行修补，则可能需要手动合并文件。在这种情况下，恢复已备份的区域文件时必须谨慎，因为备份文件可能与新安装的区域（此区域在将修补程序应用到全局区域之后生成）不兼容。在这种情况下，必须逐个检查文件并将它们与新安装的区域中的副本进行比较。在多数情况下，会发现可以直接在区域中复制文件，但在某些情况下，必须将最初对文件所做的更改合并到区域中新安装或修补的副本中。

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 Solaris 系统上使用的命令

表 26–3 中列出的命令提供了区域功能的主要管理接口。

表 26–3 用于管理区域的命令

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
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 26–4 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被修改为可在安装了区域的 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 26–5 修改为可在安装了区域的 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。
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权限。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*。

表 26–5 修改为可在安装了区域的 Solaris 系统上使用的命令 （续）

命令参考	说明
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。
ifconfig(1M)	添加了 zone 选项以用于全局区域（缺省设置），添加了 -zone *zonename* 以用于非全局区域。
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。
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 127 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。
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。

表 26–5 修改为可在安装了区域的 Solaris 系统上使用的命令 （续）

命令参考	说明
auditon(2)	添加了 AUDIT_ZONENAME 以便生成每个审计记录的区域 ID 标记。
priocntl(2)	添加了 P_ZONEID *id* 参数。
processor_info(2)	如果调用方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)	如果调用方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)	如果调用方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)	如果调用方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)	如果调用方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)	如果调用方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)	针对调用方区域内的所有可用权限的集合添加了 "zone" 字符串。
pset_getloadavg(3C)	如果调用方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)	如果调用方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) 将返回绑定有区域的池的处理器集中的处理器数。
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。
pkginfo(4)	现在，提供了可选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。

表 26–5 修改为可在安装了区域的 Solaris 系统上使用的命令 （续）

命令参考	说明
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 27 章

Solaris Zones 管理（任务）

本章介绍一般管理任务并提供用法示例。

本章新增内容

本节列出新增产品功能以及本指南中的新增内容。

有关 Solaris 10 新增功能的完整列表以及 Solaris 发行版的说明，请参见《Solaris 10 新增功能》。

本章中针对 Solaris 10 1/06 的新增内容

添加了一种访问介质的新过程。请参见第 338 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”。

添加了一种在区域中备份和恢复文件的新过程。请参见第 346 页中的 “备份安装了区域的 Solaris 系统”和第 349 页中的 “恢复非全局区域”。

本章中针对 Solaris 10 6/06 的新增内容

添加了一些新过程。请参见第 338 页中的 “如何将文件系统从全局区域挂载到非全局区域”和第 340 页中的 “如何在非全局区域中的 /usr 下添加可写目录”。

本章中针对 Solaris 10 8/07 的新增内容

添加了一些新过程。请参见第 334 页中的 “如何使用 DTrace”、第 343 页中的 “Solaris 10 8/07：在专用 IP 非全局区域中管理数据链路”、第 334 页中的 “检查非全局区域中的 SMF 服务的状态”。

使用 ppriv 实用程序

使用 ppriv 实用程序可以显示区域的权限。

▼ 如何列出全局区域中的 Solaris 权限

可使用 ppriv 实用程序的 -l 选项列出该系统中可用的权限。

- **在提示符下，键入 ppriv -l zone 报告区域中的一组可用权限。**

```
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权限集

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权限。

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权限。**

```
my-zone# ppriv -l zone
```

将显示以下类似信息：

```
contract_event
contract_observer
file_chown

.
.
.
```

▼ 如何列出带有详细输出的非全局区域的权限集

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权限。

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权限，同时给出每个权限的说明。**

```
my-zone# ppriv -l -v 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_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 DTrace

执行以下步骤，以使用第 323 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` 权限。**

```
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权限或同时添加这两个权限。

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

要检查本地非全局区域中的 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 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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 22 章和 `svcs`(1)。

▼ 如何从区域内检查 SMF 服务的状态

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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 22 章和 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。

- 作为全局区域中的全局管理员，您可以将原始设备和块设备导入非全局区域。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，您也可以将文件系统从全局区域挂载到非全局区域。

▼ 如何使用 zonecfg 导入原始设备和块设备

此过程使用 lofi 文件驱动程序，此驱动程序可以将文件导出为块设备。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **转到 /usr/tmp 目录。**

```
global# cd /usr/tmp
```

3 **创建一个新的 UFS 文件系统。**

```
global# mkfile 10m fsfile
```

4 **将文件作为块设备进行连接。**

如果没有创建其他 `lofi` 设备，则会使用第一个可用插槽 `/dev/lofi/1` 。

```
global# lofiadm -a ‘pwd‘/fsfile
```

您也将获得所需的字符设备。

5 **将设备导入区域** `my-zone` 。

```
global# zonecfg -z my-zone
zonecfg:my-zone> add device
zonecfg:my-zone:device> set match=/dev/rlofi/1
zonecfg:my-zone:device> end
zonecfg:my-zone> add device
zonecfg:my-zone:device> set match=/dev/lofi/1
zonecfg:my-zone:device> end
```

6 **重新引导区域。**

```
global# zoneadm -z my-zone boot
```

7 **登录到区域并检验设备是否成功导入。**

```
my-zone# ls  -l /dev/*lofi/*
```

将显示以下类似信息：

```
brw-------   1 root     sys      147,  1 Jan  7 11:26 /dev/lofi/1
crw-------   1 root     sys      147,  1 Jan  7 11:26 /dev/rlofi/1
```

另请参见 有关更多信息，请参见 `lofiadm`(1M) 和 `lofi`(7D) 手册页。

▼ 如何手动挂载文件系统

要执行此过程，您必须是区域管理员并且拥有区域管理配置文件。此过程使用在 `newfs`(1M) 手册页中介绍的 `newfs` 命令。

1 **成为超级用户，或者在您的配置文件列表中具有区域管理权限配置文件。**

2 **在区域** `my-zone` **中，在磁盘上创建一个新的文件系统。**

```
my-zone# newfs /dev/lofi/1
```

3 **出现提示时回答是。**

```
newfs: construct a new file system /dev/rlofi/1: (y/n)? y
```

将显示以下类似信息：

```
/dev/rlofi/1:   20468 sectors in 34 cylinders of 1 tracks, 602 sectors
        10.0MB in 3 cyl groups (16 c/g, 4.70MB/g, 2240 i/g)
super-block backups (for fsck -F ufs -o b=#) at:
 32, 9664, 19296,
```

4 检查文件系统是否有错误。

```
my-zone# fsck -F ufs /dev/rlofi/1
```

将显示以下类似信息：

```
** /dev/rlofi/1
** Last Mounted on
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Counts
** Phase 5 - Check Cyl groups
2 files, 9 used, 9320 free (16 frags, 1163 blocks, 0.2% fragmentation)
```

5 挂载文件系统。

```
my-zone# mount -F ufs /dev/lofi/1 /mnt
```

6 检验挂载。

```
my-zone# grep /mnt /etc/mnttab
```

将显示以下类似信息：

```
/dev/lofi/1     /mnt    ufs
rw,suid,intr,largefiles,xattr,onerror=panic,zone=foo,dev=24c0001
1073503869
```

▼ 如何将文件系统放入 /etc/vfstab 以在引导区域时挂载

此过程用于在文件系统路径 `/mnt` 上挂载块设备 `/dev/lofi/1`。块设备包含一个 UFS 文件系统。将使用以下选项：

- `logging`，用作挂载选项。
- `yes`，告知系统在引导区域时自动挂载文件系统。
- `/dev/rlofi/1`，是字符（或原始）设备。如果需要，会在原始设备上运行 `fsck` 命令。

1 成为超级用户，或者在您的配置文件列表中具有区域管理权限配置文件。

2 **在区域 my-zone 中，向 /etc/vfstab 添加以下行：**

```
/dev/lofi/1 /dev/rlofi/1  /mnt   ufs  2  yes logging
```

▼ 如何将文件系统从全局区域挂载到非全局区域

假设区域中有 zonepath /export/home/my-zone。您需要将磁盘 /dev/lofi/1 从全局区域挂载到非全局区域中的 /mnt。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **要将磁盘挂载到非全局区域中的 /mnt，请在全局区域中键入以下内容：**

```
global# mount -F ufs /dev/lofi/1 /export/home/my-zone/root/mnt
```

另请参见 有关 lofi 的信息，请参见 lofiadm(1M) 和 lofi(7D) 手册页。

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 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **确定 Volume Management 文件系统是否正在全局区域中运行。**

```
global# svcs volfs
STATE          STIME    FMRI
online         Sep_29   svc:/system/filesystem/volfs:default
```

3 **（可选）如果 Volume Management 文件系统没有在全局区域中运行，则启动它。**

```
global# svcadm volfs 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 /cdrom
```

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      /                              native     shared
 1  my-zone  running      /export/home/my-zone           native     shared
```

10 登录到非全局区域。

```
global# zlogin 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）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
```

▼ 如何在非全局区域中的 /usr 下添加可写目录

在稀疏根区域中，/usr 从全局区域挂载为只读状态。可以使用此过程添加一个可写目录，如区域中 /usr 下的 /usr/local。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **在全局区域中创建目录 /usr/local。**

```
global# mkdir -p /usr/local
```

3 **在全局区域中指定一个目录作为该区域中 /usr/local 目录的后备存储。**

```
global# mkdir -p /storage/local/my-zone
```

4 **编辑区域 *my-zone* 的配置。**

```
global# zonecfg -z my-zone
```

5 **添加回送挂载的文件系统。**

```
zonecfg:my-zone> add fs
zonecfg:my-zone:fs> set dir=/usr/local
        zonecfg:my-zone:fs> set special=/storage/local/my-zone
        zonecfg:my-zone:fs> set type=lofs
        zonecfg:my-zone:fs> end
        zonecfg:my-zone> commit
        zonecfg:my-zone> exit
```

6 **引导区域。**

▼ 如何将全局区域中的起始目录导出到非全局区域

此过程用于将起始目录或其他文件系统从全局区域导出到同一系统上的非全局区域。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **添加回送挂载的文件系统。**

```
global# zonecfg -z my-zone
zonecfg:my-zone> add fs
zonecfg:my-zone:fs> set dir=/export/home
zonecfg:my-zone:fs> set special=/export/home
zonecfg:my-zone:fs> set type=lofs
zonecfg:my-zone:fs> set options=nodevices
zonecfg:my-zone:fs> end
zonecfg:my-zone> commit
zonecfg:my-zone> exit
```

3 **向区域的 `/etc/auto_home` 文件添加以下行：**

```
$HOST:/export/home/&
```

在安装了区域的 Solaris 系统上使用 IP 网络多路径

▼ Solaris 10 8/07：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)。

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。在配置 IPMP 之后，系统会自动监视 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。这些地址的接收方是故障接口所在的 IPMP 组中的功能接口。IPMP 的故障转移功能可保持连通性，防止任何现有连接发生中断。此外，IPMP 可通过 IPMP 组中的接口集自动分配网络通信，从而提高整体网络性能。此过程称作负荷分配。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **按照《 系统管理指南：IP 服务 》中的"配置 IPMP 组"中的说明配置 IPMP 组。**

▼ 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 204 页中的 "使用 zonecfg 命令"和第 221 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 ifconfig 命令来建立关联。请参见第 312 页中的 "共享 IP 网络接口"和 ifconfig(1M) 手册页。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的"使用 RBAC 和 Solaris 管理工具（任务图）"。

2 **在全局区域中，按照《 系统管理指南：IP 服务 》中的"配置 IPMP 组"中的说明配置 IPMP 组。**

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，则 *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。

Solaris 10 8/07 ： 在专用 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。

要管理数据链路，您必须是全局区域中的全局管理员。

1 成为超级用户或承担主管理员角色。

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 显示系统中数据链路的分配。

```
global# dladm show-linkprop
```

示例 27–1 将 dladm 与 show-linkprop 子命令一起使用

1. 在第一个屏幕中，没有引导分配了 bge0 的区域 49bge。

```
global# dladm show-linkprop
LINK         PROPERTY        VALUE          DEFAULT        POSSIBLE
bge0         zone            --             --             --
ath0         channel         6              --             --
ath0         powermode       ?              off            off,fast,max
ath0         radio           ?              on             on,off
ath0         speed           11             --
1,2,5.5,6,9,11,12,18,24,36,48,54
ath0         zone            --             --             --
```

2. 引导区域 49bge。

```
global# zoneadm -z 49bge boot
```

3. 再次运行 dladm show-linkprop 命令。请注意，bge0 链路现在分配给了 49bge。

```
global# dladm show-linkprop
LINK         PROPERTY        VALUE          DEFAULT        POSSIBLE
bge0         zone            49bge          --             --
ath0         channel         6              --             --
ath0         powermode       ?              off            off,fast,max
ath0         radio           ?              on             on,off
ath0         speed           11             --
1,2,5.5,6,9,11,12,18,24,36,48,54
```

```
ath0          zone            --              --              --
```

▼ 如何使用 dladm set-linkprop

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 区域分配数据链路。必须使用 zonecfg 命令进行持久性分配。

要管理数据链路，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **使用带有 -t 的 dladm set-linkprop 将 bge0 添加到正在运行的名为 excl 的区域。**

```
global# dladm set-linkprop -t -p zone=excl bge0
LINK          PROPERTY        VALUE           DEFAULT         POSSIBLE
bge0          zone            excl            --              --
```

