在 Oracle® Solaris 11.2 中确保用户和进程的安全

文件号码 E53956
2014 年 7 月

ORACLE

目录

示例

使用本文档

- **概述** – 说明如何为用户指定额外权限，创建并使用角色，并将权限指定给 Oracle Solaris 系统上的程序和特定资源。
- **目标读者** – 安全管理员。
- **必备知识** – 站点安全要求。

产品文档库

位于 http://www.oracle.com/pls/topic/lookup?ctx=E56344 的文档库中包含此产品的最新信息和已知问题。

获得 Oracle 支持

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

反馈

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

♦ ♦ ♦ 第 1 章

使用权限控制用户和进程

Oracle Solaris 提供了可以指定给用户、角色、进程和所选资源的权限。这些权限提供了一个比 superuser model（超级用户模型）更安全的管理方法。

本章提供了有关支持用户和进程权限管理的元素的信息，讨论了扩展用户权限、限制用户权限、向命令添加特权以及将应用程序特权限制为其所需特权的方法：

Oracle Solaris 11.2 中新增的权限功能

本节为现有客户重点介绍用户权限方面的重要新功能，称为基于权限的访问控制 (rights based access control, RBAC) 和进程权限，也称为特权。

- 管理员作为需要验证权限配置文件指定的权限配置文件强制用户在运行特权命令之前提供口令。如果用户不提供口令，则在没有特权的情况下运行该命令。该口令在一定时间（可配置）内保持有效。请参见例 3-11 “要求用户在管理 DHCP 前键入口令”。

  通过在 `policy.conf` 文件中添加配置文件（作为 `AUTH_PROFS_GRANTED` 关键字的值），您可以为登录到系统的任何人指定需要验证权限配置文件。
- 通过指定 `access_times` 和 `access_tz` 权限，可以按照时间和时区限制用户和组对主机的访问。有关示例，请参见 `user_attr(4)` 手册页。
- Oracle Solaris 在 `armor` 软件包中提供了基于 RBAC 管理的授权角色 (Authorization Roles Managed on RBAC, ARMOR) 标准化角色集。有关更多信息，请参见“用户和进程权限提供了一种替代超级用户模型的方法” [14]和例 3-1 “使用 ARMOR 角色”。
- 提供了一个用户管理器 GUI 来管理用户和角色的权限。有关更多信息，请参见《在 Oracle Solaris 11.2 中管理用户帐户和用户环境 》中的第 3 章 “使用用户管理器 GUI 管理用户帐户”。

用户权限管理

用户权限管理是一种安全功能，用于控制用户对通常仅限于 root 角色执行的任务的访问。通过对进程和用户应用安全属性（或权限），站点可以将超级用户特权划分给多个管理员。进程权限管理通过特权实现。用户权限管理通过权限配置文件实现，此配置文件收集的权限稍后将指定给用户或角色。用户权限也可以进行限制，例如针对资讯服务站或来宾用户进行限制。

- 有关内核进程权限的讨论，请参见“进程权限管理” [21]。
- 有关管理权限的过程，请参见第 3 章 在 Oracle Solaris 中指定权限。
- 有关参考信息，请参见第 8 章 Oracle Solaris 权限参考信息。

用户和进程权限提供了一种替代超级用户模型的方法

在传统 UNIX 系统中，root 用户（也称为超级用户）具有最高权限。作为 root 运行的程序（如许多 setuid 程序）也可以执行所有功能。root 用户可以读取和写入任何文件，运行所有程序，以及向任何进程发送中止信号。实际上，任何可成为超级用户的用户都能够修改站点的防火墙，更改审计迹，读取机密记录以及关闭整个网络。被劫持的 setuid root 程序可以在系统上执行任何操作。

为用户、资源和进程指定权限是一种比超级用户模型（要么具有全部控制权要么毫无控制权）更为安全的替代方案。通过权限，您可以在更精细的级别上强制执行安全策略。权限采用最小特权安全原则。最小特权表示用户仅具有执行某项作业所必需的 privilege（特权）。一般用户具有足够特权来使用其应用程序、检查其作业状态、打印文件、创建新文件等。超出一般用户权限以外的权限将划分到权限配置文件中。如果用户将要执行的作业要求具有某些超级用户权限，可以为他们指定一个权限配置文件。

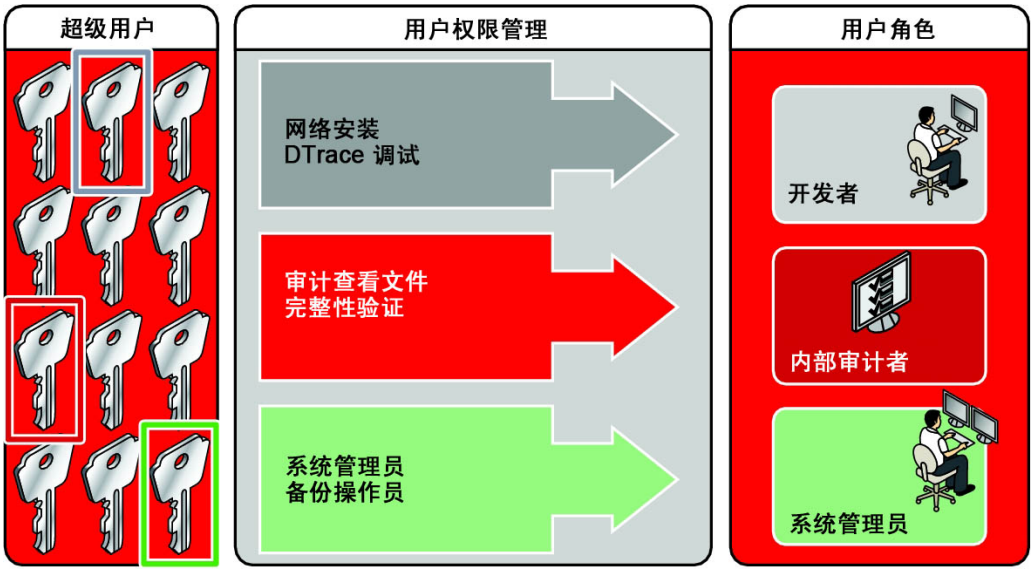
划分到一个配置文件的权限可以直接指定给用户。也可以通过创建称为角色的特殊帐户间接指定这些权限。然后用户可以承担一个角色，完成需要部分管理特权的工作。Oracle Solaris 提供了许多预定义的权限配置文件。您可以创建角色并指定相应的配置文件。

ARMOR 软件包提供了一组标准化角色。通过自动安装此软件包，并将角色指定给用户，您可以创建在引导时就职责分离的系统。有关更多信息，请参见 Authorization Rules Managed On RBAC (ARMOR)（基于 RBAC 管理的授权规则 (ARMOR)）、“遵循选择的权限模型” [36]以及例 3-1 “使用 ARMOR 角色”。

权限配置文件可以提供广泛的管理权限。例如，"System Administrator"（系统管理员）权限配置文件允许帐户执行与安全无关的任务，如打印机管理和 cron 作业管理。也可将权限配置文件定义为提供限定功能。例如，"Cron Management"（计时程序管理）权限配置文件可管理 at 和 cron 作业。创建角色时，可以为这些角色指定广泛管理权限或有限权限。

下图说明了 Oracle Solaris 如何通过创建角色将权限分配给多个 trusted users（可信用户）。超级用户也可以通过将权限配置文件直接指定给可信用户来分配权限。

图 1-1 权限分配



在图示的权限模型中，超级用户创建了三个角色。这些角色基于权限配置文件。然后，超级用户会将这些角色指定给可以执行角色任务的可信用户。这些用户使用自己的用户名进行登录。登录之后，用户将承担可以运行管理命令和图形用户界面 (graphical user interface, GUI) 工具的角色。

由于可以灵活地设置角色，因此可实现各种安全策略。尽管 Oracle Solaris 附带的角色很少，但是可以轻松配置角色。例 3-1 “使用 ARMOR 角色” 显示如何使用基于 ARMOR 标准的用户角色。要补充或替代 ARMOR 角色，可以基于 Oracle Solaris 提供的权限配置文件创建自己的角色。

- root – 等效于 `root` 用户的功能强大的角色。但是与所有角色一样，`root` 角色无法登录。一般用户必须先登录，再承担指定的 `root` 角色。缺省情况下，该角色配置并指定给初始用户。
- System Administrator（系统管理员） – 可执行与安全性无关的管理任务的角色，其权力相对较弱。此角色可以管理文件系统、邮件以及软件安装，但是不能设置口令。
- Operator（操作员） – 可执行备份和打印机管理等操作的初级管理员角色。

注 - "Media Backup"（介质备份）权限配置文件提供对整个根文件系统的访问权限。因此，面向初级管理员设计 "Media Backup"（介质备份）和 "Operator"（操作员）权限配置文件时，必须确保用户可信。

可能还希望配置一个或多个安全角色。三个权限配置文件及其补充的配置文件处理安全性：信息安全、用户安全和区域安全。网络安全是信息安全权限配置文件中的补充配置文件。

请注意，不一定要实施角色。角色是一种满足组织安全性需要的功能。一种策略是为安全、网络或防火墙管理等领域中的特殊用途管理员设置角色。另一种策略是创建一个功能强大的管理员角色以及一个高级用户角色。高级用户角色将用于那些获许修复其自己的系统部分的用户。您也可以直接为用户指定权限配置文件，而不创建任何角色。

超级用户模型与权限模型可以共存。下表汇总了权限模型中可以存在的各个等级（从超级用户到受限的一般用户）。该表包含可以在这两种模型中跟踪的管理操作。有关进程权限（即特权）作用的概述，请参见表 1-2 “具有特权的系统与不具有特权的系统之间的明显差别”。

表 1-1 超级用户模型与权限模型的对比

系统上的用户功能	超级用户模型	权限模型
是否可成为具有完全超级用户特权的超级用户	可以	可以
是否可作为具有完全用户权限的用户登录	可以	可以
是否可成为具有有限权限的超级用户	不可以	可以
是否可作为用户登录，有时具有超级用户特权	可以，仅限 `setuid root` 程序	可以，使用 `setuid root` 程序和权限
是否可作为具有管理权限的用户登录，但是不具有完全超级用户特权	不可以	可以，使用权限配置文件、角色以及使用直接指定的特权和授权
是否可作为权限少于一般用户的用户登录	不可以	可以，通过删除权限
是否可跟踪超级用户操作	可以，通过审计 `su` 命令	可以，通过审计对 `pfexec（）` 的调用 另外，审计迹中会出现已经承担 `root` 角色的用户的名称

用户和进程权限基础

术语无特权或无权限在 Oracle Solaris 中不适用。Oracle Solaris 中的每个进程（包括一般用户进程）都至少有部分特权或其他用户权限，例如授权。要了解 Oracle Solaris 授予所有 UNIX 进程的基本特权集合，请参见“进程权限管理” [21]。

以下元素用于在 Oracle Solaris 中实现用户权限。可以配置这些权限来强制实施许可性安全策略或限制性安全策略。

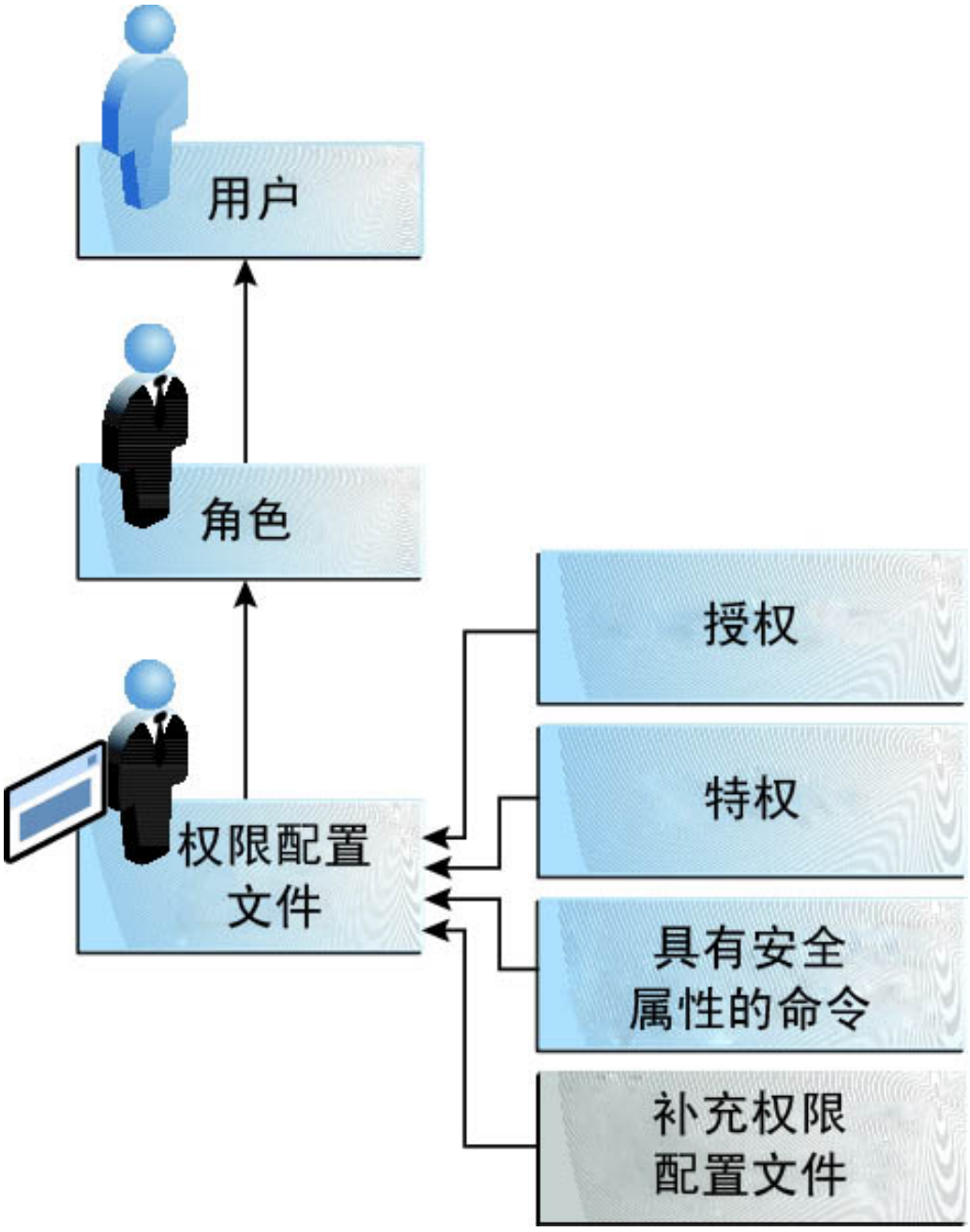
- **授权** – 一种权限，允许用户或角色执行某一类需要额外权限才能执行的操作。例如，默认安全权限为控制台用户提供 `solaris.device.cdrw` 授权。用户可使用此授权来读取和写入 CD-ROM 设备。有关授权的列表，请使用 `auths list` 命令。授权在用户应用程序级别（而不是内核级别）实施。请参见“有关用户授权的更多信息” [20]。

- **特权** – 可以授予命令、用户、角色或特定资源（例如一个端口或 SMF 方法）的权限。特权是在内核中实施的。例如，`proc_exec` 特权允许进程调用 `execve（）`。一般用户具有基本特权。要查看您的基本特权，请运行 `ppriv -vl basic` 命令。有关详细信息，请参见“进程权限管理” [21]。
- **安全属性** – 允许进程执行某个操作或实现某个权限的属性。在典型的 UNIX 环境中，安全属性允许进程执行原本禁止一般用户执行的操作。例如，`setuid` 和 `setgid` 程序具有安全属性。在权限模型中，授权和特权是除 `setuid` 和 `setgid` 程序之外的安全属性。可以将这些属性或权限指定给某位用户。例如，具有 `solaris.device.allocate` 授权的用户可以分配设备供独占使用。特权可以置于某个进程上。例如，具有 `file_flag_set` 特权的进程可以设置不变的、未解除链接的或仅附加的文件属性。

  安全属性还可以限制权限。例如，`access_times` 和 `access_tz` 安全属性可以设置允许执行特定安全相关操作的日期和时间以及（可选）时区。您可以直接限制用户或通过向他们指定包含这些关键字的需要验证权限配置文件来限制用户。有关更多信息，请参见 `user_attr(4)` 手册页。
- **特权应用程序** – 可以通过检查权限来覆盖系统控制的应用程序或命令。有关更多信息，请参见“检查权限的应用程序” [32]和《面向开发者的 Oracle Solaris 11 安全性指南》。
- **权限配置文件** – 可以指定给角色或用户的权限的集合。一个权限配置文件可以包含授权、直接指定的特权、具有安全属性的命令以及其他权限配置文件。其他配置文件中的配置文件称为补充权限配置文件。权限配置文件提供了一种便捷的权限分组方法。可以将配置文件直接指定给用户或称为角色的特殊帐户。只有当进程识别了权限后，才能使用权限配置文件中的命令。另外，可能会要求您提供口令。或者，可以提供缺省口令验证。请参见“有关权限配置文件的更多信息” [20]。
- **角色** – 用于运行特权应用程序的特殊身份。这种特殊身份只能由指定的用户承担。在按角色运行的系统中，初始配置后可能就不再需要超级用户。请参见“有关角色的更多信息” [21]。

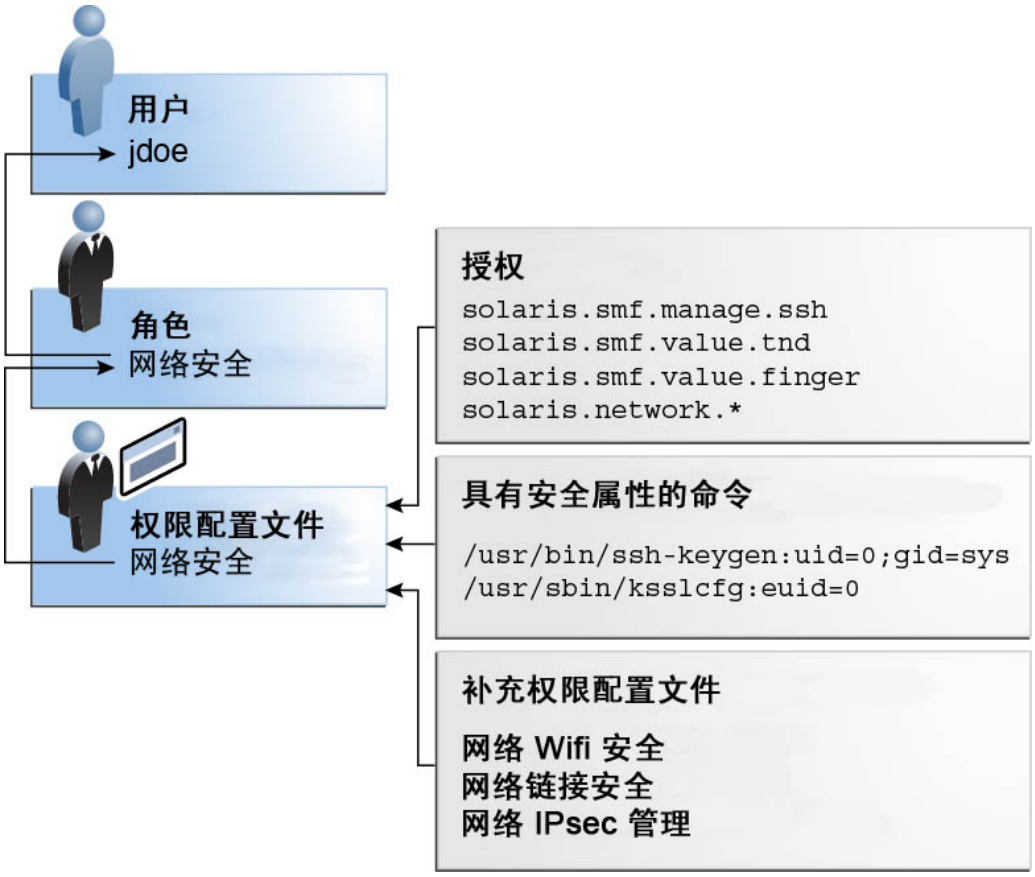
下图显示了用户权限和进程权限协同工作的方式。

图 1-2 用户权限和进程权限协同工作



下图使用 "Network Security"（网络安全）角色和 "Network Security"（网络安全）权限配置文件说明了如何使用指定的权限。

图 1-3 用户权限和进程权限指定示例



"Network Security"（网络安全）角色用于管理 IPsec、wifi 和网络链路。该角色被指定给了用户 `jdoe`。`jdoe` 可以通过切换到该角色然后提供角色口令来承担该角色。管理员可以启用角色通过用户口令而不是角色口令进行验证。

在本图中，"Network Security"（网络安全）权限配置文件指定给 "Network Security"（网络安全）角色。"Network Security"（网络安全）权限配置文件包含一些按顺序评估的补充配置文件：Network Wifi Security（网络 Wifi 安全）、Network Link Security（网络链接安全）和 Network IPsec Management（网络 IPsec 管理）。这些补充配置文件包含完成该角色的主要任务的权限。

"Network Security"（网络安全）权限配置文件有三个直接指定的授权，没有直接指定的特权，还有两个具有安全属性的命令。补充权限配置文件有直接指定的授权，其中两个包含具有安全属性的命令。

如果 `jdoe` 承担了 "Network Security"（网络安全）角色，shell 将更改为 profile shell（配置文件 shell）。配置文件 shell 进程会评估权限的使用，这样 `jdoe` 可以管理网络安全。

有关用户权限的更多信息

本节包含有关用户级别的权限实施和使用的更多详细信息。

有关用户授权的更多信息

授权是可以授予角色、程序、区域或用户的权限。授权在用户应用程序级别强制执行策略。与特权类似，授权指定错误会导致授予的权限超过本应授予的权限。有关详细信息，请参见“特权升级和用户权限” [29]。

授权与特权之间的差别与强制执行安全策略的级别有关。如果没有适当的特权，内核可能会阻止进程执行特权操作。如果没有适当的授权，用户可能无法使用 privileged application（特权应用程序），或无法在特权应用程序内执行与安全相关的操作。有关特权的更全面介绍，请参见“进程权限管理” [21]。

遵循权限的应用程序会先检查用户的授权，然后再授予其访问应用程序或应用程序中特定操作的权限。此检查取代了传统的 UNIX 应用程序中对 `UID=0` 的检查。

有关授权的更多信息，请参见以下各节：

- “授权参考信息” [100]
- “`auth_attr` 数据库” [103]
- “需要授权的命令（摘选）” [106]

有关权限配置文件的更多信息

权限配置文件是权限的集合，可以指定给角色或用户以执行需要管理权限的任务。一个权限配置文件可以包含授权、特权、具有指定安全属性的命令以及其他权限配置文件。权限配置文件还可以包含减少或扩大初始可继承特权集合的条目和减少限制特权集合的条目。

需要验证权限配置文件是一种需要用户提供口令或重新验证的权限配置文件。管理员决定哪些配置文件无需重新验证即可使用。无需重新验证的配置文件的一个典型示例是

"Basic Solaris User"（基本 Solaris 用户）权限配置文件。根据站点的安全要求，与安全相关的任务的权限配置文件可能需要重新验证。

有关权限配置文件的参考信息，请参见以下各节：

- “权限配置文件参考信息” [99]
- “prof_attr 数据库” [104]
- “exec_attr 数据库” [104]

有关角色的更多信息

角色是一种特殊类型的用户帐户，通过此帐户可运行特权应用程序。角色的创建方式与用户帐户的常规创建方式相同。角色具有起始目录、组指定和口令等。权限配置文件和授权可为角色分配管理权限。角色无法从其他角色或从承担该角色的用户继承权限。通过角色分配超级用户特权，可以实现更安全的管理制度。

可以将一种角色指定给多个用户。所有可以承担相同角色的用户都具有同一个角色起始目录，在同一环境中运行，并且可访问相同文件。用户可以通过在命令行运行 su 命令并提供角色名称和角色口令来承担角色。管理员可以将系统配置为允许用户通过提供用户的口令来验证。请参见例 3-16 “使用户可以将自己的口令用作角色口令”。

角色无法直接登录。用户需要首先登录，然后承担角色。承担了一种角色后，如果不先退出当前角色，将无法承担其他角色。

权限配置文件将权限添加到用户环境，角色也同样会为用户提供一个干净的执行环境，此环境与其他可以承担该角色的用户共享。如果用户切换到某个角色，用户的授权或权限配置文件都不会应用于该角色。

passwd、shadow 和 user_attr 数据库存储静态角色信息。您可以而且应该审计角色的操作。

有关设置角色的详细信息，请参见以下各节：

- “遵循选择的权限模型” [36]
- “为用户指定权限” [39]

在 Oracle Solaris 中 root 是一种角色，这阻止了匿名 root 登录。如果审计配置文件 shell 命令 pfexec，则审计迹会包含登录用户的实际 UID、用户承担的角色以及执行过的特权操作。要审计系统的特权操作，请参见“审计管理操作” [73]。

进程权限管理

在 Oracle Solaris 中，进程权限管理由特权实现。特权允许在命令、用户、角色和特定系统资源四个级别上对进程加以限制。特权可以降低因某个用户或某个进程在系统中具

有完全超级用户权力而带来的安全风险。进程权限和用户权限相结合，可提供远远优于传统超级用户模型的替代模型。

通常，特权用于添加权限。但是，特权也可以用于限制权限，例如，将 `setuid root` 程序更改为可识别特权的程序。同样，通过扩展特权策略，管理员可以限制文件对象、用户 ID 或端口只能使用指定特权。这种更精细的特权指定可消除这些资源除基本特权之外的所有其他特权。

- 有关扩展特权策略和限制性特权的信息，请参见“使用扩展特权策略限制特权使用” [29]。
- 有关用户权限的信息，请参见“用户权限管理” [14]。
- 有关如何管理特权的信息，请参见第 3 章 在 Oracle Solaris 中指定权限。
- 有关特权的参考信息，请参见“特权参考信息” [107]。

特权保护内核进程

特权是进程执行某项操作所需的权限。权限是在内核中实施的。如果程序在基本特权集合界限内运行，则该程序也会在系统安全策略界限内运行。例如，`setuid root` 程序就是在系统安全策略界限外部运行的程序。通过使用特权，程序可以不必调用 `setuid root`。

特权枚举了可在系统上执行的操作的种类。程序可以使用可使其成功执行的确切特权运行。例如，对文件进行操控的程序可能需要 `file_dac_write` 和 `file_flag_set` 特权。进程有了这些特权，就不需要以 `root` 身份运行该程序了。

过去，系统没有遵循 privilege model（特权模型）（或权限模型），此模型在“用户和进程权限基础” [16]中有介绍。而是使用超级用户模型。在超级用户模型中，进程以 `root` 或用户身份运行。用户进程只能对用户的目录和文件执行操作。`root` 进程可以在系统上的任何位置创建目录和文件。需要在用户目录外部创建目录的进程将使用 `UID=0`（即以 `root` 身份）运行。安全策略依靠自主访问控制 (discretionary access control, DAC) 来保护系统文件。设备节点受到 DAC 的保护。例如，组 `sys` 拥有的设备只能由该组的成员打开。

但是，`setuid` 程序、文件权限和管理帐户很容易被误用。`setuid` 进程获许执行的操作数比该进程完成其运行所需的操作数多。如果侵入者以无所不能的 `root` 用户身份运行`setuid root` 程序，该程序会受到侵害。同样，能够访问 `root` 口令的任何用户都可能会危及整个系统的安全。