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 链路值重置为未分配状态。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **使用带有 -t 的 dladm reset-linkprop 撤消 bge0 设备的区域分配。**

```
global# dladm set-linkprop -t -p zone=excl bge0
LINK          PROPERTY        VALUE           DEFAULT         POSSIBLE
bge0          zone            excl            --              --
```

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，则重新分配将失败，并显示一条错误消息。请参见第 353 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 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 221 页中的 “如何配置区域”和第 232 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。

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**
   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。
2. **使用 prctl 实用程序为全局区域分配两个份额：**
   ```
   # prctl -n zone.cpu-shares -v 2 -r -i zone global
   ```
3. **（可选）要检验为全局区域分配的份额数，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

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**
   有关如何创建角色并将角色指定给用户的信息，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。
2. **使用 prctl 命令为 cpu-shares 指定一个新值。**
   ```
   # prctl -i idtype -n zone.cpu-shares -r -v value
   ```
   idtype 为 *zonename* 或 *zoneid*。*value* 为新值。

在区域管理中使用权限配置文件

本节包含与在非全局区域中使用权限配置文件关联的任务。

▼ 如何分配区域管理配置文件

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。

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**

   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **创建一个包括区域管理权限配置文件的角色并将其指定给用户。**

   - 要使用 Solaris Management Console 创建并指定角色，请参见《系统管理指南：安全性服务》中的“配置 RBAC（任务列表）”。请参阅“如何使用 GUI 创建并指定角色”任务。
   - 要在命令行上创建并指定角色，请参见《系统管理指南：安全性服务》中的“管理 RBAC”。请参阅“如何从命令行创建角色”任务。

示例－结合使用配置文件 Shell 和区域命令

可以使用 `pfexec` 程序在配置文件中执行区域命令。该程序使用 `exec_attr` 数据库中的用户配置文件指定的属性执行命令。该程序由配置文件 shell `pfksh`、`pfcsh` 和 `pfsh` 进行调用。

使用 `pfexec` 程序可以登录到区域（例如 `my-zone`）。

```
machine$ pfexec zlogin my-zone
```

备份安装了区域的 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 ufsdump 命令执行备份

可以使用 `ufsdump` 命令执行完整备份或增量备份。此过程将区域 `/export/my-zone` 备份到 `/backup/my-zone.ufsdump`，其中 *my-zone* 将被替换为您的系统上的区域的名称。您可能需要一个单独的文件系统（例如，在 `/backup` 上挂载的文件系统），来保存备份。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **（可选）关闭区域以使其处于停顿状态，从而避免创建共享文件系统的备份。**

```
global# zlogin -S my-zone init 0
```

3 **查看区域状态。**

```
global# zoneadm list -cv
```

将显示以下类似信息：

```
ID  NAME     STATUS       PATH                           BRAND      IP
 0  global   running      /                              native     shared
 -  my-zone  installed    /export/home/my-zone           native     shared
```

4 **执行备份。**

```
global# ufsdump 0f /backup/my-zone.ufsdump /export/my-zone
```

将显示以下类似信息：

```
DUMP: Date of this level 0 dump: Wed Aug 10 16:13:52 2005
DUMP: Date of last level 0 dump: the epoch
DUMP: Dumping /dev/rdsk/c0t0d0s0 (bird:/) to /backup/my-zone.ufsdump.
DUMP: Mapping (Pass I) [regular files]
DUMP: Mapping (Pass II) [directories]
DUMP: Writing 63 Kilobyte records
DUMP: Estimated 363468 blocks (174.47MB).
DUMP: Dumping (Pass III) [directories]
DUMP: Dumping (Pass IV) [regular files]
DUMP: 369934 blocks (180.63MB) on 1 volume at 432 KB/sec
DUMP: DUMP IS DONE
```

5 **引导区域。**

```
global# zoneadm -z my-zone boot
```

▼ 如何使用 fssnap 创建 UFS 快照

此方法使用 `fssnap` 命令，此命令会创建用于备份操作的文件系统的临时映像。

此方法只能用于提供一个全新、一致的区域文件备份，并且可以在区域运行时执行。但是，最好在创建快照时暂停或检查正在更新文件的活动应用程序。在创建快照时更新文件的应用程序可能会使这些文件内部不一致、被截断或不可用。

在下面的示例过程中，请注意以下内容：

- 在 /export/home 下有一个名为 my-zone 的区域。
- /export/home 是单独的文件系统。

开始之前 目标备份为 /backup/my-zone.ufsdump。您必须在 / 下创建目录 backup。

1 成为超级用户或承担主管理员角色。

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 创建快照。

```
global# fssnap -o bs=/export /export/home
```

将显示以下类似信息：

```
dev/fssnap/0
```

3 挂载快照。

```
global# mount -o ro /dev/fssnap/0 /mnt
```

4 通过快照备份 my-zone。

```
global# ufsdump 0f /backup/my-zone.ufsdump /mnt/my-zone
```

将显示以下类似信息：

```
DUMP: Date of this level 0 dump: Thu Oct 06 15:13:07 2005
   DUMP: Date of last level 0 dump: the epoch
   DUMP: Dumping /dev/rfssnap/0 (pc2:/mnt) to /backup/my-zone.ufsdump.
   DUMP: Mapping (Pass I) [regular files]
   DUMP: Mapping (Pass II) [directories]
   DUMP: Writing 32 Kilobyte records
   DUMP: Estimated 176028 blocks (85.95MB).
   DUMP: Dumping (Pass III) [directories]
   DUMP: Dumping (Pass IV) [regular files]
   DUMP: 175614 blocks (85.75MB) on 1 volume at 2731 KB/sec
   DUMP: DUMP IS DONE
```

5 卸载快照。

```
global# umount /mnt
```

6 删除快照。

```
global# fssnap -d /dev/fssnap/0
```

请注意，快照也会在重新引导系统时从系统中删除。

▼ 如何使用 find 和 cpio 执行备份

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **转到根目录。**

```
global# cd /
```

3 **备份没有回送挂载到 /backup/my-zone.cpio 的 my-zone 文件。**

```
global# find export/my-zone -fstype lofs -prune -o -local
 | cpio -oc -O /backup/my-zone.cpio      type as one line
```

4 **检验结果。**

```
global# ls -l backup/my-zone.cpio
```

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 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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
```

恢复非全局区域

▼ 如何恢复单个非全局区域

如有必要，可以使用非全局区域配置的备份文件来恢复非全局区域。此过程使用名为 my-zone 的区域和名为 my-zone.config 的备份文件来说明恢复区域的过程。

1. **成为超级用户或承担主管理员角色。**

   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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
   ```

4. **为了防止系统显示在初始登录区域时出现的 `sysidtool` 问题，请删除文件 *zonepath*`/root/etc/.UNCONFIGURED`，例如：**

   ```
   global# rm /export/home/my-zone/root/etc/.UNCONFIGURED
   ```

5. **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 28 章

28 升级安装了非全局区域的 Solaris 10 系统

本章提供有关如何在运行 Solaris Zones 时将 Solaris™ 10 系统升级到更高发行版的信息，同时还提供指向相应 Solaris 安装文档的链接。

本章中针对 Solaris 10 8/07 的新增内容

安装了区域的系统现在支持 Solaris Live Upgrade。

在执行升级之前备份系统

在执行升级之前，您应该对 Solaris 系统上的全局区域和非全局区域进行备份。有关信息，请参见第 323 页中的 “关于备份安装了区域的 Solaris 系统”和第 346 页中的 “备份安装了区域的 Solaris 系统”。

将安装了区域的系统升级到 Solaris 10 8/07 及更高的更新发行版

您可以使用 Solaris Live Upgrade（标准 Solaris 交互式安装程序）或自定义 JumpStart 安装程序来升级安装了区域的 Solaris 系统。有关信息，请参见《Solaris 10 8/07 Installation Guide: Planning for Installation and Upgrade》中的“Upgrading With Non-Global Zones”。

将 Solaris Live Upgrade 用于 Solaris Zones 的原则

在安装了区域的系统上使用 Live Upgrade 时有许多注意事项。在 lucreate 和 lumount 操作期间应避免区域状态转换，这一点很关键。

- 在使用 `lucreate` 命令生成备用引导环境 (alternate boot environment, ABE) 时，如果给定区域未运行，则在 `lucreate` 完成之前，将无法引导该区域。
- 在使用 `lucreate` 命令生成 ABE 时，如果给定区域正在运行，则在 `lucreate` 完成之前，不应停止或重新引导该区域。
- 在通过 `lumount` 挂载 ABE 时，不能引导或重新引导区域，但在 `lumount` 操作之前已在运行的区域可以继续运行。

由于非全局区域管理员和全局区域管理员均可控制非全局区域，因此在 `lucreate` 或 `lumount` 操作期间最好停止所有区域。

当 Live Upgrade 操作正在执行时，非全局区域管理员的参与很关键。升级会影响管理员的工作，管理员将忙于处理因升级而带来的各种变化。区域管理员应确保任何本地软件包在整个操作序列期间都保持稳定，处理所有升级后任务（如配置文件调整），通常还应针对系统故障做出安排。

将安装了区域的系统升级到 Solaris 10 6/06 或 Solaris 10 11/06

升级系统前，请阅读第 356 页中的 “具有以 `lofs` 类型定义的 `fs` 资源的区域无法升级到 Solaris 10 11/06 发行版” 。

您可以使用标准 Solaris 交互式安装程序或自定义 JumpStart 安装程序来升级安装了区域的 Solaris 系统。此发行版不支持 Solaris Live Upgrade。有关信息，请参见《Solaris 10 11/06 Installation Guide: Solaris Live Upgrade and Upgrade Planning》和《Solaris 10 11/06 Installation Guide: Custom JumpStart and Advanced Installations》。

- 所有类型的安装和升级的整体规划信息和要求在《Solaris 10 11/06 Installation Guide: Planning for Installation and Upgrade》中的第 4 章 “System Requirements, Guidelines, and Upgrade (Planning)”中进行了介绍。请注意，用于安装的介质必须是 DVD 或从 DVD 创建的网络安装映像。
- Solaris 10 发行版界面在《Solaris 10 11/06 Installation Guide: Basic Installations》中进行了介绍。
- 自定义 JumpStart 安装的特殊注意事项和限制在《Solaris 10 11/06 Installation Guide: Custom JumpStart and Advanced Installations》中的第 8 章 “Custom JumpStart (Reference)”中进行了介绍。
- 有关通过网络执行安装或升级的信息，请参见《Solaris 10 11/06 Installation Guide: Network-Based Installations》。

◆ ◆ ◆ 第 29 章

29 解答各种 Solaris Zones 疑难问题

本章为 Solaris 10 6/06 发行版的新增内容。

有关 Solaris 10 新增功能的完整列表以及 Solaris 发行版的说明，请参见《Solaris 10 新增功能》。

Solaris 10 6/06、Solaris 10 11/06、Solaris 10 8/07 和 Solaris 10 5/08：不要将非全局区域的根文件系统放置在 ZFS 上

对于此发行版，非全局区域的 zonepath 不应该驻留在 ZFS 上。此操作可能会导致修补问题，并可能会阻止系统升级到更高的 Solaris 10 更新发行版。

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 344 页中的 “如何使用 dladm reset-linkprop”。正在运行的区域 excl 在使用该设备，这是在区域内通过执行 ifconfig bge0 plumb 分配的。

要重置值，请在区域内执行 ifconfig bge0 unplumb 过程，然后重新运行 dladm 命令。

区域管理员通过全局区域填充的文件系统进行挂载

首次引导非全局区域时文件系统分层结构内存在的文件表明文件系统数据由全局区域管理。安装非全局区域时，全局区域中的许多打包文件都复制到此区域内。这些文件必须直接驻留在 zonepath 下。如果文件驻留在区域管理员在磁盘设备上创建的文件系统下，或者驻留在添加到此区域的 ZFS 数据集下，则会出现打包和修补问题。

对于在本地区域文件系统中存储任何由全局区域管理的文件系统数据的问题，可以使用 ZFS 作为示例进行说明。如果已将 ZFS 数据集委托到非全局区域，则区域管理员不应使用此数据集来存储任何由全局区域管理的文件系统数据。否则，无法正确地修补或升级配置。

例如，已委托的 ZFS 数据集不应用作 /var 文件系统。Solaris 操作系统提供了将组件安装到 /var 的核心软件包。在升级或修补时，这些软件包必须访问 /var，如果 /var 挂载到委托的 ZFS 数据集，则无法对其进行访问。

支持在全局区域控制的某些分层结构下挂载文件系统。例如，如果全局区域中存在空的 /usr/local 目录，则区域管理员可以在此目录下挂载其他内容。

对于在修补或升级期间不需要访问的文件系统（例如非全局区域中的 /export），您可以使用委托的 ZFS 数据集。

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 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 会完全停止区域。

在区域配置中指定的权限集不正确

如果区域的权限集包含不允许的权限、缺少必需权限或包含未知权限名称，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
```

引导区域时显示 netmasks 警告

如果在按照第 248 页中的 “如何引导区域”中的说明引导区域时看到以下消息：

```
# zoneadm -z my-zone boot
zoneadm: zone 'my-zone': WARNING: hme0:1: no matching subnet
    found in netmasks(4) for 192.168.0.1; using default of
    255.255.255.0.
```

此消息只是警告，而命令已成功执行。此消息表明系统无法找到要用于在区域配置中指定的 IP 地址的 netmask。

要在后续重新引导时停止显示该警告，请确保在全局区域的 /etc/nsswitch.conf 文件中列出正确的 netmasks 数据库，并且至少有一个数据库包含要用于区域 my-zone 的子网和 netmasks。

例如，如果 /etc/inet/netmasks 文件和本地 NIS 数据库用于解析全局区域中的 netmasks，则 /etc/nsswitch.conf 的相应条目为：

```
netmasks: files nis
```

然后，可以将区域 my-zone 的子网和相应网络掩码信息添加到 /etc/inet/netmasks，供以后使用。

有关 netmasks 命令的更多信息，请参见 netmasks(4) 手册页。

使用 zoneadm attach 操作解决问题

▼ 修补程序和软件包不同步

目标系统上运行的下列必需的操作系统软件包和修补程序的版本必须与原始主机上安装的软件包和修补程序的版本相同。

- 在 inherit-pkg-dir 资源下提供文件的软件包
- SUNW_PKG_ALLZONES=true 的软件包

1 如果原始主机和新主机之间的软件包和修补程序不同，则可能会显示以下类似信息：

```
host2# zoneadm -z my-zone attach
    These packages installed on the source system are inconsistent with this system:
            SUNWgnome-libs (2.6.0,REV=101.0.3.2005.12.06.20.27) version mismatch
```

```
                (2.6.0,REV=101.0.3.2005.12.19.21.22)
        SUNWudaplr (11.11,REV=2005.12.13.01.06) version mismatch
                (11.11,REV=2006.01.03.00.45)
        SUNWradpu320 (11.10.0,REV=2005.01.21.16.34) is not installed
        SUNWaudf (11.11,REV=2005.12.13.01.06) version mismatch
                (11.11,REV=2006.01.03.00.45)
        NCRos86r (11.10.0,REV=2005.01.17.23.31) is not installed
These packages installed on this system were not installed on the source system:
        SUNWukspfw (11.11,REV=2006.01.03.00.45) was not installed
        SUNWsmcmd (1.0,REV=2005.12.14.01.53) was not installed
These patches installed on the source system are inconsistent with this system:
        120081 is not installed
        118844 is not installed
        118344 is not installed
These patches installed on this system were not installed on the source system:
        118669 was not installed
        118668 was not installed
        116299 was not installed
```

2 **要成功迁移区域，请使用正确的软件包和修补程序更新新主机，以便两个系统中的这些内容相同。有关更多信息，请参见第 24 章和第 25 章。**

▼ 操作系统发行版或计算机体系结构不匹配

要成功迁移区域，请在具有相同体系结构的系统上安装与原始主机上所运行的版本相同的 Solaris 发行版。

1 **检验原始系统上运行的 Solaris 发行版和系统的体系结构。**

```
host1# uname -a
```

2 **在具有相同体系结构的新主机上安装相同的发行版。**

请参阅 docs.sun.com 上的 Solaris 安装文档。

具有以 lofs 类型定义的 fs 资源的区域无法升级到 Solaris 10 11/06 发行版

注 – 此问题已在 Solaris 10 8/07 发行版中得到更正。

如果使用 lofs fs 资源配置的所有非全局区域正在挂载 miniroot 中存在的目录，则系统可以通过标准升级从早期 Solaris 10 发行版升级到 Solaris 10 11/06 发行版。例如，对以 lofs 方式挂载的 /opt 目录进行升级不会出现任何问题。

但是，如果有任何非全局区域是通过非标准 lofs 挂载进行配置的（例如以 lofs 方式挂载的 /usr/local 目录），则会显示以下错误消息：

```
The zones upgrade failed and the system needs to be restored
from backup.  More details can be found in the file
/var/sadm/install_data/upgrade_log on the upgrade root file
system.
```

尽管此错误消息表明，系统必须从备份恢复，但系统实际上是完好的，可以使用以下解决方法成功升级系统：

1. 使用安装的 OS 重新引导系统。
2. 重新配置区域，删除使用 lofs 类型定义的 fs 资源。
3. 在删除这些资源之后，将系统升级到 Solaris 10 11/06。
4. 升级后，可以再次重新配置区域，以恢复您删除的其他 fs 资源。

第 3 部分

标记区域

Solaris 10 8/07：从此发行版开始，可以使用标记区域。

BrandZ 为创建包含非本机操作环境的非全局标记区域提供了框架。在 Solaris 操作系统上使用标记区域来运行应用程序。第一个可用的标记是 `lx` 标记，即 Solaris Containers for Linux Applications。`lx` 标记为应用程序提供 Linux 环境，并可在 x86 和 x64 计算机上运行。

◆ ◆ ◆ 第 30 章

关于标记区域和 Linux 标记区域

从 Solaris 10 8/07 发行版开始，可以使用标记区域。后续更新版本中添加的功能会注明相应的发行版本。

Solaris™ 操作系统中的标记区域功能是 Solaris Zones 的简单扩展。本章讨论标记区域概念和实现 Linux 标记区域功能的 lx 标记。Linux 标记区域也称为 Linux 应用程序的 Solaris Containers。

注 – 虽然可以在启用标签的 Trusted Solaris™ 系统上配置和安装标记区域，但不能在此系统配置中引导标记区域。

关于在 Solaris 系统上使用区域

有关在 Solaris 系统上使用区域的一般信息，请参见第 16 章。

您应熟悉以下区域和资源管理概念：

- 全局区域和非全局区域，在第 187 页中的 “区域如何工作” 中介绍。
- 全局管理员和区域管理员，在第 189 页中的 “如何管理非全局区域” 和第 189 页中的 “如何创建非全局区域” 中介绍。
- 区域状态模型，在第 189 页中的 “非全局区域状态模型” 中介绍。
- 区域隔离特征，在第 191 页中的 “非全局区域特征” 中介绍。
- 权限，在第 318 页中的 “非全局区域中的权限” 中介绍。
- 联网，在第 311 页中的 “共享 IP 非全局区域中的联网” 中介绍。
- Solaris Container 概念，即将资源管理功能（如资源池）用于区域。区域以及资源管理功能的使用和交互，在第 192 页中的 “将资源管理功能用于非全局区域” 、第 201 页中的 “设置区域范围的资源控制” 、第 26 章以及本手册第 1 部分 “资源管理” 中介绍每个资源管理功能的各章中介绍。例如，资源池在第 12 章和第 13 章中介绍。

- 公平份额调度器 (fair share scheduler, FSS)，一个允许您基于份额来分配 CPU 时间的调度类，在第 8 章和第 9 章中介绍。
- 资源上限设置守护进程 (`rcapd`)，可以用来从全局区域中控制标记区域驻留集大小 (resident set size, RSS) 的使用。`zonecfg capped-memory` 资源的属性可为区域设置 `max-rss`。此值由在全局区域内运行的 `rcapd` 执行。有关更多信息，请参见第 10 章、第 11 章和 `rcapd`(1M) 手册页。

词汇表为区域和资源管理功能涉及的术语提供定义。

在系统上使用标记区域所需的任何其他信息均在指南的本部分中提供。

注 – 本指南的下列章节不适用于标记区域：

- 第 24 章
- 第 25 章

标记区域技术

标记区域 (BrandZ) 框架对本手册第 2 部分中介绍的 Solaris Zones 基础结构进行了扩展，从而包括标记的创建。术语**标记**可以指各种操作环境。通过 BrandZ 可以创建包含用于运行应用程序的非本机操作环境的非全局区域。标记类型用来确定安装和引导区域时执行的脚本。此外，区域的标记还可用来在应用程序启动时确定正确的应用程序类型。所有标记管理都通过扩展当前区域结构来执行。

标记能够提供简单或复杂环境。例如，简单环境可以用相应的 GNU 等效项来替换标准 Solaris 实用程序。复杂环境可以提供支持执行 Linux 应用程序的完整 Linux 用户空间。

每个区域都配有一个关联标记。缺省为 `native` 标记 Solaris。一个标记区域只支持一个非本地二进制命令标记，这意味着一个标记区域只提供一种操作环境。

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。

注 – 可在已配置状态下更改区域标记。一旦安装标记区域后，便不能更改或删除标记。

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 Solaris 行为。

标记还能为 `librtld_db` 提供插件库。通过插件库，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。

标记区域设备支持

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。设备支持由相应标记定义。标记可以选择不允许添加任何不受支持或无法识别的设备。

标记区域文件系统支持

标记区域所需的文件系统由相应标记定义。

标记区域中的权限

标记区域中可用的权限由相应标记定义。有关权限的更多信息，请参见第 318 页中的“非全局区域中的权限”和第 374 页中的 “lx 标记区域中的可配置权限”。

关于 lx 标记

`lx` 标记使用标记区域框架，使得 Linux 二进制应用程序无需修改，便可在具有 Solaris 操作系统内核的计算机上运行。

计算机的处理器类型必须为以下受支持的 i686 处理器类型之一：

- Intel
  - Pentium Pro
  - Pentium II
  - Pentium III
  - Celeron
  - Xeon
  - Pentium 4
  - Pentium M
  - Pentium D
  - Pentium Extreme Edition
  - Core
  - Core 2

  AMD
  - Opteron

- Athlon XP
- Athlon 64
- Athlon 64 X2
- Athlon FX
- Duron
- Sempron
- Turion 64
- Turion 64 X2

支持的 Linux 分发

lx 标记包括在非全局区域中安装 CentOS 3.*x* 或 Red Hat Enterprise Linux 3.*x* 分发版所必需的工具。每种分发的3.5 到 3.8 版本均受支持。该标记支持在以 32 位或 64 位模式运行 Solaris 系统的 x86 和 x64 计算机上执行 32 位 Linux 应用程序。

lx 标记模仿由 Linux 2.4.21 内核提供的系统调用接口，Red Hat 在 RHEL 3.*x* 分发中对这些接口进行了修改。此内核提供由 Red Hat 发行的 `glibc` 2.3.2版本所使用的系统调用接口。

此外，lx 标记还部分模仿 Linux `/dev` 和 `/proc` 接口。

注意 – 请注意，如果向 lx 标记区域中添加软件包，必须保留支持的配置。有关更多信息，请参见第 427 页中的 “关于维护支持的配置” 。

应用程序支持

Solaris 系统对于可在 lx 标记区域中运行的 Linux 应用程序数没有任何限制。必须提供足够的内存。另请参见第 369 页中的 “系统和空间要求” 。

无论基础内核如何，都只能运行 32 位 Linux 应用程序。

lx 区域只支持用户级别的 Linux 应用程序。在 lx 区域中不能使用 Linux 设备驱动程序、Linux 内核模块或 Linux 文件系统。

有关已在 lx 标记下成功运行的一些应用程序的列表，请参见 `http://opensolaris.org/os/community/brandz/applications`。有关安装应用程序的示例，请参见第 428 页中的 “如何在 lx 标记区域中安装应用程序” 。

不能在 lx 区域内运行 Solaris 应用程序。但通过 lx 区域可以使用 Solaris 系统来开发、测试和部署 Linux 应用程序。例如，可将一个 Linux 应用程序置于 lx 区域中并使用从全局区域运行的 Solaris工具对其进行分析。然后，可以在本地 Linux 系统上改进并部署优化后的应用程序。

调试工具

Solaris 调试工具（如 DTrace 和 mdb）可以应用于在区域内执行的 Linux 进程，但工具本身必须在全局区域中运行。生成的任何核心转储文件都以 Solaris 格式生成，并且只能用 Solaris 工具进行调试。

Dtrace 由 DTrace lxsyscall 动态跟踪提供器为 Linux 应用程序启用。该提供器的行为与 DTrace syscall 提供器的行为相似。lxsyscall 提供器可提供在线程进入 Linux 系统调用入口点或从此处返回时触发的探测。

有关调试选项的更多信息，请参见《Solaris 动态跟踪指南》以及 dtrace(1M) 和 mdb(1) 手册页。《Solaris 动态跟踪指南》介绍了 DTrace 功能可以使用的公共记录接口。有关 syscall 提供器的文档也可用于 lxsyscall 提供器。

注 – 因为 NFS 依赖于区域特定的名称服务，所以不能访问在当前区域外挂载的任何 NFS 文件系统。因此，不能从全局区域中调试基于 NFS 的 Linux 进程。

命令和其他接口

下表中列出的命令可提供区域功能的主要管理接口。

表 30–1 用于 lx 标记区域的命令和其他接口

命令参考	说明
zlogin(1)	登录到非全局区域
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
brands(5)	提供标记区域功能的说明
lx(5)	提供 Linux 标记区域的说明
zones(5)	提供区域功能的说明
lx_systrace(7D)	DTrace Linux 系统调用跟踪提供器
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。

注 – 表 26–5 介绍了可在全局区域中用来显示所有非全局区域（包括标记区域）的相关信息的命令。表 26–4 介绍了用于资源上限设置守护进程的命令。

在系统上设置 lx 标记区域（任务图）

下表简要介绍了首次在系统上设置 lx 区域所涉及的任务。

任务	说明	参考
确定要在区域中运行的每个 32 位 Linux 应用程序。	评估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： ■ 要运行的 Linux 应用程序数。 ■ Linux 标记区域的磁盘空间要求。 ■ 是否需要使用脚本。	请参见第 364 页中的 “应用程序支持”、第 369 页中的 “系统和空间要求”、第 217 页中的 “评估当前的系统设置”、第 390 页中的 “配置多个 lx 标记区域的脚本”。
确定是否将资源池与区域结合使用，以便创建容器。	如果要使用资源池，请在配置区域前先配置资源池。 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 386 页中的 “如何配置 lx 标记区域”和第 13 章。
执行预配置任务。	确定每个区域的区域名称和区域路径。如果需要网络连接，请获取 IP 地址。确定用于区域的调度类。确定在标准缺省权限集不充足的情况下，应为区域内的进程设置的权限集。	有关区域名称、区域路径、IP 地址和调度类的信息，请参见第 370 页中的 “lx 标记区域配置组件”。有关缺省权限和可在非全局区域中配置的权限的列表，请参见第 318 页中的 “非全局区域中的权限”。 有关资源池关联的信息，请参见第 187 页中的 “区域如何工作”和第 386 页中的 “如何配置 lx 标记区域”。
开发配置。	配置非全局区域。	请参见第 221 页中的 “配置、检验并提交区域”和 zonecfg(1M) 手册页。
以全局管理员身份检验和安装已配置的区域。	必须在引导区域之前检验和安装区域。在安装 Linux 标记区域前，必须获取 Linux 分发。	请参见第 33 章和第 34 章。

任务	说明	参考
以全局管理员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 34 章。
为生产使用准备新区域。	使用标准 Linux 系统管理工具和方法在区域内创建用户帐户、添加其他软件并自定义区域配置。	请参阅用于设置新安装的计算机和安装应用程序的文档。本指南包含适用于已安装区域的系统的特殊注意事项。

◆ ◆ ◆ 第 31 章

规划 lx 标记区域配置（概述）

本章介绍在基于 x64 或 x86 的系统上配置 lx 标记区域之前需要执行的操作，还介绍了 zonecfg 命令的用法。

系统和空间要求

以下几点主要的计算机注意事项与 lx 标记区域的使用有关。

- 计算机必须基于 x64 或 x86。
- 必须有足够的磁盘空间来保存每个 lx 区域内特有的文件。lx 区域的磁盘空间要求由安装的 RPM 或 Linux 软件包的大小和数量决定。
- lx 标记只支持完全根模型，因此安装的每个区域都将具有自己的所有文件的副本。

不限制每个区域可占用的磁盘空间量。全局管理员负责限制空间。全局管理员必须确保本地存储足以保存非全局区域的根文件系统。如果有足够的存储空间，即使小型单处理器系统也可支持同时运行多个区域。

限制标记区域的大小

可以使用以下几种选择来限制区域大小：

- 您可以将区域放置在挂载了 lofi 的分区上。此操作会将区域占用的空间量限制为 lofi 使用的文件所占用的空间量。有关更多信息，请参见 lofiadm(1M) 和 lofi(7D) 手册页。
- 您可以使用软分区将磁盘分片或将逻辑卷分为多个分区。您可以将这些分区用作区域根目录，从而限制每个区域的磁盘占用量。软分区限制为 8192 个分区。有关更多信息，请参见《Solaris Volume Manager Administration Guide》中的第 12 章 “Soft Partitions (Overview)”。
- 您可以将磁盘的标准分区用作区域根目录，从而限制每个区域的磁盘占用量。

标记区域网络地址

每个需要网络连接的区域都有一个或多个专用 IP 地址。支持 IPv4 地址。必须为区域指定一个 IPv4 地址。有关更多信息，请参见第 370 页中的 “标记区域网络地址” 。

lx 标记区域配置过程

zonecfg 命令可用于：

- 为区域设置标记
- 为 lx 区域创建配置
- 检验配置，以确定指定的资源和属性在基于 x86 或 x64 的虚拟系统上是否合法并且在内部一致。
- 执行特定于标记的检验。该检验可确保：
  - 区域不能具有任何继承的软件包目录、ZFS 数据集或添加的设备。
  - 如果将区域配置为使用音频，则指定的设备（如果有）必须为 none、default 或单数字。

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。

lx 标记区域配置组件

本节介绍以下组件：

- 可以使用 zonecfg 命令配置的区域资源和属性
- 配置中缺省包括的资源

lx 标记区域中的区域名称和区域路径

必须为区域选择名称和路径。

lx 标记区域中的区域自动引导

autoboot 属性设置决定在引导全局区域时是否自动引导区域。

lx 标记区域中的资源池关联

如果按第 13 章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。

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。

指定 dedicated-cpu 资源

dedicated-cpu 资源指定在某个非全局区域运行时，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。

dedicated-cpu 资源可为 ncpus 以及 importance（可选）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。
  - 按照第 146 页中的 “启用和禁用池功能”中所述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**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 133 页中的 “pool.importance 属性约束”。

注 – cpu-shares rctl 与 dedicated-cpu 资源不兼容。

Solaris10 5/08：指定 capped-cpu 资源

capped-cpu 资源对某一项目或区域可占用的 CPU 资源量设立绝对限制。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。

区域中的调度类

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。

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 126 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。

capped-memory 资源

capped-memory 资源可为 physical、swap 和 locked 内存设置限制。每个限制均为可选项，但至少要设置一个限制。

- 如果计划从全局区域使用 rcapd 为区域设置内存上限，请确定此资源的各个值。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。

有关更多信息，请参见第 10 章、第 11 章和第 386 页中的 “如何配置 lx 标记区域”。

lx 标记区域中的区域网络接口

lx 标记区域中只支持共享 IP 网络配置。

每个需要网络连接的区域都必须具有一个或多个专用 IP 地址。这些地址与逻辑网络接口关联。引导区域时，将在其中自动设置并放置通过 zonecfg 命令配置的网络接口。

在 lx 标记区域中挂载的文件系统

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
- 从区域本身中挂载的文件系统集合

例如，这可以包括以下文件系统：

- automount 触发的挂载
- 区域管理员明确执行的挂载

将对在应用程序环境中执行的挂载设定特定限制。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 305 页中的 “文件系统和非全局区域” 。

lx 标记区域中区域范围的资源控制

设置区域范围的资源控制的首选方法是使用属性名称（这种方法比较简单）而不是 rctl 资源。将为全局区域和非全局区域指定这些限制。

全局管理员还可使用 rctl 资源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。有关说明，请参见第 386 页中的 “如何配置 lx 标记区域” 。

当前可以使用以下资源控制：

表 31–1 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		在 Solaris 10 5/08 发行版中，对用于此区域的 CPU 资源量设置绝对限制。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	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	大小（字节）	capped-memory 的 locked 属性
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-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）	
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

lx 标记区域中的可配置权限

limitpriv 属性用于指定预定义的缺省集之外的权限掩码。引导区域时，标记配置中将包含一组缺省权限。这些权限被视为安全权限，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。可以使用 limitpriv 属性执行以下操作：

- 将权限添加至缺省权限集，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。
- 从缺省权限集中删除权限，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权限才能运行的话）。

注 – 目前，有些权限不能从区域的缺省权限集中删除，还有一些权限不能添加到缺省权限集中。

有关更多信息，请参见第 376 页中的 “在 lx 标记区域中定义的权限” 、第 318 页中的 “非全局区域中的权限”和 `privileges(5)`。

lx 标记区域中的 attr 资源

使用 `attr` 资源类型可以启用对全局区域中存在的音频设备的访问。有关说明，请参见第 387 页中的 “如何配置、检验和提交 lx 标记区域”中的步骤 12。

还可以使用 `attr` 资源类型为区域添加注释。

配置中缺省包括的资源

lx 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。`lx` 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。

请注意，可按第 387 页中的 “如何配置、检验和提交 lx 标记区域”中的步骤 12 所示，通过 `attr` 资源属性添加对全局区域中运行的音频设备的访问。

在 lx 标记区域中定义的文件系统

标记区域所需的文件系统在标记中定义。如第 387 页中的 “如何配置、检验和提交 lx 标记区域”中的步骤 9 所示，可以使用 `fs` 资源属性将其他 Solaris 文件系统添加至 `lx` 标记区域。

注 – 不支持添加本地 Linux 文件系统。可从 Linux 服务器中进行文件系统的 NFS 挂载。

在 lx 标记区域中定义的权限

仅允许进程拥有部分权限。权限限制可防止某个区域执行可能会影响其他区域的操作。通过权限设置，可以限制区域内特权用户的功能。

缺省权限、必需的缺省权限、可选权限以及禁止权限由每个标记定义。如第 387 页中的 “如何配置、检验和提交 lx 标记区域”的步骤 8 所述，还可以使用 `limitpriv` 属性添加或删除某些权限。表 26–1 列出了区域中的所有 Solaris 权限以及每个权限的状态。

有关权限的更多信息，请参见 `ppriv`(1) 手册页和《系统管理指南：安全性服务》。

使用 zonecfg 命令创建 lx 标记区域

可以使用 `zonecfg`(1M) 手册页中所述的 `zonecfg` 命令来配置区域。此命令还可用来为全局区域持久指定资源管理设置。

`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（包括说明如何使用本章中介绍的 `zonecfg` 各组成部分的过程），请参见第 386 页中的 “如何配置 lx 标记区域”。

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) 手册页中有关选项的内容。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:lx-zone:net> help
```