相比之下，通过特权强制执行政策的系统可以在用户权限和 `root` 权限之间建立等级。可以为某个用户授予特权，允许其执行超出一般用户权限的活动，并可对 `root` 加以限制，使 `root` 拥有的特权比当前拥有的特权少。借助权限，可以将使用特权运行的命令单独放置在某个权限配置文件中，并将其指定给一个用户或角色。表 1-1 “超级用户模型与权限模型的对比” 汇总了权限模型提供的用户权限与 root 特权之间的等级。

权限模型的安全性比超级用户模型要高。已从进程中删除的特权不会被利用。进程特权可为敏感文件和设备提供额外的保护措施，而 DAC 防护功能本身可被用来获取访问权限。

之后，特权可以限制程序和进程，使其仅具备程序所需的权限。在实现最小特权的系统上，捕获某个进程的入侵者只能访问该进程所具有的那些特权，不会破坏其余部分的系统安全。

特权说明

可以根据特权范围对特权进行逻辑分组。

- FILE 特权 – 以字符串 `file` 开头的特权用于文件系统对象。例如，`file_dac_write` 特权允许在写入文件时覆盖自主访问控制。
- IPC 特权 – 以字符串 `ipc` 开头的特权用于覆盖 IPC 对象访问控制。例如，`ipc_dac_read` 特权允许进程读取受 DAC 保护的远程共享内存。
- NET 特权 – 以字符串 `net` 开头的特权授予对特定网络功能的访问权限。例如，`net_rawaccess` 特权允许设备连接到网络。
- PROC 特权 – 以字符串 `proc` 开头的特权允许进程修改进程本身的受限属性。`PROC` 特权包括影响非常有限的特权。例如，`proc_clock_highres` 特权允许进程使用高分辨率的计时器。
- SYS 特权 – 以字符串 `sys` 开头的特权为进程授予对各种系统属性的无限制访问权限。例如，`sys_linkdir` 特权允许进程建立和断开指向目录的硬链接。

其他逻辑组包括 CONTRACT、CPC、DTRACE、GRAPHICS、VIRT 和 WIN。

有些特权对系统的影响有限，而有些特权则具有广泛的影响。`proc_taskid` 特权的定义指明了其有限的影响：

```
proc_taskid
        Allows a process to assign a new task ID to the calling process.
```

`net_rawaccess` 特权的定义指明了其广泛的影响：

```
net_rawaccess
        Allows a process to have direct access to the network layer.
```

`privileges(5)` 手册页提供了每种特权的说明。另请参见“列出特权” [87]。

具有特权与不具有特权的系统之间的管理差别

具有特权的系统与不具有特权的系统存在多处明显差别。下表列出了其中的一些差别。

表 1-2 具有特权的系统与不具有特权的系统之间的明显差别

功能	不具有特权	具有特权
守护进程	以 root 身份运行守护进程。	以 daemon 用户身份运行守护进程。 例如，为以下守护进程指定了有限特权，并以 daemon 身份运行：lockd 和 rpcbind。
日志文件所有权	日志文件由 root 拥有。	日志文件由创建此日志文件的 daemon 拥有。root 用户不拥有此文件。
错误消息	错误消息涉及超级用户。 例如，chroot: not superuser。	错误消息反映特权的使用。 例如，chroot 失败的等效错误消息为 chroot: exec failed。
setuid 程序	程序使用 setuid root 来完成不允许一般用户执行的任务。	许多 setuid root 程序只使用所需的特权运行。 例如，以下命令使用特权：audit、ikeadm、ipadm、ipsecconf、ping、traceroute 和 newtask。
文件权限	设备权限由 DAC 控制。例如，sys 组的成员可以打开 /dev/ip。	文件权限 (DAC) 不会预测谁可以打开设备。设备通过 DAC 和设备策略进行保护。 例如，/dev/ip 文件具有 666 权限，但是该设备只能由具有适当特权的进程打开。
审计事件	审计 su 命令的使用会涉及许多管理功能。	审计特权的使用会涉及大多数管理功能。cusa 审计类包括监视管理功能的审计事件。
进程	进程受进程拥有者的权限保护。	进程受特权保护。进程特权和进程标志显示为 /proc/<pid>/priv 目录中的一个新项。
调试	不引用核心转储中的任何特权。	核心转储的 ELF 注释部分的 NT_PRPRIV 和 NT_PRPRIVINFO 注释中包含有关进程特权和标志的信息。 ppriv 命令和其他命令显示了大小合适的集的正确数目。这些命令会将位集中的位正确映射到特权名称。

有关特权的更多信息

本节包括特权实现、使用和指定详细信息。

如何实现特权

每个进程都有四个特权集合，用于确定进程是否可以使用特定特权。内核会自动计算有效特权集合。可以修改初始可继承特权集合。通过编码来使用特权的程序可以减小该程序的允许特权集合。可以缩小限制特权集合。

- **有效特权集合 (E)** – 当前生效的特权集合。进程可以将允许特权集合中的特权添加到有效特权集合，也可以从 E 中删除特权。
- **允许特权集合 (P)** – 可供使用的特权集合。特权可通过继承或指定供程序使用。执行配置文件便是一种将特权指定给程序的方法。setuid 命令可将 root 具有的所有特权

指定给程序。可从允许特权集合中删除特权，但不能向该集中添加特权。已从 P 中删除的特权会自动从 E 中删除。

可识别特权的程序会从该程序的允许特权集合中删除程序从不使用的特权。通过这种方法，程序或恶意进程将无法利用不必要的特权。有关 privilege-aware（可识别特权）程序的更多信息，请参见《面向开发者的 Oracle Solaris 11 安全性指南 》中的第 2 章 “开发特权应用程序”。

- **可继承特权集合 (I)** – 进程可通过调用 `exec` 继承的特权集合。调用 `exec` 之后，继承的特权被置于允许特权集合和有效特权集合中，使两个权限集相同，但 `setuid` 程序这一特殊情况除外。

  对于 `setuid` 程序，调用 `exec` 之后，可继承特权集合首先会受限制特权集合的限制。然后，系统会从继承特权集合 (I) 中删除限制特权集合 (L) 中的所有特权，并将结果指定给此进程的 P 和 E。
- **限制特权集合 (L)** – 此特权集合定义对可用于某个进程及其子进程的特权的外部限制。缺省情况下，限制特权集合为所有特权。进程可以缩小限制特权集合，但是永远不能扩展限制特权集合。L 用来限制 I。因此，在执行 `exec` 命令时，L 限制 P 和 E。

  如果为用户指定的配置文件中包含指定有特权的程序，则此用户通常可以运行该程序。在未修改的系统上，为此程序指定的特权在用户的限制特权集合中。已指定给程序的特权会成为用户的允许特权集合的一部分。要运行已指定有特权的程序，用户必须从 profile shell（配置文件 shell）运行此程序。

内核可以识别基本特权集合。在未修改的系统上，每个用户的初始可继承特权集合等同于登录时获取的基本特权集合。虽然您不能修改基本特权集合，但可以修改用户从基本特权集合中继承的特权。

在未修改的系统上，用户登录时的特权集合类似如下：

```
E (Effective): basic
I (Inheritable): basic
P (Permitted): basic
L (Limit): all
```

在登录时，所有用户的可继承特权集合、允许特权集合和有效特权集合中都包含基本特权集合。用户的限制特权集合等同于区域（全局区域或非全局区域）的缺省特权集合。

您可以将额外的特权直接指定给用户，或更加精确地指定给用户的登录过程，通过权限配置文件间接指定给许多用户，或通过向用户指定特权命令来间接指定。您也可以从用户的基本特权集合中删除特权。有关过程和示例，请参见第 3 章 在 Oracle Solaris 中指定权限。

如何使用特权

特权内置在 Oracle Solaris 中。本节说明 Oracle Solaris 如何在设备、资源管理和传统应用程序中使用特权。

进程如何获取特权

进程可以继承特权，或者为其指定特权。进程从其父进程继承特权。登录时，用户的初始可继承特权集合确定该用户的进程可用的特权。用户初始登录的所有子进程都从此特权集合继承。

您还可以直接为程序、用户、角色和特定资源指定特权。当某个程序需要特权时，可以在权限配置文件中将特权指定给该程序的可执行文件。系统会向获许运行程序的用户或角色指定包含该程序的配置文件。登录或打开配置文件 shell 时，如果在配置文件 shell 中键入了程序的可执行文件，则可使用特权运行该程序。例如，拥有 "Object Access Management"（对象访问管理）配置文件的角色可以使用 `file_chown` 特权运行 `chmod` 命令，因此可以更改不归该角色所有的文件的所有权。

当某个角色或用户运行已直接指定有附加特权的程序时，该指定特权会添加到此角色或用户的可继承特权集合中。指定有特权的程序的子进程会继承父进程的特权。如果子进程需要的特权比父进程的特权多，则必须直接为子进程指定这些额外特权。

通过编码方式使用特权的程序称为 privilege-aware（可识别特权）的程序。可识别特权的程序可在程序执行过程中启用和禁用特权。要在生产环境中成功执行程序，必须为程序指定其可启用和禁用的特权。在使可识别特权的程序可用之前，仅为程序的可执行文件指定该程序所需的特权。然后测试此程序，了解程序是否能成功执行其任务。您还需要检查程序是否误用了其特权。

有关可识别特权的代码示例，请参见《面向开发者的 Oracle Solaris 11 安全性指南 》中的第 2 章 “开发特权应用程序”。要为需要特权的程序指定特权，请参见例 4-1 “将安全属性指定给传统应用程序” 和例 5-7 “创建包含特权命令的权限配置文件”。

特权和设备

在权限模型中由特权保护系统接口，在超级用户模型中这些接口只由文件权限保护。在具有特权的系统中，文件权限太小，因此无法保护这些接口。`proc_owner` 之类的特权可以覆盖文件权限，然后获得对系统的完全访问权限。

因此，在 Oracle Solaris 中，具有设备目录的所有权不足以打开相应设备。例如，不再自动允许 `sys` 组的成员打开 `/dev/ip` 设备。`/dev/ip` 的文件权限是 `0666`，但还需要 `net_rawaccess` 特权才能打开该设备。

因为设备策略由特权控制，在授予设备打开权限方面就有更大的灵活性。可以针对设备策略和驱动程序适当地配置特权要求。您可以配置安装、添加或更新设备驱动程序时的特权要求。

有关更多信息，请参见 `add_drv(1M)`、`devfsadm(1M)`、`getdevpolicy(1M)` 和 `update_drv(1M)` 手册页。

特权和资源管理

在 Oracle Solaris 中，project.max-locked-memory 和 zone.max-locked-memory 资源控制可用于限制指定有 PRIV_PROC_LOCK_MEMORY 特权的进程的内存消耗量。此特权允许进程锁定物理内存中的页。

如果将 PRIV_PROC_LOCK_MEMORY 特权指定给某个权限配置文件，则可为具有此特权的进程提供锁定所有内存的功能。作为一项保护措施，请设置资源控制以防止具有该特权的用户锁定所有内存。对于在非全局区域中运行的特权进程，请设置 zone.max-locked-memory 资源控制。对于在系统上运行的特权进程，请创建一个项目，然后设置 project.max-locked-memory 资源控制。有关这些资源控制的信息，请参见《在 Oracle Solaris 11.2 中进行资源管理 》中的第 6 章 “关于资源控制”和《Oracle Solaris Zones 介绍 》中的第 2 章 “非全局区域配置概述”。

传统应用程序和特权的使用

为了适应传统应用程序，特权的实现使用超级用户模型和权限模型。内核会自动跟踪 PRIV_AWARE 标志，此标志指示某个程序设计为可使用特权。请考虑不能识别特权的子进程。从父进程继承的所有特权均可在子进程的允许特权集合和有效特权集合中找到。如果子进程将 UID 设置为 0，则其可能不具有完全超级用户权限。进程的有效特权集合和允许特权集合会被限制为子进程限制特权集合中包含的特权。这样，可识别特权的进程的限制特权集合会限制不能识别特权的子进程的超级用户特权。

调试特权使用

Oracle Solaris 提供了用于调试特权问题的工具。ppriv 命令和 truss 命令可提供调试输出。有关示例，请参见 ppriv(1) 手册页。有关示例，请参见“排除权限问题” [91]。您也可以使用 dtrace 命令。有关更多信息，请参见 dtrace(1M) 手册页和《Oracle Solaris 11.2 Dynamic Tracing Guide 》。

特权指定

术语“privilege（特权）”通常表示权限有所增加。由于 Oracle Solaris 系统上的每个进程都使用某些权限运行，您可以通过删除特权减少某个进程的权限。在此发行版中，您还可以使用扩展特权策略删除缺省授予特定资源的特权之外的大多数特权。

为用户和进程指定特权

安全管理员负责指定特权。现有的权限配置文件包含已经指定给配置文件中的命令的特权。您之后可以将权限配置文件指定给一个角色或用户。

特权还可以直接指定给用户、角色或权限配置文件。如果您信任某些用户能够在其整个会话中会负责地使用某种特权，则可以直接指定该特权。适合直接指定的特权是具有有限影响的特权，如 `proc_clock_highres`。不适合直接指定的特权是具有广泛影响的特权，如 `file_dac_write`。有关更全面的论述，请参见“指定权限时的安全注意事项” [34]。

还可以拒绝用户、角色或进程行使某些特权。从用户或角色的初始可继承特权集合或限制特权集合中删除特权时必须小心谨慎。

扩展用户或角色的特权

用户和角色具有可继承特权集合。只能减少限制特权集合中的特权，因为限制特权集合最初包括所有特权。通过指定不在初始可继承特权集合中的特权，可以为用户、角色和进程扩展该特权集合。

您可以通过三种方法扩展可用的特权：

- 可以将不在初始可继承特权集合中、但是在限制特权集合中的特权指定给进程。可以进行间接指定（通过权限配置文件中的特权命令），也可以进行直接指定。
- 可显式将不属于可继承特权集合的特权指定给某个进程，例如为脚本或应用程序添加特权。
- 可显式将不属于可继承特权集合但属于限制特权集合的特权指定给网络端口、UID 或文件对象。特权的这种用法称为扩展特权策略，这同样是一种限制可用特权的方法。有关详细信息，请参见“使用扩展特权策略限制特权使用” [29]。

将特权只指定给需要该特权的管理任务是扩展用户或角色的特权的最精确方法。创建包含命令或脚本及其所需特权的权限配置文件。然后将此权限配置文件指定给用户或角色。通过指定，用户或角色可以运行特权命令。否则用户将无法获得此特权。

扩展用户或角色的初始可继承特权集合是一种不太可取的特权指定方法。可继承特权集合中的所有特权都位于允许特权集合和有效特权集合中。用户或角色在 shell 中键入的所有命令都可以使用直接指定的特权。有关更全面的论述，请参见“指定权限时的安全注意事项” [34]。

要降低不必要的特权的可用性，您可以将扩展特权指定给网络端口、UID 和文件对象。这种指定可以从有效特权集合中删除不属于扩展特权指定的特权。有关讨论，请参见“使用扩展特权策略限制特权使用” [29]。

限制用户或角色的特权

特权和权限配置文件还可以应用于不可信用户，以限制他们的权限。通过删除特权，您可以防止用户和角色执行特定任务。可以从初始可继承特权集合和限制特权集合中删除特权。在分配小于缺省特权集合的初始可继承特权集合或限制特权集合之前，应谨慎地测试特权删除操作。通过从初始可继承特权集合中删除特权，可以阻止用户登录。如果从限制特权集合中删除了特权，传统的 `setuid root` 程序可能会失败，因为该程序所需的特权已被删除。有关删除特权的示例，请参见例 3-21 “从用户的限制特权集合中删除特权” 和例 5-6 “创建 "Sun Ray Users"（Sun Ray 用户）权限配置文件”。

要限制用户 ID、端口或文件对象可用的特权，请参见“使用扩展特权策略限制特权使用” [29]。

为脚本指定特权

与命令类似，脚本是可执行文件。因此，在权限配置文件中，可以为脚本添加特权，就像为命令添加特权一样。当指定有权限配置文件的用户或角色在配置文件 shell 中执行脚本时，将会使用添加的特权来运行该脚本。如果脚本包含需要特权的命令，则具有已添加特权的命令也必须在指定的权限配置文件中。有关示例，请参见“向应用程序和脚本指定权限” [57]。

使用扩展特权策略限制特权使用

通过扩展特权策略，除了基本特权和您显式授予的特权之外，对端口、用户 ID 或文件对象的访问会受到限制。由于特权较少，难以轻易使用该资源攻击系统。事实上，用户可以防止可能的恶意进程访问所拥有的文件和目录。有关扩展特权策略的示例，请参见“将应用程序、脚本和资源限定于特定权限” [57]。

特权升级和用户权限

Oracle Solaris 为管理员提供了安全性配置的极大灵活性。安装后，软件会防止特权升级。用户或进程获取的管理权限多于本应授予的权限时，会发生特权升级。在这个意义上，“特权”意味着所有权限，不仅仅是内核特权。请参见“特权升级和内核特权” [30]。

Oracle Solaris 软件包括仅指定给 `root` 角色的权限。在实施了其他安全保护措施的情况下，管理员可以将为 `root` 角色设计的属性指定给其他帐户，但必须谨慎进行此类指定。

以下权限配置文件和授权集可以升级非 root 用户帐户的特权：

- **"Media Restore"（介质恢复）权限配置文件** – 此配置文件不是任何其他权限配置文件的一部分。因为 "Media Restore"（介质恢复）配置文件提供对整个根文件系统的

访问权限，因此使用它可能是一种特权升级。可以恢复故意更改的文件或替代介质。缺省情况下，root 角色包括此权限配置文件。

- solaris.*.assign 授权 – 这些授权未指定给任何权限配置文件。具有 solaris.*.assign 授权的帐户可以将帐户本身不具有的权限指定给其他帐户。例如，具有 solaris.profile.assign 授权的角色可以将角色本身不具有的权限配置文件指定给其他帐户。缺省情况下，只有 root 角色具有 solaris.*.assign 授权。

  请指定 solaris.*.delegate 授权，而不要指定 solaris.*.assign 授权。solaris.*.delegate 授权仅允许委托方将其拥有的权限指定给其他帐户。例如，指定有 solaris.profile.delegate 授权的角色可以将角色本身具有的权限配置文件指定给其他用户和角色。

有关防止内核特权升级的信息，请参见“特权升级和内核特权” [30]。

特权升级和内核特权

在内核防止 privilege escalation（特权升级）。为防止进程获取超出其应有特权的特权，内核检查会易受攻击的系统修改是否具有完全特权集合。例如，只有具有完全特权集合的进程才能更改 root (UID=0) 拥有的文件或进程。root 帐户不需要具有特权就能更改 root 拥有的文件。但是，非 root 用户必须具有全部特权才能更改 root 用户拥有的文件。

类似地，提供设备访问的操作需要具有有效特权集合中的所有特权。

具体来说，file_chown_self 和 proc_owner 特权可能发生特权升级。

- file_chown_self 特权允许进程放弃其文件。proc_owner 特权允许进程检查不归其拥有的进程。

  可通过 rstchown 系统变量限制 file_chown_self 特权。如果将 rstchown 变量设置为 0，则将从系统映像的所有用户的初始可继承特权集合中删除 file_chown_self 特权。有关 rstchown 系统变量的更多信息，请参见 chown(1) 手册页。

  将 file_chown_self 特权非常安全地指定给特定命令，将该命令放置在权限配置文件中，而将配置文件指定给角色或可信用户。
- proc_owner 特权不足以将进程 UID 切换为 0。要将进程从任何 UID 切换为 UID=0 需要具有全部特权。由于 proc_owner 特权授予对系统上所有文件的无限制读取权限，因此可以非常安全地将该特权指定给特定命令，将该命令放置在配置文件中，而将配置文件指定给角色。

注意 - 可以配置用户帐户，将 file_chown_self 特权或 proc_owner 特权包括在该用户的初始可继承特权集合中。但是，要将这些强大的特权放置在任何用户或角色的可继承特权集合中，您应有充分的安全理由。

有关如何防止设备特权升级的信息，请参见“特权和设备” [26]。有关一般讨论，请参见 privileges(5) 手册页。

权限验证

运行进程的 shell、命名服务的范围以及搜索的顺序会影响是否评估指定的权限。无法评估权限的进程将失败。有关检查权限指定的帮助，请参见“排除权限问题” [91]。

配置文件 Shell 和权限验证

用户和角色可以从配置文件 shell 运行特权应用程序。配置文件 *shell* 是识别权限的特殊 shell。管理员可以将配置文件 shell 指定为用户的登录 shell，或者在用户运行 `pfexec` 命令或 `su` 命令承担角色时启动配置文件 shell。在 Oracle Solaris 中，每个 shell 均具有对应的配置文件 shell。有关配置文件 shell 的列表，请参见 `pfexec(1)` 手册页。

直接指定有权限配置文件及其登录 shell 不是配置文件 shell 的用户必须打开配置文件 shell 才能运行所指定的特权命令。已指定有需要验证权限配置文件的用户和角色会收到进行验证的提示，也就是说需要在执行命令前提供口令。有关可使用性和安全性方面的注意事项，请参见“指定权限时的注意事项” [33]。

名称服务范围和权限验证

名称服务范围会影响所指定的权限在什么情况下可用。角色的范围可以限定为单个主机。或者，此范围也可以包括由某种名称服务（如 LDAP）提供服务的所有主机。在名称转换服务 `svc:/system/name-service/switch` 中指定系统的名称服务范围。遇到第一个匹配项时，查找便会停止。例如，如果某个权限配置文件存在于两个名称服务范围中，则只会使用第一个名称服务范围中的各项。如果 `files` 是第一个匹配项，则角色的范围将限定为本地主机。有关命名服务的信息，请参见`nsswitch.conf(4)` 手册页、《使用 Oracle Solaris 11.2 目录和命名服务：DNS 和 NIS》以及《使用 Oracle Solaris 11.2 目录和命名服务：LDAP》。

所指定权限的搜索顺序

可以为用户或角色直接指定或通过权限配置文件指定 security attributes（安全属性）。搜索顺序影响使用的安全属性值。将使用查找到的第一个属性实例的值。

注 - 授权的顺序不重要。授权是累积的。

在用户登录时，按照以下搜索顺序指定权限：

- 使用 useradd 和 usermod 命令直接指定给用户的**权限**。有关可能的权限指定的列表，请参见“user_attr 数据库” [102]。
- 使用 useradd 和 usermod 命令指定给用户的**权限配置文件**。按顺序对这些指定进行搜索。
  - 首先搜索需要验证权限配置文件。

    顺序是首先搜索需要验证配置文件列表中的第一个配置文件，然后是其补充配置文件，之后是需要验证配置文件列表中的第二个配置文件，然后是其补充配置文件，以此类推。系统使用第一个实例的值，但 auths 值除外，该值是累积的。可以指定给权限配置文件的属性包括可以指定给用户的所有权限以及补充配置文件。有关详细信息，请参见“user_attr 数据库” [102]。
  - 然后以同样方式搜索不需要重新验证的权限配置文件。
- **"Console User"（控制台用户）权限配置文件**值。有关说明，请参见“权限配置文件参考信息” [99]。
- 如果指定了 **"Stop"（停止）权限配置文件**，将停止对安全属性的评估。指定了 "Stop"（停止）配置文件之后不再指定任何属性。"Stop"（停止）配置文件在 "Console User"（控制台用户）权限配置文件之后、policy.conf 文件中的其他安全属性（包括 AUTHS_GRANTED）之前进行评估。有关说明，请参见“权限配置文件参考信息” [99]。
- policy.conf 文件中的 "Basic Solaris User"（基本 Solaris 用户）权限配置文件值。
- policy.conf 文件中的 AUTHS_GRANTED 值。
- policy.conf 文件中的 AUTH_PROFS_GRANTED 值。
- policy.conf 文件中的 PROFS_GRANTED 值。
- policy.conf 文件中的 PRIV_DEFAULT 值。
- policy.conf 文件中的 PRIV_LIMIT 值。

检查权限的应用程序

可以覆盖系统控制的应用程序和命令被视为特权应用程序。使用安全属性（如 UID=0）、特权和授权，可以将应用程序成为特权应用程序。

检查 UID 和 GID 的应用程序

检查 root（UID=0）或其他某个特殊 UID 或 GID 的特权应用程序在 UNIX 环境中已经存在很久。通过权限配置文件机制，可以隔离需要特定 ID 的命令。您不必更改任何人都可访问的命令的 ID，而是可以将具有指定 UID 的命令放置在某个权限配置文件中。这样，拥有此权限配置文件的用户或角色不必成为超级用户便可作为该 UID 运行程序。

可以将 ID 指定为实际 *ID* 或有效 *ID*。指定有效 ID 优先于指定实际 ID。有效 ID 等效于文件权限位中的 setuid 功能，该 ID 还可以标识 UID 以进行审计。但是，由于某些

shell 脚本和程序需要 root 的实际 UID，因此也可设置实际 UID。例如，reboot 命令需要实际 UID 而不是有效 UID。

提示 - 如果使用有效 ID 不足以运行命令，将此实际 ID 指定给该命令。

检查特权的应用程序

特权应用程序可以检查特权的使用。通过权限配置文件机制，可以为需要安全属性的特定命令指定特权。然后，可以在权限配置文件中隔离具有指定安全属性的命令。这样，拥有此权限配置文件的用户或角色便可仅使用命令所需的特权来运行此命令。

检查特权的命令包括：

- Kerberos 命令，如 kadmin、kprop 和 kdb5_util
- 网络命令，如 ipadm、routeadm 和 snoop
- 文件和文件系统命令，如 chmod、chgrp 和 mount
- 控制进程的命令，如 kill、pcred 和 rcapadm

要将具有特权的命令添加到权限配置文件中，请参见如何创建权限配置文件 [74]和 profiles(1) 手册页。要确定哪些命令可检查特定配置文件中的特权，请参见第 6 章 列出 Oracle Solaris 中的权限。

检查授权的应用程序

部分 Oracle Solaris 命令检查授权，包括以下命令：

- 审计管理命令，如 auditconfig 和 auditreduce
- 打印机管理命令，如 cupsenable 和 lpadmin
- 批处理作业命令，如 at、atq、batch 和 crontab。
- 面向设备的命令，如 allocate、deallocate、list_devices 和 cdrw。

有关用于授权的脚本或程序的指南，请参见例 4-3 “检查脚本或程序中的授权”。要编写需要授权的程序，请参见《面向开发者的 Oracle Solaris 11 安全性指南 》中的“关于授权”。

指定权限时的注意事项

安全性和可使用性问题会影响管理员指定权限的方式。

指定权限时的安全注意事项

通常，用户或角色可以通过权限配置文件获得管理权限，但是也可以为其直接指定权限。

- 可以直接将特权指定给用户和角色。

  直接指定特权的做法并不安全。具有直接指定的特权的用户和角色在每次内核需要该特权时会覆盖安全策略。同样，只要内核需要该特权，入侵用户或角色进程的恶意进程就可以使用此特权。