create　开始为指定的新标记区域配置内存配置。

- 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。要创建 Linux 标记区域，请使用：

```
zonecfg:lx-zone> create -t SUNWlx
```

- 与 -b 选项一起使用时，用于创建可为其设置标记的空配置。

```
zonecfg:lx-zone> create -b
zonecfg:lx-zone> set brand=lx
```

- 与 -F 选项一起使用时，用于覆写现有配置。

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“如何配置 lx 标记区域”以及 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。

`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 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。

标记区域配置数据

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。

资源和属性类型

资源和属性类型如下所述：

区域名称

区域名称用于标识配置实用程序的区域。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 64 个字符。
- 名称 global 和所有以 SUNW 开头的名称均保留，不能使用。

zonepath

zonepath 属性是区域根目录的路径。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。它必须由 root 拥有，并且模式为 700。

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 (/) 具有相同的拥有权和权限。区域目录必须由 root 拥有，并且模式为 755。这些目录是使用正确的权限自动创建的，并且不需要区域管理员进行检验。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。

路径	说明
/home/export/lx-zone	zonecfg zonepath
/home/export/lx-zone/root	区域的根目录
/home/export/lx-zone/root/dev	为区域创建的设备目录

有关此问题的进一步讨论，请参见第 310 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 “Solaris 10 11/06：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区域。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
```

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，否则会应用该引导参数。请参见第 398 页中的 “标记区域引导参数”。

pool

此属性用于将区域与系统上的特定资源池关联。多个区域可以共享一个池的资源。另请参见第 371 页中的 “指定 dedicated-cpu 资源”。

limitpriv

此属性用于指定缺省权限集之外的权限掩码。请参见第 318 页中的 “非全局区域中的权限”。

通过指定权限名称可添加权限，权限名称中可包含或不包含前导 priv_。在权限名称前添加破折号 (-) 或感叹号 (!) 可以排除权限。权限值以逗号分隔，并放在引号 (") 内。

如 priv_str_to_set(3C) 中所述，特殊权限集 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，因此不能使用特殊权限集 zone。由于常见用法是通过添加或删除某些权限来更改缺省权限集，因此特殊权限集 default 将映射为缺省权限集。当 default 出现在 limitpriv 属性开头时，它将扩展为缺省权限集。

如果输入以下指令，则会添加系统时钟设置功能，并删除原始 Internet 控制消息协议 (Internet Control Message Protocol, ICMP) 包发送功能：

```
global# zonecfg -z userzone
zonecfg:userzone> set limitpriv="default,sys_time,!net_icmpaccess"
```

如果区域的权限集包含不允许的权限、缺少必需权限或包含未知权限，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 372 页中的 “区域中的调度类”。

`dedicated-cpu` 此资源可让系统处理器子集专供某个区域在运行时使用。`dedicated-cpu` 资源可为 `ncpus` 以及 `importance`（可选）提供限制。有关更多信息，请参见第 371 页中的 “指定 dedicated-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。

`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 305 页中的 “文件系统和非全局区域”。

`net` 网络接口资源是虚拟接口名称。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应设置的网络接口。

`lx` 标记区域中只支持共享 IP 网络配置。

`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。

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 386 页中的 “如何配置 lx 标记区域”。

`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。

lx 标记区域中的资源类型属性

资源也有要配置的属性。以下属性与所示的资源类型关联。

dedicated-cpu　ncpus、importance

指定 CPU 个数以及池的相对重要性（可选）。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 lx-zone 使用的 CPU 的 CPU 限制为 3.5 个。

```
zonecfg:lx-zone> add capped-cpu
zonecfg:lx-zone:capped-cpu> set ncpus=3.5
zonecfg:lx-zone:capped-cpu> end
```

capped-memory　swap、locked

此资源可对为区域设置内存上限时使用的属性分组。以下示例指定了区域 my-zone 的内存限制。每个限制均为可选项，但至少要设置一个限制。

```
zonecfg:my-zone> add capped-memory
zonecfg:my-zone:capped-memory> set =50m
zonecfg:my-zone:capped-memory> set swap=100m
zonecfg:my-zone:capped-memory> set locked=30m
zonecfg:my-zone:capped-memory> end
```

fs　dir、special、raw、type、options

以下示例中的各行在非全局区域中添加对 CD 或 DVD 介质的只读访问权限。在非全局区域中，文件系统是使用选项 ro,nodevices（只读并且无设备）进行回送挂载的。

```
zonecfg:lx-zone> add fs
zonecfg:lx-zone:fs> set dir=/cdrom
zonecfg:lx-zone:fs> set special=/cdrom
zonecfg:lx-zone:fs> set type=lofs
zonecfg:lx-zone:fs> add options [ro,nodevices]
zonecfg:lx-zone:fs> end
```

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。这些手册页的名称格式为 mount_*filesystem*。

net　　address、physical

在以下示例中，将 IP 地址 192.168.0.1 添加到区域。bge0 卡用于物理接口。

```
zonecfg:lx-zone> add net
zonecfg:lx-zone:net> set physical=bge0
zonecfg:lx-zone:net> set address=192.168.0.1
zonecfg:lx-zone:net> end
```

注 – 要确定将使用的物理接口，请在系统上键入 ifconfig -a。每一个输出行（回送驱动程序行除外）都以系统上安装的卡的名称开头。说明中包含 LOOPBACK 的行不适用于卡。

rctl　　name、value

第 373 页中的 “lx 标记区域中区域范围的资源控制”中介绍了可用的区域范围的资源控制。

```
zonecfg:lx-zone> add rctl
zonecfg:lx-zone:rctl> set name=zone.cpu-shares
zonecfg:lx-zone:rctl> add value (priv=privileged,limit=10,action=none)
zonecfg:lx-zone:rctl> end

zonecfg:lx-zone> add rctl
zonecfg:lx-zone:rctl> set name=zone.max-lwps
zonecfg:lx-zone:rctl> add value (priv=privileged,limit=100,action=deny)
zonecfg:lx-zone:rctl> end
```

attr　　name、type、value

在以下示例中，添加了有关区域的注释。

```
zonecfg:lx-zone> add attr
zonecfg:lx-zone:attr> set name=comment
zonecfg:lx-zone:attr> set type=string
zonecfg:lx-zone:attr> set value="Production zone"
zonecfg:lx-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。

◆ ◆ ◆ 第 32 章

配置 lx 标记区域（任务）

本章介绍如何在基于 x64 或 x86 的系统上配置 lx 标记区域。此过程与配置 Solaris 区域的过程基本相同。配置标记区域时不需要使用其中一些属性。

规划和配置 lx 标记区域（任务图）

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。以下任务图概括了如何规划和配置 lx 区域。

任务	说明	参考
规划区域策略。	■ 确定要在区域中运行哪些应用程序。 ■ 评估磁盘空间的可用性，以便可以保存区域内的文件。 ■ 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。	请参见第 369 页中的 “系统和空间要求” 和第 127 页中的 “区域中使用的资源池” 。
确定区域的名称和路径。	基于命名约定决定区域的名称。推荐使用 Zetabyte 文件系统 (Zetabyte File System, ZFS) 上的路径。当源 zonepath 和目标 zonepath 都驻留在 ZFS 上并且位于同一个池中时，zoneadm clone 命令会自动使用 ZFS 来克隆区域。	请参见第 379 页中的 “资源和属性类型” 和《Solaris ZFS 管理指南》 。

任务	说明	参考
获取或配置区域的 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 218 页中的 “确定区域主机名并获取网络地址” 和《系统管理指南：IP 服务》。
确定是否要在区域内挂载文件系统。	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 312 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。	检查权限集：缺省权限集、可以添加和删除的权限，以及目前不能使用的权限。	请参见第 379 页中的 “资源和属性类型” 和第 318 页中的 “非全局区域中的权限”。
配置区域。	使用 zonecfg 可以创建区域的配置。	请参见第 387 页中的 “如何配置、检验和提交 lx 标记区域”。
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 387 页中的 “如何配置、检验和提交 lx 标记区域”。

如何配置 lx 标记区域

使用 zonecfg(1M) 手册页中所述的 zonecfg 命令可执行以下操作。

- 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

提示 – 如果您知道将要使用 CD 或 DVD 在 lx 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。

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 233 页中的 “如何恢复区域配置”。

第 390 页中的 “配置多个 lx 标记区域的脚本” 中提供了用于在系统中配置多个区域的脚本。

要显示非全局区域配置，请参见第 392 页中的 “如何显示标记区域的配置”。

提示 – 配置完标记区域之后，最好复制该区域的配置。将来您可以使用此备份来恢复区域。以超级用户或主管理员的身份，将区域 lx-zone 的配置列显到文件。以下示例使用名为 lx-zone.config 的文件。

```
global# zonecfg -z lx-zone export > lx-zone.config
```

有关更多信息，请参见第 349 页中的 “如何恢复单个非全局区域” 。

▼ 如何配置、检验和提交 lx 标记区域

请注意，不能在启用标签的 Trusted Solaris 系统中使用 lx 标记区域。zoneadm 命令将不检验配置。

要执行此过程，您必须是全局区域中的全局管理员。

1 成为超级用户或承担主管理员角色。

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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 lx-zone。

```
global# zonecfg -z lx-zone
```

如果是第一次配置该区域，则可以看到以下系统消息：

```
lx-zone: No such zone configured
Use 'create' to begin configuring a new zone.
```

3 使用 SUNWlx 模板创建新 lx 区域配置。

```
zonecfg:lx-zone> create -t SUNWlx
```

或者，也可以创建空区域并显式设置标记：

```
zonecfg:lx-zone> create -b
zonecfg:lx-zone> set brand=lx
```

4 设置区域路径，在此过程中为 /export/home/lx-zone。

```
zonecfg:lx-zone> set zonepath=/export/home/lx-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:lx-zone> set autoboot=true
```

6 **为区域设置持久引导参数。**

```
zonecfg:lx-zone> set bootargs="-i=altinit"
```

7 **如果在系统上启用资源池，则将该池与区域进行关联。**

此示例使用名为 pool_default 的缺省池。

```
zonecfg:lx-zone> set pool=pool_default
```

因为资源池允许进行一次可选的调度类分配，所以可以使用池功能来设置一个缺省的调度程序，而不是非全局区域的系统缺省值。有关说明，请参见第 155 页中的 “如何将池与调度类关联”和第 168 页中的 “创建配置”。

8 **修改缺省权限集。**

```
zonecfg:lx-zone> set limitpriv="default,proc_priocntl"
```

proc_priocntl 权限用于在实时类中运行进程。

9 **将 CPU 份额值设置为 5。**

```
zonecfg:lx-zone> set cpu-shares=5
```

10 **添加内存上限。**

```
zonecfg:lx-zone> add capped-memory
```

a. **设置内存上限。**

```
zonecfg:lx-zone:capped-memory> set =50m
```

b. **设置交换内存上限。**

```
zonecfg:lx-zone:capped-memory> set swap=100m
```

c. **设置锁定内存上限。**

```
zonecfg:lx-zone:capped-memory> set locked=30m
```

d. **结束指定。**

```
zonecfg:lx-zone:capped-memory> end
```

11 **添加文件系统。**

```
zonecfg:lx-zone> add fs
```

a. **设置文件系统的挂载点，在此过程中为 /export/linux/local。**

```
zonecfg:lx-zone:fs> set dir=/export/linux/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:lx-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:lx-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:lx-zone:fs> end
```

可多次执行此步骤来添加多个文件系统。

12 **添加网络虚拟接口。**

```
zonecfg:lx-zone> add net
```

a. **以区域的 IP 地址/网络掩码的格式设置 IP 地址。在此过程中使用了 *10.6.10.233/24*。**

```
zonecfg:lx-zone:net> set address=10.6.10.233/24
```

b. **设置网络接口的物理设备类型，在此过程中为 bge 设备。**

```
zonecfg:lx-zone:net> set physical=bge0
```

c. **结束指定。**

```
zonecfg:lx-zone:net> end
```

可多次执行此步骤来添加多个网络接口。

13 **通过使用 attr 资源类型在此区域中启用全局区域中存在的音频设备。**

```
zonecfg:lx-zone> add attr
```

a. **将名称设置为 audio。**

```
zonecfg:lx-zone:attr> set name=audio
```

b. **将类型设置为 boolean。**

```
zonecfg:lx-zone:attr> set type=boolean
```

c. **将值设置为 true。**

```
zonecfg:lx-zone:attr> set value=true
```

d. **结束 attr 资源类型指定。**

```
zonecfg:lx-zone:attr> end
```

14 **检验区域的配置。**

```
zonecfg:lx-zone> verify
```

15 **提交区域的配置。**

```
zonecfg:lx-zone> commit
```

16 **退出 zonecfg 命令。**

```
zonecfg:lx-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 lx-zone "create -t SUNWlx; set zonepath=/export/home/lx-zone"
```

下一步执行的操作

请参见第 402 页中的 “安装和引导 lx 标记区域”来安装已提交的区域配置。

配置多个 lx 标记区域的脚本

可以使用此脚本在系统中配置和引导多个区域。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

要执行此脚本，您必须是全局区域中的全局管理员。全局管理员在全局区域中拥有超级用户权限或承担主管理员角色。

```
#!/bin/ksh
#
# Copyright 2006 Sun Microsystems, Inc. All rights reserved.
# Use is subject to license terms.
#
#ident    "%Z%%M%   %I%    %E% SMI"
if [[ -z "$1" || -z "$2" || -z "$3" || -z "$4" ]]; then
    echo "usage: $0 <#-of-zones> <zonename-prefix> <basedir> <template zone>"
    exit 2
fi
if [[ ! -d $3 ]]; then
    echo "$3 is not a directory"
    exit 1
fi
state=‘zoneadm -z $4 list -p 2>/dev/null | cut -f 3 -d ":"‘
```

```
if [[ -z "$state" || $state != "installed" ]]; then
    echo "$4 must be an installed, halted zone"
    exit 1
fi

template_zone=$4

nprocs=‘psrinfo | wc -l‘
nzones=$1
prefix=$2
dir=$3

ip_addrs_per_if=‘ndd /dev/ip ip_addrs_per_if‘
if [ $ip_addrs_per_if -lt $nzones ]; then
        echo "ndd parameter ip_addrs_per_if is too low ($ip_addrs_per_if)"
        echo "set it higher with ’ndd -set /dev/ip ip_addrs_per_if <num>"
        exit 1
fi

i=1
while [ $i -le $nzones ]; do
    zoneadm -z $prefix$i clone $template_zone > /dev/null 2>&1
    if [ $? != 0 ]; then
        echo configuring $prefix$i
        F=$dir/$prefix$i.config
        rm -f $F
        echo "create -t SUNWlx" > $F
        echo "set zonepath=$dir/$prefix$i" >> $F
        zonecfg -z $prefix$i -f $dir/$prefix$i.config 2>&1 | \
            sed ’s/^/    /g’
    else
        echo "skipping $prefix$i, already configured"
    fi
    i=‘expr $i + 1‘
done

i=1
while [ $i -le $nzones ]; do
    j=1
    while [ $j -le $nprocs ]; do
        if [ $i -le $nzones ]; then
            if [ ‘zoneadm -z $prefix$i list -p | \
                cut -d’:’ -f 3‘ != "configured" ]; then
                echo "skipping $prefix$i, already installed"
            else
                echo installing $prefix$i
                mkdir -pm 0700 $dir/$prefix$i
                chmod 700 $dir/$prefix$i
```

```
                zoneadm -z $prefix$i install -s -d /path/to/ISOs > /dev/null 2>&1 &
                sleep 1    # spread things out just a tad
            fi
        fi
        i=‘expr $i + 1‘
        j=‘expr $j + 1‘
    done
    wait
done

i=1
para=‘expr $nprocs \* 2‘
while [ $i -le $nzones ]; do
    date
    j=1
    while [ $j -le $para ]; do
        if [ $i -le $nzones ]; then
            echo booting $prefix$i
            zoneadm -z $prefix$i boot &
        fi
        j=‘expr $j + 1‘
        i=‘expr $i + 1‘
    done
    wait
done
```

▼ 如何显示标记区域的配置

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **显示区域配置。**

```
global# zonecfg -z zonename info
```

修改、恢复或删除区域配置

以下各节包含用于修改、恢复或删除区域配置的过程。

- 第 229 页中的 “如何修改区域配置中的资源类型”
- 第 230 页中的 “Solaris 10 8/07：如何清除区域配置中的属性类型”
- 第 231 页中的 “Solaris 10 8/07：如何重命名区域”

- 第 233 页中的 “如何恢复区域配置”
- 第 235 页中的 “如何删除区域配置”

◆ ◆ ◆ 第 33 章

33 关于安装、引导、停止、克隆和卸载 lx 标记区域（概述）

本章讨论以下主题：

- 在系统上安装 lx 区域
- 停止、重新引导和卸载区域
- 在系统上克隆区域

标记区域安装和管理概念

zoneadm(1M) 手册页中介绍的 zoneadm 命令是用于安装和管理非全局区域的主要工具。使用 zoneadm 命令的操作必须从全局区域中运行。可以使用 zoneadm 命令执行以下任务：

- 检验区域
- 安装区域
- 引导区域
- 显示有关正在运行的区域的信息
- 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过程的信息，请参见第 34 章和 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。有关区域配置过程的信息，请参见第 32 章和 zonecfg(1M) 手册页。区域状态在第 189 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 Solaris 审计记录，请在安装非全局区域之前先阅读第 322 页中的 “在区域中使用 Solaris 审计”。