  更安全的做法是在权限配置文件中将特权指定为某个命令的安全属性。这样，该特权只可由拥有此权限配置文件的用户用于该命令。
- 可以直接将授权指定给用户和角色。

  由于授权在用户级别进行评估，因此，与直接指定特权相比，直接指定授权的危险性小一些。但是，用户可以使用授权来执行安全级别很高的任务，如指定审计标志。为提高安全性，请在需要验证权限配置文件中指定授权，对于此类配置文件，用户必须提供口令，然后才能执行命令。

指定权限时的可使用性注意事项

直接指定权限会影响可使用性。

- 直接指定的授权以及用户权限配置文件中的命令和授权必须由配置文件 shell 解释才能生效。缺省情况下，未将配置文件 shell 指定给用户。因此，用户必须记得打开某个配置文件 shell，然后在该 shell 中执行命令。
- 单独指定授权是不可伸缩的。而且，直接指定的授权可能不足以执行某个任务。该任务可能需要特权命令。

  权限配置文件用于将授权和特权命令捆绑在一起。权限配置文件还可以顺利扩展到用户组。

♦ ♦ ♦ 第 2 章

2 规划管理权限配置

本章提供的信息可帮助您确定在管理系统时使用传统权限模型还是充分利用 Oracle Solaris 权限模型。本章涵盖以下主题：

- “确定用于管理的权限模型” [35]
- “遵循选择的权限模型” [36]

有关权限的概述，请参见“用户权限管理” [14]。有关参考信息，请参见第 8 章 Oracle Solaris 权限参考信息。

确定用于管理的权限模型

Oracle Solaris 中的权限包括权限配置文件、授权和特权。Oracle Solaris 提供了在系统上配置管理权限的多种方法。

以下列表按照最安全到不太安全的传统 superuser model（超级用户模型）的顺序排序。

1. 将管理任务划分给多个 trusted users（可信用户），每个用户具有有限权限。此方法为 Oracle Solaris 权限模型。

   有关如何执行此方法的信息，请参见“遵循选择的权限模型” [36]。

   有关此方法的优势的介绍，请参见第 1 章 使用权限控制用户和进程。
2. 使用缺省权限配置。此方法使用权限模型，但不根据您的站点对其进行定制。

   缺省情况下，初始用户具有一些管理权限，并且可以承担 `root` 角色。`root` 角色还可以将 `root` 角色指定给其他可信用户。要提高安全性，`root` 角色需要启用管理命令审计。

   对于使用此模型的管理员，以下任务非常有用：

   - “使用所指定的管理权限” [70]
   - “为用户指定权限” [39]
   - “审计管理操作” [73]
   - “更改角色口令” [45]
   - 第 6 章 列出 Oracle Solaris 中的权限
3. 使用 `sudo` 命令。

熟悉 sudo 命令的管理员可以配置并使用 sudo。还可以配置 /etc/sudoers 文件，以便 sudo 用户能够在一定时间段内运行管理命令而无需重新验证。

对于 sudo 用户，以下任务非常有用：

- “使用所指定的管理权限” [70]
- “审计管理操作” [73]
- 缓存验证 – 例 5-2 “缓存验证以便简化角色使用”

sudo 命令挂接到内核的方式与权限配置文件不同。该命令以具有所有特权的 root 用户运行，因此可以授予在当前用户的 /etc/sudoers 文件中为每个程序指定的权限。尽管 sudo 无法指定程序的后续子进程的属性，但它可以阻塞这些进程的执行。Oracle Solaris 版本的 sudo 从进程中删除了 PRIV_PROC_EXEC 特权。有关更多信息，请参见 Oracle Solaris 版本的 sudo(1M) 手册页。

4. 通过将 root 角色更改为用户来使用超级用户模型。

   使用传统 UNIX 模型的管理员必须完成如何将 root 角色更改为用户 [80]。root 用户还可以配置审计（可选）。

遵循选择的权限模型

用户和进程权限管理是系统部署管理密不可分的一部分。进行规划时，需要全面了解组织的安全要求并了解 Oracle Solaris 中的权限。本节介绍了规划站点的权限使用的一般过程。

1. 了解权限的基本概念。

   阅读第 1 章 使用权限控制用户和进程。使用权限来管理系统与使用常规的 UNIX 管理做法大不相同。

2. 检查安全策略。

   您组织的安全策略应详细说明系统面临的潜在威胁，衡量每种威胁的风险并提供应对这些威胁的策略。通过权限来分离与安全相关的任务可以作为该策略的一部分。

   例如，站点可能要求将安全管理与非安全管理分开。要实现职责分离，请参见例 3-3 “创建角色以实现职责分离”。

   如果安全策略依赖基于 RBAC 管理的授权规则 (Authorization Rules Managed On RBAC, ARMOR)，则必须使用 ARMOR 软件包。有关其在 Oracle Solaris 中的使用，请参见例 3-1 “使用 ARMOR 角色”。

3. 查看缺省权限配置文件。

   缺省权限配置文件收集了完成任务所需的权限。要查看可用的权限配置文件，请参见“列出权限配置文件” [84]

4. 确定要使用角色还是将权限配置文件直接指定给用户。

   角色可简化权限的管理。角色名称标识该角色可以执行的任务并将角色权限与用户权限隔离。如果要使用角色，您有三个选择：

- 可以安装 ARMOR 软件包，该软件包将安装基于 RBAC 管理的授权角色 (Authorization Roles Managed on RBAC, ARMOR) 标准定义的七个角色。请参见例 3-1 “使用 ARMOR 角色”。
- 可以定义自己的角色，并同时使用 ARMOR 角色。请参见“创建角色” [40]和例 3-1 “使用 ARMOR 角色”。
- 可以定义自己的角色而不使用 ARMOR 角色。请参见“创建角色” [40]。

如果站点不需要使用角色，则可以直接将权限配置文件指定给用户。如果需要用户在从其权限配置文件执行管理任务时提供口令，请使用需要验证权限配置文件。请参见例 3-11 “要求用户在管理 DHCP 前键入口令”。

5. 确定是否需要创建其他权限配置文件。

   请在站点上查找可能从受限制访问中受益的其他应用程序或应用程序系列。适合使用权限的应用程序包括：影响安全的应用程序、可能导致拒绝服务问题的应用程序，或需要对管理员进行特殊培训的应用程序。例如，Sun Ray 系统的用户不需要所有基本特权。有关限制用户的权限配置文件的示例，请参见例 3-22 “从权限配置文件中删除基本特权”。

   a. 确定新任务所需的权限。
   b. 确定现有权限配置文件是否适用于此任务。
   c. 对权限配置文件进行排序，以便命令能够使用所需的特权执行。

      有关排序的信息，请参见“所指定权限的搜索顺序” [31]。

6. 确定将哪些权限指定给哪些用户。

   根据 principle of least privilege（最小特权原则），应向用户指定适合该用户信任级别的角色。如果禁止用户执行用户无需执行的任务，可以减少潜在的问题。

注 - `/etc/security/policy.conf` 文件中指定了应用于系统映像的所有用户的权限。

制定好规划后，为可以向其指定权限配置文件或角色的 trusted users（可信用户）创建登录帐户。有关创建用户的详细信息，请参见《在 Oracle Solaris 11.2 中管理用户帐户和用户环境 》中的“使用 CLI 设置和管理用户帐户的任务列表”。

要指定权限，请从“为用户指定权限” [39]中的过程开始。以下各节提供了扩展权限、限制权限、向资源指定权限以及排除权限指定问题的示例。

♦ ♦ ♦ 第 3 章

3 在 Oracle Solaris 中指定权限

本章说明为用户和角色指定权限的任务。本章涵盖以下主题：

- “为用户指定权限” [39]
- “扩展用户权限” [46]
- “限制用户的权限” [51]

有关权限的概述，请参见“用户权限管理” [14]。有关参考信息，请参见第 8 章 Oracle Solaris 权限参考信息。

为用户指定权限

Oracle Solaris 中的每个进程都存在权限。您可以将权限添加到用户和角色，也可以删除权限。权限包括用户进程的特权、用户运行命令所用的特权或特殊 ID 以及执行特定操作的授权。为减轻管理员指定权限的负担，Oracle Solaris 将服务和管理活动的权限收集到权限配置文件中。您无需将单个权限指定给用户或角色，而是可以指定包含管理任务需要的所有授权和特权的权限配置文件。

角色为用户可以执行的管理任务指定了名称，例如 `auditadm`。要执行管理操作，用户需承担指定的角色来执行该操作。安全策略可能需要角色，也可为方便使用而定义角色。您可以创建角色，也可以安装 `armor` 软件包，它可以创建 7 个角色以及相应的本地起始目录。有关角色的详细信息，请参见“用户和进程权限提供了一种替代超级用户模型的方法” [14]。

权限指定者

最初，您必须为 `root` 角色才能创建添加了权限的用户。

如果 `root` 角色已经将管理任务分配给您（将您作为可信用户或者通过为您指定角色），以下权限配置文件指定可用于创建用户和角色或为其指定权限：

- 要创建用户或角色，您必须成为指定有 "User Management"（用户管理）权限配置文件的管理员。

- 要为用户或角色指定大多数权限，您必须成为指定有 "User Security"（用户安全）权限配置文件的管理员。

  您无法指定审计标志。只有 root 可以为用户或角色指定审计标志。

  您无法更改角色的口令。只有 root 角色可以更改角色的口令。

如果为您指定了管理权限，在尝试运行管理命令之前请查看“使用所指定的管理权限” [70]。

为用户和角色指定权限

本节说明了创建和修改角色和用户的命令。要创建或修改权限配置文件，请参见如何创建权限配置文件 [74]和如何克隆和修改系统权限配置文件 [76]。

有关角色的信息，请参见“用户和进程权限基础” [16]。

创建和修改角色和用户的主要操作如下所示：

- 创建角色
- 创建可信用户
- 修改角色的权限
- 修改用户的权限
- 允许用户使用自己的口令承担角色。
- 更改角色口令
- 删除角色

创建角色

如果您要使用角色，有多个方案可以选择。您可以通过 ARMOR 安装预定义角色并以独占方式使用这些角色。还可以创建角色并为其指定口令。您可以将 ARMOR 角色与您创建的角色结合使用。

要使用 ARMOR 角色，请参见例 3-1 “使用 ARMOR 角色”。

要创建自己的角色，请使用 roleadd 命令。有关此命令的参数的完整列表，请参见 roleadd(1M) 手册页。

例如，以下命令创建 "User Administrator"（用户管理员）本地角色以及起始目录和 pfbash 登录 shell，并为该角色创建一个口令：

```
# roleadd -c "User Administrator role, local" \
-m -K profiles="User Security,User Management"  useradm
80 blocks
# ls /export/home/useradm
local.bash_profile     local.login     local.profile
```

```
# passwd useradm
Password: xxxxxxxx
Confirm Password: xxxxxxxx
```

其中，

-c *comment*	描述角色。
-m	创建起始目录。
-K profiles=	为该角色指定一个或多个权限配置文件。有关权限配置文件的列表，请参见“列出权限配置文件” [84]。
rolename	角色的名称。有关可接受字符串的限制，请参见 roleadd(1M) 手册页。

注 - 可以将一个角色帐户指定给多个用户。因此，管理员通常都会创建一个角色口令，并单独为用户提供角色口令。有关角色口令的替代方式，请参见“使用户可以将自己的口令用作角色口令” [45]、例 3-16 “使用户可以将自己的口令用作角色口令” 和例 3-17 “修改权限配置文件以使用户将自己的口令用作角色口令”。

例 3-1 使用 ARMOR 角色

在此示例中，安全管理员安装 ARMOR 标准定义的角色。管理员首先确认角色名称与任何现有帐户都不冲突，然后安装该软件包、查看角色定义并为可信用户指定角色。

首先，管理员确保命名服务中不存在以下 UID 和名称：

- 57 auditadm
- 55 fsadm
- 58 pkgadm
- 53 secadm
- 56 svcadm
- 59 sysop
- 54 useradm

确认这些 UID 和名称未被使用后，管理员可以安装此软件包。

```
# pkg install system/security/armor
```

该软件包在 /export/home 目录中创建 7 个角色和本地起始目录。

要查看每个角色的权限，管理员可以列出指定给每个角色的配置文件。

```
# profiles auditadm
# profiles fsadm
```

```
# profiles pkgadm
# profiles secadm
# profiles svcadm
# profiles sysop
# profiles useradm
```

无法修改这些权限指定。要创建不同的权限配置，您必须创建新角色，然后按照如何克隆和修改系统权限配置文件 [76]中的步骤创建新权限配置文件。

最后，管理员为可信用户指定角色。用户自己的口令可用于对角色进行验证。为部分用户指定了多个角色。任务时效性强的角色将指定给多个可信用户。

```
# usermod -R=auditadm adal
# usermod -R=fsadm,pkgadm bdewey
# usermod -R=secadm,useradm cfoure
# usermod -R=svcadm ghamada
# usermod -R=svcadm yjones
# usermod -R=sysop hmurtha
# usermod -R=sysop twong
```

例 3-2 在 LDAP 系统信息库中创建 "User Administrator"（用户管理员）角色

管理员在 LDAP 中创建了 "User Administrator"（用户管理员）角色。承担该角色时用户需要提供口令，然后不需要为单个命令提供口令。

```
# roleadd -c "User Administrator role, LDAP" -m -S ldap \
-K profiles="User Security,User Management"  useradm
```

例 3-3 创建角色以实现职责分离

管理员创建了两个角色。`usermgt` 角色可以创建用户，为其指定起始目录以及执行其他非安全任务。`usersec` 角色无法创建用户，但是可以指定口令，并更改其他权限分配。这两个角色都无法为用户或角色设置审计标志，也无法更改角色的口令。必须由 `root` 角色执行这些操作。

```
# roleadd -c "User Management role, LDAP" -s /usr/bin/pfksh \
-m -S ldap -K profiles="User Management"  usermgt
# roleadd -c "User Security role, LDAP" -s /usr/bin/pfksh \
-m -S ldap -K profiles="User Security"  usersec
```

管理员确保创建例 3-5 “为用户添加角色”中的每个一般用户都需要这两个角色。

例 3-4 为管理加密服务创建并指定角色

在以下示例中，LDAP 网络的管理员会创建一个角色以管理加密框架，并将该角色指定给 UID 1111。

```
# roleadd -c "Cryptographic Services manager" \
-g 14 -m -u 104 -S ldap -K profiles="Crypto Management" cryptmgt
# passwd cryptmgt
New Password: xxxxxxxx
Confirm password: xxxxxxxx
# usermod -u 1111 -R +cryptmgt
```

具有 UID 1111 的用户将登录，然后承担该角色并显示指定的权限。

```
% su - cryptmgt
Password: xxxxxxxx
# profiles -l
      Crypto Management
          /usr/bin/kmfcfg            euid=0
          /usr/sbin/cryptoadm        euid=0
          /usr/sfw/bin/CA.pl         euid=0
          /usr/sfw/bin/openssl       euid=0
#
```

有关加密框架的更多信息，请参见《在 Oracle Solaris 11.2 中管理加密和证书 》中的第 1 章 “加密框架”。要管理该框架，请参见《在 Oracle Solaris 11.2 中管理加密和证书 》中的“管理加密框架”。

为可信用户创建登录帐户

使用 useradd 命令创建登录帐户。有关 useradd 命令的参数的完整列表，请参见 useradd(1M) 手册页。该命令的与权限相关的参数与 roleadd 命令类似，但增加了 -R *rolename* 选项。

如果您为用户指定一个角色，该用户在承担该角色后可以使用该角色的权限。例如，以下命令创建一个可信用户，该用户可以承担您在“为可信用户创建登录帐户” [43]中创建的 useradm 角色。

```
# useradd -c "Trusted Assistant User Manager user" -m -R useradm jdoe
80 blocks
# ls /export/home/jdoe
local.bash_profile     local.login     local.profile
```

其中，

-s *shell*
: 确定 *username* 的登录 shell。此 shell 可以是配置文件 shell，例如 pfbash。有关为可信用户指定配置文件 shell 的原因，请参见“指定权限时的可使用性注意事项” [34]。有关配置文件 shell 的列表，请参见 pfexec(1) 手册页。

-R *rolename*
: 指定一个现有角色的名称。

有关更多示例，请参见《在 Oracle Solaris 11.2 中管理用户帐户和用户环境 》中的“使用 CLI 设置和管理用户帐户的任务列表”。

修改用户权限

您可以使用 `usermod` 命令修改用户帐户。有关 `usermod` 命令的参数的完整列表，请参见 `usermod(1M)` 手册页。该命令的与权限相关的参数与 `useradd` 命令类似。

如果您为用户指定一个权限配置文件，该用户可以在打开配置文件 shell 后使用这些权限。例如，为用户指定权限配置文件。

```
# usermod -K profiles="User Management" kdoe
```

此更改将在用户下次登录时生效。用户要了解如何使用所指定的权限，请参阅“使用所指定的管理权限” [70]。

例 3-5 为用户添加角色

在此示例中，管理员确保创建一般用户需要两个可信用户。这些角色是在例 3-3 “创建角色以实现职责分离” 中创建的。

```
# usermod -R +useradm jdoe
# usermod -R +usersec mdoe
```

修改角色的权限

使用 `rolemod` 命令修改角色帐户。有关 `rolemod` 命令的参数的完整列表，请参见 `rolemod(1M)` 手册页。该命令的与权限相关的参数与 `roleadd` 命令类似。

可以通过减号 (-) 或加号 (+) 修改 *key*=*value* 对的值以及 `-A`、`-P` 和 `-R` 选项。- 号表示从当前指定的值中减去该值。+ 号表示将该值加到当前指定的值。对于权限配置文件，该值追加到当前配置文件列表之前。对于成为靠前的权限配置文件的影响，请参见“所指定权限的搜索顺序” [31]。

例 3-6 将某个权限配置文件添加为角色的第一个权限配置文件

例如，为 `useradm` 角色追加一个权限配置文件。

```
# rolemod -K profiles+="Device Management" useradm
# profiles useradm
```

```
useradm:
Device Management
User Management
User Security
```

例 3-7 替换本地角色的指定配置文件

在以下示例中，管理员将修改 prtmgt 角色，在 "Printer Management"（打印机管理）配置文件后包括 "VSCAN Management"（VSCAN 管理）权限配置文件。

```
# rolemod -c "Handles printers and virus scanning" \
-P "Printer Management,VSCAN Management,All" prtmgt
```

例 3-8 将特权直接指定给角色

在以下示例中，安全管理员向 realtime 角色授予一个非常特定的特权，该特权可以影响系统时间。要将特权指定给用户，请参见例 3-14 “将特权直接指定给用户”。

```
# rolemod -K defaultpriv+='proc_clock_highres' realtime
```

defaultpriv 关键字的值始终都位于角色进程的特权列表中。

使用户可以将自己的口令用作角色口令

要使用户可以在承担角色时使用自己的口令，而不是使用角色口令，请修改该角色。

以下命令使所有指定了 useradm 角色的用户可以在承担任何指定角色（包括 useradm 角色）时使用自己的口令。

```
# rolemod -K roleauth=user useradm
```

更改角色口令

因为一个角色可能会被指定给许多用户，所以承担角色的用户无法更改角色的口令。您必须承担 root 角色才能更改角色口令。

```
# passwd useradm
Enter useradm's password: xxxxxxxx
New: xxxxxxxx
Confirm: xxxxxxxx
```

如果没有指定系统信息库，所有系统信息库中的口令都将发生更改。

有关更多命令选项，请参见 passwd(1) 手册页。

例 3-9 更改特定系统信息库中的角色口令

在以下示例中，root 角色更改了本地 devadmin 角色的口令。

```
# passwd -r files  devadmin
New password: xxxxxxxx
Confirm password: xxxxxxxx
```

在以下示例中，root 角色更改了 LDAP 命名服务中 devadmin 角色的口令。

```
# passwd -r ldap devadmin
New password: xxxxxxxx
Confirm password: xxxxxxxx
```

删除角色

删除角色后，该角色立刻无法使用。

```
# roledel useradm
```

当前以该角色执行管理任务的用户无法继续执行操作。profiles 命令显示以下输出结果：

```
useradm # profiles
Unable to get user name
```

扩展用户权限

本节中的任务和示例向用户缺省接收的权限添加权限。有关权限的信息，请参见第 1 章 使用权限控制用户和进程。

- 为可信用户指定角色 – 例 3-1 “使用 ARMOR 角色”、例 3-4 “为管理加密服务创建并指定角色”、例 3-5 “为用户添加角色”
- 为可信用户指定权限配置文件 – 例 3-10 “创建可以管理 DHCP 的用户”、例 3-19 “使可信用户可以读取扩展记帐文件”、例 4-1 “将安全属性指定给传统应用程序”
- 为可信用户指定需要验证权限配置文件 – 例 3-11 “要求用户在管理 DHCP 前键入口令”、例 4-2 “使用指定的权限运行应用程序”
- 为可信用户或角色指定授权 – 例 3-12 “将授权直接指定给用户”、例 3-13 “将授权指定给角色”
- 为用户或角色直接指定特权 – 例 3-8 “将特权直接指定给角色”、例 3-14 “将特权直接指定给用户”、例 3-15 “添加到角色的基本特权”

注意 - 不当使用直接指定的特权和授权可能导致无意中破坏安全性。有关讨论，请参见“指定权限时的安全注意事项” [34]。

- 使用户可以在承担某个角色时使用自己的口令 – 例 3-16 “使用户可以将自己的口令用作角色口令”、例 3-17 “修改权限配置文件以使用户将自己的口令用作角色口令”
- 修改权限配置文件 – 例 3-22 “从权限配置文件中删除基本特权”
- 将安全属性添加到权限配置文件中的命令 – 例 3-26 “防止所选应用程序大量生成新进程”、例 3-27 “防止来宾大量生成新子进程”、例 5-7 “创建包含特权命令的权限配置文件”
- 使用户可以读取归 `root` 所有的文件 – 例 3-19 “使可信用户可以读取扩展记帐文件”、例 3-20 “使非 `root` 帐户可以读取 `root` 所有的文件”
- 使用户或角色可以编辑归 `root` 所有的文件 – 例 5-9 “从权限配置文件中克隆和删除所选权限”
- 指定包含新授权的权限配置文件 – 例 5-11 “向权限配置文件中添加授权”

例 3-10 创建可以管理 DHCP 的用户

安全管理员创建可以管理 DHCP 的用户。

```
# useradd -P "DHCP Management" -s /usr/bin/pfbash -S ldap  jdoe
```

由于已为用户指定了 `pfbash` 作为登录 shell，将始终评估 "DHCP Management"（DHCP 管理）权限配置文件中的权限，因此 DHCP 管理命令可以成功运行。

例 3-11 要求用户在管理 DHCP 前键入口令

在此示例中，安全管理员要求 `jdoe` 提供口令，然后才能管理 DHCP。

```
# usermod -K auth_profiles="DHCP Management" profiles="Edit Administrative Files" jdoe
```

`jdoe` 键入 DHCP 命令时，将出现输入口令提示。在验证 `jdoe` 后，DHCP 命令将完成。对于搜索顺序，需要验证权限配置文件在一般配置文件之前处理。

```
jdoe% dhcpconfig -R 120.30.33.7,120.30.42.132
Password: xxxxxxxx
    /** Command completes **/
```

例 3-12 将授权直接指定给用户

在本示例中，安全管理员将创建一个可以控制屏幕亮度的本地用户。

```
# useradd -c "Screened KDoe, local" -s /usr/bin/pfbash \
-A solaris.system.power.brightness  kdoe
```

该授权将添加到用户的现有授权指定中。

例 3-13 将授权指定给角色

在此示例中，安全管理员创建一个角色，该角色可以更改 DNS 服务器服务的配置信息。

```
# roleadd -c "DNS administrator role" -m -A solaris.smf.manage.bind" dnsadmin
```

例 3-14 将特权直接指定给用户

在本示例中，安全管理员会向 kdoe 用户授予一个非常特殊的特权，该特权可以影响系统时间。要向角色直接指定特权，请参见例 3-8 “将特权直接指定给角色”。

```
# usermod -K defaultpriv='basic,proc_clock_highres' kdoe
```

defaultpriv 关键字的值将替换现有的值。因此，对于要保留 basic 特权的用户，指定了值 basic。在缺省配置中，所有用户都拥有基本特权。有关基本特权的列表，请参见“列出特权” [87]。

该用户可以查看添加的特权及其定义。

```
kdoe% ppriv -v $$
1800:   pfksh
flags = <none>
        E: file_link_any,…,proc_clock_highres,sys_ib_info
        I: file_link_any,…,proc_clock_highres,sys_ib_info
        P: file_link_any,…,proc_clock_highres,sys_ib_info
        L: cpc_cpu,dtrace_kernel,dtrace_proc,dtrace_user,...,win_upgrade_sl
% ppriv -vl proc_clock_highres
        Allows a process to use high resolution timers.
```

例 3-15 添加到角色的基本特权

在以下示例中，已经直接将处理日期和时间程序的特权指定给角色 realtime。例 3-8 “将特权直接指定给角色” 中将 proc_clock_highres 指定给 realtime。

```
# rolemod -K defaultpriv='basic,sys_time' realtime

% su - realtime
Password: xxxxxxxx
# ppriv -v $$
```

```
1600:   pfksh
flags = <none>
        E: file_link_any,...,proc_clock_highres,sys_ib_info,sys_time
        I: file_link_any,...,proc_clock_highres,sys_ib_info,sys_time
        P: file_link_any,...,proc_clock_highres,sys_ib_info,sys_time
        L: cpc_cpu,dtrace_kernel,dtrace_proc,dtrace_user,...,sys_time
```

例 3-16 使用户可以将自己的口令用作角色口令

缺省情况下，用户只有键入角色的口令才能承担角色。通过要求提供用户口令，管理员可使在 Oracle Solaris 中承担角色的过程类似于在 Linux 环境中承担角色的过程。

```
# rolemod -K roleauth=user auditrev
```

要承担该角色，指定的用户现在可以使用自己的口令，而非专为该角色创建的口令。

如果已为用户指定了其他角色，则用户的口令也可以验证这些角色。

例 3-17 修改权限配置文件以使用户将自己的口令用作角色口令

```
# profiles -p "Local System Administrator"
profiles:Local System Administrator> set roleauth="user"
profiles:Local System Administrator> end
profiles:Local System Administrator> exit
```

当指定了 "Local System Administrator"（本地系统管理员）权限配置文件的用户需要承担该角色时，系统将提示用户输入口令。在以下序列中，角色名称为 `admin`：

```
% su - admin
Password: xxxxxxxx
#       /** You are now in a profile shell with administrative rights**/
```

例 3-18 更改 LDAP 系统信息库中角色的 `roleauth` 值

在以下示例中，`root` 角色允许可以承担角色 `secadmin` 的所有用户在承担角色时使用自己的口令。对于由 LDAP 服务器管理的所有系统上的这些用户，都授予了该能力。

```
# rolemod -S ldap -K roleauth=user secadmin
```

例 3-19 使可信用户可以读取扩展记帐文件

您可以使可信用户或用户组读取 `root` 帐户所有的文件。如果用户可以运行包括 `root` 所有的文件的管理应用程序，此权限会很有用。此示例为 "Extended Accounting Net Management"（扩展记帐网络管理）权限配置文件添加一个或多个 Perl 脚本。