注 – 安装区域后，所有软件配置和管理工作都必须由区域管理员在区域内使用 Linux 工具来执行。

lx 标记区域安装方法

可以使用 tarball、CD-ROM 或 DVD 光盘或者 ISO 映像来安装 lx 标记区域。如果从光盘或 ISO 映像安装，则可指定 Sun 软件包簇类别。类别是累积的。如果不指定簇，则缺省值为 desktop。

表 33–1 软件包簇类别

Sun 类别	内容
core	构建区域需要的软件包的最小集合。
server	core 和面向服务器的软件包，如 httpd、mailman、imapd 和 spam-assassin。
desktop	server 和面向用户的软件包，如 evolution、gimp、mozilla 和 openoffice。
developer	desktop 和开发者软件包，如 bison、emacs、gcc、vim-X11 和许多库开发软件包。
all	安装介质上已知不干扰区域运行的所有内容。某些软件包在 Linux 区域中可能不起作用。

要安装已配置的 lx 标记区域，请参见第 402 页中的 “如何安装 lx 标记区域” 。

lx 标记区域构建

本节仅适用于初始区域构建，不适用于现有区域的克隆。

在配置了非全局区域之后，应检验是否可以在系统配置中安全地安装该区域。然后您可以安装该区域。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。如第 402 页中的 “如何安装 lx 标记区域”中所述，将从 CD、ISO 映像或 tarball 中填充 Linux 区域。

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。

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。

在就绪状态下，会启动 zsched 和 zoneadmd 进程来管理虚拟平台。

zoneadmd 区域管理守护进程

区域管理守护进程 zoneadmd 是管理区域虚拟平台的主要进程。有关更多信息，请参见第 240 页中的 “zoneadmd 守护进程” 。

zsched 区域调度进程

第 240 页中的 “zsched 区域调度程序”中介绍了用于管理应用程序环境的进程 zsched。

标记区域应用程序环境

zoneadm 命令用于创建区域应用程序环境。

所有其他配置均由区域管理员在区域中使用 Linux 工具完成。

口令

请注意，如果区域是从 Sun tarball 安装的，则 root 用户（超级用户）口令将为 root。从 ISO 映像或 CD 安装区域时，root（超级用户）口令将复位（为空）。

关于停止、重新引导、卸载和克隆 lx 标记区域

本节概述了停止、重新引导、卸载和克隆区域的过程。

停止标记区域

zoneadm halt 命令用于删除区域的应用程序环境和虚拟平台。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 266 页中的 “如何使用 zlogin 关闭区域” 。

停止操作失败时，请参见第 354 页中的 “区域无法停止” 。

重新引导标记区域

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。

标记区域引导参数

区域支持对 zoneadm boot 和 reboot 命令使用以下引导参数：

- -i *altinit*
- -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。

-s　将区域引导至 init 级别 s。

有关用法示例，请参见第 407 页中的 “如何引导 lx 标记区域”和第 408 页中的 “如何在单用户模式下引导 lx 标记区域”。

有关 init 命令的信息，请参见 init(1M)。

标记区域 autoboot

如果您在区域配置中将 autoboot 资源属性设置为 true，则引导全局区域时便会自动引导此区域。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。

卸载标记区域

zoneadm uninstall 命令可删除区域根文件系统中的所有文件。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 lx 标记区域

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。有关克隆进程的更多信息，请参见第 411 页中的 “在同一系统上克隆 lx 标记区域”。

引导和重新引导 lx 标记区域

有关引导和重新引导区域的过程，请参见第 407 页中的 “如何引导 lx 标记区域”和第 410 页中的 “如何重新引导 lx 标记区域”。

◆ ◆ ◆ 第 34 章

安装、引导、停止、卸载和克隆 lx 标记区域（任务）

本章介绍如何安装和引导 lx 标记区域，还介绍了以下其他任务：

- 使用克隆在同一系统上安装区域
- 停止、重新引导和卸载区域
- 从系统中删除区域

lx 标记区域安装（任务图）

任务	说明	参考
获取 Linux 归档文件。	在安装 lx 标记区域之前，必须先获取 Linux 归档文件。	第 402 页中的 “如何获取 Linux 归档文件”
安装已配置的 lx 标记区域。	安装处于已配置状态的区域。	第 402 页中的 “如何安装 lx 标记区域”
（可选）安装可用软件包的子集。	在从 CD 或 ISO 映像中进行安装时，可以安装安装介质中的软件包的子集。	第 404 页中的 “如何安装软件包的子集”
（可选）在区域中启用联网功能。	联网功能在缺省情况下处于禁用状态，如果需要此功能，则必须先启用此功能。	第 405 页中的 “如何在 lx 标记区域中启用联网”
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 405 页中的 “如何获取已安装标记区域的 UUID”
（可选）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 407 页中的 “（可选）将已安装的 lx 标记区域置于就绪状态”

任务	说明	参考
引导 lx 标记区域。	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 407 页中的 “如何引导 lx 标记区域”
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 249 页中的 “如何在单用户模式下引导区域”

安装和引导 lx 标记区域

使用 zoneadm(1M) 手册页中所述的 zoneadm 命令，可以执行非全局区域的安装任务。

▼ 如何获取 Linux 归档文件

在安装 lx 标记区域之前，必须先获取 Linux 归档文件。归档文件通过以下形式分发：

- 压缩的 tar 归档文件 (*tarball*)
- 一组 CD-ROM 或 DVD 光盘
- 一组 ISO 映像

- **使用以下方法之一获取 Linux 分发：**
  - 要下载 tarball，请转到 http://opensolaris.org/os/community/brandz/downloads。按照下载站点中的说明操作。
  - 要获取一组 CD-ROM 或 DVD 光盘，请转到位于 http://www.centos.org 的 CentOS 站点或位于 http://www.redhat.com 的 Red Hat 站点。
  - 要获取 ISO 映像，请转到位于 http://www.centos.org 的 CentOS 站点或位于 http://www.redhat.com 的 Red Hat 站点。

▼ 如何安装 lx 标记区域

此过程用于安装已配置的 lx 标记区域。安装区域后，所有软件配置和管理工作都必须由区域管理员在区域内使用 Linux 工具来执行。

有关使用不同分发路径的区域安装命令行示例，请参见示例 34–1、示例 34–2 和示例 34–3。如果从光盘或 ISO 映像安装，则必须指定 Sun 软件包簇类别。有关软件包簇类别的信息，请参见第 396 页中的 “lx 标记区域安装方法” 。

请注意，可以在安装一个区域之前检验该区域。如果您跳过此过程，则会在安装区域时自动执行检验。第 244 页中的 “（可选）如何在安装已配置的区域之前检验该区域” 中介绍了此过程。

要执行此过程，您必须是全局区域中的全局管理员。

注 – 在步骤 2 中，**如果** zonepath 在 ZFS 上，则 zoneadm install 命令将在安装区域时自动为 zonepath 创建一个 ZFS 文件系统（数据集）。可以通过添加 -x nodataset 参数阻止此操作。

---

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **（可选）如果要从 DVD 或 CD 进行安装，请在系统中启用 volfs 并检验它是否正在运行。**

```
global# svcadm enable svc:/system/filesystem/volfs:default

global# svcs | grep volfs
```

将显示以下类似信息：

```
online  17:30 svc:/system/filesystem/volfs:default
```

3 **通过将 zoneadm 命令与 install 选项和归档文件路径结合使用来安装已配置区域 lx-zone。**

- **安装区域，如果 zonepath 在 ZFS 上，则自动创建一个 ZFS 文件系统。**

  ```
  global# zoneadm -z lx-zone install -d archive_path
  ```

  系统将显示：

  ```
  A ZFS file system has been created for this zone.
  ```

- **安装 zonepath 在 ZFS 上的区域，但不自动创建 ZFS 文件系统。**

  ```
  global# zoneadm -z lx-zone install -x nodataset -d archive_path
  ```

当区域的根文件系统所需的文件和目录以及软件包文件安装在区域的根路径下时，您将看到各种消息。

注 – 如果不指定 archive_path，则缺省值为 CD。

---

4 **（可选）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 -z lx-zone list -iv
```

- 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 lx-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。

5 当安装完成时，使用带有 -i 和 -v 选项的 list 子命令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      /                              native     shared
 -  lx-zone  installed    /export/home/lx-zone           lx         shared
```

示例 34–1 使用 CentOS 压缩 tar 归档文件的安装命令

```
global# zoneadm -z lx-zone install -d /export/centos_fs_image.tar.bz2
```

示例 34–2 使用 CentOS CD 的安装命令

对于 CD 或 DVD 安装，必须在系统中启用 volfs。必须指定一个软件簇软件包。例如，使用 development 安装完整环境，或键入特定簇的名称。如果未指定簇软件包，缺省情况下将安装 desktop。CD 设备为 /cdrom/cdrom0。

```
global# zoneadm -z lx-zone install -d /cdrom/cdrom0 development
```

示例 34–3 使用 CentOS ISO 映像的安装命令

必须指定一个软件簇软件包。使用 development 安装完整环境，或指定特定簇。如果不指定簇软件包，缺省情况下将安装 desktop。CentOS ISO 映像驻留在目录 /export/centos_3.7 中。

```
global# zoneadm -z lx-zone install -d /export/centos_3.7 development
```

另请参见 有关数据集的更多信息，请参见《Solaris ZFS 管理指南》。

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。

▼ 如何安装软件包的子集

在从 CD 或 ISO 映像中进行安装时，可以安装安装介质中的软件包的子集。可用子集有 core、server、desktop、developer 和 all。

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**
   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **只安装服务器软件包：**

```
global# zoneadm -z lx-zone install -d archive_path server
```

▼ 如何在 lx 标记区域中启用联网

安装 lx 标记区域时，会禁用联网。使用类似如下的过程可以启用联网。

只有区域管理员可以执行此过程。

1. **编辑区域中的 /etc/sysconfig/network 文件。**

```
NETWORKING=yes
HOSTNAME=your.hostname
```

2. **要设置 NIS 域，请添加类似于以下内容的行：**

```
NISDOMAIN=domain.Sun.COM
```

更多信息 **配置联网和命名服务**

有关配置联网或命名服务的更多信息，请查看有关 Linux 分发的相关文档。

▼ 如何获取已安装标记区域的 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 -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 -p
```

将显示以下类似信息：

```
0:global:running:/::native
    1:centos38:running:/zones/centos38:27fabdc8-d8ce-e8aa-9921-ad1ea23ab063:lx
```

示例 34–4 如何在命令中使用 UUID

```
global# zoneadm -z lx-zone -u 61901255-35cf-40d6-d501-f37dc84eb504 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 lx 标记区域标记为未完成

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。

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**

   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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       /                        native     shared
   -   testzone   incomplete    /export/home/testzone    lx         shared
   ```

更多信息 **将区域标记为未完成**

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 410 页中的 “如何卸载标记区域”。

（可选）将已安装的 lx 标记区域置于就绪状态

转换为就绪状态之后，虚拟平台便可开始运行用户进程。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。

请参见第 247 页中的 “（可选）如何将已安装区域转换为就绪状态” 。

▼ 如何引导 lx 标记区域

引导区域会将该区域置于运行状态。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。

要执行此过程，您必须是全局区域中的全局管理员。

提示 – 请注意，不能在启用标签的 Trusted Solaris 系统中引导标记区域。

---

1. **成为超级用户或承担主管理员角色。**

   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **使用带有 -z 选项、区域名称 lx-zone 以及 boot 子命令的 zoneadm 命令引导区域。**

   ```
   global# zoneadm -z lx-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      /                      native     shared
    1  lx-zone  running      /export/home/lx-zone   lx         shared
   ```

示例 34–5 为区域指定引导参数

使用 -i altinit 选项引导区域：

```
global# zoneadm -z lx-zone boot -- -i /path/to/process
```

故障排除 如果显示一条消息，指出系统找不到要用于在区域配置中指定的 IP 地址的网络掩码，请参见第 355 页中的 “引导区域时显示 netmasks 警告” 。请注意，此消息只是警告，而命令已成功执行。

▼ 如何在单用户模式下引导 lx 标记区域

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**

   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **在单用户模式下引导区域。**

   ```
   global# zoneadm -z lx-zone boot -- -s
   ```

下一步执行的操作

要登录到区域并查看配置，请参见第 416 页中的 “登录到 lx 标记区域” 。

停止、重新引导、卸载、克隆和删除 lx 标记区域（任务图）

任务	说明	参考
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 420 页中的 “如何使用 zlogin 关闭 lx 标记区域” 。	第 409 页中的 “如何停止 lx 标记区域”
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 410 页中的 “如何重新引导 lx 标记区域”
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 410 页中的 “如何卸载标记区域”
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 411 页中的 “在同一系统上克隆 lx 标记区域”

任务	说明	参考
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 414 页中的 “从系统中删除 lx 标记区域”

停止、重新引导和卸载 lx 标记区域

▼ 如何停止 lx 标记区域

停止过程用于删除 lx 标记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 “如何使用 zlogin 关闭 lx 标记区域” 。

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**

   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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      /                     native     shared
    1  lx-zone  running      /export/home/lx-zone  lx         shared
   ```

3. **使用带有 -z 选项、区域名称（例如 lx-zone）以及 halt 子命令的 zoneadm 命令停止给定区域。**

   ```
   global# zoneadm -z lx-zone halt
   ```

4. **再次列出系统上的区域，以检验是否已停止 lx-zone。**

   ```
   global# zoneadm list -iv
   ```

   将显示以下类似信息：

   ```
   ID  NAME       STATUS        PATH                  BRAND      IP
   0  global     running       /                     native     shared
   -  lx-zone    installed     /export/home/lx-zone  lx         shared
   ```