承担 root 角色后，管理员创建了一个权限配置文件，该配置文件添加了读取名称以 network 开头的记帐文件的能力。

以下配置文件使用扩展特权策略，将 file_dac_read 特权授予一个脚本，该脚本然后只可以访问 /var/adm/exacct/network* 文件。此配置文件将现有的 "Extended Accounting Net Management"（扩展记帐网络管理）权限配置文件添加为补充配置文件。

```
# profiles -p "Extended Accounting Perl Scripts"
profiles:Extended Accounting Perl Scripts >
set desc="Perl Scripts for Extended Accounting"
... Scripts> add profiles="Extended Accounting Net Management"
... Scripts> add cmd=/usr/local/bin/exacctdisp.pl
... Scripts:exacctdisp.pl> set privs={file_dac_read}:/var/adm/exacct/network*
... Scripts:exacctdisp.pl> end
... Scripts> commit
... Scripts> exit
```

有关示例脚本，请参见《在 Oracle Solaris 11.2 中进行资源管理 》中的“使用 libexacct 的 Perl 接口”。

在检查权限配置文件条目有无错误（如拼写错误、缺失或重复）后，管理员将 "Extended Accounting Perl Scripts"（扩展记帐 Perl 脚本）权限配置文件指定给角色或用户。

```
# profiles -p "Extended Accounting Perl Scripts" info
Found profile in files repository.
name=Extended Accounting Perl Scripts
desc=Perl Scripts for Extended Accounting
profiles=Extended Accounting Net Management
cmd=/usr/local/bin/exacctdisp.pl
privs={file_dac_read}:/var/adm/exacct/network*

# rolemod -K profiles+="Extended Accounting Perl Scripts" rolename
# usermod -K profiles+="Extended Accounting Perl Scripts" username
```

例 3-20 使非 root 帐户可以读取 root 所有的文件

在此示例中，管理员创建了一个权限配置文件，该配置文件使用扩展特权策略使授权用户和角色可以读取 root 所有的 /var/adm/sulog 文件。管理员添加了用户可以用于读取该文件的命令。无法使用未列出的命令，例如 head 命令。

```
# profiles -p "Read sulog File"
profiles:Read sulog File
set desc="Read sulog File"
... File> add profiles="Read Log Files"
... File> add cmd=/usr/bin/cat
... File:cat> set privs={file_dac_read}:/var/adm/sulog
... File:cat> end
... File> add cmd=/usr/bin/less
```

```
... File:less> set privs={file_dac_read}:/var/adm/sulog
... File:less> end
... File> add cmd=/usr/bin/more
... File:more> set privs={file_dac_read}:/var/adm/sulog
... File:more> end
... File> add cmd=/usr/bin/page
... File:page> set privs={file_dac_read}:/var/adm/sulog
... File:page> end
... File> add cmd=/usr/bin/tail
... File:tail> set privs={file_dac_read}:/var/adm/sulog
... File:tail> end
... File> add cmd=/usr/bin/view
... File:head> set privs={file_dac_read}:/var/adm/sulog
... File:head> end
... File> commit
... File> exit
```

`view` 命令使用户可以读取文件但是无法编辑。

限制用户的权限

本节中的示例限制了一般用户的权限，或者删除了管理员的部分管理权限。说明了如何修改用户、角色和权限配置文件。有关权限的信息，请参见第 1 章 使用权限控制用户和进程。

- 从用户删除限制特权 – 例 3-21 “从用户的限制特权集合中删除特权”
- 从用户删除基本特权 – 例 3-22 “从权限配置文件中删除基本特权”
- 从归您所有的 shell 进程删除基本特权 – 例 3-23 “从自身删除基本特权”
- 创建使用受限的系统 – 例 3-24 “修改系统以使权限仅对其用户可用”
- 限制管理员使用显式指定的权限 – 例 3-25 “限制管理员使用显式指定的权限”
- 从系统的所有用户删除权限 – 例 3-24 “修改系统以使权限仅对其用户可用”、例 3-28 “将 "Editor Restrictions"（编辑器限制）权限配置文件指定给所有用户”
- 防止应用程序创建子进程 – 例 3-26 “防止所选应用程序大量生成新进程”
- 防止用户大量生成子进程例 3-27 “防止来宾大量生成新子进程”
- 为来宾创建受限编辑器 – 例 3-27 “防止来宾大量生成新子进程”
- 将受限编辑器指定给一个公共系统 – 例 3-28 “将 "Editor Restrictions"（编辑器限制）权限配置文件指定给所有用户”
- 从权限配置文件的限制特权集合中删除特权 – 例 5-6 “创建 "Sun Ray Users"（Sun Ray 用户）权限配置文件”
- 为 Sun Ray 用户创建权限配置文件 – 例 5-6 “创建 "Sun Ray Users"（Sun Ray 用户）权限配置文件”
- 从权限配置文件删除权限 – 例 5-6 “创建 "Sun Ray Users"（Sun Ray 用户）权限配置文件”、例 5-9 “从权限配置文件中克隆和删除所选权限”

- 从用户删除授权 – 例 5-10 “测试新授权”
- 删除角色指定 – 例 5-13 “防止 root 角色用于维护系统”

例 3-21 从用户的限制特权集合中删除特权

在以下示例中，防止所有源自 jdoe 初始登录的会话使用 sys_linkdir 特权。即使在用户运行 su 命令之后，仍然不能生成指向目录的硬链接，并且也不能解除目录链接。

```
# usermod -K 'limitpriv=all,!sys_linkdir' jdoe
# userattr limitpriv jdoe
all,!sys_linkdir
```

例 3-22 从权限配置文件中删除基本特权

在以下示例中，在详尽测试后，安全管理员从 "Sun Ray User"（Sun Ray 用户）权限配置文件中删除了另一个基本特权。管理员在例 5-6 “创建 "Sun Ray Users"（Sun Ray 用户）权限配置文件” 中创建配置文件时，从限制特权集合删除了一个特权。此时，管理员删除了两个基本特权。指定有此配置文件的用户无法检查其当前会话外部的任何进程，也无法添加其他会话。

```
# profiles -p "Sun Ray Users"
profiles:Sun Ray Users> set defaultpriv="basic,!proc_session,!proc_info"
profiles:Sun Ray Users> end
profiles:Sun Ray Users> exit
```

例 3-23 从自身删除基本特权

在以下示例中，一般用户修改 .bash_profile 以删除 proc_info 基本特权。ps 和 prstat 这类程序的输出仅包含用户自己的进程，可以突出显示有用信息。

```
##  .bash_profile
## Remove proc_info privilege from my shell
##
ppriv -s EI-proc_info $$
```

ppriv 行在当前 shell 进程 ($$) 中删除了用户的有效和可继承特权集合 (EI-) 中的 proc_info 特权。

在以下 prstat 输出中，总进程从 74 缩减为 3 个进程：

```
## With all basic privileges
Total: 74 processes, 527 lwps, load averages: 0.01, 0.00, 0.00

## With proc_info removed from the effective and inheritable set
Total: 3 processes, 3 lwps, load averages: 0.00, 0.00, 0.00
```

例 3-24 修改系统以使权限仅对其用户可用

在以下示例中，管理员将创建一个仅可以用来管理网络的系统。管理员将从 `policy.conf` 文件中删除 "Basic Solaris User"（基本 Solaris 用户）权限配置文件和任何授权。不删除 "Console User"（控制台用户）权限配置文件。在生成的 `policy.conf` 文件中，受影响的行如下所示：

```
...
##AUTHS_GRANTED=
##AUTH_PROFS_GRANTED=
##PROFS_GRANTED=Basic Solaris User
CONSOLE_USER=Console User
...
```

只有显式为其指定了授权、命令或权限配置文件的用户才能使用该系统。登录后，已授权的用户可以执行管理任务。如果授权的用户位于该系统控制台上，则该用户将拥有 "Console User"（控制台用户）权限。

例 3-25 限制管理员使用显式指定的权限

您可以通过下列两种方式限制角色或用户，使其执行有限数量的管理操作。

- 将 "Stop"（停止）权限配置文件指定为用户配置文件列表中的最后一个配置文件。

  "Stop"（停止）权限配置文件是创建受限 shell 的最简单方式。`policy.conf` 文件中的授权和权限配置文件未指定给用户或角色。
- 修改系统的 `policy.conf` 文件，要求角色或用户使用该系统执行管理任务。请参见例 3-24 “修改系统以使权限仅对其用户可用”。

以下命令限制 `auditrev` 角色只能执行审计查看。

```
# rolemod -P "Audit Review,Stop" auditrev
```

由于 `auditrev` 角色不具有 "Console User"（控制台用户）权限配置文件，因此审计者无法关闭系统。由于该角色不包含 `solaris.device.cdrw` 授权，因此审计者无法读取或写入 CD-ROM 驱动器。由于该角色不具有 "Basic Solaris User"（基本 Solaris 用户）权限配置文件，因此该角色不能运行配置文件中的任何命令。因为没有指定 "All"（所有）权限配置文件，`ls` 命令不会运行。此角色使用 "File Browser"（文件浏览器）选择要查看的审计文件。

有关更多信息，请参见“所指定权限的搜索顺序” [31]和“权限配置文件参考信息” [99]。

例 3-26 防止所选应用程序大量生成新进程

在此示例中，管理员为无需子进程即可正常运行的应用程序创建了一个权限配置文件。为方便起见，管理员创建了一个目录保存这些可执行文件。添加了无需子进程的新应

用程序后，可以将其可执行文件添加到此目录。或者，如果要求可执行文件位于特定目录，管理员可以通过 /opt/local/noex/*app-executable* 进行链接。

```
# profiles -p "Prevent App Subprocess"
profiles:Prevent App Subprocess> set desc="Keep apps from execing processes"
profiles:Prevent App Subprocess> add cmd=/opt/local/noex/mkmod
... Subprocess:mkmod> set limitprivs=all,!proc_exec
... Subprocess:mkmod> end
... Subprocess> add cmd=/opt/local/noex/gomap
... Subprocess:gomap> set limitprivs=all,!proc_exec
... Subprocess:gomap> end
... Subprocess> commit
... Subprocess> exit
```

例 3-27 防止来宾大量生成新子进程

在此示例中，管理员通过从编辑器命令中删除 proc_exec 基本特权，防止用户通过一个或多个编辑器创建子 shell。

1. 管理员创建了一个权限配置文件，从 vim 编辑器的限制特权集合删除了 proc_exec。

   ```
   # profiles -p -S ldap "Editor Restrictions"
   profiles:Editor Restrictions> set desc="Site Editor Restrictions"
   ... Restrictions> add cmd=/usr/bin/vim
   ... Restrictions:vim> set limitprivs=all,!proc_exec
   ... Restrictions:vim> end
   ... Restrictions> commit
   ... Restrictions> exit
   ```

2. 管理员向权限配置文件添加其他常用编辑器。

   ```
   # profiles -p "Editor Restrictions"
   profiles:Editor Restrictions> add cmd=/usr/bin/gedit
   ... Restrictions:gedit> set limitprivs=all,!proc_exec
   ... Restrictions:gedit> end
   ... Restrictions> add cmd=/usr/bin/gconf-editor
   ... Restrictions:gconf-editor> set limitprivs=all,!proc_exec
   ... Restrictions:gconf-editor> end
   ... Restrictions> add cmd=/usr/bin/ed
   ... Restrictions:ed> set limitprivs=all,!proc_exec
   ... Restrictions:ed> end
   ... Restrictions> add cmd=/usr/bin/ex
   ... Restrictions:ex> set limitprivs=all,!proc_exec
   ... Restrictions:ex> end
   ... Restrictions> add cmd=/usr/bin/edit
   ... Restrictions:edit> set limitprivs=all,!proc_exec
   ... Restrictions:edit> end
   ```

```
... Restrictions> commit
... Restrictions> exit
```

3. 管理员检查权限配置文件条目有无错误，如拼写错误、缺失或重复。

```
# profiles -p "Editor Restrictions" info
Found profile in files repository.
name=Editor Restrictions
desc=Site Editor Restrictions
cmd=/usr/bin/vim
limitprivs=all,!proc_exec
...
```

4. 管理员将 "Editor Restrictions"（编辑器限制）权限配置文件指定给 `guest` 用户。

```
# usermod -K profiles+="Editor Restrictions" guest
```

使用 `profiles+`，管理员将此权限配置文件添加到帐户的当前权限配置文件。

5. 为了验证已经限制编辑器特权，管理员在单独窗口中打开编辑器，检查编辑器进程的特权。

```
# ppriv -S $(pgrep vi)
2805:   vi .bash_profile
flags = PRIV_PFEXEC       User is running a profile shell
        E: basic,!proc_info       proc_info is removed from basic set
        I: basic,!proc_info
        P: basic,!proc_info
        L: all,!proc_exec       proc_exec is removed from limit set
```

例 3-28 将 "Editor Restrictions"（编辑器限制）权限配置文件指定给所有用户

在此示例中，管理员将 "Editor Restrictions"（编辑器限制）权限配置文件添加到 `policy.conf` 文件。管理员确保将此文件分配给了来宾可以登录的所有公共系统。

```
# cd /etc/security; cp policy.conf policy.conf.orig
# pfedit /etc/security/policy.conf
...
AUTHS_GRANTED=
AUTH_PROFS_GRANTED=
#PROFS_GRANTED=Basic Solaris User
PROFS_GRANTED=Editor Restrictions,Basic Solaris User
```

"User Security"（用户安全）管理员为每位用户指定了一个配置文件 shell。有关原因和过程，请参见“为用户指定权限” [39]。

♦ ♦ ♦ 第 4 章

向应用程序、脚本和资源指定权限

本章介绍了将特权、扩展特权策略和其他权限应用于用户、端口和应用程序的任务：

- “向应用程序和脚本指定权限” [57]
- “使用扩展特权锁定资源” [59]
- “用户锁定其运行的应用程序” [65]

有关权限的概述，请参见“用户权限管理” [14]。

将应用程序、脚本和资源限定于特定权限

本节中的任务和示例将特权指定给可执行文件和系统资源。通常，可将特权指定给可执行文件，以便可信用户能够运行该可执行文件。在“向应用程序和脚本指定权限” [57]中，特权指定使可信用户能够在配置文件 shell 中运行应用程序或脚本。在“使用扩展特权锁定资源” [59]中，扩展特权策略将用户 ID、端口或文件对象限定为比缺省有效特权集合小的特权集合。不允许该用户的进程、端口或对象行使未指定的特权。此类指定接近最小特权策略。

向应用程序和脚本指定权限

应用程序和脚本执行一个命令或一系列命令。要指定权限，请为权限配置文件中的每个命令设置安全属性，例如设置 ID 或特权。应用程序可检查授权（如果适用）。

注 - 如果脚本中的某个命令需要设置 `setuid` 位或 `setgid` 位才能成功运行，则必须在权限配置文件中为该脚本的可执行文件和该命令添加安全属性。在配置文件 shell 中执行该脚本时，该命令将使用安全属性运行。

- 运行需要权限的脚本 – 如何运行具有特权命令的 Shell 脚本 [58]
- 使非 `root` 用户能够运行可识别特权的应用程序 – 例 4-1 “将安全属性指定给传统应用程序”
- 使非 `root` 用户能够运行归 `root` 所有的应用程序 – 例 4-2 “使用指定的权限运行应用程序”

- 检查脚本中的授权 – 例 4-3 “检查脚本或程序中的授权”

▼ 如何运行具有特权命令的 Shell 脚本

要运行特权 shell 脚本，需要向脚本以及脚本中的命令添加特权。然后，相应权限配置文件必须包含指定有特权的命令。

开始之前 您必须承担 `root` 角色。有关详细信息，请参见“使用所指定的管理权限” [70]。

1. **将 `/bin/pfsh` 或任何其他配置文件 shell 放在第一行来创建脚本。**

```
#!/bin/pfsh
# Copyright (c) 2013 by Oracle
```

2. **以一般用户身份运行脚本，确定脚本中的命令所需的特权。**

通过运行没有特权的脚本，`ppriv` 命令的 `debug` 选项可列出缺少的特权。

```
% ppriv -eD script-full-path
```

有关详细信息，请参见如何确定程序所需的特权 [96]。

3. **创建或修改脚本的权限配置文件。**

将 shell 脚本、该 shell 脚本中的命令以及所需的安全属性添加到权限配置文件中。请参见如何创建权限配置文件 [74]。

4. **将权限配置文件指定给可信用户或角色。**

有关示例，请参见“为用户指定权限” [39]。

5. **要运行脚本，请执行以下操作之一：**

- **如果以用户方式将脚本指定给您，请打开配置文件 shell 并运行脚本。**

```
% pfexec script-full-path
```

- **如果以角色方式将脚本指定给您，请承担该角色并运行脚本。**

```
 % su - rolename
Password: xxxxxxxx
# script-full-path
```

例 4-1 将安全属性指定给传统应用程序

由于传统应用程序不识别特权，管理员需要在权限配置文件中将 `euid=0` 安全属性指定给应用程序可执行文件。然后，管理员将权限配置文件指定给可信用户。

```
# profiles -p LegacyApp
```

```
profiles:LegacyApp> set desc="Legacy application"
profiles:LegacyApp> add cmd=/opt/legacy-app/bin/legacy-cmd
profiles:LegacyApp:legacy-cmd> set euid=0
profiles:LegacyApp:legacy-cmd> end
profiles:LegacyApp> exit
# profiles -p LegacyApp 'select cmd=/opt/legacy-app/bin/legacy-cmd;info;end'
 id=/opt/legacy-app/bin/legacy-cmd
 euid=0

# usermod -K profiles+="Legacy application" jdoe
```

例 4-2 使用指定的权限运行应用程序

在本示例中，管理员将例 5-7 “创建包含特权命令的权限配置文件” 中的权限配置文件指定给可信用户。用户在执行脚本时必须提供口令。

```
# usermod -K auth_profiles+="Site application" jdoe
```

例 4-3 检查脚本或程序中的授权

要检查授权，请编写一个基于 `auths` 命令的测试。有关此命令的详细信息，请参见 `auths(1)` 手册页。

例如，以下行会测试用户是否具有作为 `$1` 参数提供的授权：

```
if [ `/usr/bin/auths|/usr/xpg4/bin/grep $1` ]; then
        echo Auth granted
else
        echo Auth denied
fi
```

更加完整的测试包括某种逻辑来检查使用通配符的情况。例如，要测试用户是否具有 `solaris.system.date` 授权，需要检查以下字符串：

- `solaris.system.date`
- `solaris.system.*`
- `solaris.*`

如果您要编写程序，请使用函数 `getauthattr（）` 对授权进行测试。

使用扩展特权锁定资源

在对应用程序的攻击成功的情况下，扩展特权策略可以限制攻击者对系统的访问权限。扩展策略规则将特权指定的作用范围限定到该规则中的资源。扩展策略规则的表示方法为：将特权括在花括号中，后跟一个冒号和关联的资源。有关更多介绍，请参见“扩展用户或角色的特权” [28]。有关语法的示例，请参见 `ppriv(1)` 和 `privileges(5)` 手册页。

管理员和一般用户都可以使用扩展特权锁定资源。管理员可以为用户、端口和应用程序创建扩展特权。一般用户可以使用命令行或者编写使用 `ppriv -r` 命令的脚本，防止应用程序将文件写入用户指定的目录之外。

- 限制通过端口进入的恶意用户可使用的权限 – 如何将扩展特权策略应用于端口 [60]
- 将数据库作为非 `root` 守护进程运行 – 如何锁定 MySQL 服务 [61]
- 将 Apache Web 服务器作为非 `root` 守护进程运行 – 如何将特定特权指定给 Apache Web 服务器 [63]
- 验证 Apache Web 服务器是否使用特权运行 – 如何确定 Apache Web 服务器正在使用的特权 [64]
- 防止 Firefox 将数据写入系统上的目录 – 例 4-4 “在受保护的环境中运行浏览器”
- 将应用程序限定于访问系统上的特定目录 – 例 4-5 “保护系统上的目录免受应用程序进程访问”

▼ 如何将扩展特权策略应用于端口

用于网络时间协议 (Network Time Protocol, NTP) 的服务使用特权端口 `123` 进行 `udp` 通信。运行此服务需要使用特权。如果恶意用户获得了指定给此端口的特权，以下示例过程可修改服务清单以保护其他端口免受恶意用户访问。

开始之前 您必须承担 `root` 角色。有关详细信息，请参见“使用所指定的管理权限” [70]。

1. **读取此端口的缺省服务清单项。**

   在下面的 `/lib/svc/manifest/network/ntp.xml` `start` 方法项中，可能其他进程也使用 `net_privaddr`、`proc_lock_memory` 和 `sys_time` 特权：

   ```
   privileges='basic,!file_link_any,!proc_info,!proc_session,net_privaddr,
   proc_lock_memory,sys_time'
   ```

   删除 `!file_link_any,!proc_info,!proc_session` 指定的特权可防止服务接收信号或观察任何其他进程以及创建硬链接来重命名文件。也就是说，由该服务启动的进程只能绑定到 NTP 的端口 `123`，而不能绑定到任何其他特权端口。

   如果黑客能够利用该服务启动其他进程，则该进程将会受到类似限制。

2. **修改 `start` 和 `restart` 方法以使 `net_privaddr` 特权仅限于该端口。**

   ```
   # svccfg -s ntp editprop
   ```

   a. **搜索字符串 `net_privaddr`。**

   b. **取消对包含 `net_privaddr` 的项的注释。**

   c. **在这两项中，将 `net_privaddr` 替换为 `{net_privaddr}:123/udp`。**

扩展特权策略将从该服务中删除指定特权以及未指定的基本特权以外的所有特权。因此，特权集合中有可能加以利用的特权由超过八十个减少为不到八个。

3. 重新启动服务以使用扩展特权策略。

```
# svcadm restart ntp
```

4. 验证该服务是否在使用扩展特权。

```
# svccfg -s ntp listprop | grep privileges
start/privileges    astring  basic,!file_link_any,!proc_info,!proc_session,
                    {net_privaddr}:123/udp,proc_lock_memory,sys_time
restart/privileges  astring  basic,!file_link_any,!proc_info,!proc_session,
                    {net_privaddr}:123/udp,proc_lock_memory,sys_time
```

▼ 如何锁定 MySQL 服务

安装时，MySQL 数据库配置为通过不受保护的端口使用 root 的全部特权运行。在本任务中，需要在权限配置文件中向 MySQL 服务指定扩展特权策略。在权限配置文件成为服务的 exec 方法之后，MySQL 通过受保护的端口作为用户 mysql 运行，非 MySQL 进程对数据库的访问将受到限制。

开始之前 初始用户可以安装软件包。其余步骤必须由 root 角色来执行。有关详细信息，请参见“使用所指定的管理权限” [70]。

1. 安装 MySQL 软件包。

```
# pkg search basename:mysql
...
basename ... pkg:/database/mysql-51@version
# pfexec pkg install mysql-51
```

注 - 如果升级到 MySQL 数据库的版本 5.5，则需要将所有步骤修改为使用 5.5 和 55，而非 5.1 和 51。

---

2. 显示 MySQL 服务的 FMRI 和状态。

```
# svcs mysql
STATE          STIME    FMRI
disabled       May_15   svc:/application/database/mysql:version_51
```

3. 创建修改服务的执行方法的权限配置文件。

该服务的服务清单指定执行方法为 shell 脚本包装 /lib/svc/method/mysql_51。

```
# svccfg -s mysql listprop | grep manifest
... astring     /lib/svc/manifest/application/database/mysql_51.xml
```

```
# grep exec= /lib/svc/manifest/application/database/mysql_51.xml
               exec='/lib/svc/method/mysql_51 start'
               exec='/lib/svc/method/mysql_51 stop'
```

在配置文件中使用命令的 /lib/svc/method/mysql_51 包装。

```
% su -
Password: xxxxxxxx
# profiles -p "MySQL Service"
MySQL Service> set desc="Locking down the MySQL Service"
MySQL Service> add cmd=/lib/svc/method/mysql_51
MySQL Service:mysql_51> set privs=basic
MySQL Service:mysql_51> add privs={net_privaddr}:3306/tcp
MySQL Service:mysql_51> add privs={file_write}:/var/mysql/5.1/data/*
MySQL Service:mysql_51> add privs={file_write}:/tmp/mysql.sock
MySQL Service:mysql_51> add privs={file_write}:/var/tmp/ib*
MySQL Service:mysql_51> end
MySQL Service> set uid=mysql
MySQL Service> set gid=mysql
MySQL Service> exit
```

file_write 特权是缺省情况下授予所有进程的基本特权。通过显式枚举可写路径，将写入访问仅限于这些路径。该约束应用于指定的可执行文件及其子进程。

4. 使 MySQL 的缺省端口成为特权端口。

```
# ipadm set-prop -p extra_priv_ports+=3306 tcp
# ipadm show-prop -p extra_priv_ports tcp
PROTO PROPERTY            PERM CURRENT     PERSISTENT   DEFAULT    POSSIBLE
tcp   extra_priv_ports    rw   2049,4045,  3306         2049,4045  1-65535
                                 3306
```

net_privaddr 特权需要绑定到特权端口。对于 MySQL，绑定到缺省端口号 3306 通常不需要此特权。

5. 将权限配置文件指定给 MySQL 服务并通知服务使用该配置文件。

```
# svccfg -s mysql:version_51
...version_51> setprop method_context/profile="MySQLService"
...version_51> setprop method_context/use_profile=true
...version_51> refresh
...version_51> exit
```

6. 启用服务。

   FMRI 的最后组成部分 mysql:version_51 足以唯一指定服务。

```
# svcadm enable mysql:version_5
```

7. （可选） 验证服务是否正在使用 MySQL 服务权限配置文件中指定的权限运行。

```
# ppriv $(pgrep mysql)
103697:   /usr/mysql/5.1/bin/mysqld --basedir=/usr/mysql/5.1
```

```
                                    --datadir=/var/mysql/5.1/data
flags =   PRIV_XPOLICY
    Extended policies:
             {net_privaddr}:3306/tcp
             {file_write}:/var/mysql/5.1/data/*
             {file_write}:/tmp/mysql.sock
             {file_write}:/var/tmp/ib*
        E: basic,!file_write
        I: basic,!file_write
        P: basic,!file_write
        L: all
103609:  /bin/sh /usr/mysql/5.1/bin/mysqld_safe --user=mysql
                               --datadir=/var/mysql/5.1/data
flags =  PRIV_XPOLICY
    Extended policies:
             {net_privaddr}:3306/tcp
             {file_write}:/var/mysql/5.1/data/*
             {file_write}:/tmp/mysql.sock
             {file_write}:/var/tmp/ib*
        E: basic,!file_write
        I: basic,!file_write
        P: basic,!file_write
        L: all
```

▼ 如何将特定特权指定给 Apache Web 服务器

此过程通过仅向 Web 服务器守护进程指定需要的特权来锁定该守护进程。Web 服务器只能绑定到端口 80，并且只能写入 `webservd` 守护进程所拥有的文件。没有 `apache22` 服务进程以 `root` 身份运行。

开始之前 您必须承担 `root` 角色。有关详细信息，请参见“使用所指定的管理权限” [70]。

1. **创建 Web 服务器权限配置文件。**

```
# profiles -p "Apache2"
profiles:Apache2> set desc="Apache Web Server Extended Privilege"
profiles:Apache2> add cmd=/lib/svc/method/http-apache22
profiles:Apache2:http-apache22> add privs={net_privaddr}:80/tcp
...http-apache22> add privs={zone}:/system/volatile/apache2
...http-apache22> add privs={zone}:/var/apache2/2.2/logs/*
...http-apache22> add privs={zone}:/var/user
...http-apache22> add privs={file_write}:/var/user/webserv*
...http-apache22> add privs={file_write}:/tmp/*
...http-apache22> add privs={file_write}:/system/volatile/apache*
...http-apache22> add privs={file_write}:/proc/*
...http-apache22> add privs=basic,proc_priocntl
...http-apache22> set uid=webservd
...http-apache22> set gid=webservd
...http-apache22> end
---Apache2> exit
```