5. **如果您要重新启动区域，请引导它。**

   ```
   global# zoneadm -z lx-zone boot
   ```

故障排除 如果区域没有正常停止，请参见第 354 页中的 “区域无法停止”以获得疑难解答提示。

▼ 如何重新引导 lx 标记区域

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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      /                     native     shared
 1  lx-zone  running      /export/home/lx-zone  lx         shared
```

3 **使用带有 -z reboot 选项的 zoneadm 命令重新引导区域 lx-zone。**

```
global# zoneadm -z lx-zone reboot
```

4 **再次列出系统上的区域，以检验是否已重新引导 lx-zone。**

```
global# zoneadm list -v
```

将显示以下类似信息：

```
ID  NAME     STATUS       PATH                  BRAND      IP
 0  global   running      /                     native     shared
 2  lx-zone  running      /export/home/lx-zone  lx         shared
```

提示 – 请注意，lx-zone 的区域 ID 已更改。区域 ID 通常会在重新引导后更改。

▼ 如何卸载标记区域

注意 – 此过程会删除区域根文件系统中的所有文件。此操作是无法恢复的。

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。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

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       /                    native     shared
-  lx-zone    installed     /export/home/lx-zone  lx         shared
```

3 **使用带有 -z uninstall 选项的 zoneadm 命令删除区域 lx-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 lx-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，以检验是否不再列出 lx-zone。**

```
global# zoneadm list -v
```

将显示以下类似信息：

```
ID  NAME     STATUS       PATH              BRAND      IP
0  global   running      /                 native     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。

使用 uninstall 命令时应谨慎，因为此操作是无法恢复的。

在同一系统上克隆 lx 标记区域

克隆用于通过从源 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时，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 lx 标记区域

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。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **停止要克隆的源区域，在此过程中该区域为 lx-zone。**

```
global# zoneadm -z lx-zone halt
```

3 **通过将源区域 lx-zone 的配置导出到文件（例如 master），开始配置新区域。**

```
global# zonecfg -z lx-zone export -f /export/zones/master
```

注 – 也可以通过使用第 221 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。至少要为新区域设置一个不同的 zonepath 和 IP 地址。**

5 **通过使用文件 *master* 中的命令创建新区域 zone1。**

```
global# zonecfg -z zone1 -f /export/zones/master
```

6 **通过克隆 lx-zone 安装新区域 zone1。**

```
global# zoneadm -z zone1 clone lx-zone
```

系统将显示：

```
Cloning zonepath /export/home/lx-zone...
```

如果源 zonepath 在 ZFS 池（例如 zeepool）中，系统会显示：

```
Cloning snapshot zeepool/zones/lx-zone@SUNWzone1
Instead of copying, a ZFS clone has been created for this zone.
```

7 **列出系统上的区域。**

```
global# zoneadm list -iv
  ID  NAME          STATUS          PATH                    BRAND      IP
   0  global        running         /                       native     shared
   -  lx-zone       installed       /export/home/lx-zone    lx         shared
   -  zone1         installed       /export/home/zone1      lx         shared
```

更多信息 克隆 ZFS 文件系统上的源 zonepath 时

如果 zoneadm 命令克隆位于自己的 ZFS 文件系统上的源 zonepath，则会执行以下操作：

- zoneadm 命令提取软件清单。
- zoneadm 命令捕获 ZFS 快照并将其命名为 SUNWzone*X*，例如 SUNWzone*1*。
- zoneadm 命令使用 ZFS 克隆来克隆快照。

▼ 如何从现有快照克隆区域

可以从克隆区域时最初捕获的现有快照多次克隆源区域。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **配置区域** zone2。

3 **指定要使用现有快照来创建** new-zone2。

```
global# zoneadm -z zone2 clone -s zeepool/zones/lx-zone@SUNWzone1 lx-zone
```

系统将显示：

```
Cloning snapshot zeepool/zones/lx-zone@SUNWzone1
```

zoneadm 命令从快照 SUNWzone*1* 中验证软件并克隆快照。

4 **列出系统上的区域。**

```
global# zoneadm list -iv
  ID NAME             STATUS         PATH                     BRAND      IP
   0 global           running        /                        native     shared
   - lx-zone          installed      /zeepool/zones/lx-zone   lx         shared
   - zone1            installed      /zeepool/zones/zone1     lx         shared
   - zone2            installed      /zeepool/zones/zone1     lx         shared
```

▼ 如何使用复制代替 ZFS 克隆

可以使用此过程指定复制 zonepath，从而阻止在 ZFS 文件系统上自动克隆区域。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **指定复制 ZFS 上的** zonepath **而不是进行 ZFS 克隆。**

```
global# zoneadm -z zone1 clone -m copy lx-zone
```

从系统中删除 lx 标记区域

本节中所述的过程会从系统中完全删除区域。

▼ 如何删除 lx 标记区域

1 **关闭区域 lx-zone。**

```
global# zlogin lx-zone shutdown
```

2 **删除 lx-zone 的根文件系统。**

```
global# zoneadm -z lx-zone uninstall -F
```

3 **删除 lx-zone 的配置。**

```
global# zonecfg -z lx-zone delete -F
```

4 **列出系统上的区域，以检验是否不再列出 lx-zone。**

```
global# zoneadm list -iv
```

将显示以下类似信息：

```
ID  NAME     STATUS       PATH   BRAND      IP
 0  global   running      /      native     shared
```

◆ ◆ ◆ 第 35 章

登录到 lx 标记区域（任务）

本章提供以下信息：

- 有关区域登录的介绍性信息
- 完成已安装 lx 标记区域的内部配置
- 从全局区域登录到区域
- 关闭区域
- 使用 zonename 命令列显当前区域的名称

zlogin 命令概述

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。

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。

如第 418 页中的 “如何使用非交互模式访问 lx 标记区域”中所述，可以通过提供要在区域内运行的命令，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。

只有全局区域中的全局管理员才能使用 zlogin 命令。有关更多信息，请参见 zlogin(1) 手册页。

lx 标记区域登录方法

第 256 页中的 “非全局区域登录方法”中提供了区域控制台和用户登录方法的概述。

当出现登录问题使您无法使用 zlogin 命令或带 -C 选项的 zlogin 命令来访问区域时，可以使用故障安全模式。该模式在第 257 页中的 “故障安全模式”中介绍。

有关远程登录区域的信息在第 257 页中的 “远程登录”中介绍。

交互模式分配新的伪终端，以供在区域内部使用。非交互模式用于运行可管理区域的 shell 脚本。有关更多信息，请参见第 257 页中的 “交互模式与非交互模式” 。

标记区域的登录过程（任务图）

任务	说明	参考
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 416 页中的 “登录到 lx 标记区域”
退出标记区域。	从标记区域断开。	第 419 页中的 “如何退出 lx 标记区域”
关闭标记区域。	使用 shutdown 实用程序或脚本来关闭标记区域。	第 420 页中的 “如何使用 zlogin 关闭 lx 标记区域”

登录到 lx 标记区域

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 257 页中的 “远程登录”中所述。

▼ 如何登录到 lx 标记区域控制台

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **使用带有 -C 选项和区域名称（例如 lx-zone）的 zlogin 命令。**

```
global# zlogin -C lx-zone
[Connected to zone 'lx-zone' console]
```

注 – 如果您在发出 zoneadm boot 命令之后立即启动 zlogin 会话，则会显示区域的引导消息：

```
INIT: version 2.85 booting
                    Welcome to CentOS
                    Press 'I' to enter interactive startup.
    Configuring kernel parameters:  [  OK  ]
    Setting hostname lx-zone:  [  OK  ]
    [...]
    CentOS release 3.6 (Final)
    Kernel 2.4.21 on an i686
```

---

3 **当显示区域控制台时，以 root 身份登录，按回车键，并在提示时键入超级用户口令。**

```
lx-zone console login: root
Password:
```

注 – 回顾一下，当从 Sun tarball 安装区域时，root（超级用户）口令为 root。当从 ISO 映像或 CD 安装区域时，root（超级用户）口令未设置（为空）。

▼ 如何使用交互模式访问标记区域

在交互模式下，会分配新的伪终端以在区域内部使用。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **从全局区域登录到某个区域（例如 lx-zone）。**

```
global# zlogin lx-zone
```

将显示以下类似信息：

```
[Connected to zone 'lx-zone' pts/2]
Last login: Wed Jul  3 16:25:00 on console
Sun Microsystems Inc. SunOS 5.10 Generic July 2006
```

3 **键入 exit 关闭连接。**

将显示以下类似消息：

```
[Connection to zone 'lx-zone' pts/2 closed]
```

▼ 如何检验运行环境

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **登录到某个区域，例如 lx-zone。**

```
global# zlogin lx-zone
```

3 **检验当前是否在 Solaris 操作系统下的 Linux 环境中运行。**

```
[root@lx-zone root]# uname -a
```

将显示以下类似信息：

```
Linux lx-zone 2.4.21 BrandZ fake linux i686 i686 i386 GNU/Linux
```

▼ 如何使用非交互模式访问 lx 标记区域

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。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **从全局区域登录到 lx-zone 区域并提供命令名称。**

将命令替换为要在区域内运行的命令的名称。

```
global# zlogin lx-zone command
```

示例 **35–1** 在区域 lx_master 中使用命令 uptime

```
global#  zlogin lx_master uptime
    21:16:01  up  2:39,  0 users,  load average: 0.19, 0.13, 0.11
    fireball#
```

▼ 如何退出 lx 标记区域

- **要从非全局区域断开，请使用波浪号 (~) 字符和句点：**

```
zonename# ~.
```

将显示以下类似信息：

```
[Connection to zone 'lx-zone' pts/6 closed]
```

  - **您还可以键入 exit 来退出区域。**

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 lx 标记区域

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。

要执行此过程，您必须是全局区域中的全局管理员。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **在全局区域中，使用带有 -S 选项的 zlogin 命令访问区域（例如 lx-zone）。**

```
global# zlogin -S lx-zone
```

▼ 如何使用 zlogin 关闭 lx 标记区域

注 – 如果在全局区域中运行 `init 0` 干净地关闭 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 250 页中的 “如何停止区域”。

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**

   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **登录到要关闭的区域（例如 `lx-zone`），并将 `shutdown` 指定为实用程序的名称，将 init 0 指定为状态。**

   ```
   global# zlogin lx-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。

更多信息

在非交互模式下使用 shutdown

目前，不能在非交互模式下使用 `shutdown` 命令将区域置于单用户状态。有关更多信息，请参见 CR 6214427。

可以使用交互式登录，如第 417 页中的 “如何使用交互模式访问标记区域”中所述。

◆ ◆ ◆ 第 36 章

移动和迁移 lx 标记区域（任务）

本章介绍如何：

- 将现有的 lx 标记区域移动到同一计算机上的新位置。
- 在执行实际迁移前验证将在 lx 标记区域迁移中发生的情况。
- 将现有的 lx 标记区域迁移到新计算机中。

移动 lx 标记区域

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。新 zonepath 必须位于本地文件系统中。需要满足第 379 页中的 “资源和属性类型”中所述的标准 zonepath 条件。

▼ 如何移动区域

1 **成为超级用户或承担主管理员角色。**

《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”介绍了角色。

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 /export/zones/db-zone。**

```
global# zoneadm -z db-zone move /export/zones/db-zone
```

4 **检验路径。**

```
global# zoneadm list -iv
  ID NAME             STATUS         PATH                    BRAND      IP
```

```
0 global          running        /                    native     shared
- lx-zone         installed      /export/home/lx-zone lx         shared
- db-zone         installed      /export/zones/db-zone lx        shared
```

将 lx 标记区域迁移到其他计算机

关于迁移 lx 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。

以下要求适用于 lx 标记区域迁移：

- 目标系统上的全局区域必须与原始主机运行相同的 Solaris 发行版。
- 为确保区域可以正常运行，目标系统上所需安装的操作系统软件包和修补程序必须与原始主机上安装的软件包和修补程序具有相同的版本。
- 原始主机和目标系统上的标记必须相同。
- 目标系统的处理器类型必须为以下受支持的 i686 处理器类型之一：
  - Intel
    - Pentium Pro
    - Pentium II
    - Pentium III
    - Celeron
    - Xeon
    - Pentium 4
    - Pentium M
    - Pentium D
    - Pentium Extreme Edition
    - Core
    - Core 2

    AMD

    - Opteron
    - Athlon XP
    - Athlon 64
    - Athlon 64 X2
    - Athlon FX
    - Duron
    - Sempron
    - Turion 64
    - Turion 64 X2

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。

▼ 如何迁移 lx 标记区域

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **停止要迁移的区域，在此过程中为 lx-zone。**

```
host1# zoneadm -z lx-zone halt
```

3 **分离该区域。**

```
host1# zoneadm -z lx-zone detach
```

分离的区域现在处于已配置状态。

4 **将 lx-zone 的 zonepath 移至新主机。**

有关更多信息，请参见第 424 页中的 “如何将 zonepath 移动到新主机”。

5 **在新主机上，对该区域进行配置。**

```
host2# zonecfg -z lx-zone
```

您会看到以下系统消息：

```
lx-zone: No such zone configured
Use 'create' to begin configuring a new zone.
```