2. （可选） 如果要将 SSL 内核代理与 Apache2 配合使用，则必须将 SSL 端口添加到 **webservd** 扩展策略。

```
# profiles -p "Apache2"
profiles:Apache2> add privs={net_privaddr}:443/tcp
profiles:Apache2> add privs={net_privaddr}:8443/tcp
profiles:Apache2:http-apache22> end
```

《在 Oracle Solaris 11.2 中确保网络安全 》中的“如何配置 Apache 2.2 Web 服务器以使用 SSL 内核代理”对 SSL 内核代理过程进行了介绍。

3. 将权限配置文件添加到 apache22 SMF 启动方法。

```
# svccfg -s apache22
svc:/network/http:Apache2> listprop start/exec
start/exec  astring "/lib/svc/method/http-apache22 start"
...
svc:/network/http:Apache2> setprop start/profile="Apache2"
svc:/network/http:Apache2> setprop start/use_profile=true
svc:/network/http:Apache2> refresh
svc:/network/http:Apache2> exit
```

启用 `apache22` 服务后，将使用 Apache2 配置文件。

4. 启用 **apache22** 服务。

```
# svcadm enable apache22
```

5. 验证 Web 服务器是否正常工作。

打开浏览器并在 Firefox URL 字段中键入 `localhost`。

接下来的步骤 要验证特权是否已正确应用，请继续执行如何确定 Apache Web 服务器正在使用的特权 [64]这一过程。

▼ 如何确定 Apache Web 服务器正在使用的特权

在该任务中，通过创建 Apache2 权限配置文件的调试版本确定 Web 服务器正在使用的特权。

开始之前 您已完成如何将特定特权指定给 Apache Web 服务器 [63]。`apache22` 服务处于禁用状态。您现在处于 `root` 角色。

1. 克隆 Apache2 配置文件以调用其他命令。

调试命令比调试 SMF 服务简单。`apachectl` 以交互方式启动 Apache 服务。

```
# profiles -p "Apache2"
profiles:Apache2> set name="Apache-debug"
```

```
profiles:Apache-debug> sel <Tab><Tab>
profiles:Apache-debug:http-apache22> set id=/usr/apache2/2.2/bin/apachectl
profiles:Apache-debug:apachectl> end
profiles:Apache-debug> exit
```

有关更多信息，请参见 apachectl(8) 手册页。

2. 将克隆的配置文件指定给 **webservd** 帐户。

```
# usermod -K profiles+=Apache-debug webservd
```

3. 切换为 **webservd** 身份。

```
# su - webservd
```

4. （可选） 验证身份。

```
# id
uid=80(webservd) gid=80(webservd)
```

5. 在配置文件 shell 中以调试模式启动 Web 服务。
请勿直接使用 SMF。使用 Apache-debug 权限配置文件中的命令。

```
% pfbash
# ppriv -De /usr/apache2/2.2/bin/apachectl start
```

6. 在 **root** 角色中，检查第一个 **http** 守护进程的特权。

```
# ppriv $(pgrep httpd|head -1)
2999:   httpd
flags = PRIV_DEBUG|PRIV_XPOLICY|PRIV_EXEC
  5         Extended policies:
  6                 {net_privaddr}:80/tcp
  7                 {zone}:/system/volatile/apache2
  8                 {zone}:/var/apache2/2.2/logs/*
  9                 {zone}:/var/user
 10                 {file_write}:/var/user/webserv*
 11                 {file_write}:/tmp/*
 12                 {file_write}:/system/volatile/apache*
 13                 {file_write}:/proc/*
 14         E: basic,!file_write,!proc_info,proc_priocntl
 15         I: basic,!file_write,!proc_info,proc_priocntl
 16         P: basic,!file_write,!proc_info,proc_priocntl
 17         L: all
```

用户锁定其运行的应用程序

用户可以使用扩展特权策略从应用程序中删除基本特权。该策略可防止应用程序访问其不应访问的目录。

注 - 顺序非常重要。必须在为大多数 $HOME/.* 目录指定限制性强的特权之后为 $HOME/Download* 等目录指定限制性弱的特权。

例 4-4 在受保护的环境中运行浏览器

本示例说明了用户如何在受保护的环境中运行 Firefox 浏览器。在此配置中，用户的 Documents 目录对 Firefox 隐藏。

通过使用以下命令，用户可以从 /usr/bin/firefox 命令中删除基本特权。ppriv -r 命令的扩展特权参数可将浏览器限定为仅在用户指定的目录中执行读取和写入。-e 选项及其参数使用扩展特权策略打开浏览器。

```
% ppriv -r "\
{file_read}:/dev/*,\
{file_read}:/etc/*,\
{file_read}:/lib/*,\
{file_read}:/usr/*,\
{file_read}:/var/*,\
{file_read}:/proc,\
{file_read}:/proc/*,\
{file_read}:/system/volatile/*,\
{file_write}:$HOME,\
{file_read}:$HOME/.*,\
{file_read,file_write}:$HOME/.mozill*,\
{file_read,file_write}:$HOME/.gnome*,\
{file_read,file_write}:$HOME/Downloa*,\
{file_read,file_write}:/tmp,\
{file_read,file_write}:/tmp/*,\
{file_read,file_write}:/var/tmp,\
{file_read,file_write}:/var/tmp/*,\
{proc_exec}:/usr/*\
" -e /usr/bin/firefox file:///$HOME/Desktop
```

在扩展策略中使用 file_read 和 file_write 特权时，必须向应当读取或写入的每个文件授予显式访问权限。在此类策略中需要使用通配符 (*)。

要处理自动挂载的起始目录，用户需要为自动挂载路径添加显式条目，例如：

```
{file_read,file_write}:/export/home/$USER
```

如果站点未使用 automount 工具，则只需受保护目录的初始列表即可。

用户可以通过创建 shell 脚本自动运行此受命令行保护的浏览器。之后，要启动浏览器，用户需要调用脚本，而非 /usr/bin/firefox 命令。

例 4-5 保护系统上的目录免受应用程序进程访问

在本示例中，一般用户使用 shell 脚本包装为应用程序创建沙箱。脚本的第一部分限制应用程序只能访问特定目录。例外情况（如 Firefox）在脚本的后面部分进行处理。脚本后跟有关其各个部分的注释。

```
1 #!/bin/bash
2
3 # Using bash because ksh misinterprets extended policy syntax
4
5 PATH=/usr/bin:/usr/sbin:/usr/gnu/bin
6
7 DENY=file_read,file_write,proc_exec,proc_info
8
9 SANDBOX="\
10 {file_read}:/dev/*,\
11 {file_read}:/etc/*,\
12 {file_read}:/lib/*,\
13 {file_read,file_write}:/usr/*,\
14 {file_read}:/proc,\
15 {file_read,file_write}:/proc/*,\
16 {file_read}:/system/volatile/*,\
17 {file_read,file_write}:/tmp,\
18 {file_read,file_write}:/tmp/*,\
19 {file_read,file_write}:/var/*,\
20 {file_write}:$HOME,\
21 {file_read}:$HOME/.*,\
22 {file_read,file_write}:$PWD,\
23 {file_read,file_write}:$PWD/*,\
24 {proc_exec}:/usr/*\
25 "
26
27 # Default program is restricted bash shell
28
29 if [[ ! -n $1 ]]; then
30     program="/usr/bin/bash --login --noprofile
       --restricted"
31 else
32     program="$@"
33 fi
34
35
36 # Firefox needs more file and network access
37 if [[ "$program" =~ firefox ]]; then
38     SANDBOX+=",\
39 {file_read,file_write}:$HOME/.gnome*,\
40 {file_read,file_write}:$HOME/.mozill*,\
41 {file_read,file_write}:$HOME/.dbu*,\
42 {file_read,file_write}:$HOME/.puls*\
43 "
44
45 else
```

```
46     DENY+=",net_access"
47 fi
48
49 echo Starting $program in sandbox
50 ppriv -s I-$DENY -r $SANDBOX -De $program
```

可对策略进行调整，以授予特定应用程序更多或更少的访问权限。一项调整位于第 38-42 行，其中授予 Firefox 对多个点文件（在用户的起始目录中维护会话信息）的写入访问权限。此外，Firefox 不受第 46 行的约束，该行删除了网络访问权限。但是，不允许 Firefox 读取用户的起始目录中的任意文件，并且只能将文件保存在其当前目录中。

额外一级保护位于第 30 行，即缺省程序使用受限 Bash shell。受限 shell 无法更改其当前目录或执行用户的点文件。因此，从该 shell 启动的命令会以类似方式锁定在沙箱中。

在脚本的最后一行，为 `ppriv` 命令传递了两个特权集合作为 shell 变量，即 `$DENY` 和 `$SANDBOX`。

第一个特权集合 `$DENY` 可防止进程读取或写入任何文件、执行任何子进程、观察其他用户的进程以及（有条件地）访问网络。这些限制过于严格，因此在第二个特权集合 `$SANDBOX` 中，通过枚举允许读取、写入和执行的目录对策略进行了细化。

另外，第 50 行中指定了调试选项 `-D`。访问失败实时显示在终端窗口中，并显示成功访问所需的特定对象以及相应特权。此调试信息可帮助用户为其他应用程序定制策略。

♦ ♦ ♦ 第 5 章

管理权限的使用

本章介绍使用权限模型进行管理的系统的维护任务。在一些任务中，通过创建新权限配置文件和授权来扩展 Oracle Solaris 提供的权限。

本章涵盖以下主题：

有关权限的信息，请参见第 1 章 使用权限控制用户和进程。有关维护为用户和角色指定的权限的信息，请参见第 3 章 在 Oracle Solaris 中指定权限。

管理权限的使用

本节中的任务和示例说明如何使用已经指定给您的权限，以及如何更改缺省提供的权限配置。

注 - 有关故障排除帮助，请参见“排除权限问题” [91]。

- 使用所指定的权限 – “使用所指定的管理权限” [70]
- 审计管理操作 – 例 5-5 “使用两个角色配置审计”
- 添加权限配置文件和授权 – “创建权限配置文件和授权” [74]
- 将 root 配置为用户 – 如何将 root 角色更改为用户 [80]
- 将 root 改回角色 – 例 5-12 “将 root 用户更改为 root 角色”
- 防止 root 管理系统 – 例 5-13 “防止 root 角色用于维护系统”

使用所指定的管理权限

在 root 角色中，初始用户具有所有管理权限。root 用户可以为可信用户指定管理权限（例如角色、权限配置文件或特定特权）和授权。本节说明这类用户可以如何使用指定给他的权限。

注 - Oracle Solaris 提供了用于编辑管理文件的特殊编辑器。编辑管理文件时，使用 pfedit 命令。例 5-1 “编辑系统文件” 显示如何允许非 root 用户编辑指定系统文件。

要执行管理任务，请打开终端窗口，然后从以下选项进行选择：

- 如果您使用 sudo，键入 sudo 命令。

  对于熟悉 sudo 命令的管理员，使用在 sudoers 中指定给他的管理命令名运行该命令。有关更多信息，请参见 sudo(1M) 和 sudoers(4) 手册页。
- 如果您要执行的任务需要超级用户特权，请成为 root 用户。

```
% su -
Password: xxxxxxxx
#
```

注 - 无论 root 是用户还是角色，此命令都可以运行。井号 (#) 提示符表明您现在已是 root 用户。

- 如果您的任务是指定给某个角色，承担可以执行该任务的角色。

  在以下示例中，您承担了审计配置角色。此角色包括 "Audit Configuration"（审计配置）权限配置文件。您已经从管理员处收到了角色口令。

```
% su - audadmin
Password: xxxxxxxx
#
```

提示 - 如果您没有收到角色口令，您的管理员已经将角色配置为需要您的用户口令。键入您的用户口令以承担该角色。有关此选项的更多信息，请参见例 3-16 “使用户可以将自己的口令用作角色口令”。

您之前键入该命令的 shell 现在成为配置文件 shell。在该 shell 中，您可以运行 auditconfig 命令。有关配置文件 shell 的更多信息，请参见“配置文件 Shell 和权限验证” [31]。

提示 - 要查看您的角色的权限，请参见“列出权限配置文件” [84]。

- 如果您的任务是以用户方式直接指定给您的，请通过以下方法之一创建配置文件 shell：
  - 使用 `pfbash` 命令创建评估管理权限的 shell。

    在以下示例中，已经将 "Audit Configuration"（审计配置）权限配置文件直接指定给您。以下命令集允许您在 `pfbash` 配置文件 shell 中查看审计预选值和审计策略：

    ```
    % pfbash
    # auditconfig -getflags
    active user default audit flags = ua,ap,lo(0x45000,0x45000)
    configured user default audit flags = ua,ap,lo(0x45000,0x45000)
    # auditconfig -getpolicy
    configured audit policies = cnt
    active audit policies = cnt
    ```
  - 使用 `pfexec` 命令运行一个管理命令。

    在以下示例中，已经将 "Audit Configuration"（审计配置）权限配置文件作为需要验证权限配置文件直接指定给您。您可以运行此配置文件中的特权命令，即运行带该命令名称的 `pfexec` 命令。例如，您可以查看用户预选的审计标志：

    ```
    % pfexec auditconfig -getflags
    Enter password:       Type your user password
    active user default audit flags = ua,ap,lo(0x45000,0x45000)
    configured user default audit flags = ua,ap,lo(0x45000,0x45000)
    ```

    通常，要运行包含在您的权限中的其他特权命令，必须在键入该特权命令之前再次键入 `pfexec`。有关更多信息，请参见 `pfexec(1)` 手册页。如果配置有口令缓存，您可以在一定间隔（可配置）内运行后续命令而不提供口令，如例 5-2 “缓存验证以便简化角色使用”中所示。

例 5-1 编辑系统文件

如果您不是 UID 为 `0` 的 `root`，则缺省情况下您无法编辑系统文件。不过，如果为您指定了 `solaris.admin.edit/` *path-to-system-file* 授权，则您可以编辑 *system-file*。例如，如果为您指定了 `solaris.admin.edit/etc/security/audit_warn` 授权，则您可以使用 `pfedit` 命令编辑 `audit_warn` 文件。

```
# pfedit /etc/security/audit_warn
```

有关更多信息，请参见 `pfedit`(4) 手册页。所有管理员都可以使用此命令。

例 5-2 缓存验证以便简化角色使用

在此示例中，管理员配置了一个用于管理审计配置的角色，但通过缓存该用户的验证来简化使用。首先，管理员创建并指定角色。

```
# roleadd -K roleauth=user -P "Audit Configuration" audadmin
# usermod -R +audadmin jdoe
```

当切换到该角色时，如果 jdoe 使用 -c 选项，则在显示 auditconfig 输出之前需要输入口令。

```
% su - audadmin -c auditconfig option
Password: xxxxxxxx
    auditconfig output
```

如果没有缓存验证，jdoe 再次运行该命令时，将提示输入口令。

管理员在 pam.d 目录中创建一个文件，以保留启用验证缓存的 su 栈。如果缓存了验证，只需要在初次运行时输入口令，以后都不再需要输入口令，直到超过一定的时间。

```
# pfedit /etc/pam.d/su
## Cache authentication for switched user
#
auth required           pam_unix_cred.so.1
auth sufficient         pam_tty_tickets.so.1
auth requisite          pam_authtok_get.so.1
auth required           pam_dhkeys.so.1
auth required           pam_unix_auth.so.1
```

创建文件之后，管理员将对这些项进行检查，确定是否出现拼写、缺失或重复问题。

管理员必须提供完整的上述 su 栈。pam_tty_tickets.so.1 模块实现高速缓存。有关 PAM 的更多信息，请参见pam_tty_tickets(5) 和 pam.conf(4) 手册页以及《在 Oracle Solaris 11.2 中管理 Kerberos 和其他验证服务 》中的第 1 章 “使用可插拔验证模块”。

在管理员添加 su PAM 文件并重新引导系统之后，对于所有角色（包括 audadmin 角色），当运行一系列命令时仅会提示他们输入一次口令。

```
% su - audadmin -c auditconfig option
Password: xxxxxxxx
    auditconfig output
% su - audadmin -c auditconfig option
    auditconfig output
...
```

例 5-3 承担 root 角色

在以下示例中，初始用户承担 root 角色并列出该角色的 shell 中的特权。

```
% roles
root
% su - root
Password: xxxxxxxx
#       Prompt changes to root prompt
# ppriv $$
1200:   pfksh
flags = <none>
        E: all
        I: basic
        P: all
        L: all
```

有关特权的信息，请参见“进程权限管理” [21]以及 ppriv(1) 手册页。

例 5-4 承担 ARMOR 角色

在本示例中，用户承担了管理员指定的 ARMOR 角色。

在终端窗口中，用户确定指定了哪些角色。

```
% roles
fsadm
sysop
```

然后该用户承担 fsadm 角色，并提供用户的口令。

```
% su - fsadm
Password: xxxxxxxx
#
```

su - *rolename* 命令将终端的 shell 更改为配置文件 shell。现在，该用户在此终端窗口中是 fsadm 角色。

要确定此角色可以运行的命令，用户需按照“列出权限配置文件” [84]中的说明操作。

审计管理操作

站点安全策略通常要求您审计管理操作。116:AUE_PFEXEC:execve(2) with pfexec enabled:ps,ex,ua,as 审计事件捕获这些操作。cusa 元类提供适合与角色结合使用的一组事件，是审计管理操作时的另一个选项。有关更多信息，请查看 /etc/security/audit_class 文件中的注释。

例 5-5 使用两个角色配置审计

在本示例中，通过两个管理员来实现站点安全管理人员的审计配置计划。在该计划中，对所有用户使用 pf 类，为单个角色指定 cusa 元类。root 角色将审计标志指定给各个角色。第一个管理员配置审计，第二个管理员启用新配置。

第一个管理员指定有 "Audit Configuration"（审计配置）权限配置文件。此管理员查看当前审计配置：

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

因为 pf 类不包括 lo 类，该管理员将该类添加到系统配置。

```
# auditconfig -setflags lo,pf
```

要将新的审计配置读入内核，指定有 "Audit Control"（审计控制）权限配置文件的管理员需要刷新审计服务。

```
# audit -s
```

创建权限配置文件和授权

当提供的权限配置文件不包含所需的权限集合时，您可以创建或更改权限配置文件。您可以为具有有限权限的用户、新应用程序或因为各种其他原因而创建权限配置文件。

Oracle Solaris 提供的权限配置文件是只读的。如果所提供的某个权限配置文件的权限集合不足以满足要求，您可以克隆该配置文件并进行修改。例如，您可能希望将 solaris.admin.edit/ *path-to-system-file* 授权添加到所提供的某个权限配置文件中。有关背景的信息，请参见“有关权限配置文件的更多信息” [20]。

如果提供的授权不包括在您的特权应用程序中编写的授权，您可以创建一个授权。您无法更改现有的授权。有关背景的信息，请参见“有关用户授权的更多信息” [20]。

▼ 如何创建权限配置文件

开始之前 要创建权限配置文件，您必须是指定有 "File Security"（文件安全）权限配置文件的管理员。有关详细信息，请参见“使用所指定的管理权限” [70]。

1. **创建权限配置文件。**

   # profiles -p [-S *repository***]** *profile-name*

   系统会提示您输入说明。

2. 向权限配置文件添加内容。

   对具有单个值的配置文件属性使用 `set` 子命令，如 `set desc`。对可以具有多个值的属性使用 `add` 子命令，如 `add cmd`。

   以下命令创建《在 Oracle Solaris 11.2 中管理 Kerberos 和其他验证服务 》中的“如何分配修改后的 PAM 策略”中的定制 PAM 权限配置文件。为方便显示，名称进行了缩写。

```
# profiles -p -S LDAP "Site PAM LDAP"
profiles:Site PAM LDAP> set desc="Profile which sets pam_policy=ldap"
...LDAP> set pam_policy=ldap
...LDAP> commit
...LDAP> end
...LDAP> exit
```

例 5-6 创建 "Sun Ray Users"（Sun Ray 用户）权限配置文件

在此示例中，管理员在 LDAP 系统信息库中为 Sun Ray 用户创建权限配置文件。管理员已创建了 "Basic Solaris User"（基本 Solaris 用户）权限配置文件的 Sun Ray 版本，并从 Sun Ray 服务器上的 `policy.conf` 文件中删除了所有权限配置文件。

```
# profiles -p -S LDAP "Sun Ray Users"
profiles:Sun Ray Users> set desc="For all users of Sun Rays"
... Ray Users> add profiles="Sun Ray Basic User"
... Ray Users> set defaultpriv="basic,!proc_info"
... Ray Users> set limitpriv="basic,!proc_info"
... Ray Users> end
... Ray Users> exit
```

管理员验证内容。

```
# profiles -p "Sun Ray Users" info
Found profile in LDAP repository.
        name=Sun Ray Users
        desc=For all users of Sun Rays
        defaultpriv=basic,!proc_info,
        limitpriv=basic,!proc_info,
        profiles=Sun Ray Basic User
```

例 5-7 创建包含特权命令的权限配置文件

在以下示例中，安全管理员在管理员创建的权限配置文件中添加某应用程序的特权。此应用程序能够识别特权。

```
# profiles -p SiteApp
profiles:SiteApp> set desc="Site application"
profiles:SiteApp> add cmd="/opt/site-app/bin/site-cmd"
profiles:SiteApp:site-cmd> add privs="proc_fork,proc_taskid"
profiles:SiteApp:site-cmd> end
profiles:SiteApp> exit
```

要进行验证，管理员可以选择 `site-cmd`。

```
# profiles -p SiteApp "select cmd=/opt/site-app/bin/site-cmd; info;end"
Found profile in files repository.
  id=/opt/site-app/bin/site-cmd
  privs=proc_fork,proc_taskid
```

接下来的步骤 将权限配置文件指定给可信用户或角色。有关示例，请参见例 3-10 “创建可以管理 DHCP 的用户”和例 3-19 “使可信用户可以读取扩展记帐文件”。

另请参见 要对权限指定进行故障排除，请参见如何排除权限指定问题 [91]。有关背景的信息，请参见“所指定权限的搜索顺序” [31]。

▼ 如何克隆和修改系统权限配置文件

开始之前 要创建或更改权限配置文件，您必须是指定有 "File Security"（文件安全）权限配置文件的管理员。有关详细信息，请参见“使用所指定的管理权限” [70]。

1. **通过现有的配置文件创建新的权限配置文件。**

   ```
   # profiles -p [-S repository] existing-profile-name
   ```

   - **要向某个现有的权限配置文件添加内容，请创建一个新配置文件。**

     将现有权限配置文件作为补充权限配置文件添加到新配置文件，然后添加增强功能。请参见例 5-8 “克隆并增强 "Network IPsec Management"（网络 IPsec 管理）权限配置文件”。

   - **要删除某个现有权限配置文件中的内容，请克隆该配置文件然后重命名并进行更改。**

     请参见例 5-9 “从权限配置文件中克隆和删除所选权限”。

2. **通过添加或删除补充权限配置文件、授权以及其他权限修改新的权限配置文件。**

例 5-8 克隆并增强 "Network IPsec Management"（网络 IPsec 管理）权限配置文件

在此示例中，管理员将 `solaris.admin.edit` 授权添加到站点的 "IPsec Management"（IPsec 管理）权限配置文件，这样就不再需要 `root` 角色。此权限配置文件只指定给可以修改 `/etc/hosts` 文件的可信用户。

1. 管理员验证是否无法修改该 "Network IPsec Management"（网络 IPsec 管理）权限配置文件。

   ```
   # profiles -p "Network IPsec Management"
   profiles:Network IPsec Management> add auths="solaris.admin.edit/etc/hosts"
   Cannot add. Profile cannot be modified
   ```

2. 管理员创建包含该 "Network IPsec Management"（网络 IPsec 管理）配置文件的权限配置文件。

```
# profiles -p "Total IPsec Mgt"
... IPsec Mgt> set desc="Network IPsec Mgt plus /etc/hosts"
... IPsec Mgt> add profiles="Network IPsec Management"
... IPsec Mgt> add auths="solaris.admin.edit/etc/hosts"
... IPsec Mgt> end
... IPsec Mgt> exit
```

3. 管理员验证内容。

```
# profiles -p "Total IPsec Mgt" info
        name=Total IPsec Mgt
        desc=Network IPsec Mgt plus /etc/hosts
        auths=solaris.admin.edit/etc/hosts
        profiles=Network IPsec Management
```

例 5-9 从权限配置文件中克隆和删除所选权限

在此示例中，管理员将管理 VSCAN 服务的属性这一功能与启用和禁用该服务的功能相分离。

首先，管理员列出 Oracle Solaris 提供的权限配置文件的内容。

```
# profiles -p "VSCAN Management" info
        name=VSCAN Management
        desc=Manage the VSCAN service
        auths=solaris.smf.manage.vscan,solaris.smf.value.vscan,
              solaris.smf.modify.application
        help=RtVscanMngmnt.html
```

然后，管理员创建可以启用和禁用该服务的权限配置文件。

```
# profiles -p "VSCAN Management"
profiles:VSCAN Management> set name="VSCAN Control"
profiles:VSCAN Control> set desc="Start and stop the VSCAN service"
... VSCAN Control> remove auths="solaris.smf.value.vscan"
... VSCAN Control> remove auths="solaris.smf.modify.application"
... VSCAN Control> end
... VSCAN Control> exit
```

然后，管理员创建可更改该服务的属性的权限配置文件。

```
# profiles -p "VSCAN Management"
profiles:VSCAN Management> set name="VSCAN Properties"
profiles:VSCAN Properties> set desc="Modify VSCAN service properties"
... VSCAN Properties> remove auths="solaris.smf.manage.vscan"
... VSCAN Properties> end
... VSCAN Properties> exit
```

管理员验证新权限配置文件的内容。

```
# profiles -p "VSCAN Control" info
        name=VSCAN Control
        desc=Start and stop the VSCAN service
        auths=solaris.smf.manage.vscan
# profiles -p "VSCAN Properties" info
        name=VSCAN Properties
        desc=Modify VSCAN service properties
        auths=solaris.smf.value.vscan,solaris.smf.modify.application
```

接下来的步骤 将权限配置文件指定给可信用户或角色。有关示例，请参见例 3-10 “创建可以管理 DHCP 的用户”和例 3-19 “使可信用户可以读取扩展记帐文件”。

另请参见 要对权限指定进行故障排除，请参见如何排除权限指定问题 [91]。有关背景的信息，请参见“所指定权限的搜索顺序” [31]。

▼ 如何创建授权

开始之前 开发者在您要安装的应用程序中定义和使用了授权。有关说明，请参见《面向开发者的 Oracle Solaris 11 安全性指南》和《面向开发者的 Oracle Solaris 11 安全性指南》中的“关于授权”。

1. （可选） 为您的新授权创建帮助文件。

   例如，为使用户能够修改应用程序中数据的授权创建帮助文件。

```
# pfedit /docs/helps/NewcoSiteAppModData.html
<HTML>
-- Copyright 2013 Newco.  All rights reserved.
-- NewcoSiteAppModData.html
-->
<HEAD>
     <TITLE>NewCo Modify SiteApp Data Authorization</TITLE>
</HEAD>
<BODY>
The com.newco.siteapp.data.modify authorization authorizes you
to modify existing data in the application.
<p>
Only authorized accounts are permitted to modify data.
Use this authorization with care.
<p>
</BODY>
</HTML>
```

2. 通过使用 **auths add** 命令创建授权。

   例如，以下命令在本地系统上创建 `com.newco.siteapp.data.modify` 授权。

```
# auths add -t "SiteApp Data Modify Authorized" \
-h /docs/helps/NewcoSiteAppModData.html com.newco.siteapp.data.modify
```

现在，您可以测试授权，然后将该授权添加到权限配置文件中，并将该配置文件指定给角色或用户。

例 5-10 测试新授权

在此示例中，管理员使用例 5-7 “创建包含特权命令的权限配置文件” 的 SiteApp 权限配置文件测试 `com.newco.siteapp.data.modify` 授权。

```
# usermod -A com.newco.siteapp.data.modify -P SiteApp tester1
```

如果测试成功，管理员删除该授权。

```
# rolemod -A-=com.newco.siteapp.data.modify siteapptester
```

为简化维护工作，管理员将该授权添加到例 5-11 “向权限配置文件中添加授权” 中的 SiteApp 权限配置文件。

例 5-11 向权限配置文件中添加授权

测试授权可以正常使用后，安全管理员将 `com.newco.siteapp.data.modify` 授权添加到现有权限配置文件。例 5-7 “创建包含特权命令的权限配置文件” 显示了管理员创建配置文件的方法。

```
# profiles -p "SiteApp"
profiles:SiteApp> add auths="com.newco.siteapp.data.modify"
profiles:SiteApp> end
profiles:SiteApp> exit
```

为进行验证，管理员列出了该配置文件的内容。

```
# profiles -p SiteApp
Found profile in files repository.
  id=/opt/site-app/bin/site-cmd
  auths=com.newco.siteapp.data.modify
```

接下来的步骤 将权限配置文件指定给可信用户或角色。有关示例，请参见例 3-10 “创建可以管理 DHCP 的用户”和例 3-19 “使可信用户可以读取扩展记帐文件”。

另请参见 要对权限指定进行故障排除，请参见如何排除权限指定问题 [91]。有关背景的信息，请参见“所指定权限的搜索顺序” [31]。

将 root 更改为用户或角色

缺省情况下，`root` 是 Oracle Solaris 中的一个角色。您可以选择将它更改为一个用户，更改回角色，或者禁止使用。

如果您使用 Oracle Enterprise Manager 或采用传统的超级用户管理模型而不是权限模型，必须将 root 更改为一个用户。有关背景的信息，请参见“确定用于管理的权限模型” [35]。

如果您采用权限模型，在淘汰已从网络中删除的系统时，可能需要将 root 更改为用户。在此情况下，以 root 身份登录系统可简化清除过程。

注 - 如果管理员远程使用 root 角色，有关保护远程登录安全的说明，请参见《在 Oracle Solaris 11.2 中管理安全 Shell 访问 》中的“如何使用安全 Shell 远程管理 ZFS”。

在某些站点上，root 不是生产系统上的合法帐户。要禁止使用 root 帐户，请参见例 5-13 “防止 root 角色用于维护系统”。

▼ 如何将 root 角色更改为用户

在 root 必须可以直接登录系统的系统上，需要执行此过程。

开始之前 您必须承担 root 角色。

1. **从本地用户中删除 root 角色指定。**

   例如，从两个用户中删除角色指定。

```
% su -
Password: xxxxxxxx
# roles jdoe
root
# roles kdoe
root
# roles ldoe
secadmin
# usermod -R "" jdoe
# usermod -R "" kdoe
#
```

2. **将 root 角色更改为用户。**

```
# rolemod -K type=normal root
```

   当前承担 root 角色的用户保持不变，而具有 root 访问权限的其他用户可以使用 su 命令成为 root 或以 root 用户身份登录系统。

3. **检验更改。**

   您可以使用以下命令之一。

   - **检查 root 的 user_attr 项。**

```
# getent user_attr root
root::::auths=solaris.*;profiles=All;audit_flags=lo\:no;lock_after_retries=no;
min_label=admin_low;clearance=admin_high
```

如果输出中缺少 type 关键字或此关键字等于 normal，则此帐户不是一个角色。

- **查看 userattr 命令的输出。**

  ```
  # userattr type root
  ```

  如果输出为空或列出了 normal，则此帐户不是一个角色。

例 5-12 将 root 用户更改为 root 角色

在此示例中，root 用户将 root 用户恢复为角色。

首先，root 用户将 root 帐户更改为角色并验证此更改。

```
# usermod -K type=role root
# getent user_attr root
root::::type=role...
```

然后，root 将 root 角色指定给一个本地用户。

```
# usermod -R root jdoe
```

例 5-13 防止 root 角色用于维护系统

在以下示例中，站点安全策略要求应防止 root 帐户维护系统。管理员已创建和测试维护系统的角色。这些角色包括每个安全配置文件和 "System Administrator"（系统管理员）权限配置文件。已为一位可信用户指定了可以恢复备份的角色。没有任何角色可以更改用户、角色或权限配置文件的审计标志，或更改角色的口令。

为防止使用 root 帐户维护系统，安全管理员删除了 root 角色指定。由于 root 帐户必须能够以单用户模式登录到系统，所以该帐户保留了一个口令。

```
# usermod -K roles= jdoe
# userattr roles jdoe
```

故障排除 在桌面环境中，如果 root 为角色，则您无法以 root 身份直接登录。一条诊断消息会指出 root 在您的系统中为角色。

如果您没有可以承担 root 角色的本地帐户，请执行以下步骤：

- 以 root 身份在单用户模式下登录到系统，创建一个本地用户帐户和口令。
- 将 root 角色指定给新帐户。
- 以新用户的身份登录并承担 root 角色。

♦ ♦ ♦ 第 6 章

列出 Oracle Solaris 中的权限

本章介绍了如何列出系统上的所有权限、指定给特定用户的权限以及您自己的权限：

- “列出授权” [83]
- “列出权限配置文件” [84]
- “列出角色” [87]
- “列出特权” [87]
- “列出限定属性” [90]

有关权限的概述，请参见“用户权限管理” [14]。有关参考信息，请参见第 8 章 Oracle Solaris 权限参考信息。

列出权限及其定义

使用本节中的命令，您可以查找系统上定义的权限，并列出对用户进程有效的权限。有关本节中的命令的完整说明，请参见以下手册页：

- `auths(1)`
- `getent(1M)`
- `ppriv(1)`
- `profiles(1)`
- `privileges(5)`
- `roles(1)`

列出授权

- `auths` – 列出当前用户的授权
- `auths list` – 列出当前用户的授权
- `auths list -u` *username* – 列出 *username* 的授权

- `auths list -x` – 列出当前用户的需要验证的授权
- `auths list -xu` *username* – 列出 *username* 的需要验证的授权
- `auths info` – 列出命名服务中的所有授权名称
- `getent auth_attr` – 列出命名服务中所有授权的完整定义

例 6-1 列出所有授权

```
$ auths info
    solaris.account.activate
    solaris.account.setpolicy
    solaris.admin.edit
...
    solaris.zone.login
    solaris.zone.manage
```

例 6-2 列出授权数据库的内容

```
$ getent auth_attr | more
solaris.:::All Solaris Authorizations::help=AllSolAuthsHeader.html
solaris.account.:::Account Management::help=AccountHeader.html
...
solaris.zone.login:::Zone Login::help=ZoneLogin.html
solaris.zone.manage:::Zone Deployment::help=ZoneManage.html
```

例 6-3 列出用户的缺省授权

以下授权包含在缺省情况下指定给所有用户的权限配置文件中。

```
$ auths
solaris.device.cdrw,solaris.device.mount.removable,solaris.mail.mailq
solaris.network.autoconf.read,solaris.admin.wusb.read
solaris.smf.manage.vbiosd,solaris.smf.value.vbiosd
```

列出权限配置文件

- `profiles` – 列出当前用户的权限配置文件
- `profiles -a` – 列出所有权限配置文件名称
- `profiles -l` – 列出当前用户的权限配置文件的完整定义
- `profiles` *username* – 列出 *username* 的权限配置文件
- `profiles -x` – 列出当前用户的需要验证的权限配置文件
- `profiles -x` *username* – 列出 *username* 的需要验证的权限配置文件
- `profiles -p` *profile-name* `info` – 以优质打印方式输出指定权限配置文件的内容
- `getent prof_attr` – 列出命名服务中所有权限配置文件的完整定义

例 6-4 列出所有权限配置文件的名称

```
$ profiles -a
        Console User
        CUPS Administration
        Desktop Removable Media User
...
        VSCAN Management
        WUSB Management
```

例 6-5 列出权限配置文件数据库的内容

```
$ getent prof_attr | more
All:::Execute any command as the user or role:help=RtAll.html
Audit Configuration:::Configure Solaris Audit:auths=solaris.smf.value.audit;
help=RtAuditCfg.html
...
Zone Management:::Zones Virtual Application Environment Administration:
help=RtZoneMngmnt.html
Zone Security:::Zones Virtual Application Environment Security:auths=solaris.zone.*,
solaris.auth.delegate;help=RtZoneSecurity.html ...
```

例 6-6 列出用户的缺省权限配置文件

列出权限配置文件。缺省情况下会将以下权限配置文件指定给所有用户。

```
$ profiles
Basic Solaris User
All
```

例 6-7 列出初始用户的权限配置文件

为初始用户指定了多个权限配置文件。

```
$ profiles Initial user
System Administrator
Audit Review
...
CPU Power Management
Basic Solaris User
All
```

要显示指定给初始用户的配置文件的所有安全属性，请使用 -l 选项。

```
$ profiles -l Initial user | more
Initial user:
System Administrator
 profiles=Install Service Management,Audit Review,Extended Accounting
Flow Management,Extended Accounting Net Management,Extended Accounting Process
Management,Extended Accounting Task Management,Printer Management,Cron Managem
```

```
ent,Device Management,File System Management,Log Management,Mail Management,
Maintenance and Repair,Media Catalog,Name Service Management,Network Management,
Project Management,RAD Management,Service Operator,Shadow Migration Monitor,So
Software Installation,System Configuration,User Management,ZFS Storage Management
          /usr/sbin/gparted          uid=0
Install Service Management
 auths=solaris.autoinstall.service
 profiles=Install Manifest Management,Install Profile Management,
Install Client Management
...
```

例 6-8 列出指定的权限配置文件的内容

初始用户列出了由 "Audit Review"（审计查看）配置文件授予的权限。

```
$ profiles -l
Audit Review
  solaris.audit.read

  /usr/sbin/auditreduce  euid=0
  /usr/sbin/auditstat    privs=proc_audit
  /usr/sbin/praudit      privs=file_dac_read
```

例 6-9 列出权限配置文件中命令的安全属性

`profiles` 命令的变体可用于查看未指定给您的权限配置文件中命令的安全属性。

首先，列出配置文件中的命令。

```
% profiles -p "Audit Review" info
 name=Audit Review
 desc=Review Solaris Auditing logs
 help=RtAuditReview.html
 cmd=/usr/sbin/auditreduce
 cmd=/usr/sbin/auditstat
 cmd=/usr/sbin/praudit
```

然后，列出配置文件中某一个命令的安全属性。

```
% profiles -p "Audit Review" "select cmd=/usr/sbin/praudit ; info; end;"
select: command is read-only
 id=/usr/sbin/praudit
 privs=file_dac_read
end: command is read-only
```

例 6-10 列出最近创建的权限配置文件的内容

`less` 选项首先显示最近添加的权限配置文件。在您的站点创建或修改了权限配置文件时，`profiles` 命令的变体非常有用。以下输出显示了例 4-1 “将安全属性指定给传统应用程序” 中添加的配置文件的内容。一般用户可以运行此命令。

```
$ profiles -la | less
LegacyApp
          /opt/legacy-app/bin/legacy-cmd
                                      euid=0
OpenLDAP...
```

列出角色

- `roles` – 列出当前用户的角色
- `roles` *username* – 列出 *username* 的角色
- `logins -r` – 列出所有可用角色

例 6-11 列出指定的角色

`root` 角色在缺省情况下会指定给初始用户。`No roles` 表明没有为您指定任何角色。

```
$ roles
root
```

列出特权

- `man privileges` – 列出开发者使用的特权的定义及其名称
- `ppriv -vl` – 列出管理员使用的特权的定义及其名称
- `ppriv -vl basic` – 列出基本特权集合中特权的名称和定义
- `ppriv $$` – 列出当前 shell (`$$`) 中的特权
- `getent exec_attr` – 按照权限配置文件名称列出具有安全属性（`setuid` 或特权）的所有命令

```
$ getent exec_attr | more
All:solaris:cmd:::*:
Audit Configuration:solaris:cmd:::/usr/sbin/auditconfig:privs=sys_audit
...
Zone Security:solaris:cmd:::/usr/sbin/txzonemgr:uid=0
Zone Security:solaris:cmd:::/usr/sbin/zonecfg:uid=0 ...
```

例 6-12 列出所有特权及其定义

privileges(5) 手册页中说明的特权格式由开发者使用。

```
$ man privileges
```

```
Standards, Environments, and Macros                 privileges(5)

NAME
     privileges - process privilege model
...
     The defined privileges are:

     PRIV_CONTRACT_EVENT

         Allow a process to request reliable delivery  of  events
         to an event endpoint.

         Allow a process to include events in the critical  event
         set  term  of  a  template  which  could be generated in
         volume by the user.
...
```

例 6-13 列出特权指定中使用的特权

ppriv 命令按照名称列出所有特权。有关定义，请使用 -v 选项。

此特权格式用于通过使用 useradd、roleadd、usermod 和 rolemod 命令将特权指定给用户和角色，以及使用 profiles 命令将特权指定给权限配置文件。

```
$ ppriv -lv | more
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.
...
win_upgrade_sl
        Allows a process to set the sensitivity label of a window
        resource to a sensitivity label that dominates the existing
        sensitivity label.
        This privilege is interpreted only if the system is configured
        with Trusted Extensions.
```

例 6-14 列出当前 shell 中的特权

缺省情况下，将为每个用户指定基本的特权集合。缺省限制特权集合为所有特权。

输出中的单个字母指代以下特权集合：

E 有效特权集合

I 可继承特权集合

P 允许特权集合

L 有限特权集合

```
$ ppriv $$
1200:   -bash
flags = <none>
        E: basic
        I: basic
        P: basic
        L: all
$ ppriv -v $$
1200:   -bash
flags = <none>
 E: file_link_any,file_read,file_write,net_access,proc_exec,proc_fork,
          proc_info,proc_session,sys_ib_info
 I: file_link_any,file_read,...,sys_ib_info
 P: file_link_any,file_read,...,sys_ib_info
 L: contract_event,contract_identity,...,sys_time
```

双美元符号 (`$$`) 可将父 shell 的进程号传递给命令。此列表不包括在指定的权限配置文件中为命令限定的特权。

例 6-15 列出基本特权及其定义

```
$ ppriv -vl basic
file_link_any
 Allows a process to create hardlinks to files owned by a uid
 different from the process' effective uid.
file_read
 Allows a process to read objects in the filesystem.
file_write
 Allows a process to modify objects in the filesystem.
net_access
 Allows a process to open a TCP, UDP, SDP or SCTP network endpoint.
proc_exec
 Allows a process to call execve().
proc_fork
 Allows a process to call fork1()/forkall()/vfork()
proc_info
 Allows a process to examine the status of processes other
 than those it can send signals to.  Processes which cannot
 be examined cannot be seen in /proc and appear not to exist.
proc_session
 Allows a process to send signals or trace processes outside its
 session.
sys_ib_info
 Allows a process to perform read InfiniBand MAD (Management Datagram)
 operations.
```

例 6-16 列出权限配置文件中具有安全属性的命令

基本 Solaris 用户配置文件包括允许用户读取和写入 CD-ROM 的命令。

```
$ profiles -l
  Basic Solaris User
...
   /usr/bin/cdrecord.bin   privs=file_dac_read,sys_devices,
     proc_lock_memory,proc_priocntl,net_privaddr
   /usr/bin/readcd.bin     privs=file_dac_read,sys_devices,net_privaddr
   /usr/bin/cdda2wav.bin   privs=file_dac_read,sys_devices,
     proc_priocntl,net_privaddr
  All
   *
```

列出限定属性

- `man user_attr` – 定义安全属性的限定符
- `getent` – 列出运行命令的系统上用户或角色的限定安全属性
- `ldapaddent` – 列出用户或角色的所有限定安全属性

例 6-17 列出该系统上用户的限定属性

```
machine1$ getent user_attr | jdoe:
jdoe:machine1:::profiles=System Administrator
```

例 6-18 列出 LDAP 中用户的所有限定属性

```
machine1$ ldapaddent -d user_attr | grep ^jdoe:
jdoe:machine1:::profiles=System Administrator
jdoe:sysopgroup:::profiles=System Operator
```

♦ ♦ ♦ 第 7 章

7 排除 Oracle Solaris 中的权限问题

本章提供有关在 Oracle Solaris 中管理和使用管理权限时进行故障排除的建议：

- 如何排除权限指定问题 [91]
- 如何对指定的权限重新排序 [95]
- 如何确定程序所需的特权 [96]

有关使用权限的信息，请查看以下内容：

- 第 3 章 在 Oracle Solaris 中指定权限
- “权限指定者” [39]
- “用户权限管理” [14]
- “进程权限管理” [21]

排除权限问题

本节中的任务和示例将推荐几种解决权限指定问题的方法。有关背景信息，请参见“权限验证” [31]。

▼ 如何排除权限指定问题

若干因素会导致无法评估和正确应用权限。此过程有助于调试所指定的权限为何不能用于用户、角色或进程的问题。其中有多个步骤基于“所指定权限的搜索顺序” [31]。

开始之前 您必须承担 `root` 角色。有关详细信息，请参见“使用所指定的管理权限” [70]。

1. **确认并重新启动命名服务。**

   a. **验证用户或角色的安全指定是否位于在系统上启用的命名服务中。**

   ```
   # svccfg -s name-service/switch

   svc:/system/name-service/switch>
   listprop config
   ```

```
config                      application
config/value_authorization  astring  solaris.smf.value.name-service.switch
config/default              astring  files ldap
config/host                 astring  "files dns mdns ldap"
config/netgroup             astring  ldap
config/printer              astring  "user files"
```

在此输出中，所有未明确提及的服务均继承缺省值 `files ldap`。因此，将先在文件中搜索 `passwd` 和其相关属性数据库（`user_attr`、`auth_attr` 及 `prof_attr`），然后再在 LDAP 中搜索。

b. **重新启动名称服务高速缓存 `svc:/system/name-service/cache`。**

`nscd` 守护进程可以具有很长的生存时间间隔。通过重新启动此守护进程，可使用当前数据更新该命名服务。

```
# svcadm restart name-service/cache
```

2. **通过运行 `userattr -v` 命令确定为用户指定权限的位置。**

例如，以下命令指明了为用户 `jdoe` 指定的权限以及指定此分配的位置。无输出表示 `jdoe` 使用缺省值。

```
% userattr -v access_times jdoe
% userattr -v access_tz jdoe
% userattr -v auth_profiles jdoe
% userattr -v defaultpriv jdoe
% userattr -v limitpriv jdoe
% userattr -v idlecmd jdoe
% userattr -v idletime jdoe
% userattr -v lock_after_retries jdoe
% userattr -v pam_policy jdoe

% userattr -v auths jdoe       Output indicates authorizations from rights profiles
Basic Solaris User :solaris.mail.mailq,solaris.network.autoconf.read,
solaris.admin.wusb.read
Console User :solaris.system.shutdown,solaris.device.cdrw,
solaris.device.mount.removable,solaris.smf.manage.vbiosd,solaris.smf.value.vbiosd
% userattr -v audit_flags jdoe
user_attr: fw:no      Output indicates jdoe is individually assigned audit flags
# userattr -v profiles jdoe
user_attr: Audit Review,Stop      Output indicates two assigned rights profiles
# userattr roles jdoe
user_attr : cryptomgt,infosec      Output indicates two assigned roles
```

输出指示 `jdoe` 已直接分配有审计标志、两个权限配置文件和两个角色。指定的授权来自 `policy.conf` 文件中的缺省权限配置文件。

- 由于直接为 `jdoe` 指定了审计标志，因此将不使用权限配置文件中的审计标志值。
- 将按顺序评估权限配置文件，首先是 "Audit Review"（审计查看）权限配置文件，然后是 "Stop"（停止）配置文件。

- 指定给 jdoe 的其他所有权限位于角色 cryptomgt 和 infosec 中。要查看这些权限，jdoe 必须承担每个角色，然后列出这些权限。

如果权限未直接指定给用户，请继续以下检查。

3. **确认指定的授权是否拼写正确。**

   授权指定的来源并不重要，因为会为用户累积授权。不过，拼写错误的授权将失败，且不会进行提示。

4. **对于所创建的权限配置文件，请验证是否将相应的安全属性指定给该配置文件中的命令。**

   例如，要成功运行某些命令，需要 uid=0，而非 euid=0。要判断命令或其任何选项是否需要授权，请查看命令的手册页。

5. **检查用户权限配置文件中的权限。**

   a. **按顺序检查需要验证权限配置文件列表中的权限。**

      该列表中最早的权限配置文件中的属性值是内核中的值。如果该值不正确，请在该权限配置文件中更改该值，或者按正确顺序重新指定配置文件。请参见如何对指定的权限重新排序 [95]。

      对于特权命令，检查未使用 defaultpriv 或 limitpriv 关键字删除特权。

   b. **按顺序检查常规权限配置文件列表中的权限。**

      执行的检查与对需要验证权限配置文件执行的检查相同。

   c. **如果未列出您搜索的权限，请检查为用户指定的角色。**

      如果权限已指定给一个角色，则该用户只有承担该角色才能获取权限。

6. **检查失败的命令是否需要授权才能成功执行。**

   a. **检查现有的权限配置文件是否包含需要的授权。**

      如果存在，请使用该配置文件。将其作为需要验证权限配置文件或常规权限配置文件指定给用户。将该配置文件放在需要该授权才能成功执行的命令所在的其他所有权限配置文件之前。

   b. **检查命令的选项是否需要授权。**

      将特权指定给需要它的命令，添加所需的授权，将命令和授权置于一个权限配置文件中，然后将该配置文件指定给用户。

7. **如果用户仍无法成功执行命令，请确认用户是否在配置文件 shell 中执行该命令。**

   必须在配置文件 shell 中执行管理命令。例 7-1 “判断是否使用配置文件 shell” 说明如何测试配置文件 shell。

要降低用户错误的可能性，可尝试以下操作：

- 指定一个配置文件 shell 作为用户登录 shell。
- 告知用户在所有特权命令之前加上 `pfexec` 命令。
- 提醒用户在配置文件 shell 中运行管理命令。
- 如果站点使用角色，提醒用户在运行管理命令之前承担相应角色。有关以角色而非用户成功执行命令的示例，请参见例 7-3 “运行您的角色中的特权命令”。

8. **如果命令因角色问题而失败，请承担该角色并执行在检查用户权限时执行的相同步骤。**

例 7-1 判断是否使用配置文件 shell

当特权命令不起作用时，用户可测试 `PRIV_PFEXEC` 标志，然后运行该命令。错误消息可能不会指明该问题为特权问题。

```
% praudit 20120814200247.20120912213421.example-system
praudit: Cannot associate stdin with 20120814200247.20120912213421.example-system:
Permission denied

% ppriv $$
107219: bash
flags = <none>
...

% pfbash
# ppriv $$
1072232: bash
flags = PRIV_PFEXEC
...

# praudit 20120814200247.20120912213421.example-system
     /** Command succeeds **/
```

例 7-2 确定角色的特权命令

在此示例中，用户将承担指定的角色并列出其中一个权限配置文件中包括的权限。已将权限截去以强调命令。

```
% roles
devadmin

% su - devadmin
Password: xxxxxxxx

# profiles -l
Device Security
        ...
 profiles=Service Configuration
          /usr/sbin/add_drv          uid=0
```

```
          /usr/sbin/devfsadm         uid=0
                                     privs=sys_devices,sys_config,
                                           sys_resource,file_owner,
                                           file_chown,file_chown_self,
                                           file_dac_read
          /usr/sbin/eeprom           uid=0
          /usr/bin/kbd
          /usr/sbin/list_devices     euid=0
          /usr/sbin/rem_drv          uid=0
          /usr/sbin/strace           euid=0
          /usr/sbin/update_drv       uid=0
          /usr/sbin/add_allocatable  euid=0
          /usr/sbin/remove_allocatable   euid=0
Service Configuration
          /usr/sbin/svcadm
          /usr/sbin/svccfg
```

例 7-3 运行您的角色中的特权命令

在以下示例中，admin 角色可以更改 useful.script 文件的权限。

```
% whoami
jdoe
% ls -l useful.script
-rwxr-xr-- 1 elsee eng 262 Apr 2 10:52 useful.script

% chgrp admin useful.script
chgrp: useful.script: Not owner

% su - admin
Password: xxxxxxxx

# chgrp admin useful.script
# chown admin useful.script
# ls -l useful.script
-rwxr-xr-- 1 admin admin 262 Apr 2 10:53 useful.script
```

▼ 如何对指定的权限重新排序

对用户有效的是非特权命令而不是其特权版本时，您必须对用户的权限配置文件指定进行重新排序。有关详细信息，请参见“所指定权限的搜索顺序” [31]。

开始之前 您必须是指定有 "User Security"（用户安全）权限配置文件的管理员。有关详细信息，请参见“使用所指定的管理权限” [70]。

1. 查看当前指定给用户或角色的权限配置文件的列表。
   该列表将按顺序显示。

   ```
   % profiles username | rolename
   ```

2. 按正确的顺序指定权限配置文件。

```
# usermod | rolemod -P "list-of-profiles"
```

例 7-4 按特定的顺序指定权限配置文件

在此示例中，管理员决定将包含特权命令的权限配置文件列出在角色 devadmin 的所有权限配置文件之后。

```
# profiles devadmin

    Basic Solaris User
    All
    Device Management
```

因此，devadmin 角色无法使用为角色指定的特权运行设备管理命令。

管理员重新为 devadmin 指定权限配置文件。按照新的指定顺序，可以使用指定的特权运行设备管理命令。

```
# rolemod -P "Device Management,Basic Solaris User,All"

# profiles devadmin

    Device Management
    Basic Solaris User
    All
```

▼ 如何确定程序所需的特权

可在命令或进程失败时使用此调试过程。在找到并修复第一个特权问题之后，您可能需要再次运行 ppriv -eD *command* 命令以查找其他特权要求。

1. 键入失败的命令作为 **ppriv** 调试命令的参数。

```
% ppriv -eD touch /etc/acct/yearly

touch[5245]: missing privilege "file_dac_write"
     (euid = 130, syscall = 224) needed at zfs_zaccess+0x258
touch: cannot create /etc/acct/yearly: Permission denied
```

2. 使用调试输出的 **syscall** 编号来确定哪个系统调用失败。

   在 /etc/name_to_sysnum 文件中查找 syscall 编号的名称。

```
% grep 224 /etc/name_to_sysnum

creat64                 224
```