6 **要在新主机上创建区域 lx-zone，请使用带有 -a 选项以及新主机上的 zonepath 的 zonecfg 命令。**

```
zonecfg:lx-zone> create -a /export/zones/lx-zone
```

7 **查看配置。**

```
zonecfg:lx-zone> info
zonename: lx-zone
zonepath: /export/zones/lx-zone
brand: lx
autoboot: false
bootargs:
```

```
pool:
limitpriv:
net:
         address: 192.168.0.90
         physical: bge0
```

8 （可选）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可能有所不同，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。

```
zonecfg:lx-zone> select net physical=bge0
zonecfg:lx-zone:net> set physical=e1000g0
zonecfg:lx-zone:net> end
```

9 提交配置并退出。

```
zonecfg:lx-zone> commit
zonecfg:lx-zone> exit
```

10 在新主机上附加区域。

- **附加区域，并进行验证检查。**

  ```
  host2# zoneadm -z lx-zone attach
  ```

  当发生下列一种或两种情况时，将向系统管理员通知所需执行的操作：

  - 新计算机中不存在所需软件包和修补程序。
  - 计算机之间的软件级别不同。

- **强制执行附加操作，而不执行验证。**

  ```
  host2# zoneadm -z lx-zone attach -F
  ```

注意 – 通过 -F 选项可以在不进行验证的情况下强制执行 attach。这在某些情况下（例如在群集环境中或在执行备份和恢复操作时）很有用，但要求对系统进行托管区域所需的正确配置。不正确的配置以后可能会导致未定义的行为。

▼ 如何将 zonepath 移动到新主机

创建 zonepath 的归档的方法有很多种。例如，可以使用 cpio(1) 和 pax(1) 手册页中所述的 cpio 或 pax 命令。

将归档传送至新主机的方法也有很多种。用于将 zonepath 从源主机传送到目标主机的机制取决于本地配置。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。在其他情况下，可能要将 zonepath 写入磁带，再将磁带邮寄至新站点。

由于上述原因，此步骤不能自动执行。系统管理员必须选择最合适的方法来将 zonepath 移动到新主机。

1 **成为超级用户或承担主管理员角色。**

有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2 **将 zonepath 移动到新主机。您可以使用本过程中介绍的方法，也可以使用您选择的其他方法。**

示例 36–1 使用 tar 命令归档和移动 zonepath

1. 在 host1 上创建 zonepath 的 tar 文件，并使用 sftp 命令将其传送到 host2。

```
host1# cd /export/zones
host1# tar cf lx-zone.tar lx-zone
host1# sftp host2
Connecting to host2...
Password:
sftp> cd /export/zones
sftp> put lx-zone.tar
Uploading lx-zone.tar to /export/zones/lx-zone.tar
sftp> quit
```

2. 在 host2上，解压缩 tar 文件。

```
host2# cd /export/zones
host2# tar xf lx-zone.tar
```

有关更多信息，请参见 sftp(1) 和 tar(1)。

故障排除 有关以下问题的疑难解答信息，请参见第 355 页中的 “使用 zoneadm attach 操作解决问题”：

- 修补程序和软件包不同步。
- 操作系统发行版不匹配。

用户必须验证新计算机中的处理器类型是否受支持。有关更多信息，请参见第 422 页中的 “关于迁移 lx 标记区域”。

Solaris 10 5/08：关于在执行迁移之前验证 lx 标记区域迁移

可以在将区域移动到新计算机之前使用“不执行”选项 -n 执行试验。

zoneadm detach 子命令与 -n 选项结合使用，可在运行的区域上生成清单，而不实际分离该区域。源系统中区域的状态不会改变。区域清单会被发送到 stdout。全局管理员可以将此输出定向到某一文件，或将此输出传输到远程命令以便立即在目标主机上进行验证。zoneadm attach 子命令与 -n 选项结合使用，可读取该清单并检验目标计算机是否具有托管区域所需的正确配置，而不实际执行附加。

在执行试验性附加之前，不必在新主机上配置目标系统中的区域。

▼ Solaris 10 5/08：如何在执行迁移之前验证 lx 标记区域迁移

要执行此过程，您必须是全局区域中的全局管理员。

1. **成为超级用户或承担主管理员角色。**

   有关如何创建该角色并将其指定给用户，请参见《系统管理指南：基本管理》中的“使用 RBAC 和 Solaris 管理工具（任务图）”。

2. **使用以下方法之一。**

   - **在名为 lx-zone 的源主机上生成清单，并将输出传输到将立即验证目标主机的远程命令：**

     ```
     global# zoneadm -z lx-zone detach -n | ssh remotehost zoneadm attach -n -
     ```

     行尾的连字符 (–) 指定了路径为 stdin。

   - **在名为 lx-zone 的源主机上生成清单，并将输出定向到某一文件：**

     ```
     global# zoneadm -z lx-zone detach -n
     ```

     如第 424 页中的 “如何将 zonepath 移动到新主机” 中所述将清单复制到新主机系统，并执行验证：

     ```
     global# zoneadm attach -n path_to_manifest
     ```

     路径可以是 –，以便指定 stdin。

◆ ◆ ◆ 第 37 章

在 lx 标记区域中管理和运行应用程序（任务）

本章包含有关在 lx 标记区域中运行应用程序所需的材料。

关于维护支持的配置

在使用支持的 CentOS 或 Red Hat Enterprise Linux 分发安装区域时，就创建了一个支持的区域。如果向此区域中添加不同版本的软件包，则可能会创建一个不支持的标记区域。

升级分发版本和添加软件包

▼ 如何升级 CentOS 3.*x* 分发

要执行此过程，您必须是 lx 标记区域中的区域管理员。

- **使用 yum upgrade 或 up2date 将 CentOS 3.*x* 分发升级到不同版本。**
  有关说明，请参见位于 http://www.centos.org 的文档。

▼ 如何升级 Red Hat 3.*x* 分发

要执行此过程，您必须是 lx 标记区域中的区域管理员。

- **使用 up2date 将 Red Hat Enterprise Linux 3.*x* 分发更新到不同版本。**
  有关说明，请参见位于 http://www.redhat.com 的文档。

▼ 如何升级软件包

要执行此过程，您必须是 lx 标记区域中的区域管理员。

- **要更新软件包，请使用以下方法之一。**
  - **yum update** *package_name*
  - **rpm -U** *package_name*

更多信息 **使用 yum 和 rpm**

yum：

- 《Software Management with Yum》文档中包含有关从隔离软件包中安装软件的一章。请参见 http://fedora.redhat.com/docs/yum。
- yum.conf(5)
- yum(8)

rpm：

- 请参见 http://kbase.redhat.com/faq/FAQ_35_198.shtm 中的《How do I install or upgrade an RPM package?》。
- rpm(8)

如何在 lx 标记区域中安装应用程序

通过装入 CD 并运行安装程序来安装应用程序，就像在 Linux 系统中进行安装一样。本节对 lx 标记区域中典型的应用程序安装进行了说明。

提示 – 如果您知道将要使用 CD 或 DVD 在 lx 标记区域中安装应用程序，请在最初配置标记区域时，在全局区域内添加对 CD 或 DVD 介质的只读访问权限。请参见第 429 页中的 “如何使用 CD 安装 MATLAB 7.2”中的步骤 7。

关于 MATLAB

MATLAB 是一种可快速执行计算密集型任务的高级语言和交互式环境。该产品由 The MathWorks 开发。有关更多信息，请参见 http://www.mathworks.com。

▼ 如何使用 CD 安装 MATLAB 7.2

1. **获取 MATLAB 7.2 CD。**

   MATLAB/Simulink 软件包中有三张 CD。简单的 MATLAB 安装只需要使用光盘 1 和光盘 3。

2. **按照第 387 页中的“ 如何配置、检验和提交 lx 标记区域 ”和第 402 页中的“ 安装和引导 lx 标记区域 ”中的说明创建和安装 lx 标记区域。**

3. **如果 Volume Management 文件系统未在全局区域中运行，则启动它。**

   ```
   global# svcadm volfs enable
   ```

4. **插入介质。**

5. **检查驱动器中的介质。**

   ```
   global# volcheck
   ```

6. **测试 CD 是否自动装入。**

   ```
   global# ls /cdrom
   ```

   将显示以下类似信息：

   ```
   cdrom   cdrom1   mathworks_2006a1
   ```

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 lx-zone
   zonecfg:lx-zone> add fs
   zonecfg:lx-zone:fs> set dir=/cdrom
   zonecfg:lx-zone:fs> set special=/cdrom
   zonecfg:lx-zone:fs> set type=lofs
   zonecfg:lx-zone:fs> add options [ro,nodevices]
   zonecfg:lx-zone:fs> end
   zonecfg:lx-zone> commit
   zonecfg:lx-zone> exit
   ```

8. **重新引导非全局区域。**

   ```
   global# zoneadm -z lx-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      /                              native     shared
 1  lx-zone  running      /export/home/lx-zone           lx         shared
```

10 **登录到 lx 区域。**

```
global# zlogin lx-zone
```

11 **检验 CD-ROM 挂载。**

```
lx-zone# ls /cdrom
```

将显示以下类似信息：

```
cdrom   cdrom1   mathworks_2006a1
```

12 **按照 MATLAB 文档中的说明创建许可证文件。**

13 **按照产品安装指南中的介绍安装产品。**

```
lx-zone# /mnt/install
```

14 **退出区域。**

```
lx-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。

---

15 **（可选）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 lx-zone
zonecfg:lx-zone> remove fs dir=/cdrom
zonecfg:lx-zone> commit
zonecfg:lx-zone> exit
```

▼ 如何使用 ISO 镜像安装 MATLAB 7.2

开始之前 请注意，此方法需要占用相当大的磁盘空间。

1 **获取 MATLAB 7.2 CD。**

MATLAB/Simulink 软件包中有三张 CD。简单的 MATLAB 安装只需要使用光盘 1 和光盘 3。

2 **按照第 387 页中的“ 如何配置、检验和提交 lx 标记区域 ”和第 402 页中的“ 安装和引导 lx 标记区域 ”中的说明创建和安装 lx 标记区域。**

3 **将每张 CD 中的数据复制到 .iso 文件中。**

```
global# /usr/bin/dd if=/dev/rdsk/c1d0s2 of=disk1.iso
```

上述命令可将第一张 CD 中的数据复制到文件 disk1.iso 中。可对第三张 CD 重复此操作，但使用不同的文件名，如 disk3.iso。

4 **从全局区域中，将第一个 .iso 文件按 lofi 方式挂载到 lx 区域。**

```
global# lofiadm -a /zpool/local/disk1.iso
global# mount -F hsfs /dev/lofi/1 /zones/lx-zone/root/mnt
```

5 **登录到 lx 区域。**

```
global# zlogin lx-zone
```

6 **使用 X 转发来将显示重定向到您的桌面：**

```
lx-zone# ssh -X root@lx-zone
```

7 **按照 MATLAB 文档中的说明创建许可证文件。**

8 **按照产品安装指南中的介绍安装产品。**

```
lx-zone# /mnt/install
```

9 **当提示插入 CD 3 时，请返回全局区域终端窗口，然后挂载 disk3.iso 文件来替代第一个文件。**

```
global# umount /zones/lx-zone/root/mnt
    global# lofiadm -d /dev/lofi/1
    global# lofiadm -a /zpool/local/disk3.iso
    global# mount -F hsfs /dev/lofi/1 /zones/lx-zone/root/mnt
```

将完成安装。

备份 lx 标记区域

有关区域备份的信息，请参见第 323 页中的 “关于备份安装了区域的 Solaris 系统”、第 325 页中的 “确定在非全局区域中备份的内容”、第 326 页中的 “关于恢复非全局区域”和第 349 页中的 “恢复非全局区域”。

lx 标记区域中不支持的功能

lx 标记区域中只支持共享 IP 网络配置。

Linux 区域中不支持 chroot 命令。如果在某一进程中使用，则该进程将无法再访问其运行时所需的 Solaris 库。

尽管您可以在启用了标记的 Trusted Solaris 系统上配置和安装 lx 标记区域，但无法在此系统配置中引导 lx 标记区域。

您不能使用 zonecfg 命令的 fs 资源属性来添加本地 Linux 文件系统。

词汇表

bless 在 Perl 中，用于创建对象的关键字。

blessed 在 Perl 中，用于表示类成员关系的术语。

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。

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） 拥有超级用户权限或主管理员角色的管理员。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。

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 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） 对试图超过控制值的进程执行的本地操作。

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 Solaris 操作系统的单个实例中创建的虚拟操作系统环境。Solaris Zones 软件分区技术用于虚拟化操作系统服务。

page in（页入） 以一次一页的方式将数据从文件读入物理内存。

page out（页出） 将页面重新放置到物理内存以外的区域。

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（项目） 相关工作在网络范围内的管理标识符。

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） 识别不常用的页面并将这些页面重新放置到物理内存以外区域的内核线程。

Solaris Container 一个完整的应用程序运行时环境。资源管理和 Solaris Zones 软件分区技术是该容器的两个组成部分。

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。

sparse root zone（稀疏根区域） 具有 `inherit-pkg-dir` 资源并优化对象共享的非全局区域类型。

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） 不具有 `inherit-pkg-dir` 资源的非全局区域类型。

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） 拥有区域管理配置文件的管理员。区域管理员的权限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”、“正在运行”或“正在关闭”中的一种。

索引

P

R

S