在此示例中，creat64（） 调用失败。要成功执行进程，必须为该进程指定可在 /etc/acct/yearly 目录中创建文件的权限。

例 7-5 使用 truss 命令检查特权使用

truss 命令可以在常规 shell 中调试特权使用。例如，以下命令调试失败的 touch 进程：

```
% truss -t creat touch /etc/acct/yearly

creat64("/etc/acct/yearly", 0666)
                      Err#13 EACCES [file_dac_write
]
touch: /etc/acct/yearly cannot create
```

扩展的 /proc 接口在 truss 输出中的错误代码后面报告缺少 file_dac_write 特权。

例 7-6 使用 ppriv 命令在配置文件 Shell 中检查特权使用

在此示例中，jdoe 用户可以承担角色 objadmin。objadmin 角色拥有 "Object Access Management"（对象访问管理）权限配置文件。使用此权限配置文件，objadmin 角色可以更改不属于 objadmin 的文件的权限。

在以下摘录中，jdoe 无法更改 useful.script 文件的权限：

```
jdoe% ls -l useful.script

-rw-r--r--  1 aloe  staff  2303 Apr 10 10:10 useful.script
jdoe%
chown objadmin useful.script

chown: useful.script: Not owner
jdoe%
ppriv -eD chown objadmin useful.script

chown[11444]: missing privilege "file_chown"
            (euid = 130, syscall = 16) needed at zfs_zaccess+0x258
chown: useful.script: Not owner
```

当 jdoe 承担 objadmin 角色时，更改了该文件的权限：

```
jdoe% su - objadmin
Password: xxxxxxxx

# ls -l useful.script
-rw-r--r--  1 aloe  staff  2303 Apr 10 10:10 useful.script

# chown objadmin useful.script
# ls -l useful.script
-rw-r--r--  1 objadmin  staff  2303 Apr 10 10:10 useful.script
```

```
# chgrp admin useful.script

# ls -l objadmin.script
-rw-r--r--  1 objadmin  admin  2303 Apr 10 10:11 useful.script
```

例 7-7 更改 root 用户拥有的文件

此示例说明了防止特权升级的方法。有关讨论，请参见“特权升级和内核特权” [30]。此文件归 root 用户所有。由于权限较低的 objadmin 角色需要所有特权才能更改文件的所有权，因此操作失败。

```
jdoe% su - objadmin
Password: xxxxxxxx

# cd /etc; ls -l system
-rw-r--r--  1 root  sys   1883 Oct 10 10:20 system

# chown objadmin system
chown: system: Not owner
# ppriv -eD chown objadmin system
chown[11481]: missing privilege "ALL"
     (euid = 101, syscall = 16) needed at zfs_zaccess+0x258
chown: system: Not owner
```

♦ ♦ ♦ 第 8 章

Oracle Solaris 权限参考信息

本章提供了使用 Oracle Solaris 中的管理权限的参考资料：

- “权限配置文件参考信息” [99]
- “授权参考信息” [100]
- “权限数据库” [101]
- “权限管理命令” [105]
- “特权参考信息” [107]

有关使用权限（包括特权）的信息，请参见第 3 章 在 Oracle Solaris 中指定权限。有关概述信息，请参见“用户权限管理” [14]和“进程权限管理” [21]。

权限配置文件参考信息

本节介绍了一些典型的权限配置文件。权限配置文件是以下对象的便利集合：授权和其他安全属性、具有安全属性的命令以及补充权限配置文件。Oracle Solaris 提供了多种权限配置文件。如果这些文件无法满足您的需要，您可以修改现有权限配置文件，创建新的权限配置文件。

权限配置文件必须按权限从高到低的顺序指定。有关详细信息，请参见“所指定权限的搜索顺序” [31]。

要查看以下权限配置文件的内容，请参见“查看权限配置文件的内容” [100]。

- **"System Administrator"（系统管理员）权限配置文件** – 提供与安全性无关的大多数任务的访问权限。此配置文件包括一些可用于创建功能强大的角色的其他配置文件。请注意，将在补充权限配置文件列表的末尾指定 "All"（所有）权限配置文件。
- **"Operator"（操作员）权限配置文件** – 提供用于管理文件和脱机介质的有限权限。此配置文件包括可用于创建简单角色的补充权限配置文件。
- **"Printer Management"（打印机管理）权限配置文件** – 提供用于处理打印的数量有限的命令和授权。此配置文件是涉及单个管理区域的若干个配置文件之一。
- **"Basic Solaris User"（基本 Solaris 用户）权限配置文件** – 使用此配置文件，用户可以在安全策略范围内使用系统。缺省情况下，会在 `policy.conf` 文件中列出此配置文件。请注意，"Basic Solaris User"（基本 Solaris 用户）权限配置文件提供

的便利必须与站点的安全要求平衡。需要更严格的安全性的站点可能更倾向于从 `policy.conf` 文件中删除此配置文件，或者指定 "Stop"（停止）权限配置文件。有关 "Basic Solaris User"（基本 Solaris 用户）权限配置文件的信息，请参见例 6-16 “列出权限配置文件中具有安全属性的命令”。

- **"Console User"（控制台用户）权限配置文件** – 针对工作站所有者，为计算机前的用户提供授权、命令和操作的访问权限。
- **"All"（所有）权限配置文件** – 针对角色，提供访问不具有安全属性的命令的权限。此配置文件适用于具有有限权限的用户。
- **"Stop"（停止）权限配置文件** – 是一个特殊的权限配置文件，可停止对其他权限配置文件的评估。此配置文件可阻止对 `policy.conf` 文件中的 `AUTHS_GRANTED`、`PROFS_GRANTED` 和 `CONSOLE_USER` 变量进行评估。通过此配置文件，可为角色和用户提供受限制的配置文件 shell。

注 - "Stop"（停止）权限配置文件间接影响特权指定。列在 "Stop"（停止）配置文件之后的权限配置文件不会被评估。因此，使用这些配置文件中特权的命令不会生效。请参见例 3-25 “限制管理员使用显式指定的权限”。

每个权限配置文件都有关联的帮助文件。这些帮助文件以 HTML 格式提供，并且可定制。这些文件位于 `/usr/lib/help/profiles/locale/C` 目录下。

查看权限配置文件的内容

您可以通过三种方式查看权限配置文件的内容：

- 使用 `getent` 命令，可以查看系统中所有权限配置文件的内容。有关样例输出，请参见第 6 章 列出 Oracle Solaris 中的权限。
- 使用 `profiles -p "`*Profile Name*`" info` 命令，可以查看特定权限配置文件的内容。
- 使用 `profiles -l` *account-name* 命令，可以查看指定给特定用户或角色的权限配置文件的内容。

有关更多信息，请参见第 6 章 列出 Oracle Solaris 中的权限和 `getent`(1M) 及 `profiles`(1) 手册页。

授权参考信息

授权是可以授予角色或用户的独立权限。在用户获取对应用程序或应用程序内特定操作的访问权限之前，将通过合规应用程序检查授权。

授权是用户级的，因此可以扩展。可以编写需要授权的程序，为系统添加授权，为这些授权创建权限配置文件，并将该权限配置文件指给允许使用该程序的用户或角色。

授权命名约定

授权具有内部使用的名称。例如，`solaris.system.date` 是一个授权的名称。授权具有简短说明，此说明显示在图形用户界面 (graphical user interface, GUI) 中。例如，`Set Date & Time` 是 `solaris.system.date` 授权的说明。

根据约定，授权名称由供应商 Internet 名称（顺序颠倒过来）、主题区域、任何子区域以及功能组成。授权名称的各个部分以点分隔。例如，`com.xyzcorp.device.access` 便是一个授权名称。此约定的例外是 Oracle 的授权，它使用前缀 `solaris` 而不是 Internet 名称。使用命名约定，管理员可以用层次化方式应用授权。通配符 (*) 可以表示点右侧的所有字符串。

以下示例说明了授权使用方式："Network Link Security"（网络链路安全）权限配置文件只有 `solaris.network.link.security` 授权，而 "Network Security"（网络安全）权限配置文件以 "Network Link Security"（网络链路安全）配置文件为补充配置文件，另外加上了 `solaris.network.*` 和 `solaris.smf.manage.ssh` 授权。

授权中的委托授权

使用以后缀 `delegate` 结尾的授权，用户或角色可将以相同前缀开头的任何指定授权委托给其他用户。

使用 `solaris auth.delegate` 授权，用户或角色可将指定给授权用户或角色的任何授权委托给其他用户。例如，拥有 `solaris auth.delegate` 和 `solaris.network.wifi.wep` 授权的角色可将 `solaris.network.wifi.wep` 授权委托给其他用户或角色。

权限数据库

以下数据库存储 Oracle Solaris 中的权限数据：

- **扩展用户属性数据库** (`user_attr`) – 与其他关键字结合使用，将用户和角色与授权、特权和权限配置文件关联。
- **权限配置文件属性数据库** (`prof_attr`) – 定义权限配置文件，列出配置文件的指定授权、特权和关键字，并指明关联的帮助文件
- **授权属性数据库** (`auth_attr`) – 定义授权及其属性，并指明关联的帮助文件

- **执行属性数据库** (`exec_attr`) – 标识指定给特定权限配置文件的具有安全属性的命令

`policy.conf` 数据库包含应用于所有用户的授权、特权和权限配置文件。有关更多信息，请参见“`policy.conf` 文件” [104]。

权限数据库和命名服务

在 SMF 服务中使用命名服务切换参数 `svc:/system/name-service/switch` 定义权限数据库的名称服务范围。此服务中针对权限数据库的属性有 `auth_attr`、`password` 和 `prof_attr`。`password` 属性设置 `passwd` 和 `user_attr` 数据库的命名服务优先级。`prof_attr` 属性设置 `prof_attr` 和 `exec_attr` 数据库的命名服务优先级。

在以下输出中，`auth_attr`、`password` 和 `prof_attr` 条目未列出。因此，权限数据库使用的是 `files` 命名服务。

```
# svccfg -s name-service/switch listprop config
config                          application
config/value_authorization      astring       solaris.smf.value.name-service.switch
config/default                  astring       files
config/host                     astring       "files ldap dns"
config/printer                  astring       "user files ldap"
```

`user_attr` 数据库

`user_attr` 数据库包含补充 `passwd` 和 `shadow` 数据库的用户和角色信息。*attr* 字段包含安全属性，*qualifier* 字段包含限定或限制安全属性对系统或系统组的影响的属性。

使用 `roleadd`、`rolemod`、`useradd`、`usermod` 和 `profiles` 命令可以设置 `attr` 字段中的安全属性。可以在本地和 LDAP 命名范围中设置这些属性。

- 对于用户，`roles` 关键字指定一个或多个已定义的角色。
- 对于角色，对 `roleauth` 关键字使用 `user` 值可允许角色使用用户口令而不是角色口令进行验证。缺省情况下，值为 `role`。
- 对于用户或角色，可设置以下属性：
  - `access_times` 关键字 – 指定日期和时间，用于指定应用程序和服务何时可以访问。有关更多信息，请参见 `getaccess_times(3C)` 手册页。
  - `access_tz` 关键字 – 指定解释 `access_times` 条目中的时间时使用的时区。有关更多信息，请参见 `pam_unix_account(5)` 手册页。
  - `audit_flags` 关键字 – 修改审计掩码。有关更多信息，请参见 `audit_flags(5)` 手册页。

- auths 关键字 – 指定授权。有关更多信息，请参见 auths(1) 手册页。
- auth_profiles 关键字 – 指定需要验证权限配置文件。有关参考信息，请参见 profiles(1) 手册页。
- defaultpriv 关键字 – 添加特权或者从缺省的基本特权集合中删除特权。
- limitpriv 关键字 – 添加特权或者从缺省的限制特权集合中删除特权。

  因为 defaultpriv 和 limitpriv 特权会指定给用户的初始进程，因此它们始终有效。有关更多信息，请参见 privileges(5) 手册页和“如何实现特权” [24]。
- idlecmd 关键字 – 达到 idletime 后注销用户或锁定屏幕。
- idletime 关键字 – 设置没有键盘活动后系统仍处于可用状态的时间。如果为 idlecmd 指定值，请设置 idletime。
- lock_after_retries 关键字 – 如果值为 yes，当重试次数超出 /etc/default/login 文件中允许的次数后，系统会锁定。有关更多信息，请参见 login(1) 手册页。
- profiles 关键字 – 指定权限配置文件。有关更多信息，请参见 profiles(1) 手册页。
- project 关键字 – 添加缺省项目。有关更多信息，请参见 project(4) 手册页。

注 - 因为 access_times 和 access_tz 属性是 PAM 属性，因此在验证过程中将检查这些属性。因此，必须直接将它们指定给用户或角色，或者在一个需要验证权限配置文件中指定。在一般权限配置文件中将忽略这些属性。

只能为在 LDAP 命名范围中的用户和角色设置这些限定属性。这些限定符将用户或角色的属性指定（例如权限配置文件）限定为一个或多个系统。有关示例，请参见 useradd(1M) 和 user_attr(4) 手册页。

限定符包括 host 和 netgroup。

- host 限定符 – 标识用户或角色可以执行指定操作的系统。
- netgroup 限定符 – 列出用户或角色可以执行指定操作的系统。host 指定优先于 netgroup 指定。

有关更多信息，请参见 user_attr(4) 手册页。要查看此数据库的内容，请使用 getent user_attr 命令。有关更多信息，请参见 getent(1M) 手册页和第 6 章 列出 Oracle Solaris 中的权限。

auth_attr 数据库

auth_attr 数据库存储授权定义。可以将授权指定给用户、角色或权限配置文件。首选方法是将授权添加到一个权限配置文件，然后将该权限配置文件指定给角色或用户。

要查看此数据库的内容，请使用 getent auth_attr 命令。有关更多信息，请参见 getent(1M) 手册页和第 6 章 列出 Oracle Solaris 中的权限。

prof_attr 数据库

prof_attr 数据库存储指定给权限配置文件的名称、说明、帮助文件位置、特权以及授权。指定给权限配置文件的命令和安全属性存储在 exec_attr 数据库中。有关更多信息，请参见 “exec_attr 数据库” [104]。

有关更多信息，请参见 prof_attr(4) 手册页。要查看此数据库的内容，请使用 getent exec_attr 命令。有关更多信息，请参见 getent(1M) 手册页和第 6 章 列出 Oracle Solaris 中的权限。

exec_attr 数据库

exec_attr 数据库定义需要安全属性才能成功运行的命令。这些命令是权限配置文件的一部分。具有安全属性的命令可以由为其指定了此配置文件的角色或用户运行。

有关更多信息，请参见 exec_attr(4) 手册页。要查看此数据库的内容，请使用 getent 命令。有关更多信息，请参见 getent(1M) 手册页和第 6 章 列出 Oracle Solaris 中的权限。

policy.conf 文件

/etc/security/policy.conf 文件提供了一种向系统的所有用户授予特定权限配置文件、特定授权和特定特权的方法。文件中的相关项由 *key*=*value* 对组成：

- AUTHS_GRANTED=*authorizations* – 指一个或多个授权。
- AUTH_PROFS_GRANTED=*rights profiles* – 指一个或多个需要验证权限配置文件。
- PROFS_GRANTED=*rights profiles* – 指一个或多个无验证权限配置文件。
- CONSOLE_USER=Console User – 指 "Console User"（控制台用户）权限配置文件。此配置文件附带了一组便利的控制台用户授权。可以定制此配置文件。
- PRIV_DEFAULT=*privileges* – 指一个或多个特权。
- PRIV_LIMIT=*privileges* – 指所有特权。

以下示例显示了 policy.conf 数据库中的一些权限值：

```
##
AUTHS_GRANTED=
AUTH_PROFS_GRANTED=
CONSOLE_USER=Console User
PROFS_GRANTED=Basic Solaris User
#PRIV_DEFAULT=basic
#PRIV_LIMIT=all
```

权限管理命令

本节列出了用于管理权限的命令，还包括一个命令表，其中命令的访问可以由授权控制。

管理授权、权限配置文件和角色的命令

下表中列出的命令检索和设置用户进程的权限。

表 8-1 权限管理命令

命令	说明
auths(1)	显示用户的授权。创建新授权。
getent(1M)	列出权限数据库的内容。
nscd(1M)	名称服务高速缓存守护进程，适用于缓存权限数据库。使用 `svcadm` 命令重新启动守护进程。
pam_roles(5)	PAM 的角色帐户管理模块。检查承担角色的授权。
pam_unix_account(5)	PAM 的 UNIX 帐户管理模块。检查帐户限制，例如时间限制和不活动时间。
pfbash(1)	用于创建可以评估权限的配置文件 shell 进程。
pfedit(1M)	用于编辑管理文件。
pfexec(1)	用于执行具有安全属性的命令。
policy.conf(4)	系统安全策略的配置文件。列出授予的授权、授予的特权和其他安全信息。
profiles(1)	显示某个指定用户的权限配置文件。创建或修改权限配置文件。
roles(1)	显示指定用户可以承担的角色。
roleadd(1M)	向本地系统或 LDAP 网络中添加角色。
roleadd(1M)	向本地系统或 LDAP 网络中添加角色。
rolemod(1M)	修改本地系统或 LDAP 网络中角色的属性。
userattr(1)	显示指定给用户或角色帐户的特定权限的值。

命令	说明
useradd(1M)	向系统或 LDAP 网络中添加用户帐户。-R 选项将角色指定给用户帐户。
userdel(1M)	从系统或 LDAP 网络中删除用户登录帐户。
usermod(1M)	修改系统中的用户帐户属性。

需要授权的命令（摘选）

下表提供了在 Oracle Solaris 系统上如何使用授权限制命令选项的示例。有关授权的更多讨论，请参见“授权参考信息” [100]。

表 8-2 命令和关联的授权

命令	授权要求
at(1)	所有选项都需要 solaris.jobs.user（at.allow 和 at.deny 文件都不存在时）
atq(1)	所有选项都需要 solaris.jobs.admin
cdrw(1)	所有选项都需要 solaris.device.cdrw，缺省情况下在 policy.conf 文件中授予
crontab(1)	提交作业的选项需要 solaris.jobs.user（crontab.allow 和 crontab.deny 文件都不存在时） 列出或修改其他用户的 crontab 文件的选项需要 solaris.jobs.admin
allocate(1)	分配设备需要 solaris.device.allocate（或在 device_allocate 文件中指定的其他授权） 将设备分配给其他用户（-F 选项）需要 solaris.device.revoke（或在 device_allocate 文件中指定的其他授权）
deallocate(1)	取消其他用户的设备分配需要 solaris.device.allocate（或在 device_allocate 文件中指定的其他授权） 强制取消指定设备的分配（-F 选项）或所有设备的分配（-I 选项）需要 solaris.device.revoke（或在 device_allocate 中指定的其他授权）
list_devices(1)	列出其他用户的设备（-U 选项）需要 solaris.device.revoke
roleadd(1M)	创建角色需要 solaris.user.manage。设置初始口令需要 solaris.account.activate。设置口令策略（如帐户锁定和口令有效期）需要 solaris.account.setpolicy。
roledel(1M)	删除口令需要 solaris.passwd.assign 授权。
rolemod(1M)	更改口令需要 solaris.passwd.assign 授权。更改口令策略（如帐户锁定和口令有效期）需要 solaris.account.setpolicy。
sendmail(1M)	访问邮件子系统功能需要 solaris.mail；查看邮件队列需要 solaris.mail.mailq
useradd(1M)	创建用户需要 solaris.user.manage。设置初始口令需要 solaris.account.activate。设置口令策略（如帐户锁定和口令有效期）需要 solaris.account.setpolicy。
userdel(1M)	删除口令需要 solaris.passwd.assign 授权。
usermod(1M)	更改口令需要 solaris.passwd.assign 授权。更改口令策略（如帐户锁定和口令有效期）需要 solaris.account.setpolicy。

特权参考信息

特权限制进程是在内核中实现的，可以在命令、用户、角色和系统级别对进程加以限制。

用于处理特权的命令

下表列出了处理特权可以使用的命令。

表 8-3 用于处理特权的命令

用途	命令	手册页
对特权应用失败进行调试	`ppriv -eD` *failed-operation*	ppriv(1)
列出系统上的特权	`ppriv -l`	ppriv(1)
列出特权及其说明	`ppriv -lv` *priv*	ppriv(1)
列出 UID、进程或端口上的扩展的特权策略	`ppriv -lv` *extended-policy*	ppriv(1)
检查进程特权	`ppriv -v` *pid*	ppriv(1)
向 UID、进程或端口添加扩展特权策略	`ppriv -r` *rule*	privileges(5)
设置进程特权	`ppriv -s` *spec*	ppriv(1)
删除扩展的特权策略规则	`ppriv -X` *rule*	privileges(5)
为权限配置文件指定特权	`profiles -p` *profile-name*	profiles(1)
为新角色指定特权	`roleadd -K defaultpriv=`	roleadd(1M)
为现有角色添加特权	`rolemod -K defaultpriv+=`	rolemod(1M)
为新用户指定特权	`useradd -K defaultpriv=`	useradd(1M)
为现有用户添加特权	`usermod -K defaultpriv+=`	usermod(1M)
向设备添加设备策略	`add_drv -p` *policy driver*	add_drv(1M)
设置设备策略	`devfsadm`	devfsadm(1M)
查看设备策略	`getdevpolicy`	getdevpolicy(1M)
在打开设备上更新设备策略	`update_drv -p` *policy driver*	update_drv(1M)

包含特权信息的文件

`policy.conf` 和 `syslog.conf` 文件包含特权相关信息。

- `/etc/security/policy.conf` 包含以下特权信息：

- PRIV_DEFAULT – 系统的可继承特权集合
- PRIV_LIMIT – 系统的限制特权集合

有关更多信息，请参见 policy.conf(4) 手册页。

- /etc/syslog.conf 是包含与特权调试相关的调试消息的系统日志文件。在 priv.debug 项中设置调试消息的路径。

  有关更多信息，请参见 syslog.conf(4) 手册页。

审计记录中的特权操作

可以对特权的使用进行审计。只要进程使用特权，就会在 upriv 审计标记的审计迹中记录该特权的使用。如果记录中包含特权名称，则将使用特权名称的文本表示形式。以下审计事件记录特权的使用：

- AUE_SETPPRIV **审计事件** – 更改了特权集合时，生成一条审计记录。AUE_SETPPRIV 审计事件在 pm 类中。
- AUE_MODALLOCPRIV **审计事件** – 从内核外部添加了特权时，生成一条审计记录。AUE_MODALLOCPRIV 审计事件在 ad 类中。
- AUE_MODDEVPLCY **审计事件** – 更改了设备策略时，生成一条审计记录。AUE_MODDEVPLCY 审计事件在 ad 类中。
- AUE_PFEXEC **审计事件** – 如果在启用 pfexec（） 时调用了 execve（），生成一条审计记录。AUE_PFEXEC 审计事件在 as、ex、ps 和 ua 审计类中。特权的名称包含在审计记录中。

不会审计基本特权集合中特权的成功使用。但是如果尝试使用已从用户基本特权集合中删除的基本特权，将会生成审计记录。

安全词汇表

Access Control List, ACL（访问控制列表）	与传统的 UNIX 文件保护相比，访问控制列表 (access control list, ACL) 可提供更为精细的文件安全性。例如，通过 ACL 可以让组获得对某个文件的读取权限，而仅允许该组中的一个成员获得对该文件的写入权限。
admin principal（admin 主体）	名称形式为 *username*`/admin` 的用户主体（如 `jdoe/admin`）。与一般用户主体相比，管理主体可以拥有更多特权（例如，可以更改策略）。另请参见 principal name（主体名称）和 user principal（用户主体）。
AES	Advanced Encryption Standard（高级加密标准）。一种对称的 128 位块数据加密技术。美国政府在 2000 年 10 月采用该种算法的 Rijndael 变体作为其加密标准。AES 从而取代了 user principal（用户主体）加密方法成为政府的加密标准。
algorithm（算法）	加密算法。这是一种确立的递归计算过程，用于对输入执行加密或散列操作。
application server（应用服务器）	请参见 network application server（网络应用服务器）。
asynchronous audit event（异步审计事件）	异步事件在系统事件中属于少数。这些事件不与任何进程关联，因此没有任何进程可供阻塞并在以后唤醒。例如，初始系统引导和 PROM 进入和退出事件都是异步事件。
audit files（审计文件）	二进制审计日志。审计文件单独存储在一个审计文件系统中。
audit policy（审计策略）	决定要记录的审计事件的全局设置和按用户设置。通常，应用于审计服务的全局设置会影响审计迹所包括的可选信息。`cnt` 和 `ahlt` 这两个设置会影响系统在填充审计队列时执行的操作。例如，审计策略可能要求每条审计记录都包含一个序列号。
audit trail（审计迹）	来自所有主机的所有审计文件的集合。
authenticated rights profile（需要验证权限配置文件）	指定有这类 rights profile（权限配置文件）的用户或角色执行配置文件中的操作之前需要键入口令。此行为类似于 `sudo` 行为。口令的有效期可配置。

authentication（验证）	验证主体所声明的身份的过程。
authenticator（验证者）	当客户机从 KDC 请求票证以及从服务器请求服务时，会传递验证者。这些验证者包含使用仅对客户机和服务器公开的会话密钥所生成的信息，这些信息可以作为最新来源进行检验，从而表明事务是安全的。验证者可与票证一起使用来验证用户主体。验证者中包括用户的主体名称、用户主机的 IP 地址，以及时间戳。与票证不同，验证者只能使用一次，通常在请求访问服务时使用。验证者是使用特定客户机和服务器的会话密钥进行加密的。
authorization（授权）	1. 在 Kerberos 中，是指决定主体是否可以使用服务，允许主体访问哪些对象，以及可对每个对象执行的访问操作类型的过程。 2. 在用户权限管理中，是指可以指定给角色或用户（或嵌入权限配置文件中）的权限，此权限用于执行安全策略原本禁止的一类操作。授权在用户应用程序级别（而不是内核级别）实施。
basic set（基本特权集合）	登录时为用户进程指定的特权集合。在未修改的系统上，每个用户的初始可继承特权集合等同于登录时获取的基本特权集合。
Blowfish	一种对称块加密算法，它采用 32 位到 448 位的可变长度密钥。其作者 Bruce Schneier 声称 Blowfish 已针对密钥不经常更改的应用程序进行优化。
client principal（客户机主体）	(RPCSEC_GSS API) 是指使用受 RPCSEC_GSS 保护的网络服务的客户机（用户或应用程序）。客户机主体名称将以 `rpc_gss_principal_t` 结构的形式进行存储。
client（客户机）	狭义上讲，是指代表用户使用网络服务的进程，例如，使用 `rlogin` 的应用程序。在某些情况下，服务器本身即可是其他某个服务器或服务的客户机。 广义上讲，是指 a) 接收 Kerberos 凭证的主机，以及 b) 使用由服务器提供的服务的主机。 非正式地讲，是指使用服务的主体。
clock skew（时钟相位差）	所有参与 Kerberos 验证系统的主机上的内部系统时钟可以相差的最大时间量。如果任意两台参与主机之间的时间偏差超过了时钟相位差，则请求会被拒绝。可以在 `krb5.conf` 文件中指定时钟相位差。
confidentiality（保密性）	请参见 privacy（保密性）。
consumer（使用者）	在 Oracle Solaris 的加密框架功能中，使用者是指使用提供者提供的加密服务的用户。使用者可以是应用程序、最终用户或内核操作。例如，Kerberos、IKE 和 IPsec 便属于使用者。有关提供者的示例，请参见 provider（提供者）。

credential cache（凭证高速缓存）	包含从 KDC 接收的凭证的存储空间（通常为文件）。
credential（凭证）	包括票证及匹配的会话密钥的信息软件包。用于验证主体的身份。另请参见 ticket（票证）和 session key（会话密钥）。
cryptographic algorithm（密码算法）	请参见 algorithm（算法）。
DES	Data Encryption Standard（数据加密标准）。一种对称密钥加密方法，开发于 1975 年，1981 年由 ANSI 标准化为 ANSI X.3.92。DES 使用 56 位密钥。
device allocation（设备分配）	用户级别的设备保护。设备分配强制规定一次只能由一个用户独占使用一台设备。重用设备之前，将清除设备数据。可以使用授权来限制允许分配设备的用户。
device policy（设备策略）	内核级别的设备保护。设备策略在设备上作为两个特权集合实现。一个特权集合控制对设备的读取权限，另一个特权集合控制对设备的写入权限。另请参见 policy（策略）。
Diffie-Hellman protocol（Diffie-Hellman 协议）	也称为公钥密码学。Diffie 和 Hellman 于 1976 年开发的非对称密钥一致性协议。使用该协议，两个用户可以在以前没有任何密钥的情况下通过不安全的介质交换密钥。Diffie-Hellman 由 Kerberos 使用。
digest（摘要）	请参见 message digest（消息摘要）。
DSA	Digital Signature Algorithm（数字签名算法）。一种公钥算法，采用大小可变（512 位到 4096 位）的密钥。美国政府标准 DSS 可达 1024 位。DSA 的输入依赖于 SHA1。
ECDSA	Elliptic Curve Digital Signature Algorithm（椭圆曲线数字签名算法）。一种基于椭圆曲线数学运算的公钥算法。在生成相同长度的签名时，所需的 ECDSA 密钥大小明显小于 DSA 公钥大小。
effective set（有效特权集合）	当前对进程有效的特权集合。
flavor（特性）	以前，*security flavor*（安全特性）和 *authentication flavor*（验证特性）具有相同的含义，都是表示验证类型 (AUTH_UNIX, AUTH_DES, AUTH_KERB) 的特性。RPCSEC_GSS 也是一种安全特性，虽然它除了验证之外还提供完整性和保密性服务。
forwardable ticket（可转发票证）	一种票证，可供客户机在不需要完成远程主机上的完整验证过程的情况下用于请求此主机票证。例如，如果用户 `david` 登录到用户 `jennifer` 的计算机时获取了一张可转发票证，则 `david` 不必获取新的票证（从而对自身进行重新验证）即可登录到自己的计算机。另请参见 proxiable ticket（可代理票证）。
FQDN	Fully qualified domain name（全限定域名）。例如，`central.example.com`（与简单的 `denver` 相对）。

GSS-API	Generic Security Service Application Programming Interface（通用安全服务应用编程接口）。为各种模块化安全服务（包括 Kerberos 服务）提供支持的网络层。GSS-API 可用于安全验证服务、完整性服务和保密性服务。另请参见 authentication（验证）、integrity（完整性）和 privacy（保密性）。
hardening（强化）	为了删除主机中固有的安全漏洞而对操作系统的缺省配置进行的修改。
hardware provider（硬件提供者）	在 Oracle Solaris 的加密框架功能中，是指设备驱动程序及其硬件加速器。硬件提供者使计算机系统不必执行开销很大的加密操作，从而可释放 CPU 资源以用于其他用途。另请参见 provider（提供者）。
host principal（host 主体）	服务主体的一个特定实例，其中将主体（由主名称 `host` 表示）设置为提供一系列网络服务，如 `ftp`、`rcp` 或 `rlogin`。例如，`host/central.example.com@EXAMPLE.COM` 便是一个主机主体。另请参见 server principal（服务器主体）。
host（主机）	可通过网络进行访问的系统。
inheritable set（可继承特权集合）	进程可以通过调用 `exec` 而继承的特权集合。
initial ticket（初始票证）	直接颁发（即，不基于现有的票证授予票证）的票证。某些服务（如用于更改口令的应用程序）可能需要将票证标记为 `initial`，以便使其自身确信客户机知晓其密钥。这种保证非常重要，因为初始票证表明客户机最近已进行了自我验证（而并非依赖于存在时间可能较长的票证授予票证）。
instance（实例）	实例是主体名称的第二个部分，用于限定主体的主名称。对于服务主体，实例是必需的。实例就是主机的全限定域名，例如 `host/central.example.com`。对于用户主体，实例是可选的。但是请注意，`jdoe` 和 `jdoe/admin` 都是唯一的主体。另请参见 primary（主）、principal name（主体名称）、service principal（服务主体）和 user principal（用户主体）。
integrity（完整性）	一种安全服务，除了用于用户验证之外，还用于通过加密校验和来验证传输数据的有效性。另请参见 authentication（验证） 和 privacy（保密性）。
invalid ticket（无效票证）	尚未成为可用票证的以后生效的票证。应用服务器将拒绝无效票证，直到此票证生效为止。要使无效票证生效，必须在其开始时间已过后，由客户机通过 TGS 请求将其提供给 KDC，同时设置 `VALIDATE` 标志。另请参见 postdated ticket（以后生效的票证）。
KDC	Key Distribution Center（密钥分发中心）。具有以下三个 Kerberos V5 组件的计算机：

- 主体和密钥数据库
- 验证服务
- 票证授予服务

每个领域都具有一个主 KDC，并且应该具有一个或多个从 KDC。

Kerberos

是指一种验证服务、此服务所使用的协议或者用于实现此服务的代码。

Oracle Solaris 中的 Kerberos 实现主要基于 Kerberos V5 实现。

虽然在技术方面有所不同，但是在 Kerberos 文档中经常会互换使用 "Kerberos" 和 "Kerberos V5"。

Kerberos（也可写成 Cerberus）在希腊神话中是指守护地狱之门的三头凶悍猛犬。

Kerberos policy（Kerberos 策略）

管理 Kerberos 服务中口令的使用的规则集合。这些策略可以控制主体的访问权限或票证参数（如生命周期）。

key（密钥）

1. 通常是指以下两种主要密钥类型之一：

- 对称密钥 – 与解密密钥相同的加密密钥。对称密钥用于对文件进行加密。
- 非对称密钥或公钥 – 在公钥算法（如 Diffie-Hellman 或 RSA）中使用的密钥。公钥包括仅对一个用户公开的私钥、服务器或通用资源所使用的公钥，以及包含这两者的私钥/公钥对。私钥 (private key) 也称为密钥 (secret key)。公钥也称为共享密钥或公用密钥。

2. 密钥表文件中的项（主体名称）。另请参见 keytab file（密钥表文件）。

3. 在 Kerberos 中，是指加密密钥，此类密钥分为以下三种类型：

- 私钥 – 由主体和 KDC 共享并在系统范围之外分发的加密密钥。另请参见 private key（私钥）。
- 服务密钥 – 此密钥与私钥的用途相同，但由服务器和服务使用。另请参见 service key（服务密钥）。
- 会话密钥 – 在两个主体之间使用的临时加密密钥，其生命周期仅限于单个登录会话的持续时间。另请参见 session key（会话密钥）。

keystore（密钥库）

密钥库包含用于应用程序检索的口令、口令短语、证书，以及其他验证对象。密钥库可特定于一种技术，或特定于多个应用程序使用的一个位置。

keytab file（密钥表文件）

包含一个或多个密钥（主体）的密钥表文件。主机或服务使用密钥表文件的方式与用户使用口令的方式大致相同。

kvno	Key version number（密钥版本号）。按照生成顺序跟踪特定密钥的序列号。kvno 最高则表示密钥最新。
least privilege（最小特权）	一种安全模型，该模型仅向指定进程提供超级用户功能的某个子集。最小特权模型为一般用户指定可以用来执行个人管理任务（如挂载文件系统和更改文件的所有权）的足够特权。另一方面，仅使用完成该任务所需的特权运行进程，而不是使用超级用户的完全功能模式（即所有特权）。对非 root 用户而言，可以包含由于编程错误而导致的损坏（如缓冲区溢出），该用户对重要功能（如读取或写入受保护的系统文件或停止计算机）没有访问权限。
limit set（限制特权集合）	对哪些特权可用于进程及其子进程的外部限制。
MAC	1. 请参见 message authentication code, MAC（消息验证代码）。 2. 也称为标签设置操作。在政府安全术语中，MAC 是指 Mandatory Access Control（强制访问控制）。例如，Top Secret（绝密）和 Confidential（机密）之类的标签便是 MAC。MAC 与 DAC 相对，后者是指 Discretionary Access Control（自主访问控制）。例如，UNIX 权限便是一个 DAC。 3. 在硬件中，是指 LAN 中的唯一系统地址。如果系统位于以太网中，则 MAC 是指以太网地址。
master KDC（主 KDC）	每个领域中的主要 KDC，包括 Kerberos 管理服务器 `kadmind`，以及验证和票证授予守护进程 `krb5kdc`。每个领域至少都必须具有一个主 KDC，可以具有多个 KDC 副本或从 KDC，这些 KDC 为客户机提供验证服务。
MD5	一种重复加密散列函数，用于进行消息验证（包含数字签名）。该函数于 1991 年由 Rivest 开发。其使用已过时。
mechanism（机制）	1. 指定加密技术以实现数据验证或保密的软件包。例如：Kerberos V5、Diffie-Hellman 公钥。 2. 在 Oracle Solaris 的加密框架功能中，是指用于特殊用途的算法的实现。例如，应用于验证的 DES 机制（如 CKM_DES_MAC）与应用于加密的 DES 机制（如 CKM_DES_CBC_PAD）不同。
message authentication code, MAC（消息验证代码）	MAC 可确保数据的完整性，并验证数据的来源。MAC 不能防止窃听。
message digest（消息摘要）	消息摘要是从消息中计算所得的散列值。此散列值几乎可唯一地标识消息。摘要对检验文件的完整性非常有用。
minimization（最小安装）	运行服务器所需的最小操作系统安装。不安装与服务器操作不直接相关的任何软件，或者在安装之后即删除。
name service scope（名称服务范围）	允许角色在其中执行操作的范围，即，由指定的命名服务（如 NIS 或 LDAP）提供服务的单个主机或所有主机。

network application server（网络应用服务器）	提供网络应用的服务器，如 `ftp`。一个领域可以包含多个网络应用服务器。
network policies（网络策略）	网络实用程序为了保护网络通信而配置的设置。有关网络安全性的信息，请参见《在 Oracle Solaris 11.2 中确保网络安全 》。
nonattributable audit event（无归属审计事件）	无法确定其触发者的审计事件，如 AUE_BOOT 事件。
NTP	Network Time Protocol（网络时间协议）。由特拉华大学开发的软件，可用于在网络环境中管理准确时间或网络时钟同步，或者同时管理这两者。可以使用 NTP 在 Kerberos 环境中维护时钟相位差。另请参见 clock skew（时钟相位差）。
PAM	Pluggable Authentication Module（可插拔验证模块）。一种框架，允许使用多种验证机制而不必重新编译运行这些机制的服务。PAM 可用于在登录时初始化 Kerberos 会话。
passphrase（口令短语）	一种短语，用于验证某个私钥是否是由口令短语用户创建。理想的口令短语应包含 10-30 个字符，请混合使用字母和数字字符，并且避免简单的文本结构和名称。使用私钥对通信执行加密和解密操作时，系统会提示您提供口令短语进行验证。
password policy（口令策略）	可用于生成口令的加密算法，还可以指与口令有关的更普遍的问题，如必须对口令进行更改的频率，允许的口令尝试次数以及其他安全注意事项。安全策略需要口令。口令策略可能要求使用 AES 算法对口令进行加密，并可能对口令强度提出进一步要求。
permitted set（允许特权集合）	可供进程使用的特权集合。
policy for public key technologies（公钥技术的策略）	在密钥管理框架 (Key Management Framework, KMF) 中，所实现的策略是管理证书的使用。KMF 策略数据库可以对由 KMF 库管理的密钥和证书的使用施加约束。
policy in the Cryptographic Framework（加密框架中的策略）	在 Oracle Solaris 的加密框架功能中，所实现的策略是禁用现有的加密机制。从而使这些机制不可使用。加密框架中的策略可能会阻止使用提供者（如 DES）提供的特殊机制，如 `CKM_DES_CBC`。
policy（策略）	一般而言，是指影响或决定决策和操作的操作规划或操作过程。对于计算机系统，策略通常表示安全策略。站点的安全策略是规则集合和相关措施，可用于定义所处理信息的敏感度并防止信息受到未经授权的访问。例如，安全策略可能要求对系统进行审计，必须分配设备才能使用，以及每六周必须更改一次口令。 有关在 Oracle Solaris OS 特定区域中实施策略的信息，请参见 audit policy（审计策略）、policy in the Cryptographic Framework（加密框架中的策略）、device policy（设备策略）、Kerberos

	policy（Kerberos 策略）、password policy（口令策略）和 rights policy（权限策略）。
postdated ticket（以后生效的票证）	以后生效的票证直到创建之后的某一指定时间才能开始生效。此类票证对于计划在深夜运行的批处理作业等情况非常有用，因为在运行批处理作业之前无法使用该票证（即使被盗）。颁发以后生效的票证时，将以 `invalid` 状态颁发该票证，并在出现以下情况之前一直保持此状态：a) 票证开始时间已过，并且 b) 客户机请求 KDC 进行验证。通常，以后生效的票证在票证授予票证的截止时间之前会一直有效。但是，如果将以后生效的票证标记为 `renewable`，则通常会将其生命周期设置为等于票证授予票证的整个生命周期的持续时间。另请参见 invalid ticket（无效票证）和 renewable ticket（可更新票证）。
primary（主）	主体名称的第一部分。另请参见 instance（实例）、principal name（主体名称）和 Realm（领域）。
principal name（主体名称）	1. 主体的名称，格式为 *primary/instance@REALM*。另请参见 instance（实例）、primary（主）和 Realm（领域）。 2.(RPCSEC_GSS API) 请参见 client principal（客户机主体）和 server principal（服务器主体）。
principal（主体）	1. 参与网络通信并且具有唯一名称的客户机/用户或服务器/服务实例。Kerberos 事务涉及主体之间（服务主体与用户主体）或主体与 KDC 之间的交互。换言之，主体是 Kerberos 可为其指定票证的唯一实体。另请参见 principal name（主体名称）、service principal（服务主体）和 user principal（用户主体）。 2.(RPCSEC_GSS API) 请参见 client principal（客户机主体）和 server principal（服务器主体）。
principle of least privilege（最小特权原则）	请参见 least privilege（最小特权）。
privacy（保密性）	一种安全服务，其中传输的数据加密之后才会发送。保密性还包括数据完整性和用户验证。另请参见 authentication（验证）、integrity（完整性）和 service（服务）。
private key（私钥）	为每个用户主体提供的密钥，并且只对主体的用户和 KDC 公开。对于用户主体，密钥基于用户的口令。另请参见 key（密钥）。
private-key encryption（私钥加密）	采用私钥加密时，发送者和接收者使用相同的加密密钥。另请参见 public-key encryption（公钥加密）。
privilege escalation（特权升级）	可以访问在所指定权限（包括覆盖缺省设置的权限）允许的资源范围以外的资源。特权升级的结果是某个进程可以执行未经授权的操作。
privilege model（特权模型）	计算机系统上比超级用户模型更为严格的安全模型。在特权模型中，进程需要具有相应的特权才能运行。系统管理可以分为多个独立的部

分，这些部分基于管理员在其进程中所具有的特权。可以将特权指定给管理员的登录过程。或者，可以指定特权只对特定命令有效。

privilege set（特权集合）

特权的集合。每个进程都有四个特权集合，用于确定进程是否可以使用特定特权。请参见 limit set（限制特权集合）、effective set（有效特权集合）、permitted set（允许特权集合）和 inheritable set（可继承特权集合）。

此外，特权的 basic set（基本特权集合）是指登录时为用户进程指定的特权集合。

privilege-aware（可识别特权）

在其代码中启用和禁用特权的程序、脚本和命令。在生产环境中，启用的特权必须提供给进程，例如，通过要求程序的用户使用将特权添加到程序中的权限配置文件。有关特权的完整说明，请参见 `privileges(5)` 手册页。

privilege（特权）

1. 通常是指在某个计算机系统上执行一般用户所无法执行的操作的能力。超级用户特权是向超级用户授予的所有 rights（权限）。特权用户或特权应用程序是指获得了额外权限的用户或应用程序。

2. Oracle Solaris 系统中的进程具有的独立权限。与 `root` 相比，特权可提供更为精细的进程控制。特权是在内核中定义和实施的。特权也称为进程特权或内核特权。有关特权的完整说明，请参见 `privileges(5)` 手册页。

privileged application（特权应用程序）

可以覆盖系统控制的应用程序。该应用程序可以检查安全属性（如特定的 UID、GID、授权或特权）。

privileged user（特权用户）

计算机系统上为其指定的权限高于一般用户权限的用户。另请参见 trusted users（可信用户）。

profile shell（配置文件 shell）

在权限管理中，角色（或用户）可通过该 shell 从命令行运行指定给角色权限配置文件的任何特权应用程序。配置文件 shell 版本与系统上可用的 shell 对应（例如 `bash` 的 `pfbash` 版本）。

provider（提供者）

在 Oracle Solaris 的加密框架功能中，是指为使用者提供的加密服务。例如，PKCS #11 库、内核加密模块和硬件加速器便是提供者。提供者可插入到加密框架中，因此也称为插件。有关使用者的示例，请参见 consumer（使用者）。

proxiable ticket（可代理票证）

可供服务用于代表客户机执行客户机操作的票证。因此，可以说服务充当客户机的代理。使用该票证，服务便可具有客户机的身份。服务可以使用可代理票证来获取其他服务的服务票证，但是不能获取票证授予票证。可代理票证与可转发票证之间的区别在于可代理票证只对单项操作有效。另请参见 forwardable ticket（可转发票证）。

public object（公共对象）

`root` 用户所拥有且全局可读的文件，如 `/etc` 目录中的任何文件。

public-key encryption（公钥加密）	一种加密方案，其中每个用户都有两个密钥：一个是公钥，一个是私钥。采用公钥加密时，发送者使用接收者的公钥对消息进行加密，而接收者则使用私钥对其进行解密。Kerberos 服务是一种私钥系统。另请参见 private-key encryption（私钥加密）。
QOP	Quality of Protection（保护质量）。用于选择与完整性服务或保密性服务结合使用的加密算法的参数。
RBAC	Role-based access control（基于角色的访问控制），Oracle Solaris 的一项用户权限管理功能。请参见 rights（权限）。
RBAC policy（RBAC 策略）	请参见 rights policy（权限策略）。
Realm（领域）	1. 由单个 Kerberos 数据库以及一组密钥分发中心 (Key Distribution Center, KDC) 提供服务的逻辑网络。 2. 主体名称的第三部分。对于主体名称 `jdoe/admin@CORP.EXAMPLE.COM`，领域为 `CORP.EXAMPLE.COM`。另请参见 principal name（主体名称）。
reauthentication（重新验证）	执行计算机操作需要提供口令。通常，`sudo` 操作需要重新验证。需要验证权限配置文件可包含需要重新验证的命令。请参见 authenticated rights profile（需要验证权限配置文件）。
relation（关系）	在 `kdc.conf` 或 `krb5.conf` 文件中定义的配置变量或关系。
renewable ticket（可更新票证）	由于票证的生命周期过长会存在安全风险，因此可以将票证指定为 `renewable`。可更新票证有两个截止时间：a) 票证的当前实例的截止时间，b) 任意票证的最长生命周期。如果客户机需要继续使用某票证，则可在首次失效之前更新此票证。例如，某个票证的有效期为 1 小时，所有票证的最长生命周期为 10 小时。如果持有票证的客户机希望保留此票证的时间长于 1 小时，则必须更新此票证。当某个票证达到最长票证生命周期时，便会自动到期，并且无法更新。
rights policy（权限策略）	与命令关联的安全策略。当前，`solaris` 是 Oracle Solaris 的有效策略。`solaris` 策略可识别特权和扩展特权策略、授权及 `setuid` 安全属性。
rights profile（权限配置文件）	也称为配置文件。指的是可以指定给角色或用户的安全设置覆盖值的集合。权限配置文件可包括授权、特权、具有安全属性的命令和称为补充配置文件的其他权限配置文件。
rights（权限）	对超级用户模型（管理员对系统要么具有全部控制权要么毫无控制权）的替代。通过用户权限管理和进程权限管理，组织可划分超级用户的特权并将其指定给用户或角色。Oracle Solaris 中的权限实施方式有内核特权、授权和以特定 UID 或 GID 运行进程的能力。可在 rights profile（权限配置文件）和 role（角色）中收集权限。

role（角色）	一种用于运行特权应用程序的特殊身份，仅指定用户才能承担此身份。
RSA	获取数字签名和公钥密码系统的方法。该方法于 1978 年首次由其开发者 Rivest、Shamir 和 Adleman 介绍。
scan engine（扫描引擎）	第三方应用程序，驻留在外部主机上，可检查文件中是否含有已知病毒。
SEAM	这是 Solaris 系统上的 Kerberos 初始版本的产品名。该产品基于麻省理工学院开发的 Kerberos V5 技术。SEAM 现在称为 Kerberos 服务。其特性与 MIT 版本仍稍有不同。
secret key（密钥）	请参见 private key（私钥）。
Secure Shell（安全 Shell）	一种特殊协议，用于在不安全的网络中进行安全远程登录并提供其他安全网络服务。
security attributes（安全属性）	是指当超级用户以外的用户运行管理命令时，可使此命令成功执行的安全策略覆盖项。在超级用户模型中，`setuid root` 和 `setgid` 程序都是安全属性。将这些属性应用于某命令时，此命令便会成功执行，而与运行它的用户无关。在 privilege model（特权模型）中，内核特权及其他 rights（权限）会将 `setuid root` 程序替换为安全属性。特权模型与超级用户模型兼容，因为特权模型也可将 `setuid` 和 `setgid` 程序识别为安全属性。
security flavor（安全特性）	请参见 flavor（特性）。
security mechanism（安全机制）	请参见 mechanism（机制）。
security policy（安全策略）	请参见 policy（策略）。
security service（安全服务）	请参见 service（服务）。
seed（种子）	用于生成随机数的数字起动器。当起动器来自随机源时，种子称为随机种子。
separation of duty（职责分离）	least privilege（最小特权）的部分概念。职责分离可阻止一个用户执行或批准完成事务的所有操作。例如，在RBAC中，可以将登录用户的创建与安全覆盖的指定分隔开来。一个角色创建该用户。另一个角色可以将安全属性（如权限配置文件、角色和特权）指定给现有用户。
server principal（服务器主体）	(RPCSEC_GSS API) 提供服务的主体。服务器主体以 *service*@*host* 形式的 ASCII 字符串进行存储。另请参见 client principal（客户机主体）。
server（服务器）	为网络客户机提供资源的主体。 例如，如果通过 `ssh` 远程登录到系统 `central.example.com`，则该系统便是提供 `ssh` 服务的服务器。另请参见 service principal（服务主体）。

service key（服务密钥）	由服务主体和 KDC 共享，并在系统范围之外分发的加密密钥。另请参见 key（密钥）。
service principal（服务主体）	为一项或多项服务提供 Kerberos 验证的主体。对于服务主体，主名称是服务的名称（如 `ftp`），其实例是提供服务的系统的全限定主机名。另请参见 host principal（host 主体）和 user principal（用户主体）。
service（服务）	1. 通常由多台服务器提供给网络客户机的资源。例如，如果通过 `rlogin` 远程登录到计算机 `central.example.com`，则该计算机便是提供 `rlogin` 服务的服务器。 2. 除验证之外，还提供其他保护级别的安全服务（完整性或保密性）。另请参见 integrity（完整性）和 privacy（保密性）。
session key（会话密钥）	由验证服务或票证授予服务生成的密钥。生成会话密钥的目的是在客户机与服务之间提供安全事务。会话密钥的生命周期仅限于单个登录会话的持续时间。另请参见 key（密钥）。
SHA1	Secure Hashing Algorithm（安全散列算法）。该算法可以针对长度小于 2^{64} 的任何输入进行运算，以生成消息摘要。SHA1 算法是 DSA 的输入。
single-system image（单系统映像）	单系统映像用在 Oracle Solaris 审计中来描述使用相同命名服务的一组受审计系统。这些系统将其审计记录发送给某个中心审计服务器，可在该服务器中对记录进行比较，就像这些记录来自一个系统一样。
slave KDC（从 KDC）	主 KDC 的副本，可以执行主 KDC 的大多数功能。每个领域通常都具有若干个从 KDC（但仅有一个主 KDC）。另请参见 KDC 和 master KDC（主 KDC）。
software provider（软件提供者）	在 Oracle Solaris 的加密框架功能中，是指提供加密服务的内核软件模块或 PKCS #11 库。另请参见 provider（提供者）。
stash file（存储文件）	存储文件包含 KDC 主密钥的已加密副本。当重新引导服务器以便在 KDC 启动 `kadmind` 和 `krb5kdc` 进程之前自动验证 KDC 时，将使用此主密钥。由于存储文件中包含主密钥，因此，应该保证存储文件及其任何备份的安全。如果加密受到威胁，则可以使用此密钥来访问或修改 KDC 数据库。
superuser model（超级用户模型）	计算机系统上的典型 UNIX 安全模型。在超级用户模型中，管理员对系统要么具有全部的控制权要么毫无控制权。通常，为了管理计算机，用户可成为超级用户 (`root`)，并可执行所有管理活动。
synchronous audit event（同步审计事件）	审计事件中的大多数事件属于同步审计事件。这些事件与系统中的某个进程关联。与某个进程关联的无归属事件属于同步事件，如失败的登录。
TGS	Ticket-Granting Service（票证授予服务）。负责颁发票证的那部分 KDC。

TGT	Ticket-Granting Ticket（票证授予票证）。由 KDC 颁发的票证，客户机可使用此票证来请求其他服务的票证。
ticket file（票证文件）	请参见 credential cache（凭证高速缓存）。
ticket（票证）	用于安全地将用户身份传递给服务器或服务的信息包。一个票证仅对一台客户机以及某台特定服务器上的一项特殊服务有效。票证包含服务的主体名称、用户的主体名称、用户主机的 IP 地址、时间戳以及定义此票证生命周期的值。票证是通过由客户机和服务使用的随机会话密钥创建的。一旦创建了票证，便可重复使用此票证，直到其到期为止。票证与新的验证者同时出现时，仅用于验证客户机。另请参见 authenticator（验证者）、credential（凭证）、service（服务）和 session key（会话密钥）。
trusted users（可信用户）	指的是您决定允许其在一定信任级别下执行管理任务的用户。通常，管理员先为可信用户创建登录名，并指定与此类用户的信任级别和能力匹配的管理权限。之后，这些用户便能帮助配置和维护系统。此类用户也称为特权用户。
user principal（用户主体）	属于某个特定用户的主体。用户主体的主名称是用户名，其可选实例是用于说明相应凭证预期用法的名称（例如 `jdoe` 或 `jdoe/admin`）。也称为用户实例。另请参见 service principal（服务主体）。
virtual private network, VPN（虚拟专用网络）	通过使用加密和隧道连接公共网络上的用户来提供安全通信的网络。

索引

I

J

K

Q

R

T

Z